Formale Grundlagen der Informatik

Herbstsemester 2020

Prof. Dr. Matthias Krause (Sprecher)
Dr. Matthias Hamann

Endliche Automaten und Reguläre Ausdrücke

(Stand: 11.11.2020)

Lehrstuhl für Theoretische Informatik
Universität Mannheim

Endliche Automaten

Endliche Automaten

Eine Einführung

Endliche Automaten (informale Definition)

Endliche Automaten bilden ein Modell zur formalen Spezifikation von endlichen, durch ein gewünschtes Ein-/Ausgabeverhalten definierten Systemen $\mathcal{S}$.

Sie dienen als Baupläne zur Erstellung praktischer digitaler Realisierungen von $\mathcal{S}$.

Endliche Automaten A sind taktweise arbeitende Maschinen, die in jedem Takt ...

- ... sich in einem Zustand $q \in Q$ aus einer **endlichen Zustandsmenge** Q befinden,
- ... eine Eingabeaktion σ aus einer **endlichen Menge Σ von Eingabeaktionen** entgegennehmen können,
- ... gemäß der (ggf. partiellen) Ausgabefunktion $\alpha : Q \times \Sigma \to \Omega$ eine Ausgabeaktion $\omega = \alpha(q, \sigma)$ aus einer endlichen Menge Ω von möglichen Ausgabeaktionen ausführen,
- ... gemäß der (ggf. partiellen) Zustandsüberführungsfunktion $\delta : Q \times \Sigma \to Q$ in einen neuen Zustand $q' = \delta(q, \sigma)$ wechseln.

Formal: $A = (Q, q_0, \Sigma, \Omega, \alpha, \delta)$, wobei $q_0 \in Q$ den Anfangszustand des Systems bezeichnet.

Modellierung eines einfachen Getränkeautomaten

Auswahl: Cola, Fanta (Preis jeweils zwei Euro) und Mineralwasser (Preis ein Euro).

Bezahlt werden kann nur mit Ein-Euro-Stücken.

Definieren $A = (Q, q_0, \Sigma, \Omega, \sigma, \alpha)$ durch

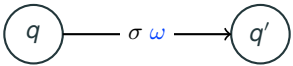
- $\Sigma = \{DrF, DrC, DrM, WE\}$, d.h., drücke Fanta bzw. Cola bzw. Mineralwasser bzw. werfe eine Euromünze ein,
- $\Omega = \{GF, GC, GM\}$ d.h., gebe Fanta bzw. Cola bzw. Mineralwasser aus.
- $Q = \{q_0, F_0, F_1, C_0, C_1, M_0\}$ (Zustand F_0 bzw. F_1 heißt zahle noch 2 bzw. 1 Euro für die Fanta, C_0, C_1, M_0 entsprechend).
- $\delta(q_0, DrF) = F_0$, $\delta(q_0, DrC) = C_0$, $\delta(q_0, DrM) = M_0$
- $\delta(C_0, WE) = C_1$, $\delta(F_0, WE) = F_1$, $\delta(M_0, WE) = q_0$
- $\delta(C_1, WE) = q_0$, $\delta(F_1, WE) = q_0$
- $\alpha(C_1, WE) = GC$, $\alpha(F_1, WE) = GF$, $\alpha(M_0, WE) = GM$.

Visualisierung endlicher Automaten

Endliche Automaten $A = (Q, q_0, \Sigma, \Omega, \alpha, \delta)$ werden oft durch **Graphen** mit Knoten- und Kantenmarkierungen visualisiert.

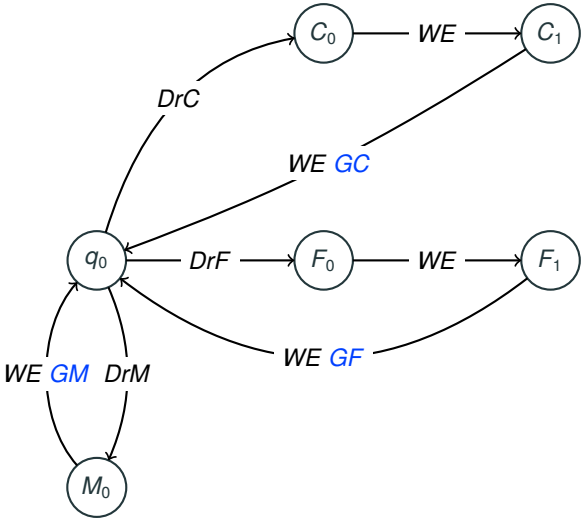
Die **Knotenmenge** entspricht der Menge der Zustände.

Für Zustandspaare q, q' und Eingabezeichen $\sigma \in \Sigma$ und Ausgabezeichen $\omega \in \Omega$ gilt, dass Zustandsübergänge $q' = \delta(q, \sigma)$ und Ausgabeaktionen $\omega = \alpha(q, \sigma)$ durch eine Kante von q nach q' visualisiert werden, die mit $\sigma\ \omega$ markiert ist.



Die partiellen Funktionen δ und α werden oft als **Zustandsüberführungs- bzw. Ausgabetabelle** notiert, deren Zeilen und Spalten mit den Elementen aus Q bzw. Σ markiert sind, und für die an Position (q, σ) der Eintrag $\delta(q, \sigma)$ bzw. $\alpha(q, \sigma)$ steht.

Beispiel Getränkeautomat



Beispiel FlipFlop

Ein FlipFlop ist ein Speicherbaustein, der eine 0 oder eine 1 speichern kann.

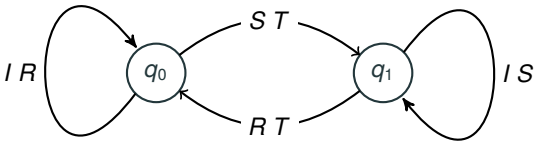
Ein FlipFlop reagiert auf folgende Befehle:

- I: behalte den gespeicherten Wert bei,
- R: (reset) setze den gespeicherten Wert auf 0 zurück,
- S: (set) setze den gespeicherten Wert auf 1 hoch,
- T: (toggle) negiere den gespeicherten Wert.

Formal: $A = (Q = \{0, 1\}, q_0, \Sigma = \{I, R, S, T\}, \delta)$ mit

$\delta(q, \sigma)$	I	R	S	T
q_0	q_0	q_0	q_1	q_1
q_1	q_1	q_0	q_1	q_0

Der FlipFlop Automat als Graph



Definition 11.1

Entscheidungsprobleme Π sind definiert durch Funktionen $\Pi : X \longrightarrow \{0, 1\}$, die Eingaben x aus einer Eingabemenge X entweder den Wert $\Pi(x) = 0$ (verwerfen, *reject*) oder den Wert $\Pi(x) = 1$ (akzeptieren, *accept*) zuordnen.

Beispiel: Entscheide, ob eine natürliche Zahl $n \in \mathbb{N}$ durch 3 teilbar ist.

Aufgabenstellung: Konstruiere für ein gegebenes Entscheidungsprobleme $\Pi : X \longrightarrow \{0, 1\}$ einen endlichen Automaten A, der für jede Eingabe $x \in X$ die Ausgabe $\Pi(x) \in \{0, 1\}$ ausgibt.

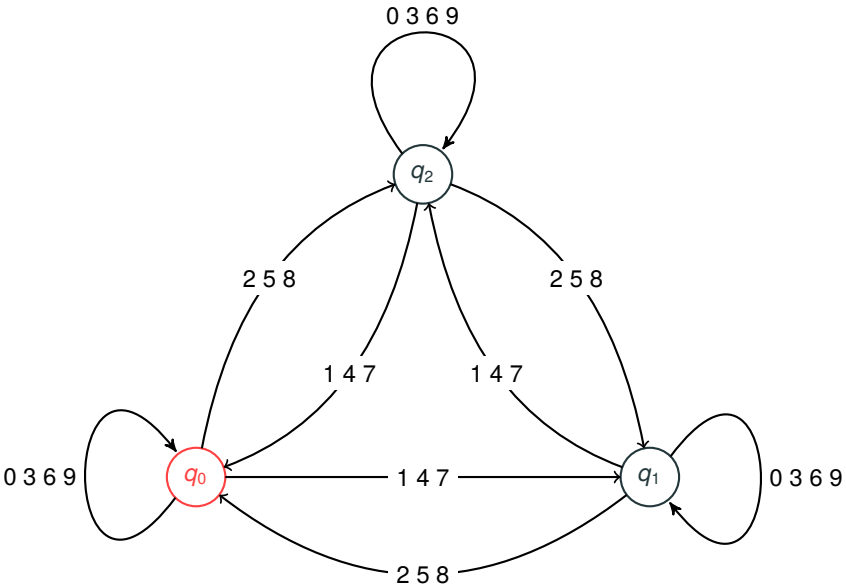
Vorgehen: Wir kodieren die Eingaben $x \in X$ als Worte über einem endlichen Alphabet Σ.

Im Beispiel: Natürliche Zahlen kodiert als Dezimalzahlen, also $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

Konstruktion: Wir konstruieren A über der Eingabemenge Σ mit verwerfenden und akzeptierenden Zuständen so, dass jede Eingabe x genau dann in einem akzeptierenden Zustand landet, wenn $\Pi(x) = 1$ ist.

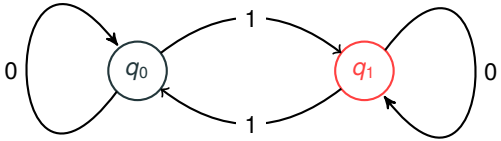
Im Beispiel: $\Pi(x) = 1$ genau dann, wenn die Quersumme durch 3 teilbar ist.

Beispiel Teilbarkeit durch 3



Weiteres Beispiel

Der folgende endliche Automat berechnet das Entscheidungsproblem *Parity*, ob eine gegebene endliche Folge aus Nullen und Einsen eine ungerade Zahl von Einsen enthält.



Wir steuern im Folgenden auf die formalen Definitionen rund um die Berechnung von Entscheidungsproblemen durch endliche Automaten zu und müssen dazu eine Reihe neuer Begriffe klären.

Buchstaben, Wörter, Konketanation

Es sei Σ ein **endliches Alphabet**, bestehend aus Buchstaben.

Definition 11.2

- Für alle natürliche Zahlen $n \geq 1$ bezeichnen wir n-Tupel $x = (x_1, \ldots, x_n)$ aus Σ^n als **Worte der Länge** $|x| = n$ **über** Σ.
- Mit ϵ bezeichnen wir das **leere Wort** der Länge 0.
- Die Menge $\Sigma^* = \bigcup_{n=1}^{\infty} \Sigma^n \cup \{\epsilon\}$ wird als **Menge der Worte über** Σ bezeichnet.
- Teilmengen $L \subseteq \Sigma^*$ von Σ^* heißen **(formale) Sprachen über dem Alphabet** Σ.
- Für natürliche Zahlen $1 \leq m \leq n$ und $x = (x_1, \ldots, x_n) \in \Sigma^n$ bezeichnet $(x_1, \ldots, x_m)$ das **Präfix** der Länge m von x, und $x = (x_{n-m+1}, \ldots, x_n)$ das **Suffix** der Länge m von x.

$$x = (x_1, \ldots, x_m, x_{m+1}, \cdots, x_{n-m}, x_{n-m+1}, \ldots, x_n).$$

- Das leere Wort ϵ bildet Präfix und Suffix der Länge 0 von x.
- Für $x, y \in \Sigma^*$ bildet die **Konketanation** xy das eindeutig bestimmte Wort der Länge $|x| + |y|$ mit Präfix x und Suffix y.

Beziehung zu natürlichen Sprachen (z.B. DEUTSCH)

Zugrundeliegendes Alphabet: Menge aller Kein- und Großbuchstaben inklusive Umlaute, alle Satzzeichen inklusive dem Leerzeichen, Ziffern

DEUTSCH besteht aus allen syntaktisch und semantisch korrekten Texten in deutscher Sprache.

Beispiele:

- *Uffbasse!* $\notin$ DEUTSCH,
- *Aufpassen!* $\in$ DEUTSCH,
- *Der wo gestern hier war.* $\notin$ DEUTSCH,
- *Der der gestern hier war.* $\in$ DEUTSCH,
- *FGDI war der interessanteste Kurs meines ganzen Studiums!* $\in$ DEUTSCH

Lemma 11.3

Σ^* *bildet einen Monoid bzgl. der Konketanation, neutrales Element ist* ϵ. $\square$

Wichtig: Die Begriffe **Sprache über** Σ und **Entscheidungsproblem** über Σ^* werden in Abhängigkeit vom gegebenen Kontext oft **synonym** verwendet.

Jede Sprache $L \subseteq \Sigma^*$ definiert ein Entscheidungsproblem $\chi_L : \Sigma^* \longrightarrow \{0, 1\}$ durch $\chi_L(x) = 1 \iff x \in L$ und umgekehrt.

Definition 11.4

Für Sprachen $L, L_1, L_2 \subseteq \Sigma^*$ seien neben den bekannten mengentheoretischen Operationen **Komplement** $\overline{L} = \Sigma^* \setminus L$, **Vereinigung** $L_1 \cup L_2$, **Durchschnitt** $L_1 \cap L_2$ die folgenden Operationen definiert:

- Die **Konketanation** $L_1 \cdot L_2 = \{uv; u \in L_1 \wedge v \in L_2\}$ von Sprachen.
- Die ***n*-fache Konketanation** $L^n = \{u_1 u_2 \cdots u_n; u_i \in L$ für alle $i = 1, \cdots, n\}$ einer Sprache.
- Der Kleene'sche[1] Abschluss $L^* = \bigcup_{n=0}^{\infty} L^n$ von Sprachen, wobei $L^0 = \{\epsilon\}$.

[1] Stephen Cole Kleene, 1909-1994, US Mathematiker und Logiker

Endliche Automaten

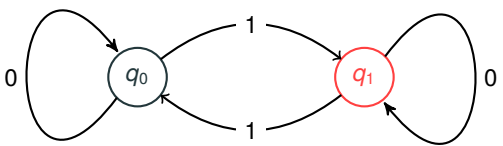
Deterministische endliche Automaten (DFAs)

Deterministische endliche Automaten (DFAs) über Σ, Definition

Definition 11.5

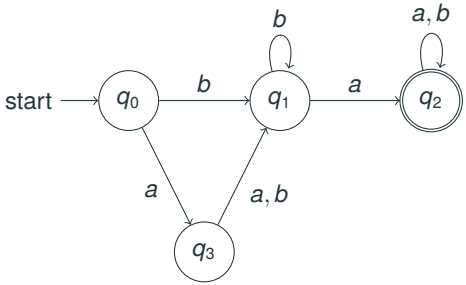
Ein **deterministischer endlicher Automat** (*deterministic finite automaton, DFA*) $A = (Q, q_0, F, \delta)$ über einem endlichen Alphabet Σ ist definiert durch eine **endliche Zustandsmenge** Q, einen **Anfangszustand** $q_0 \in Q$, eine **Menge akzeptierender Zustände** $F \subseteq Q$, und eine **Zustandsüberführungsfunktion** $\delta : Q \times \Sigma \to Q$.

Beispiel: $Q = \{q_0, q_1\}$, $F = \{q_1\}$



DFAs: Weiteres Beispiel

Beispiel: $\Sigma = \{a, b\}$



Berechnungen durch DFAs

Definition 11.6

Sei $A = (Q, q_0, F, \delta)$ ein DFA über dem Alphabet Σ und $x = (x_1, \ldots, x_n) \in \Sigma^*$ ein Wort der Länge $n \geq 1$.

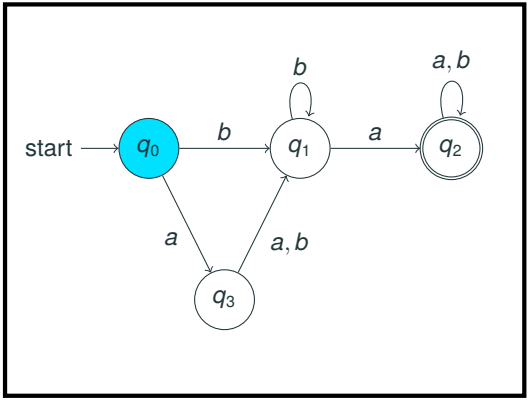
- Unter der **Berechnung von** A **auf** x **vom Zustand** $q \in Q$ **aus** versteht man die Zustandsfolge $q \xrightarrow{x_1} q_1 \xrightarrow{x_2} \cdots \xrightarrow{x_n} q_n$.
- Der Zustand q_n heißt **Zielzustand dieser Berechnung** und wird mit $\delta^*(q, x)$ bezeichnet.
- Für alle Worte $x \in \Sigma^*$ definieren wir die **Ausgabe** $A(x) \in \{0, 1\}$ von A auf Eingabe x durch $A(x) = 1 \iff \delta^*(q_0, x) \in F$.
- Die Menge $L(A) = \{x \in \Sigma^* \mid A(x) = 1\}$ heißt **die von** A **berechnete Sprache**.

Man beachte: Für die Funktion $\delta^* : Q \times \Sigma^* \longrightarrow Q$ gilt

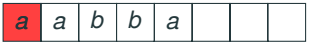
- $\delta^*(q, \epsilon) = q$
- $\delta^*(q, x\sigma) = \delta(\delta^*(q, x), \sigma)$ für $\sigma \in \Sigma$.

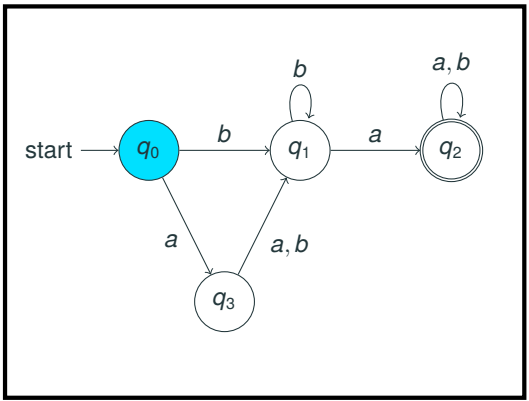
Endliche Automaten: Beispiel I (‘aabba’)

a	a	b	b	a			

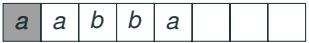


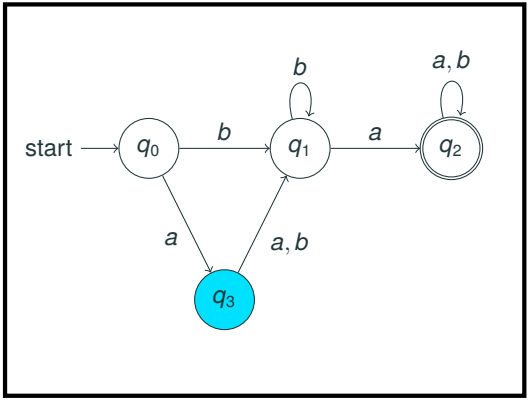
Endliche Automaten: Beispiel I (‘aabba’)



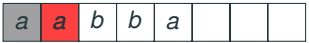


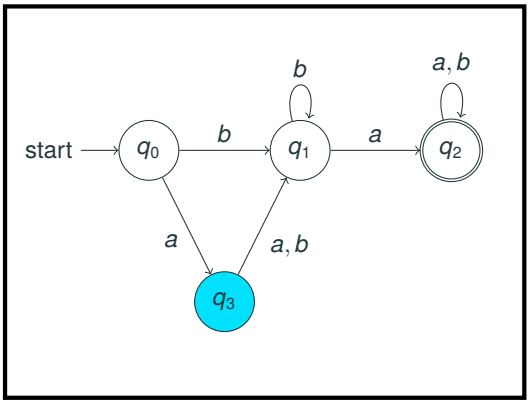
Endliche Automaten: Beispiel I (‘aabba’)



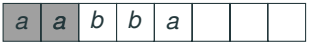


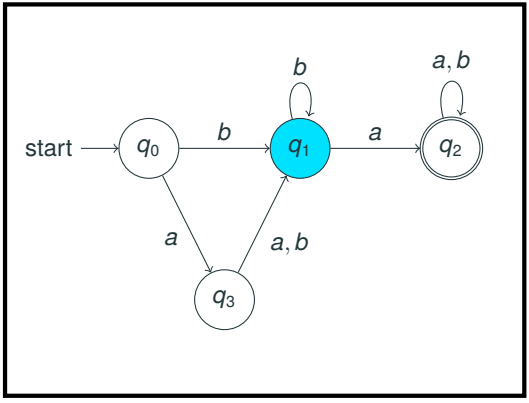
Endliche Automaten: Beispiel I (‘aabba’)



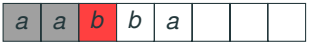


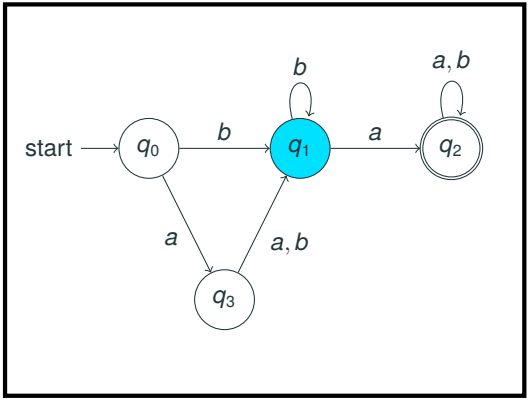
Endliche Automaten: Beispiel I (‘aabba’)



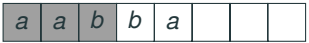


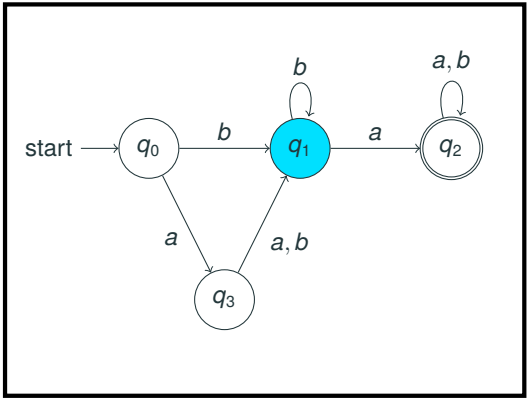
Endliche Automaten: Beispiel I (‘aabba’)



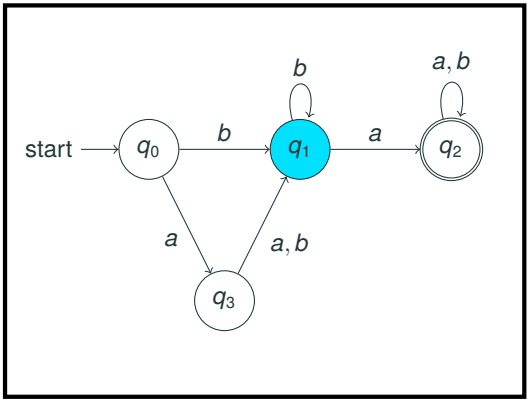


Endliche Automaten: Beispiel I (‘aabba’)

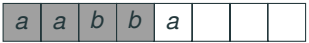


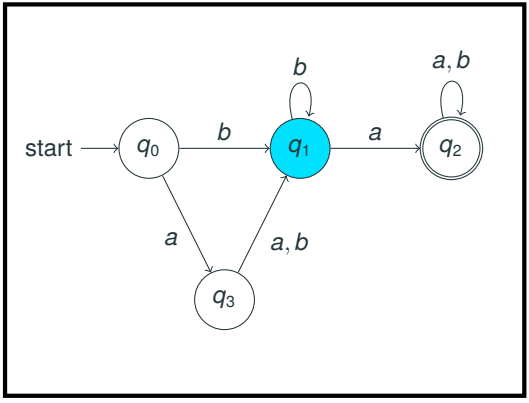


Endliche Automaten: Beispiel I (‘aabba’)

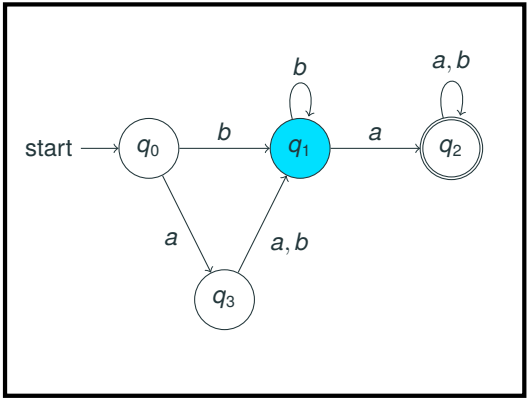


Endliche Automaten: Beispiel I (‘aabba’)

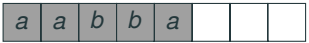


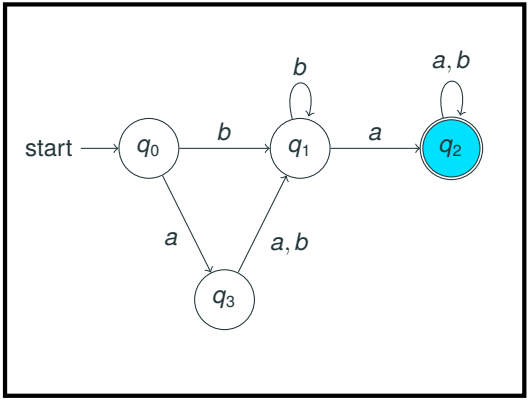


Endliche Automaten: Beispiel I (‘aabba’)

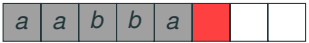


Endliche Automaten: Beispiel I (‘aabba’)

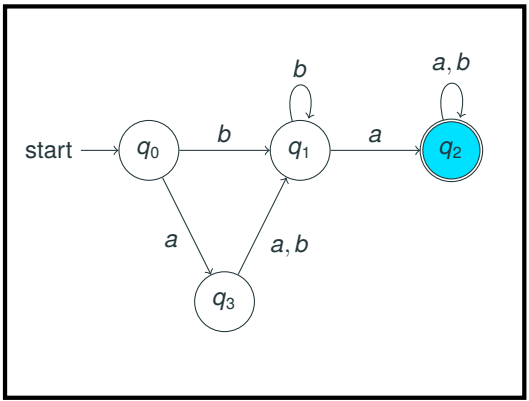




Endliche Automaten: Beispiel I (‘aabba’)

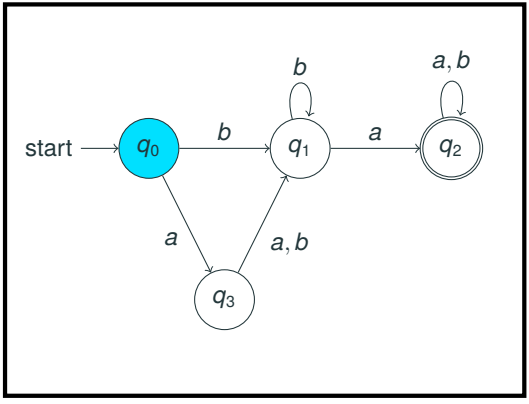


→ **‘aabba’ wird akzeptiert!**

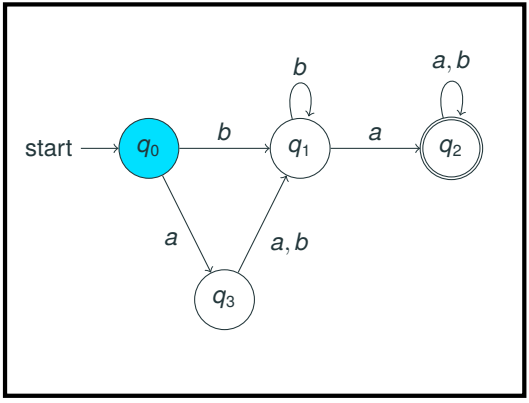


Endliche Automaten: Beispiel II (‘aab’)

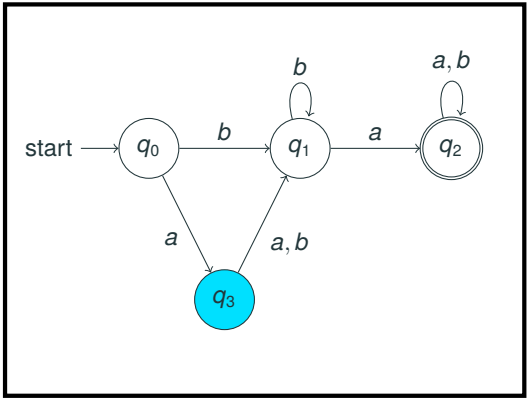
a	a	b					



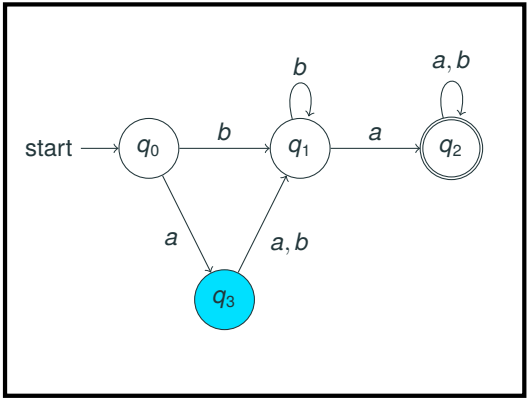
Endliche Automaten: Beispiel II (‘aab’)



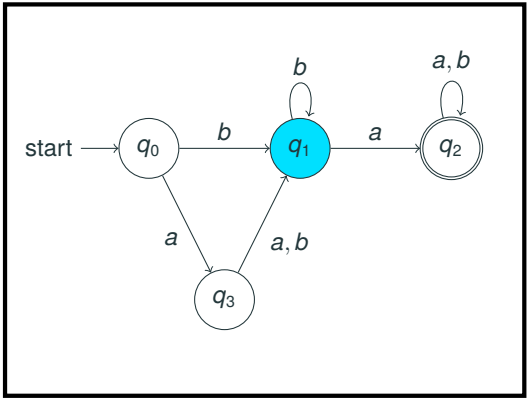
Endliche Automaten: Beispiel II (‘aab’)



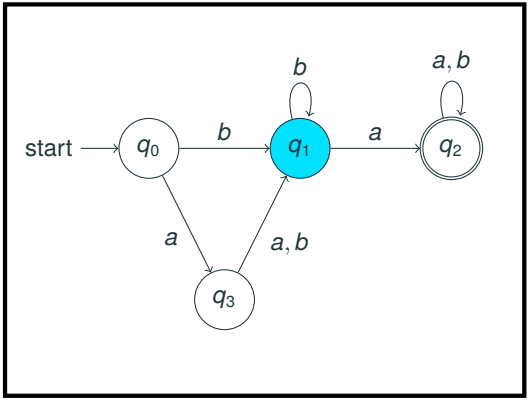
Endliche Automaten: Beispiel II (‘aab’)



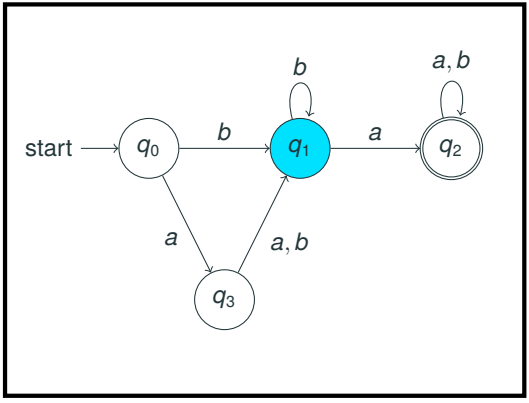
Endliche Automaten: Beispiel II (‘aab’)



Endliche Automaten: Beispiel II (‘aab’)

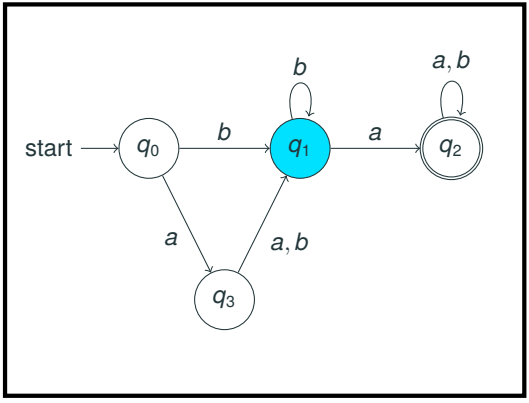


Endliche Automaten: Beispiel II (‘aab’)



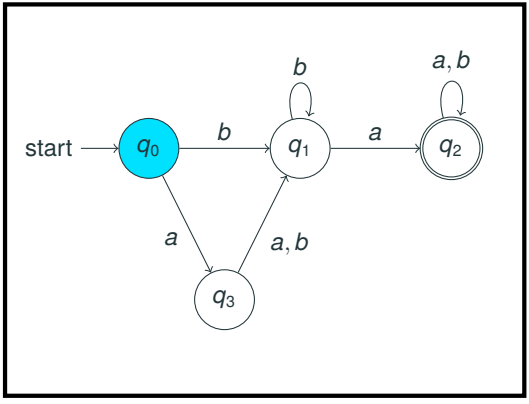
Endliche Automaten: Beispiel II (‘aab’)

→ **‘aab’ wird nicht akzeptiert!**

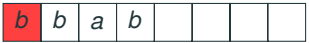


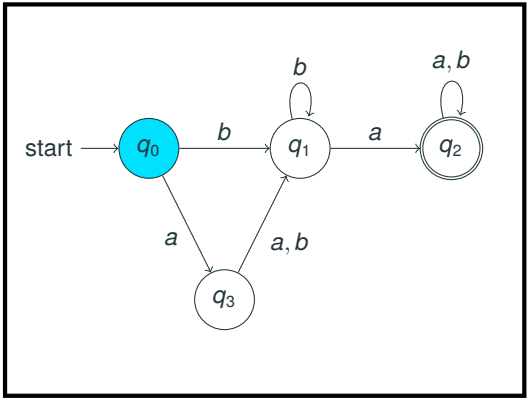
Endliche Automaten: Beispiel III (‘bbab’)

b	b	a	b				

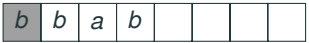


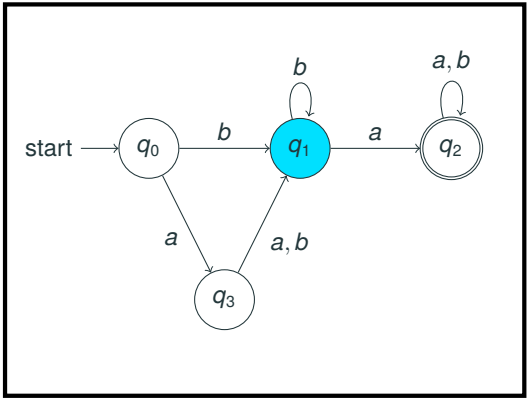
Endliche Automaten: Beispiel III (‘bbab’)



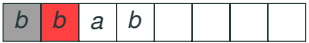


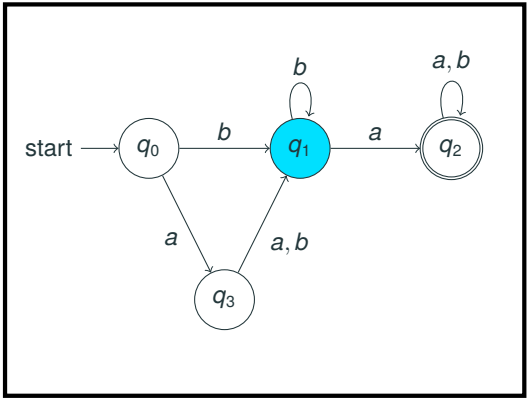
Endliche Automaten: Beispiel III (‘bbab’)



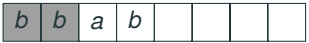


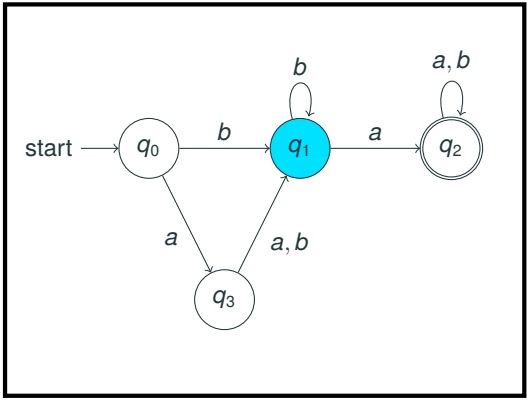
Endliche Automaten: Beispiel III (‘bbab’)



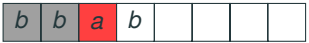


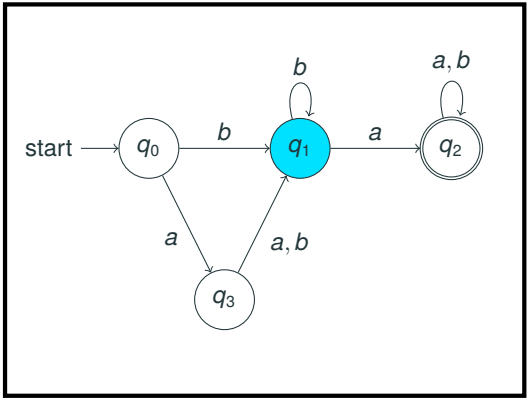
Endliche Automaten: Beispiel III (‘bbab’)



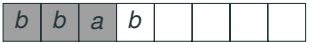


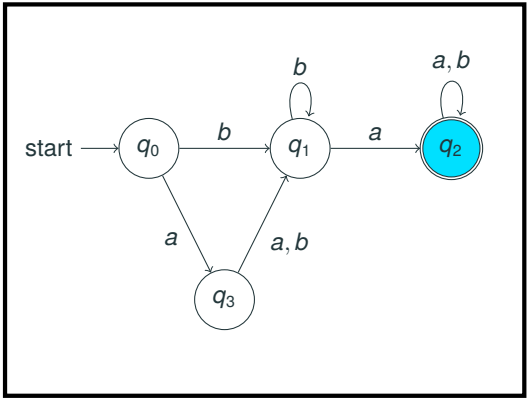
Endliche Automaten: Beispiel III (‘bbab’)



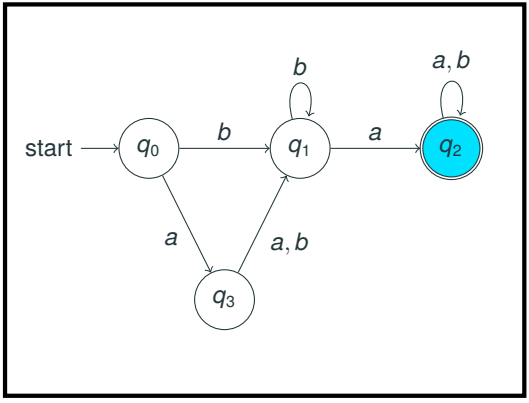


Endliche Automaten: Beispiel III (‘bbab’)

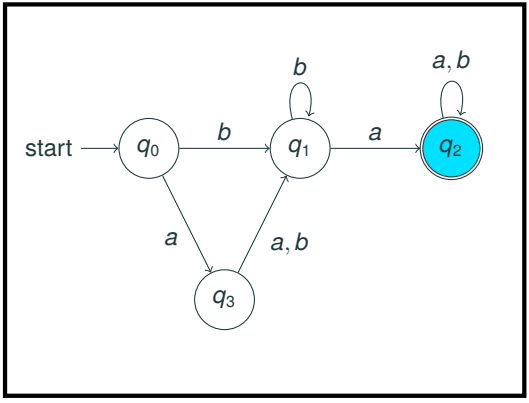




Endliche Automaten: Beispiel III (‘bbab’)

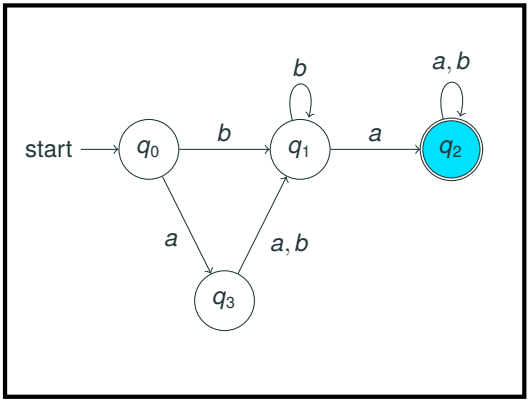


Endliche Automaten: Beispiel III (‘bbab’)



Endliche Automaten: Beispiel III (‘bbab’)

→ **‘bbab’ wird akzeptiert!**



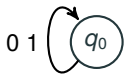
Reguläre Sprachen, Beispiele

Definition 11.7

Eine Sprache $L \subseteq \Sigma^*$ heißt **regulär**, wenn es einen DFA A über Σ gibt, der L berechnet, d. h., für den $L(A) = L$ gilt.

Beispiele für reguläre Sprachen:

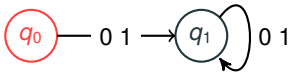
Die triviale Sprache $\emptyset \subseteq \{0,1\}^*$,



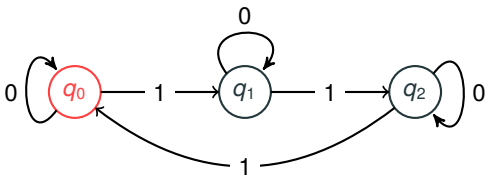
Die triviale Sprache $\{0,1\}^*$,

Reguläre Sprachen, Beispiele

Die triviale Sprache $\{\epsilon\} \subseteq \{0,1\}^*$.

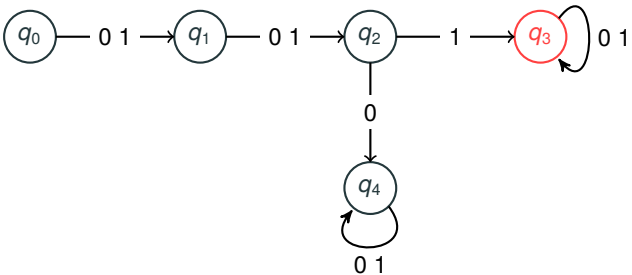


Die Sprache $MOD_3 = \{x \in \{0,1\}^*, | \sum_{i=1}^{|x|} x_i \equiv 0 \mod 3\}$.



Weitere einfache Beispiele regulärer Sprachen

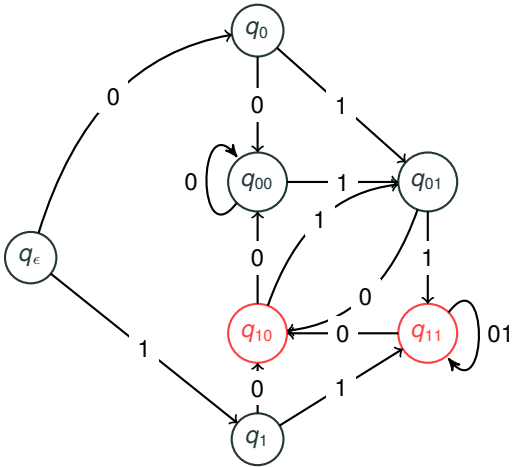
Erkennung, ob Eingaben $x \in \{0,1\}^*$ mindestens die Länge 3 haben und an der 3-ten Position **von vorne** eine Eins steht.



Weitere einfache Beispiele regulärer Sprachen

Erkennung, ob $|x| \geq 2$ und an der vorletzten Position von x eine Eins steht.

Idee: A merkt sich x (für $|x| \leq 1$) bzw. das Suffix der Länge 2 von x (Startzustand q_ϵ).



Lemma 11.8

Jede endliche Sprache ist regulär.

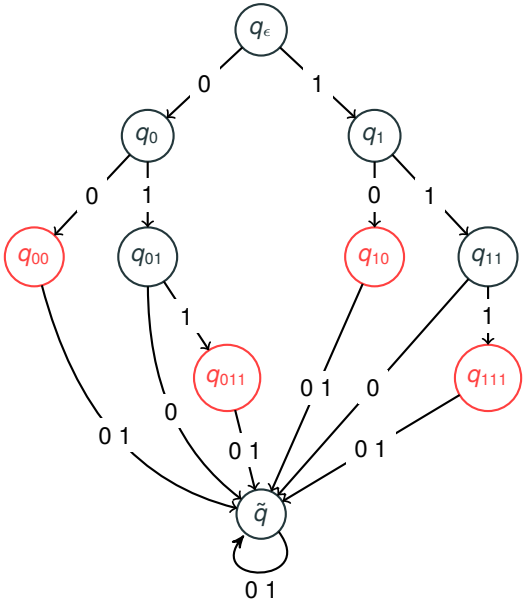
Beweis: Es sei $L \subseteq \Sigma^*$ endlich und $n = \max_{x \in L}\{|x|\}$.

Wir definieren einen DFA $A = (Q, q_0, F, \delta)$ mit $L(A) = L$:

- $Q = \{q_x \mid x \in \Sigma^*, |x| \leq n\} \cup \{\tilde{q}\}$,
- $q_0 = q_\epsilon$,
- $F = \{q_x \mid x \in L\}$,
- $\delta(q_x, \sigma) = q_{x\sigma}$ für alle $x \in \Sigma^*$, $|x| < n$, und alle $\sigma \in \Sigma$,
- $\delta(q_x, \sigma) = \tilde{q}$ für alle $x \in \Sigma^*$, $|x| = n$, und alle $\sigma \in \Sigma$.
- $\delta(\tilde{q}, \sigma) = \tilde{q}$ für alle $\sigma \in \Sigma$.

Man überlegt sich leicht, dass für alle $x \in \Sigma^*$ mit $|x| \leq n$ gilt, dass $\delta^*(q_\epsilon, x) = q_x$, und dass für alle $x \in \Sigma^*$ mit $|x| > n$ gilt, dass $\delta^*(q_\epsilon, x) = \tilde{q}$. □

Beispiel $L = \{00, 10, 011, 111\}$



Ein Beispiel für eine nichtreguläre Sprache

Wir betrachten die Sprache

$$L = \{0^n 1^n \mid n \in \mathbb{N}^+\} \subseteq \{0,1\}^*.$$

Lemma 11.9

L ist nicht regulär.

Beweis: Wir nehmen an, L wäre regulär und fixieren einen DFA $A = (Q, q_0, F, \delta)$ für L.

Da Q endlich ist, existieren natürliche Zahlen $s, t \in \mathbb{N}^+$, $s < t$, mit

$$\delta^*(q_0, 0^s) = \delta^*(q_0, 0^t).$$

Daraus folgt

$$\delta^*(q_0, 0^s 1^s) = \delta^*(q_0, 0^t 1^s).$$

Das ergibt einen Widerspruch, da $0^s 1^s \in L$ aber $0^t 1^s \notin L$. □

Das Pumping-Lemma für reguläre Sprachen

Hier ein klassisches Resultat über die Struktur regulärer Sprachen, das zugleich ein wichtiges Instrument zum Zeigen der Nichtregularität von gegebenen Sprachen ist:

Lemma 11.10 (Pumping-Lemma für reguläre Sprachen)

Sei $L \subseteq \Sigma^*$ regulär. Dann existiert eine natürliche Zahl $n_0 \in \mathbb{N}^+$, so dass jedes Wort x in L mit $|x| \geq n_0$ in folgendem Sinne ***in*** *L* ***aufpumpbar*** *ist:*

Es existiert eine Zerlegung $x = uvw$ mit $|uv| \leq n_0$ und $|v| \geq 1$, so dass für alle $i \in \mathbb{N}$ gilt, dass auch uv^iw in L liegt.

Wir beweisen zunächst mit dem Pumping Lemma, dass $L = \{0^n1^n \mid n \in \mathbb{N}^+\}$ nicht regulär ist:

Wir zeigen, dass kein Wort der Gestalt $x = 0^n1^n$ eine Zerlegung $x = uvw$, $|v| \geq 1$, hat, so dass x in L aufpumpbar ist.

Dafür müsste einerseits v die gleiche Zahl von Nullen und Einsen besitzen (weil das auch für alle Worte in L gilt), also die Gestalt $v = 0^i1^i$, $i \geq 1$, haben.

Damit wäre aber bereits $uvvw = 0^n1^i0^i1^n$ nicht in L. □

Der Beweis des Pumping Lemmas

Sei L regulär und $A = (Q, q_0, F, \delta)$ ein DFA für L.

Wir setzen $n_0 = |Q|$ und fixieren ein Wort $x \in L$ mit $|x| \geq |Q|$.

Entlang der ersten $|Q|$ Schritte der Berechnung von A auf x werden genau $|Q| + 1$ Zustände besucht.

Also existiert ein Zustand $q \in Q$, der entlang der ersten $|Q|$ Schritte der Berechnung von A auf x zweimal vorkommt.

Die Berechnung von A auf x hat also die Gestalt $q_0 \xrightarrow{u} q \xrightarrow{v} q \xrightarrow{w} q'$ und $q' \in F$.

Mit anderen Worten, es existiert eine Zerlegung $x = uvw$ mit $|uv| \leq |Q|$ und $|v| \geq 1$ und $\delta^*(q_0, u) = q$ und $\delta^*(q, v) = q$ und $\delta^*(q, w) = q'$.

Damit hat die Berechnung von A auf uw die Gestalt $q_0 \xrightarrow{u} q \xrightarrow{w} q'$ und für alle $i \geq 2$ die Gestalt

$$q_0 \xrightarrow{u} q \xrightarrow{v} q \xrightarrow{v} \cdots \xrightarrow{v} q \xrightarrow{w} q'.$$

Also gilt für alle $i \geq 0$, dass $\delta^*(q_0, uv^i w) = q' \in F$ und damit $uv^i w \in L$. $\square$

Grundlegende Problemstellungen

Wie erkennt man, ob eine Sprache regulär ist?

Wie konstruiert man minimale DFAs
für gegebene reguläre Sprachen?

Endliche Automaten

Charakterisierung von Regularität und minimale DFAs

Die Nerode-Relation und der Index einer Sprache

Definition 11.11

Es sei $L \subseteq \Sigma^*$ eine Sprache über einem endlichem Alphabet Σ und $x \in \Sigma^*$ ein Wort.

Die **Präfixsprache** $L_x \subseteq \Sigma^*$ von L ist definiert als

$$L_x = \{y \in \Sigma^* \mid xy \in L\}.$$

Worte $x, x' \in \Sigma^*$ heißen **Nerode-äquivalent** bezüglich L ($x \sim_L x'$), wenn $L_x = L_{x'}$.

Der **Index** $ind(L)$ einer Sprache L ist definiert als die Anzahl der Präfixsprachen bezüglich L, d. h.,

$$ind(L) = |\{L_x \mid x \in \Sigma^*\}|.$$

Beobachtung 1: $\sim_L$ ist eine Äquivalenzrelation auf Σ^*. Der Index von L ist die Anzahl der Äquivalenzklassen bezüglich $\sim_L$.

Beobachtung 2: Es gilt stets $L_\epsilon = L$.

Beobachtung 3: Es gilt stets, dass $x \in L \iff \epsilon \in L_x$.

Präfixsprachen - Beispiele

- **Beispiel 1:** $PARITY = \{x \in \{0,1\}^* \mid (\sum_{i=1}^{|x|} x_i) \bmod 2 = 1\}$.
  - Sei $x \in PARITY$ (d.h. $\sum_{i=1}^{|x|} x_i$ ungerade): Dann $xy \in PARITY \iff \sum_{i=1}^{|y|} y_i$ gerade.
    $$\implies PARITY_x = \{0,1\}^* \setminus PARITY$$
  - Sei $x \in \{0,1\}^* \setminus PARITY$ (d.h. $\sum_{i=1}^{|x|} x_i$ gerade): Dann $xy \in PARITY \iff \sum_{i=1}^{|y|} y_i$ ungerade.
    $$\implies PARITY_x = PARITY.$$

  Also gilt $ind(PARITY) = 2$.

- **Beispiel 2:** $L^{(k)} = \{x \in \{0,1\}^*; |x| \geq k, x_k = 1\}$ für $k \in \mathbb{N}^+$.
  - $L^{(3)}_\epsilon = L^{(3)}$,
  - $L^{(3)}_b = L^{(2)}$ für alle $b \in \{0,1\}$,
  - $L^{(3)}_{ab} = L^{(1)}$ für alle $ab \in \{0,1\}^2$,
  - $L^{(3)}_{ab0x} = \emptyset$ für alle $ab \in \{0,1\}^2$ und $x \in \{0,1\}^*$,
  - $L^{(3)}_{ab1x} = \{0,1\}^*$ für alle $ab \in \{0,1\}^2$ und $x \in \{0,1\}^*$.
  - Also gilt $ind(L^{(3)}) = 5$.

Präfixsprachen - Weitere Beispiele

- **Beispiel 3:** $L = \{0^n 1^n; n \geq 1\}$
  - $L_\epsilon = L$
  - Für alle $m \geq 1$ gilt $L_{0^m} = \{0^s 1^{m+s}, s \in \mathbb{N}\}$.
  - Für alle $m \geq 1$ und $1 \leq n < m$ gilt $L_{0^m 1^n} = \{1^{m-n}\}$.
  - Für alle $n \geq 1$ gilt $L_{0^n 1^n} = \{\epsilon\}$.
  - Für $x \notin \{0^m 1^n; m \geq 1, 0 \leq n \leq m\} \cup \{\epsilon\}$ gilt $L_x = \emptyset$.
  - Also gilt $ind(L) = \infty$.

- **Beispiel 4:** $L' = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$ (d.h., die vorletzte Stelle von x ist 1).
  - $L'_\epsilon = L'_{00} = L'_{x00} = L'$ für alle $x \in \{0,1\}^*$,
  - $L'_1 = L'_{01} = L'_{x01} = \{0,1\} \cup L'$, für alle $x \in \{0,1\}^*$,
  - $L'_{10} = L'_{x10} = \{\epsilon\} \cup L'$, für alle $x \in \{0,1\}^*$,
  - $L'_{11} = L'_{x11} = \{\epsilon, 0, 1\} \cup L'$, für alle $x \in \{0,1\}^*$.
  - Also gilt $ind(L') = 4$.

Eine Charakterisierung regulärer Sprachen

Die Bedeutung der Konzepte Präfixsprachen und Index im Kontext endlicher Automaten liegt in

Theorem 11.12

Eine Sprache $L \subseteq \Sigma^*$ ist genau dann regulär, wenn $ind(L) < \infty$.

Ist L regulär, so gibt $ind(L)$ die minimale Anzahl der Zustände eines DFAs für L an.

Wir werden uns im Folgenden schrittweise dem Beweis von Theorem 11.12 nähern und die Beziehung zwischen Präfixsprachen und DFA-Zuständen prinzipiell und an Beispielen studieren.

Definition 11.13

Es sei $A = (Q, q_0, F, \delta)$ ein DFA für eine reguläre Sprache $L \subseteq \Sigma^*$ und $q \in Q$ ein Zustand.

Dann bezeichne $A_q = (Q, q, F, \delta)$ den DFA, der aus A dadurch entsteht, dass q die Rolle des Anfangszustands übernimmt.

Man beachte, dass $y \in L(A_q)$ genau dann, wenn $\delta^*(q, y) \in F$.

Wir zeigen im Folgenden, dass jede Präfixsprache von L durch einen DFA A_q für ein $q \in Q$ berechnet wird.

Präfixsprachen und Zustände von DFAs

Lemma 11.14

Sei $L \subseteq \Sigma^*$ regulär, $A = (Q, q_0, F, \delta)$ ein DFA für L, $x \in \Sigma^*$ ein Eingabewort und $q = \delta^*(q_0, x)$ der Zielzustand der Berechnung von A auf x. Dann gilt $L(A_q) = L_x$.

Beweis: Es gilt für beliebige $y \in \Sigma^*$, dass $y \in L(A_q)$ genau dann, wenn $\delta^*(q, y) \in F$.

Das gilt genau dann, wenn $\delta^*(q_0, xy) \in F \iff xy \in L \iff y \in L_x$. $\square$

Daraus folgt eine Teilaussage von Theorem 11.12.

Lemma 11.15

Für jede reguläre Sprache L und jeden DFA $A = (Q, q_0, F, \delta)$ für L gilt, dass $ind(L) \leq |Q| < \infty$.

Beweis: Wir fixieren für jede Präfixsprache $\tilde{L}$ von L ein x mit $\tilde{L} = L_x$ und ordnen dieser den Zustand $q = \delta^*(q_0, x)$ zu.

Diese Zuordnung ist injektiv, da aus $q = \delta^*(q_0, x) = \delta^*(q_0, x')$ stets $L_x = L_{x'} = L(A_q)$ folgt.

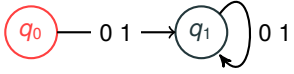
Folglich gilt $ind(L) \leq |Q|$. $\square$

Beispiele DFA-Zustände und Präfixsprachen, I

Beispiel 1: Die trivialen Sprache $\emptyset$ und $\{0,1\}^*$ ($\emptyset_x = \emptyset$ und $\{0,1\}^*_x = \{0,1\}^*$ für alle $x \in \{0,1\}^*$)

$ind(\emptyset) = ind(\{0,1\}^*) = 1$

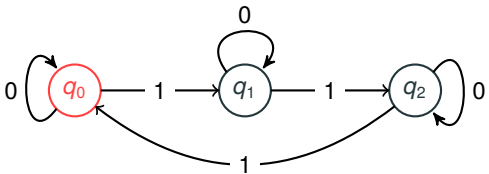
Beispiel 2: Die triviale Sprache $L = \{\epsilon\} \subseteq \{0,1\}^*$.



$L_\epsilon = \{\epsilon\}$, $L_x = \emptyset = L(A(q_1))$ für alle $x \in \{0,1\}^* \setminus \{\epsilon\} \implies ind(\{\epsilon\}) = 2$.

Beispiele DFA-Zustände und Präfixsprachen, II

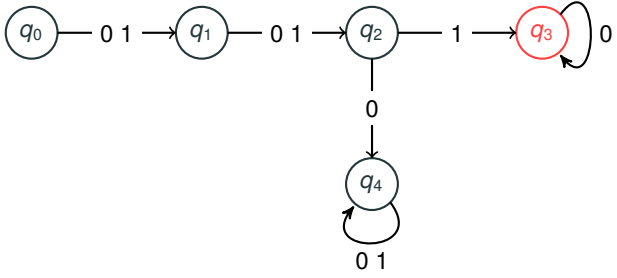
Beispiel 4: Die Sprache $MOD_3 = \{x \in \{0,1\}^*, \sum_{i=1}^{|x|} x_i \equiv 0 \mod 3\}$.



- $(MOD_3)_\epsilon = MOD_3 = L(A) = L(A(q_0))$
- $\sum_{i=1}^{|x|} x_i \equiv 0 \mod 3$: $(MOD_3)_x = MOD_3$, (d.h., $x \sim_{MOD_3} \epsilon$)
- $\sum_{i=1}^{|x|} x_i \equiv 1 \mod 3$: $(MOD_3)_x = \{y; \sum_{i=1}^{|y|} y_i \equiv 2 \mod 3\} = L(A(q_1))$.
- $\sum_{i=1}^{|x|} x_i \equiv 2 \mod 3$: $(MOD_3)_x = \{y; \sum_{i=1}^{|y|} y_i \equiv 1 \mod 3\} = L(A(q_2))$.
- **Folgerung:** $ind(MOD_3) = 3$

Beispiele DFA-Zustände und Präfixsprachen, III

Wir haben gezeigt, dass für $L^{(3)} = \{x \in \{0,1\}^*; |x| \geq 3, x_3 = 1\}$ gilt $ind(L^{(3)}) = 5$.



Präfixsprachen: $L_{\epsilon}^{(3)} = L^{(3)}$, $L_{b}^{(3)} = L^{(2)} = L(A(q_1))$ für alle $b \in \{0,1\}$,

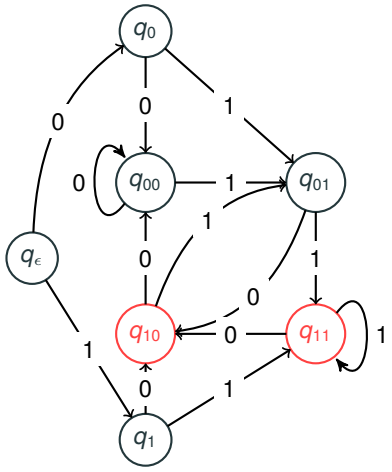
$L_{ab}^{(3)} = L^{(1)} = L(A(q_2))$ für alle $ab \in \{0,1\}^2$,

$L_{ab0x}^{(3)} = \emptyset = L(A(q_4))$ für alle $ab \in \{0,1\}^2$ und $x \in \{0,1\}^*$,

$L_{ab1x}^{(3)} = \{0,1\}^* = L(A(q_3))$ für alle $ab \in \{0,1\}^2$ und $x \in \{0,1\}^*$.

Beispiele DFA-Zustände und Präfixsprachen, IV

$L' = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$



Wir hatten gezeigt, dass $ind(L') = 4$ und L' die folgenden Präfixsprachen hat:

- $L'_\epsilon = L'_0 = L'_{x00} = L'$
  $= L(A(q_\epsilon)) = L(A(q_0)) = L(A(q_{00}))$.
- $L'_1 = L'_{01} = L'_{x01} = L' \cup \{0,1\}$
  $= L(A(q_1)) = L(A(q_{01}))$.
- $L'_{10} = L'_{x10} = L' \cup \{\epsilon\}$
  $= L(A(q_{10}))$.
- $L'_{11} = L'_{x11} = L' \cup \{\epsilon, 0, 1\}$
  $= L(A(q_{11}))$.

Hierbei bezeichnet x ein beliebiges Wort aus $\{0,1\}^*$.

Die Konstruktion von Nerode-Automaten

Wir hatte bereits gezeigt, dass jede reguläre Sprache L einen endlichen Index hat, und dass $ind(L)$ eine untere Schranke für die Anzahl der Zustände eines jeden DFA ist, der L berechnet.

Zum Beweis von Theorem 11.12 genügt es zu zeigen, dass jede reguläre Sprache L einen DFA mit $ind(L)$ Zuständen hat.

Wir konstruieren nun für eine beliebig fixierte Sprache L mit $ind(L) = n < \infty$ den sogenannten **Nerode-Automaten** $A = (Q, q_0, F, \delta)$, für den $|Q| = n$ gilt.

Wir bezeichnen mit $L_0, \ldots, L_{n-1}$ die n verschiedenen Präfixsprachen von L, wobei $L_0 = L_\epsilon = L$ gelte.

Es sei $Q = \{q_0, \ldots, q_{n-1}\}$ und $F = \{q_i \mid \epsilon \in L_i\}$.

Außerdem gelte für alle i, $0 \leq i \leq n-1$, und alle $\sigma \in \Sigma$, dass

$$\delta(q_i, \sigma) = q_j,$$

falls für $x \in \Sigma^*$ mit $L_x = L_i$ gilt, dass $L_{x\sigma} = L_j$.

Zur Rechtfertigung dieser Definition müssen wir zeigen, dass aus $x \sim_L x'$ stets $x\sigma \sim_L x'\sigma$ folgt.

(Das gilt, da für alle $y \in \Sigma^*$ gilt $y \in L_{x\sigma} \iff \sigma y \in L_x \iff \sigma y \in L_{x'} \iff y \in L_{x'\sigma}$.)

Der Beweis von Theorem 11.12

Aus der Definition von δ folgt direkt, dass für alle $x \in \Sigma^*$ gilt, dass

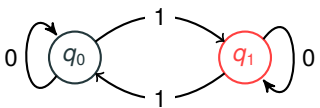
$$\delta^*(q_0, x) = q_i \iff L_x = L_i.$$

Daraus folgt, dass

$$\delta^*(q_0, x) \in F \iff \epsilon \in L_x \iff x \in L. \square$$

Beispiel: $PARITY = \{x \in \{0,1\}^*; \sum_{i=1}^{|x|} x_i \mod 2 = 1\}$ hat den Index zwei und die beiden Präfixsprachen $L_0 = L_\epsilon = PARITY$ und $L_1 = L_1 = \{0,1\}^* \setminus PARITY$.

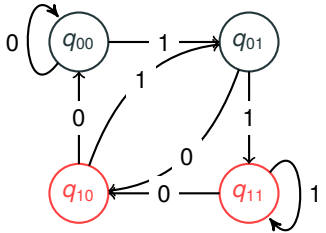
Somit ist



ein minimaler DFA für $PARITY$.

Ein minimlaer DFA für L'

$L' = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$, $ind(L) = 4$



Endliche Automaten

Konstruktionsmethoden für endliche Automaten

Konstruktionsmethoden für endliche Automaten

Wir lernen im Folgenden verschiedene Methoden kennen, wie aus DFAs für einfache Entscheidungsprobleme DFAs für komplexere Problemstellungen konstruiert werden können.

Wir starten mit der Konstruktion des sogenannten Komplementärautomaten.

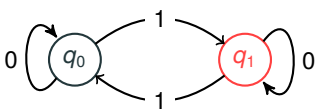
Lemma 11.16

Es sei Σ endliches Alphabet, $L \subseteq \Sigma^*$ regulär, und für den DFA $A = (Q, q_0, F, \delta)$ gelte $L(A) = L$.

Dann hat das Komplement $\bar{L} = \Sigma^* \setminus L$ der Sprache L einen DFA $\bar{A}$ gleicher Größe.

Beweis: Zur Konstruktion von $\bar{A}$ vertauschen wir in A die akzeptierenden und die nichtakzeptierenden Zustände, $\bar{A} = (Q, q_0, Q \setminus F, \delta)$. □

Beispiel: $L(A) = PARITY$



Konstruktionsmethoden für endliche Automaten

Wir lernen im Folgenden verschiedene Methoden kennen, wie aus DFAs für einfache Entscheidungsprobleme DFAs für komplexere Problemstellungen konstruiert werden können.

Wir starten mit der Konstruktion des sogenannten Komplementärautomaten.

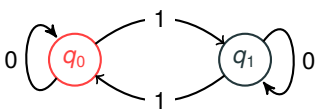
Lemma 11.17

Es sei Σ endliches Alphabet, $L \subseteq \Sigma^*$ regulär, und für den DFA $A = (Q, q_0, F, \delta)$ gelte $L(A) = L$.

Dann hat das Komplement $\bar{L} = \Sigma^* \setminus L$ der Sprache L einen DFA $\bar{A}$ gleicher Größe.

Beweis: Zur Konstruktion von $\bar{A}$ vertauschen wir in A die akzeptierenden und die nichtakzeptierenden Zustände, $\bar{A} = (Q, q_0, Q \setminus F, \delta)$. □

Beispiel: $L(\bar{A}) = \{0,1\}^* \setminus \mathit{PARITY}$



Die Produktkonstruktion

Wir betrachten nun das Problem, für gegebene DFAs A und A' mit $L = L(A)$ und $L' = L(A')$ einen DFA für die Sprache $L \cup L'$ zu konstruieren.

Schlüssel ist die **Produktkonstruktion**, die auf folgender Idee beruht.

Um für eine Eingabe $x \in \Sigma^*$ zu testen ob $x \in L \cup L'$, führen wir die Berechnung auf x in A und A' parallel aus.

Dann betrachten wir die erreichten Zielzustände $\delta_A^*(q_0, x)$ und $\delta_{A'}^*(q_0', x)$ und akzeptieren x, falls mindestens einer dieser Zielzustände akzeptierend ist.

Lemma 11.18

Gegeben seien DFAs $A = (Q, q_0, F, \delta)$ und $A' = (Q', q_0', F', \delta')$ über Σ, wobei $L = L(A)$ und $L' = L(A')$. Dann existieren DFAs der Größe $|Q| \cdot |Q'|$ für die Sprachen $L \cup L'$, $L \cap L'$ und $L \triangle L'$.

Beweis: Wir konstruieren den **Produktautomaten** $\tilde{A} = (Q \times Q', (q_0, q_0'), \tilde{F}, \tilde{\delta})$, der die parallele Ausführung der Automaten A und A' simuliert.

Der Produktautomat $\tilde{A} = (Q \times Q', (q_0, q_0'), \tilde{F}, \tilde{\delta})$

- Für alle $(q, q') \in Q \times Q'$ und $a \in \Sigma$ sei

$$\tilde{\delta}((q, q'), a) = (\delta(q, a), \delta'(q', a)).$$

- $\tilde{A}$ berechnet $L \cup L'$, falls $\tilde{F} = \{(q, q') \mid (q \in F) \vee (q' \in F')\}$.
- $\tilde{A}$ berechnet $L \cap L'$, falls $\tilde{F} = \{(q, q') \mid (q \in F) \wedge (q' \in F')\}$.
- $\tilde{A}$ berechnet $L \triangle L'$, falls $\tilde{F} = \{(q, q') \mid (q \in F) \oplus (q' \in F')\}$.

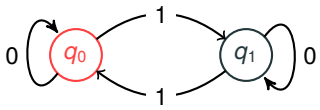
Bemerkung 1: Diese Produktkonstruktion lässt sich leicht auf andere zweistellige und sogar mehrstellige Mengenoperationen verallgemeinern.

Bemerkung 2: Der Produktautomat in der so konstruierten Form kann redundante Zustandsknoten enthalten, die vom Startzustand (q_0, q_0') aus nicht erreichbar sind.

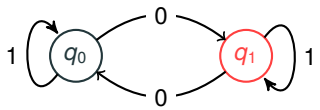
Diese können durch elementare Graphalgorithmen (z.B. durch Breitensuche, siehe Kurs CS 307, Algorithmen und Datenstrukturen) effizient berechnet und gestrichen werden.

Produktautomat - Beispiel (0)

$L = \{x \in \{0,1\}^*; |x|_1 \text{ gerade}\}$

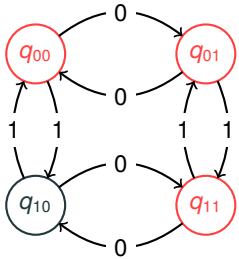


$L = \{x \in \{0,1\}^*; |x|_0 \text{ ungerade}\}$



$|x|_b = |\{i;\ 1 \leq i \leq |x|, x_i = b\}|, b \in \{0,1\}$

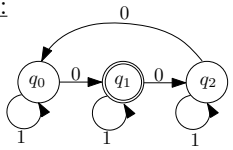
DFA für $L \cup L'$



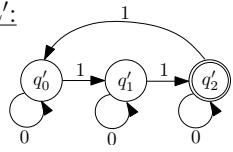
Produktautomat - Beispiel (1)

$L = \{x \in \{0,1\}^* \mid (|x|_0 \bmod 3 = 1)\}$,
$L' = \{x \in \{0,1\}^* \mid (|x|_1 \bmod 3 = 2)\}$.

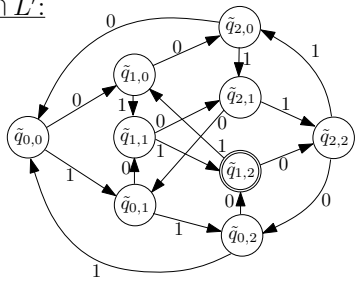
$\underline{L}$:



$\underline{L'}$:



$\underline{L \cap L'}$:

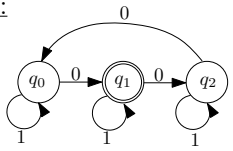


Beachte: Wir schreiben hier kurz $\tilde{q}_{i,j} := (q_i, q'_j)$, $i,j \in \{0,1,2\}$.

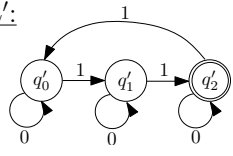
Produktautomat - Beispiel (2)

$L = \{x \in \{0,1\}^* \mid (|x|_0 \bmod 3 = 1)\}$,
$L' = \{x \in \{0,1\}^* \mid (|x|_1 \bmod 3 = 2)\}$.

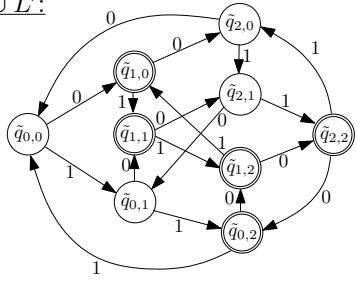
$\underline{L}$:



$\underline{L'}$:



$\underline{L \cup L'}$:



Beachte: Wir schreiben hier kurz $\tilde{q}_{i,j} := (q_i, q'_j)$, $i,j \in \{0,1,2\}$.

Verallgemeinerte Produktkonstruktion (1)

Definition 11.19

Seien $L_1, \ldots, L_k \subseteq \Sigma^*$, $k \geq 1$, Sprachen und $f : \{0,1\}^k \to \{0,1\}$ eine Boolesche Funktion. Die Sprache $L^f = L^f(L_1, \ldots, L_k) \subseteq \Sigma^*$ sei definiert durch

$$x \in L^f \iff f(L_1(x), \ldots, L_k(x)) = 1.$$

Hierbei sei für eine Sprache $L \subseteq \Sigma^*$ der Wert $L(x) \in \{0,1\}$ definiert durch $L(x) = 1 \iff x \in L$.

Offenbar gilt:

- $\bar{L_1} = L^f(L_1)$ für $f(b_1) = \neg b_1 = 1 - b_1$,
- $L_1 \cup L_2 = L^f(L_1, L_2)$ mit $f(b_1, b_2) = b_1 \vee b_2$,
- $L_1 \cap L_2 = L^f(L_1, L_2)$ mit $f(b_1, b_2) = b_1 \wedge b_2$,
- $L_1 \triangle L_2 = L^f(L_1, L_2)$ mit $f(b_1, b_2) = b_1 \oplus b_2$.

Verallgemeinerte Produktkonstruktion (2)

Theorem 11.20

Seien $L_1, \ldots, L_k \subseteq \Sigma^*$, $k \geq 1$, reguläre Sprachen mit den DFAs $A^j = (Q^j, q_0^j, F^j, \delta^j)$ und $f : \{0,1\}^k \to \{0,1\}$ eine Boolesche Funktion.

Dann ist $L^f = L^f(L_1, \ldots, L_k)$ auch regulär und hat einen DFA $A^f = (Q^f, q_0^f, F^f, \delta^f)$ der Größe $\leq |Q^1| \cdot |Q^2| \cdot \ldots \cdot |Q^k|$.

Beweis: A^f simuliert wieder die parallele Ausführung von $A^1, \ldots, A^k$, d. h., wir setzen:

- $Q^f = Q^1 \times Q^2 \times \cdots \times Q^k$.
- $q_0^f = (q_0^1, \ldots, q_0^k)$.
- $\delta^f((q^1, \ldots, q^k), \sigma) = (\delta^1(q^1, \sigma), \ldots, \delta^k(q^k, \sigma))$ für alle $1 \leq j \leq k$, $q^j \in Q^j$ und $\sigma \in \Sigma$.
- $(q^1, \ldots, q^k) \in F^f \iff f(\chi(q^1), \ldots, \chi(q^k)) = 1$.
  Hierbei gilt für alle $1 \leq j \leq k$, dass $\chi(q^j) = 1$ wenn $q^j \in F^j$ und $\chi(q^j) = 0$ wenn $q^j \notin F^j$. □

Endliche Automaten

Nichtdeterministische Endliche Automaten (NFA - Nondeterministic Finite Automata)

Definition von NFAs

NFAs N sind endliche Automaten, bei denen für gegebene Zustände q und Buchstaben $\sigma \in \Sigma$ mehrere (oder auch null) Nachfolgezustände von q bezüglich σ erlaubt sind, d. h., $\delta(q, \sigma)$ ist eine Menge von Zuständen.

Dies hat die Wirkung, dass es für Eingabeworte $x \in \Sigma^*$ mehrere in q startende Berechnungen von N auf x gibt, die sowohl akzeptierend (d. h. in einem akzeptierenden Zustand endend) als auch verwerfend sein können.

N akzeptiert x genau dann, wenn es eine im Anfangszustand q_0 startende akzeptierende Berechnung von N auf x gibt.

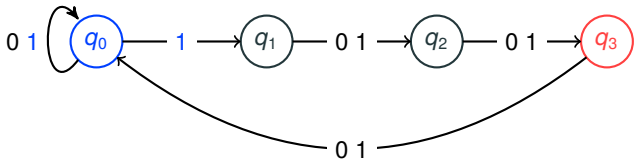
Definition 11.21

Ein NFA über Σ ist ein 4-Tupel $N = (Q, q_0, F, \delta)$, wobei, wie bei DFAs, Q eine endliche Zustandsmenge, $q_0 \in Q$ einen Anfangszustand und $F \subseteq Q$ die Menge akzeptierender Zustände bezeichnet.

Die Zustandsüberführungsfunktion δ ist jedoch definiert als $\delta : Q \times \Sigma \to \mathcal{P}(Q)$.

Beispiel NFA

$L^{[3]} = \{x \in \{0,1\}^*; |x| \leq 3,\ x_{|x|-2} = 1\}$



Der Zustand q_0 ist als einziger nichtdeterministisch, $\delta(q_0, 1) = \{q_0, q_1\}$.

Eingaben $x \in \{0,1\}^*$ haben genau dann eine akzeptierende Berechnung, wenn $|x| \geq 2$ und $x_{|x|-2} = 1$. (Bleibe bis $x_{|x|-2}$ in q_0 und gehe mit $x_{|x|-2} = 1$ nach q_1. Dann und nur dann endet die Berechnung in q_3.)

Das heißt, N berechnet $L^{[3]}$.

Anmerkung: $ind(L^{[3]}) = 8$ (Beweis später).

Berechnungen durch NFAs

Definition 11.22

Sei $N = (Q, q_0, F, \delta)$ ein NFA über Σ, $x = (x_1, \ldots, x_n) \in \Sigma^*$ ein Wort und $q \in Q$.

Eine **Berechnung von** N **auf** x **von** q **aus** ist eine Zustandsfolge $q \xrightarrow{x_1} q_1 \xrightarrow{x_2} \cdots \xrightarrow{x_n} q_n$, wobei $q_1 \in \delta(q, x_1)$ und $q_i \in \delta(q_{i-1}, x_i)$ für alle $2 \leq i \leq n$.

Die Berechnung heißt **akzeptierend**, falls der Zielzustand q_n in F liegt.

Die **von** N **berechnete Sprache** $L(N) \subseteq \Sigma^*$ enthält genau jene $x \in \Sigma^*$, für die es eine akzeptierende Berechnung von N auf x von q_0 aus gibt.

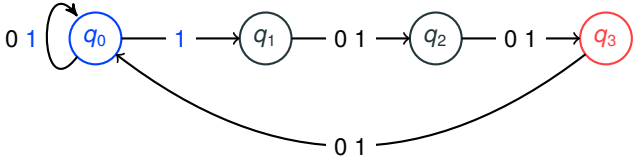
Wir verallgemeinern δ zu einer Funktion $\delta^* : Q \times \Sigma^* \to \mathcal{P}(Q)$.

Für alle $x \in \Sigma^*$ sei $\delta^*(q, x)$ die Menge der mit x von q aus erreichbaren Zustände.

Es gilt: $x \in L(N) \iff \delta^*(q_0, x) \cap F \neq \emptyset$.

Außerdem gilt für alle $\sigma \in \Sigma$, dass $\delta^*(q, x\sigma) = \bigcup_{q' \in \delta^*(q,x)} \delta(q', \sigma)$.

Beispiel δ^*



$\delta^*(q_0, 0) = \delta(q_0, 0) = \{q_0\}$

$\delta^*(q_0, 01) = \delta(q_0, 1) = \{q_0, q_1\}$

$\delta^*(q_0, 010) = \delta(q_0, 0) \cup \delta(q_1, 0) = \{q_0\} \cup \{q_2\} = \{q_0, q_2\}$

$\delta^*(q_0, 0101) = \delta(q_0, 1) \cup \delta(q_2, 1) = \{q_0, q_1\} \cup \{q_3\} = \{q_0, q_1, q_3\}$

$\delta^*(q_0, 01010) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_3, 0) = \{q_0\} \cup \{q_1\} \cup \{q_0\} = \{q_0, q_2\}.$

Die Ausdruckskraft von NFAs versus DFAs

Mit NFAs können viele Sprachen mit deutlich weniger Zuständen realisiert werden als mit DFAs.

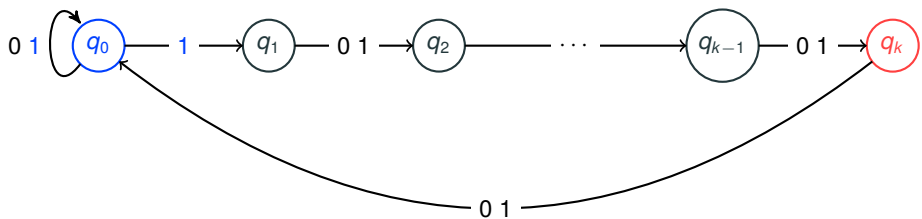
Theorem 11.23

Für alle $k \geq 1$ sei $L^{[k]} = \{x \in \{0,1\}^*; |x| \geq k, x_{|x|-k+1} = 1\}$.

Dann gilt $ind(L^{(k)}) = 2^k$, jedoch kann $L^{[k]}$ mit NFAs mit nur $k+1$ Zuständen berechnet werden.

Beweis: Dass $ind(L^{[k]}) = 2^k$ ist in den Übungen zu zeigen.

Zudem berechnet der folgende NFA $L^{[k]}$ mit $k+1$ Zuständen:



Eine effizientere Konstruktionen für die Vereinigung regulärer Sprachen

Theorem 11.24

Es seien $A = (Q, q_0, F, \delta)$ und $A' = (Q', q_0', F', \delta')$ NFAs über Σ.

Dann existiert ein NFA $N = (Q^N, q_0^N, F^N, \delta^N)$ der Größe $|Q| + |Q'| + 1$ mit $L(N) = L(A) \cup L'(A)$.

Anmerkung: Die Produktkonstruktion liefert einen DFA für $L(A) \cup L'(A)$ der Größe $|Q| \cdot |Q'|$, falls A, A' DFAs.

Beweis Theorem 11.24: Wir führen einen neuen Anfangszustand ein, in dem wir nichtdeterministisch entscheiden, ob wir in A oder in A' rechnen.

D. h., wir fixieren einen neuen Anfangszustand $q_0^N \notin Q \cup Q'$ und setzen $Q^N = Q \cup Q' \cup \{q_0^N\}$.

Außerdem sei für alle $\sigma \in \Sigma$

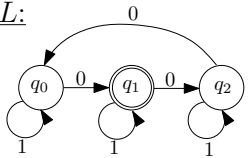
- $\delta^N(q_0^N, \sigma) = \delta(q_0, \sigma) \cup \delta'(q_0', \sigma)$,
- $\delta^N(q, \sigma) = \delta(q, \sigma)$ für alle $q \in Q$,
- $\delta^N(q', \sigma) = \delta'(q', \sigma)$ für alle $q' \in Q'$.

Letztlich gelte $F^N = F \cup F'$, falls $q_0 \notin F$ und $q_0' \notin F'$, bzw. $F^N = F \cup F' \cup \{q_0^N\}$ sonst. □

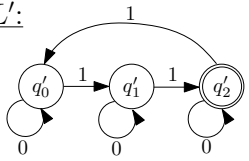
NFAs erlauben effizientere Konstruktionen - Beispiel

$L = \{x \in \{0,1\}^* \mid (|x|_0 \bmod 3 = 1)\}$,
$L' = \{x \in \{0,1\}^* \mid (|x|_1 \bmod 3 = 2)\}$.

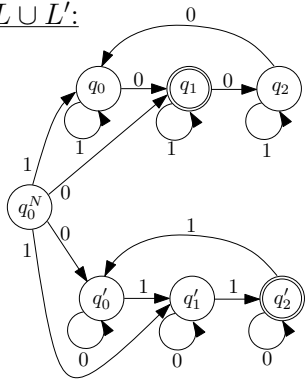
DFA zu L:



DFA zu L':



NFA zu $L \cup L'$:



Simulation von NFAs durch DFAs

Wir haben gesehen, dass NFAs deutlich kompaktere Darstellungen von regulären Sprachen erlauben als DFAs.

Frage: Können NFAs auch nichtreguläre Sprachen berechnen?

Antwort: NEIN!

Theorem 11.25

Jeder NFA $N = (Q, q_0, F, \delta)$ über Σ kann durch einen DFA A mit höchstens $2^{|Q|}$ Zuständen simuliert werden, d. h., jede von einem NFA berechnete Sprache ist regulär.

Beweis: Die Idee ist, dass die Zustände von A den Teilmengen von Q entsprechen.

Der σ-Nachfolger einer Teilmenge $Q' \subseteq Q$ ist die Menge der Zustände, die in N von einem Zustand $q' \in Q'$ mittels eines σ-Übergangs erreicht werden können.

Anfangszustand ist die Einermenge $\{q_0\}$.

Diese Simulation wird auch als **Potenzmengenkonstruktion** bezeichnet.

Potenzmengenkonstruktion

Für den NFA $N = (Q, q_0, F, \delta)$ konstruieren wir den DFA $A = (\mathcal{P}(Q), \{q_0\}, \tilde{F}, \tilde{\delta})$ mit

- $\tilde{F} = \{Q' \subseteq Q \mid F \cap Q' \neq \emptyset\}$,
- $\tilde{\delta}(Q', \sigma) = \bigcup_{q' \in Q'} \delta(q', \sigma)$ für alle $Q' \subseteq Q$ und $\sigma \in \Sigma$.

Beobachtung: Für alle Worte $x \in \Sigma^*$ und alle $Q' \subseteq Q$ gilt, dass

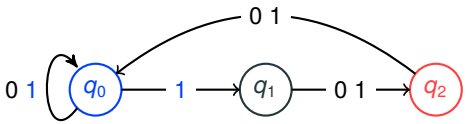
$$\tilde{\delta}^*(Q', x) = \bigcup_{q \in Q'} \delta^*(q, x).$$

D. h., $\delta^*(Q', x)$ enthält alle Zielzustände von in Q' startenden Berechnungen von N auf x.

Das wiederum heißt, A akzeptiert x genau dann, wenn es in N eine in q_0 startende, akzeptierende x-Berechnung gibt.

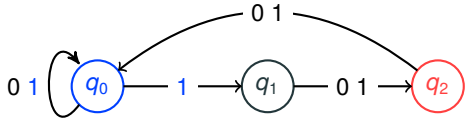
Daraus folgt nach Definition, dass $L(N) = L(A)$. □

Beispiel NFA für $L^{[2]} = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$, 0



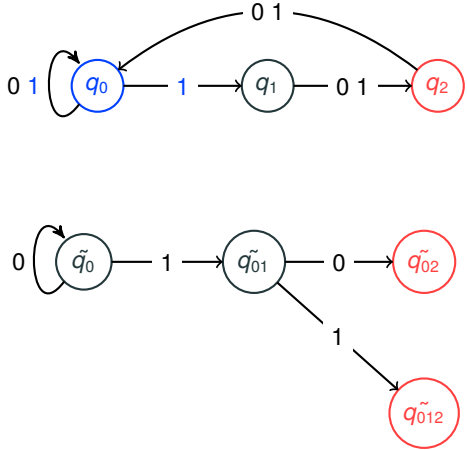
$\tilde{\delta}(\{q_0\}, 0) = \{q_0\},\ \tilde{\delta}(\{q_0\}, 1) = \{q_0, q_1\}$

Beispiel NFA für $L^{[2]} = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$, 1



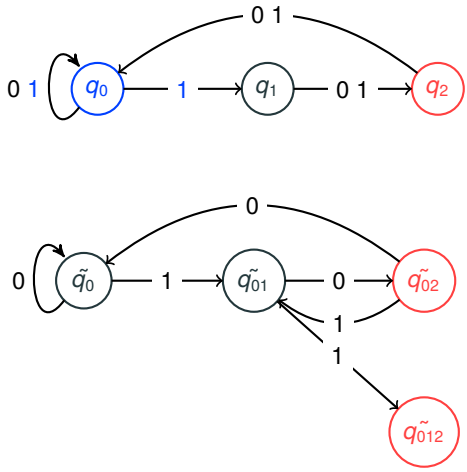
$\tilde{\delta}(\{q_0, q_1\}, 0) = \{q_0, q_2\}$, $\tilde{\delta}(\{q_0, q_1\}, 1) = \{q_0, q_1, q_2\}$

Beispiel NFA für $L^{[2]} = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$, 2



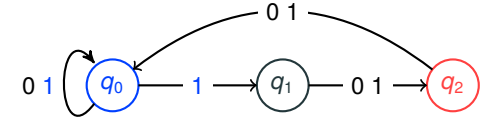
$\tilde{\delta}(\{q_0, q_2\}, 0) = \{q_0\}, \ \tilde{\delta}(\{q_0, q_2\}, 1) = \{q_0, q_1\}$

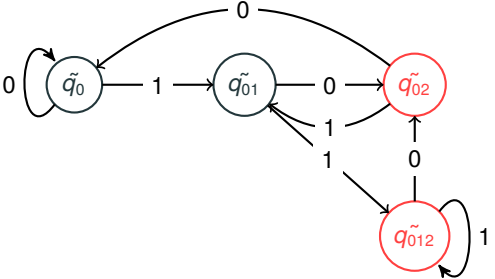
Beispiel NFA für $L^{[2]} = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$, 3



$\tilde{\delta}(\{q_0, q_1, q_2\}, 0) = \{q_0, q_2\}, \ \tilde{\delta}(\{q_0, q_1, q_2\}, 1) = \{q_0, q_1, q_2\}$

Beispiel NFA für $L^{[2]} = \{x \in \{0,1\}^*; |x| \geq 2, x_{|x|-1} = 1\}$,4





Konketanation und Kleene'scher Abschluss regulärer Sprachen

Theorem 11.26

Für alle regulären Sprachen $L, L' \subseteq \Sigma^*$ gilt, dass sowohl $L \cdot L'$ (die Konkatenation von L und L') als auch L^* (der Kleene'sche Abschluss von L) regulär sind.

Insbesondere existieren für alle NFAs $N = (Q, q_0, F, \delta)$ und $N' = (Q', q_0', F', \delta')$ ein NFA $\tilde{N}$ der Größe $|Q| + |Q'|$ für $L(N) \cdot L(N')$ und ein NFA $\hat{N}$ der Größe $|Q| + 1$ für $L(N)^*$.

Beweis: Wir konstruieren aus N und N' die NFAs $\tilde{N}$ und $\hat{N}$.

Die Idee für $\tilde{N}$ ist, dass wir in N starten und an jedem akzeptierenden Zustand von N nichtdeterministisch nach N' wechseln.

D. h., $\tilde{N} = (Q \cup Q', q_0, \tilde{F}, \tilde{\delta})$ mit:

- $\tilde{F} = F'$, falls $\epsilon \notin L'$, bzw. $\tilde{F} = F \cup F'$, falls $\epsilon \in L'$.
- Für alle $\sigma \in \Sigma$ sei:
  - $\tilde{\delta}(q, \sigma) = \delta(q, \sigma)$ für alle $q \in Q \setminus F$,
  - $\tilde{\delta}(q, \sigma) = \delta(q, \sigma) \cup \delta'(q_0', \sigma)$ für alle $q \in F$,
  - $\tilde{\delta}(q', \sigma) = \delta(q', \sigma)$ für alle $q' \in Q'$.

Fortsetzung Beweis Theorem 11.26

Ist $\epsilon \notin L'$, so ist eine in q_0 startende Berechnung auf einem Wort $z \in \Sigma^*$ in $\tilde{N}$ genau dann akzeptierend, wenn sie in einem F'-Zustand endet.

Dies geht nur, wenn sie einen Zustandsübergang von Q nach Q' der Gestalt $q \xrightarrow{z_i} q'$ mit i, $1 \leq i \leq |z|$, $q \in F$ und $q' \in \delta'(q'_0, z_i)$ hat.

Dies bedeutet jedoch wiederum, dass $(z_1, \ldots, z_{i-1}) \in L(N)$ und $(z_i, \ldots, z_{|z|}) \in L'$, also $z \in L \cdot L'$.

Andererseits hat jedes Wort der Gestalt xy mit $x \in L$ und $y \in L(N')$ eine akzeptierende, in q_0 startende Berechnung in $\tilde{N}$, die in einem Zustand in $F \cap \delta^*(q_0, x)$ nach N' wechselt und in einem Zustand in $F' \cap \delta^*(q'_0, y)$ endet.

Ist $\epsilon \in L'$, so ist eine in q_0 startende Berechnung auf einem Wort $z \in \Sigma^*$ in $\tilde{N}$ genau dann akzeptierend, wenn sie in einem Zustand aus F oder F' endet.

Im ersten Fall gilt $z \in L \subseteq L \cdot L'$, im zweiten Fall gilt das oben Gesagte.

Auch hat jedes Wort der Gestalt xy mit $x \in L$ und $y \in L'$ eine akzeptierende, in q_0 startende Berechnung in $\tilde{N}$ (siehe oben).

Folglich gilt $L(\tilde{N}) = L \cdot L'$.

Fortsetzung Beweis Theorem 11.26, II

Die Idee für $\hat{N}$ ist, dass wir in N rechnen, aber zusätzlich an jedem akzeptierenden Zustand von N nichtdeterministisch auf q_0 zurückwechseln.

Da $\epsilon \in L^*$, benötigen wir zudem einen neuen Startzustand $\hat{q}_0$, der akzeptierend ist.

$\hat{N} = (Q \cup \{\hat{q}_0\}, \hat{q}_0, F \cup \{\hat{q}_0\}, \hat{\delta})$, wobei für alle $\sigma \in \Sigma$ gilt:

- $\hat{\delta}(\hat{q}_0, \sigma) = \delta(q_0, \sigma)$,
- $\hat{\delta}(q, \sigma) = \delta(q, \sigma) \cup \delta(q_0, \sigma)$ für alle $q \in F$,
- $\hat{\delta}(q, \sigma) = \delta(q, \sigma)$ für alle $q \in Q \setminus F$.

Um zu zeigen, dass $L(\hat{N}) = L^*$, betrachten wir eine in $\hat{q}_0$ startende, akzeptierende Berechnung auf $x \in \Sigma^*$ mit Zielzustand q in $\hat{N}$.

Dann ist entweder $x = \epsilon$ und $q = \hat{q}_0$ oder es ist $x \neq \epsilon$ und $q \in F$.

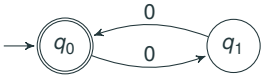
Wir betrachten alle $s \geq 0$ neu hinzugenommenen Zustandsübergänge $q_1 \xrightarrow{x_i} q_2$ mit $q_1 \in F$ und $q_2 \in \delta(q_0, x_i)$ entlang der Berechnung.

Diese Zustandsübergänge definieren eine Zerlegung $x = x^1 x^2 \dots x^{s+1}$ in Teilworte aus L, d. h., $x \in L^*$.

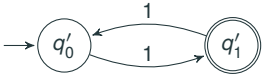
Andererseits haben alle $x \in L^*$ eine akzeptierende Berechnung in $\hat{N}$. $\square$

Theorem 11.26 - Beispiel zu $L \cdot L'$

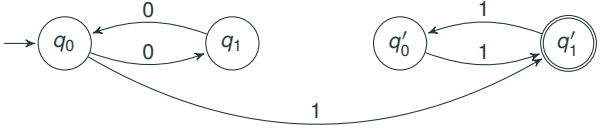
NFA zur Sprache $L = \{0^n \mid n \in \mathbb{N}, n \bmod 2 = 0\}$:



NFA zur Sprache $L' = \{1^n \mid n \in \mathbb{N}, n \bmod 2 = 1\}$:

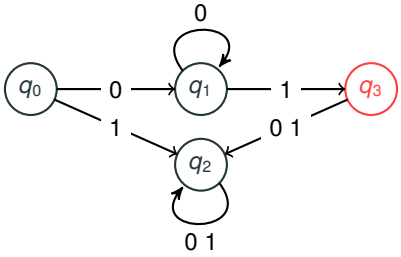


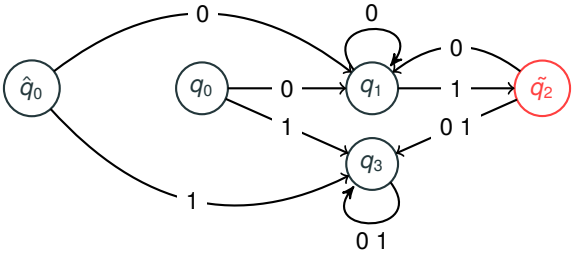
NFA zur Sprache $L \cdot L'$:



Theorem 11.26 - Beispiel zu L^*

$L = \{0^j 1; j \in \mathbb{N}^+\}$





Endliche Automaten

Reguläre Ausdrücke (kurz REs, von *regular expressions*)

Definition Regulärer Ausdrücke

Reguläre Ausdrücke über einem endlichen Alphabet Σ bilden eine Datenstruktur zur Repräsentation bestimmter Sprachen, die insbesondere im Compilerbau benutzt wird.

Definition 11.27

REs sind Worte über $\Sigma \cup \{+, \cdot, (,), ^*\}$, die mittels folgender syntaktischer Regeln gebildet werden.

- ϵ, $\emptyset$ und σ sind für alle $\sigma \in \Sigma$ REs.
- Falls R, R' REs, so sind auch $(R) \cdot (R')$, $(R) + (R')$ und $(R)^*$ REs.

Jeder RE R definiert eine Sprache $L(R) \subseteq \Sigma^*$ mittels folgender Regeln.

- $L(\epsilon) = \{\epsilon\}$, $L(\emptyset) = \emptyset$, $L(\sigma) = \{\sigma\}$ für alle $\sigma \in \Sigma$.
- $L((R) \cdot (R')) = L(R) \cdot L(R')$,
- $L((R) + (R')) = L(R) \cup L(R')$
- $L((R)^*) = L(R)^*$.

Beispiele regulärer Ausdrücke

- $\{0^j; j \in \mathbb{N}\} \implies (0)^*$
- $\{0^j 1; j \in \mathbb{N}^+\} \implies (0) \cdot (0)^* \cdot (1) = (0)(0)^*(1) = 0(0)^*1$
- $\{0,1\}^* \implies ((0)+(1))^* = (0+1)^*$
- $\{0,1\}^k \implies ((0)+(1))((0)+(1))\cdots((0)+(1)) = (0+1)^k$
- $L^{(3)} = \{x \in \{0,1\}^*; |x| \geq 3, x_3 = 1\} \implies ((0)+(1))^2(1)((0)+(1))^*$
- $L^{[3]} = \{x \in \{0,1\}^*; |x| \geq 3, x_{|x|-2} = 1\} \implies ((0)+(1))^*(1)((0)+(1))^2$
- $\{\epsilon, 0, 10, 111\} \implies (\epsilon)+(0)+(10)+(111)$
- $PARITY = \{x \in \{0,1\}^*; \sum_{i=1}^{|x|} \mod 2 \equiv 1\} \implies (0)^*(1)(0)^*((1)(0)^*(1)(0)^*)^*$
- $MOD_3^2 = \{x \in \{0,1\}^*; \sum_{i=1}^{|x|} \mod 3 \equiv 2\} \implies (0)^*(1)(0)^*(1)(0)^*((1)(0)^*(1)(0)^*(1)(0)^*)^*$

Theorem 11.28

Jede von einem regulären Ausdruck R berechnete Sprache ist regulär.

Beweis: Das gilt offensichtlich für die Sprachen $L(\epsilon)$, $L(\emptyset)$ und $L(\sigma)$, $\sigma \in \Sigma$.

Ansonsten folgt die Aussage entlang der rekursiven Struktur von R mit Hilfe der Theoreme 11.24 und 11.26. □

Theorem 11.29

Jede reguläre Sprache $L \subseteq \Sigma^*$ hat einen regulären Ausdruck R über Σ mit $L(R) = L$.

Beweis: Sei $A = (Q, q_0, F, \delta)$ ein DFA für L mit $Q = \{1, \ldots, n\}$ und $q_0 = 1$.

Für alle $1 \leq i, j \leq n$ und $0 \leq k \leq n$ definieren wir eine Sprache $L^k_{i,j}(A)$ wie folgt:

$L^k_{i,j}(A)$ enthalte alle Worte $x \in \Sigma^*$ mit $\delta^*(i, x) = j$ und alle Zwischenzustände entlang der x-Berechnung von i nach j liegen in $\{1, \ldots, k\}$.

Beweis von Theorem 11.29, Fortsetzung

Offensichtlich gilt:

- $L^0_{i,i} = \{\epsilon\} \cup \bigcup_{\sigma \in \Sigma, \delta(i,\sigma)=i} \{\sigma\}$ für alle $1 \leq i \leq n$.
- $L^0_{i,j} = \bigcup_{\sigma \in \Sigma, \delta(i,\sigma)=j} \{\sigma\}$ für alle $i \neq j$, $1 \leq i,j \leq n$.
- $L^k_{i,j} = L^{k-1}_{i,j} \cup (L^{k-1}_{i,k} \cdot (L^{k-1}_{k,k})^* \cdot L^{k-1}_{k,j})$ für alle $1 \leq i,j,k \leq n$.
- $L(A) = \bigcup_{j \in F} L^n_{1,j}$.

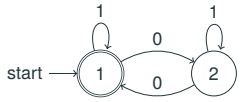
Aus diesen Regeln folgen zunächst direkt reguläre Ausdrücke für die Sprachen $L^0_{i,j}$, $1 \leq i,j \leq n$.

Ausgehend von diesen lassen sich mittels der dritten Regel für wachsende k reguläre Ausdrücke für die Sprachen $L^k_{i,j}$, $1 \leq i,j,k \leq n$ konstruieren.

Die letzte Regel liefert daraus einen regulären Ausdruck für L. □

Beispiel zu Theorem 11.29

Beispiel: $L = \{x \in \{0,1\}^* \mid (|x|_0 \bmod 2 = 0)\}$



- $\underline{k = 0:}$ $R^0_{1,1} = \epsilon + 1$, $R^0_{1,2} = 0$, $R^0_{2,1} = 0$, $R^0_{2,2} = \epsilon + 1$.
- $\underline{k = 1:}$
  - $R^1_{1,1} = R^0_{1,1} + (R^0_{1,1} \cdot (R^0_{1,1})^* \cdot R^0_{1,1}) \equiv \cdots \equiv 1^*$,
  - $R^1_{1,2} = R^0_{1,2} + (R^0_{1,1} \cdot (R^0_{1,1})^* \cdot R^0_{1,2}) \equiv \cdots \equiv 0 + (1^* \cdot 0) \equiv 1^* \cdot 0$,
  - $R^1_{2,1} = R^0_{2,1} + (R^0_{2,1} \cdot (R^0_{1,1})^* \cdot R^0_{1,1}) \equiv \cdots \equiv 0 + (0 \cdot 1^*) \equiv 0 \cdot 1^*$,
  - $R^1_{2,2} = R^0_{2,2} + (R^0_{2,1} \cdot (R^0_{1,1})^* \cdot R^0_{1,2}) \equiv \cdots \equiv (\epsilon + 1) + (0 \cdot 1^* \cdot 0)$.
- $\underline{k = 2:}$
  - $R^2_{1,1} = R^1_{1,1} + (R^1_{1,2} \cdot (R^1_{2,2})^* \cdot R^1_{2,1}) \equiv \cdots$
    $\equiv 1^* + ((1^* \cdot 0) \cdot ((\epsilon + 1) + (0 \cdot 1^* \cdot 0))^* \cdot (0 \cdot 1^*))$
    $\equiv 1^* + ((1^* \cdot 0) \cdot (1 + (0 \cdot 1^* \cdot 0))^* \cdot (0 \cdot 1^*))$.
  - Beachte: Bspw. gilt auch $R^2_{1,1} \equiv (1^* \cdot 0 \cdot 1^* \cdot 0 \cdot 1^*)^*$, benötigt kein $+$, und $R^2_{1,1} \equiv (1 + (0 \cdot 1^* \cdot 0))^*$, kompakter.
  - $R^2_{1,2}, R^2_{2,1}, R^2_{2,2}$ hier uninteressant, da nur $q_0 = 1$ akzeptierend.