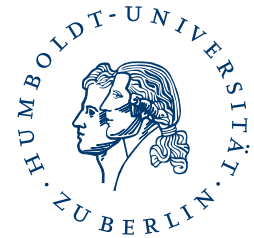
Einführung in die Theoretische Informatik

Johannes Köbler



Institut für Informatik
Humboldt-Universität zu Berlin

WS 2015/16

Vorlesung

Mo,Mi	15-17	RUD 26	0'115	Johannes Köbler

Übungen

Mo	09-11	RUD 26	0'313	Frank Fuhlbrück
Mo	09-11	RUD 26	1'303	Robert Prüfer
Mo	13-15	RUD 26	0'313	Frank Fuhlbrück
Di	11-13	RUD 26	1'306	Robert Prüfer
Mi	09-11	RUD 26	0'313	Robert Prüfer
Mi	09-11	RUD 26	1'303	Berit Grußien
Do	09-11	RUD 26	0'313	Wolfgang Kössler
Do	11-13	RUD 26	0'313	Wolfgang Kössler
Fr	09-11	RUD 26	1'305	Berit Grußien

Tutorien

Mo	13-15	RUD 26	0'310	Michael Jung
Mi	13-15	RUD 26	0'310	Michael Jung
Mi	17-19	RUD 26	0'307	Michael Jung

Weitere Infos unter

- u.hu-berlin.de/ethi15

bzw.

- www.informatik.hu-berlin.de/forschung/gebiete/algorithmenII/Lehre/ws15/einftheo

Übungen (Anmeldung über GOYA erforderlich)

Ausgabe der Aufgabenblätter

- in der VL, auf GOYA und der VL-Webseite

Bearbeitung

- in Gruppen von zwei bis drei Teilnehmern
- Teilnehmer müssen nicht in der gleichen Übungsgruppe sein
- bitte Übungsgruppe, Namen und Matrikelnr. angeben
- bitte jede Aufgabe auf einem separaten Blatt bearbeiten

Abgabe (bitte nur in Papierform)

- bis 15:10 Uhr hier im Hörsaal (Abgabetermin s. Aufgabenblatt)

Rückgabe

- in den Übungsgruppen

Schein, Klausur, Skript

Scheinkriterien

- Lösen von $\geq 50\%$ der schriftlichen Aufgaben,
- Erfolgreiches Vorrechnen von ≥ 2 mündl. Aufgaben.

Klausur

- **Termin: 19.02.2016**
- Zulassung nur mit Übungsschein
- Nachklausur: 22.03.2016

Skript

- wird wöchentlich ins Netz gestellt.

Gibt es zum organisatorischen Ablauf noch Fragen?

Inhalt der Vorlesung

Themen dieser VL:

- **Welche Rechenmodelle sind adäquat?** **Automatentheorie**
- **Welche Probleme sind lösbar?** **Berechenbarkeitstheorie**
- **Welcher Aufwand ist nötig?** **Komplexitätstheorie**

Themen der VL Algorithmen und Datenstrukturen:

- Wie lassen sich praktisch relevante Problemstellungen möglichst effizient lösen? Algorithmik

Themen der VL Logik in der Informatik:

- Mathem. Grundlagen der Informatik, Beweise führen, Modellierung Aussagenlogik, Prädikatenlogik

Maschinenmodelle

- Rechenmaschinen spielen in der Informatik eine zentrale Rolle.
- Es gibt viele unterschiedliche math. Modelle.
- Diese können sich in der Berechnungskraft unterscheiden.
- Die Turingmaschine (TM) ist ein universales Berechnungsmodell, da sie alle anderen bekannten Rechenmodelle simulieren kann.
- Wir betrachten zunächst Einschränkungen des TM-Modells, die vielfältige praktische Anwendungen haben, wie z.B.
  - endliche Automaten (DFA, NFA),
  - Kellerautomaten (PDA, DPDA) etc.

Der Algorithmenbegriff

- Der Begriff Algorithmus geht auf den persischen Gelehrten Muhammed Al Chwarizmi (8./9. Jhd.) zurück.
- Ältester bekannter nicht-trivialer Algorithmus:
  Euklidischer Algorithmus zur Berechnung des größten gemeinsamen Teilers zweier natürlicher Zahlen (300 v. Chr.).
- Von einem Algorithmus wird erwartet, dass er bei jeder zulässigen Problemeingabe nach endlich vielen Rechenschritten eine korrekte Ausgabe liefert.
- Problemeingaben können Zahlen, Formeln, Graphen etc. sein.
- Diese werden über einem Eingabealphabet Σ kodiert.

Alphabet, Wort, Sprache

Definition

- Ein Alphabet ist eine geordnete endliche Menge

  $$\Sigma = \{a_1, \ldots, a_m\}, \ m \geq 1$$

  von Zeichen a_i.
- Eine Folge $x = x_1 \ldots x_n \in \Sigma^n$ heißt Wort (der Länge n).
- Die Menge aller Wörter über Σ ist

  $$\Sigma^* = \bigcup_{n \geq 0} \Sigma^n.$$

- Das (einzige) Wort der Länge $n = 0$ ist das leere Wort, welches wir mit ε bezeichnen, d.h. $\Sigma^0 = \{\varepsilon\}$.
- Jede Teilmenge $L \subseteq \Sigma^*$ heißt Sprache über dem Alphabet Σ.

Beispiele für Sprachen

Beispiel

Sei Σ ein Alphabet.

- Dann sind $\varnothing, \Sigma^*, \Sigma$ und $\{\varepsilon\}$ Sprachen über Σ.
- $\varnothing$ enthält keine Wörter und heißt leere Sprache.
- Σ^* enthält dagegen alle Wörter über Σ.
- Σ enthält alle Wörter über Σ der Länge 1.
- $\{\varepsilon\}$ enthält nur das leere Wort, ist also einelementig.
  Solche Sprachen werden auch als Singleton-Sprachen bezeichnet.

Operationen auf Sprachen

- Da Sprachen Mengen sind, können wir sie bzgl. Inklusion vergleichen.
- Zum Beispiel gilt $\varnothing \subseteq \{\varepsilon\} \subseteq \Sigma^*$.
- Wir können Sprachen auch vereinigen, schneiden und komplementieren.
- Seien A und B Sprachen über Σ. Dann ist
  - $A \cap B = \{x \in \Sigma^* \mid x \in A \wedge x \in B\}$ der Schnitt von A und B,
  - $A \cup B = \{x \in \Sigma^* \mid x \in A \vee x \in B\}$ die Vereinigung von A und B, und
  - $\overline{A} = \{x \in \Sigma^* \mid x \notin A\}$ das Komplement von A.

Konkatenation von Wörtern

Definition

Seien $x = x_1 \dots x_n$ und $y = y_1 \dots y_m$ Wörter. Dann wird das Wort $x \circ y = x_1 \dots x_n y_1 \dots y_m$ als Konkatenation von x und y bezeichnet.
Für $x \circ y$ schreiben wir auch einfach xy.

Beispiel

- Für $x = aba$ und $y = abab$ erhalten wir $xy = abaabab$ und $yx = abababa$.
- Die Konkatenation ist also nicht kommutativ.
- Allerdings ist $\circ$ assoziativ, d.h. es gilt $x(yz) = (xy)z$.
  Daher können wir hierfür auch einfach xyz schreiben.
- Es gibt auch ein neutrales Element, da $x\varepsilon = \varepsilon x = x$ ist.
- Eine algebraische Struktur $(M, \square, e)$ mit einer assoziativen Operation $\square : M \times M \to M$ und einem neutralen Element e heißt Monoid.
- $(\Sigma^*, \circ, \varepsilon)$ ist also ein Monoid.

Spezielle Sprachoperationen

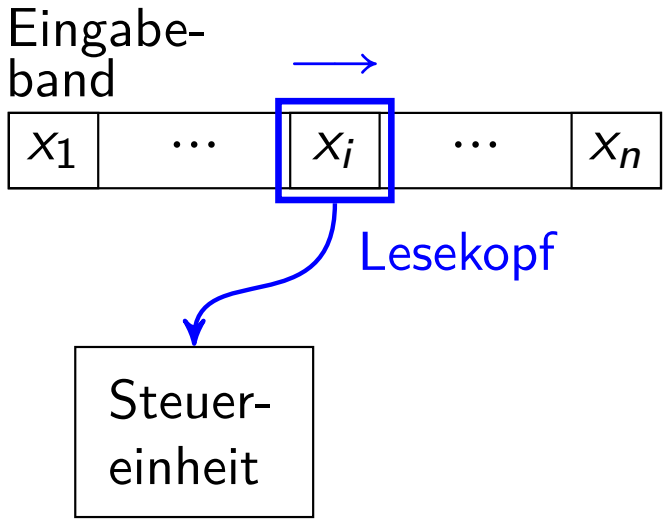
Neben den Mengenoperationen Schnitt, Vereinigung und Komplement gibt es auch spezielle Sprachoperationen.

Definition

- Das Produkt (Verkettung, Konkatenation) der Sprachen A und B ist
  $$AB = \{xy \mid x \in A, y \in B\}.$$
- Ist $A = \{x\}$ eine Singletonsprache, so schreiben wir für $\{x\}B$ auch einfach xB.
- Die n-fache Potenz A^n einer Sprache A ist induktiv definiert durch
  $$A^n = \begin{cases} \{\varepsilon\}, & n = 0, \\ A^{n-1}A, & n > 0. \end{cases}$$
- Die Sternhülle von A ist $A^* = \bigcup_{n \geq 0} A^n$.
- Die Plushülle von A ist $A^+ = \bigcup_{n \geq 1} A^n = AA^*$.

Algorithmische Erkennung von Sprachen

- Ein einfaches Rechenmodell zum Erkennen von Sprachen ist der endliche Automat.



- Ein endlicher Automat
  - nimmt zu jedem Zeitpunkt genau einen von endlich vielen Zuständen an,
  - macht bei Eingaben der Länge n genau n Rechenschritte und
  - liest in jedem Schritt genau ein Eingabezeichen.

Formale Definition eines endlichen Automaten

Definition

- Ein endlicher Automat (kurz: DFA; *Deterministic Finite Automaton*) wird durch ein 5-Tupel $M = (Z, \Sigma, \delta, q_0, E)$ beschrieben, wobei
  - $Z \neq \emptyset$ eine endliche Menge von Zuständen,
  - Σ das Eingabealphabet,
  - $\delta : Z \times \Sigma \to Z$ die Überführungsfunktion,
  - $q_0 \in Z$ der Startzustand und
  - $E \subseteq Z$ die Menge der Endzustände ist.
- Die von M akzeptierte oder erkannte Sprache ist

$$L(M) = \left\{ x_1 \dots x_n \in \Sigma^* \;\middle|\; \begin{array}{l} \text{es gibt } q_1, \dots, q_{n-1} \in Z, q_n \in E \text{ mit} \\ \delta(q_i, x_{i+1}) = q_{i+1} \text{ für } i = 0, \dots, n-1 \end{array} \right\}$$

- Eine Zustandsfolge $q_0, q_1, \dots, q_n$ heißt Rechnung von $M(x_1 \dots x_n)$, falls $\delta(q_i, x_{i+1}) = q_{i+1}$ für $i = 0, \dots, n-1$ gilt.
- Sie heißt akzeptierend, falls $q_n \in E$ ist.

Die Klasse der regulären Sprachen

Frage

Welche Sprachen lassen sich durch endliche Automaten erkennen und welche nicht?

Definition

Eine von einem DFA akzeptierte Sprache wird als regulär bezeichnet. Die zugehörige Sprachklasse ist

$$\mathrm{REG} = \{L(M) \mid M \text{ ist ein DFA}\}.$$

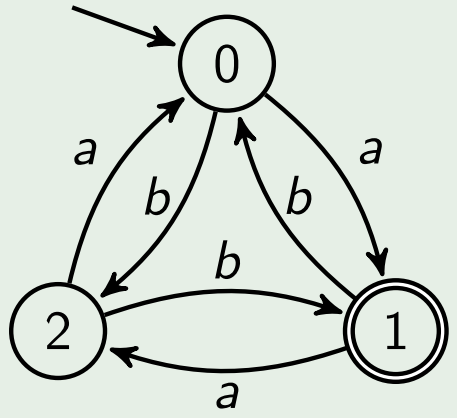
DFAs beherrschen Modulare Arithmetik

Beispiel

Sei $M_3 = (Z, \Sigma, \delta, 0, E)$ ein DFA mit $Z = \{0, 1, 2\}$, $\Sigma = \{a, b\}$, $E = \{1\}$ und der Überführungsfunktion

δ	0	1	2
a	1	2	0
b	2	0	1

Graphische Darstellung:



Endzustände werden durch einen doppelten Kreis und der Startzustand wird durch einen Pfeil gekennzeichnet. ◁

Frage: Welche Wörter akzeptiert M_3?

- $w_1 = aba$? Ja (Rechnung: $0, 1, 0, 1$).
- $w_2 = abba$? Nein (Rechnung: $0, 1, 0, 2, 0$).

DFAs beherrschen Modulare Arithmetik

Behauptung

Die von M_3 erkannte Sprache ist

$$L(M_3) = \{x \in \{a,b\}^* \mid \#_a(x) - \#_b(x) \equiv_3 1\}, \quad \text{wobei}$$

- $\#_a(x)$ die Anzahl der Vorkommen von a in x bezeichnet und
- $i \equiv_m j$ (in Worten: i ist kongruent zu j modulo m) bedeutet, dass $i - j$ durch m teilbar ist.

Beweis der Behauptung durch Induktion über die Länge von x

Wir betrachten zunächst das Erreichbarkeitsproblem für DFAs.

Das Erreichbarkeitsproblem für DFAs

Frage

Sei $M = (Z, \Sigma, \delta, q_0, E)$ ein DFA und sei $x = x_1 \dots x_n \in \Sigma^*$. Welchen Zustand erreicht M bei Eingabe x nach i Schritten?

Antwort

- nach 0 Schritten: q_0,
- nach 1 Schritt: $\delta(q_0, x_1)$,
- nach 2 Schritten: $\delta(\delta(q_0, x_1), x_2)$,
- nach i Schritten: $\delta(\dots\delta(\delta(q_0, x_1), x_2), \dots x_i)$.

Das Erreichbarkeitsproblem für DFAs

Definition

- Bezeichne $\hat{\delta}(q, x)$ denjenigen Zustand, in dem sich M nach Lesen von x befindet, wenn M im Zustand q gestartet wird.
- Dann können wir die Funktion

  $$\hat{\delta} : Z \times \Sigma^* \to Z$$

  induktiv über die Länge von x wie folgt definieren.
- Für $q \in Z$, $x \in \Sigma^*$ und $a \in \Sigma$ sei

  $$\begin{aligned} \hat{\delta}(q, \varepsilon) &= q, \\ \hat{\delta}(q, xa) &= \delta(\hat{\delta}(q, x), a). \end{aligned}$$

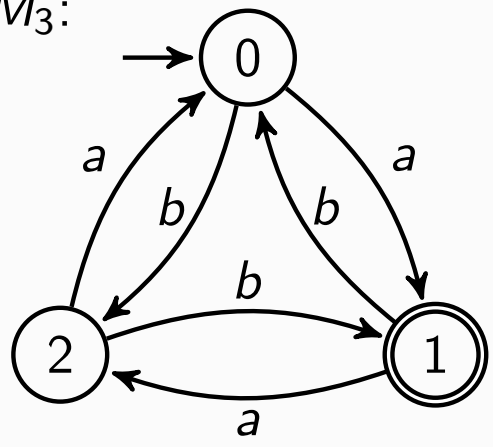
- Die von M erkannte Sprache lässt sich nun auch in der Form

  $$L(M) = \{x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in E\}$$

  schreiben.

DFAs beherrschen Modulare Arithmetik

M_3:



Behauptung

$L(M_3) = \{x \in \{a,b\}^* \mid \#_a(x) - \#_b(x) \equiv_3 1\}$.

Beweis

- 1 ist der einzige Endzustand von M.
- Daher ist $L(M_3) = \{x \in \Sigma^* \mid \hat{\delta}(0,x) = 1\}$.
- Obige Behauptung ist also äquivalent zu

$$\hat{\delta}(0,x) = 1 \Leftrightarrow \#_a(x) - \#_b(x) \equiv_3 1$$

- Folglich reicht es, folgende Kongruenzgleichung zu zeigen:

$$\hat{\delta}(0,x) \equiv_3 \#_a(x) - \#_b(x)$$

DFAs beherrschen Modulare Arithmetik

Beweis von $\hat{\delta}(0,x) \equiv_3 \#_a(x) - \#_b(x)$:

Wir führen Induktion über die Länge n von x.

Induktionsanfang $n = 0$: klar, da $\hat{\delta}(0,\varepsilon) = \#_a(\varepsilon) = \#_b(\varepsilon) = 0$ ist.

Induktionsschritt $n \rightsquigarrow n+1$:

- Sei $x = x_1 \dots x_{n+1}$ gegeben und sei $i = \hat{\delta}(0, x_1 \dots x_n)$.
- Nach IV gilt $i \equiv_3 \#_a(x_1 \dots x_n) - \#_b(x_1 \dots x_n)$.
- Wegen $\delta(i,a) \equiv_3 i+1$ und $\delta(i,b) \equiv_3 i-1$ folgt daher

$$\begin{aligned} \delta(i, x_{n+1}) &\equiv_3 i + \#_a(x_{n+1}) - \#_b(x_{n+1}) \\ &\equiv_3 \#_a(x_1 \dots x_n) - \#_b(x_1 \dots x_n) + \#_a(x_{n+1}) - \#_b(x_{n+1}) \\ &= \#_a(x) - \#_b(x) \end{aligned}$$

und somit

$$\hat{\delta}(0,x) = \delta(\hat{\delta}(0, x_1 \dots x_n), x_{n+1}) = \delta(i, x_{n+1}) \equiv_3 \#_a(x) - \#_b(x).$$

□

Singletons sind regulär

Vereinbarung

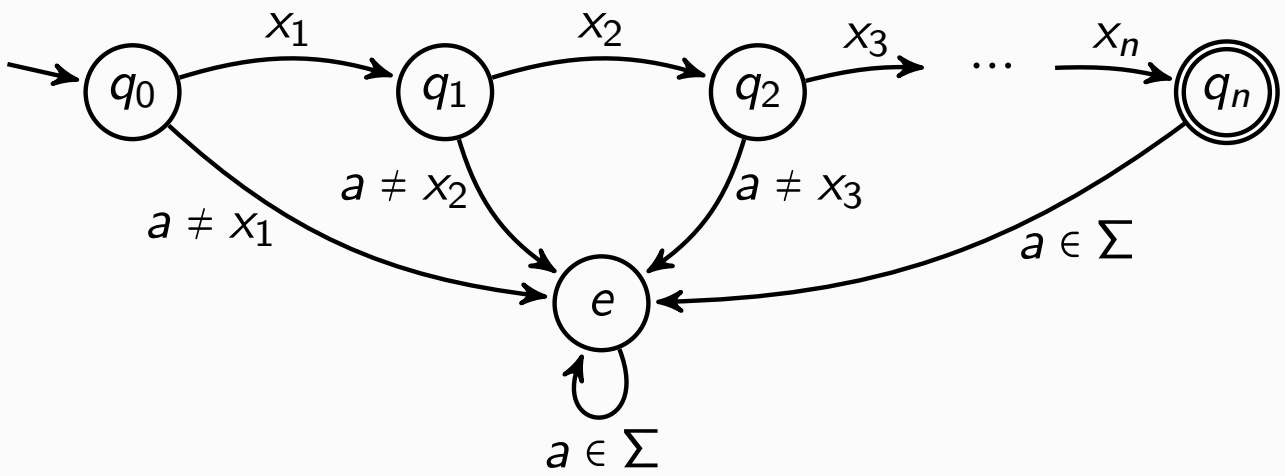
Für das Folgende sei $\Sigma = \{a_1, \ldots, a_m\}$ ein fest gewähltes Alphabet.

Beobachtung 1

Alle Sprachen, die nur ein Wort $x = x_1 \ldots x_n \in \Sigma^*$ enthalten, sind regulär.

Beweis

Folgender DFA M erkennt die Sprache $L(M) = \{x\}$:



□

REG ist unter Komplement abgeschlossen

Beobachtung 2

Ist $L \in \text{REG}$, so ist auch die Sprache $\overline{L} = \Sigma^* \setminus L$ regulär.

Beweis

- Sei $M = (Z, \Sigma, \delta, q_0, E)$ ein DFA mit $L(M) = L$.
- Dann wird das Komplement $\overline{L}$ von L von dem DFA $\overline{M} = (Z, \Sigma, \delta, q_0, Z \setminus E)$ akzeptiert.

□

Definition

Für eine Sprachklasse $\mathcal{C}$ bezeichne *co*-$\mathcal{C}$ die Klasse $\{\overline{L} \mid L \in \mathcal{C}\}$ aller Komplemente von Sprachen in $\mathcal{C}$.

Korollar

co-REG = REG.

REG ist unter Schnitt abgeschlossen

Beobachtung 3

Sind $L_1, L_2 \in \text{REG}$, so ist auch die Sprache $L_1 \cap L_2$ regulär.

Beweis

- Seien $M_i = (Z_i, \Sigma, \delta_i, q_i, E_i)$, $i = 1, 2$, DFAs mit $L(M_i) = L_i$.
- Dann wird der Schnitt $L_1 \cap L_2$ von dem DFA

$$M = (Z_1 \times Z_2, \Sigma, \delta, (q_1, q_2), E_1 \times E_2)$$

mit

$$\delta((p, q), a) = (\delta_1(p, a), \delta_2(q, a))$$

erkannt.

- M wird auch als Kreuzproduktautomat bezeichnet.

□

REG ist unter Vereinigung abgeschlossen

Beobachtung 4

Die Vereinigung $L_1 \cup L_2$ von regulären Sprachen L_1 und L_2 ist regulär.

Beweis

Es gilt $L_1 \cup L_2 = \overline{(\overline{L_1} \cap \overline{L_2})}$. □

Frage

Wie sieht der zugehörige DFA aus?

Antwort

$$M' = (Z_1 \times Z_2, \Sigma, \delta, (q_1, q_2), (E_1 \times Z_2) \cup (Z_1 \times E_2)).$$

Abschlusseigenschaften von Sprachklassen

Definition

- Ein (k-stelliger) Sprachoperator ist eine Abbildung op, die k Sprachen $L_1, \ldots, L_k$ auf eine Sprache $op(L_1, \ldots, L_k)$ abbildet.
- Eine Sprachklasse $\mathcal{K}$ heißt unter op abgeschlossen, wenn gilt:

$$L_1, \ldots, L_k \in \mathcal{K} \Rightarrow op(L_1, \ldots, L_k) \in \mathcal{K}.$$

- Der Abschluss von $\mathcal{K}$ unter op ist die (bzgl. Inklusion) kleinste Sprachklasse $\mathcal{K}'$, die $\mathcal{K}$ enthält und unter op abgeschlossen ist.

Beispiel

- Der 2-stellige Schnittoperator $\cap$ bildet L_1 und L_2 auf $L_1 \cap L_2$ ab.
- Der Abschluss der Singletonsprachen unter $\cap$ besteht aus allen Singletonsprachen und der leeren Sprache.
- Der Abschluss der Singletonsprachen unter $\cup$ besteht aus allen nichtleeren endlichen Sprachen. ◁

REG ist unter Mengenoperationen abgeschlossen

Korollar

Die Klasse REG der regulären Sprachen ist unter folgenden Operationen abgeschlossen:

- Komplement,
- Schnitt,
- Vereinigung.

Wie umfangreich ist REG?

Folgerung

- Aus den Beobachtungen folgt, dass alle endlichen und alle co-endlichen Sprachen regulär sind.
- Da die reguläre Sprache

$$L(M_3) = \{x \in \{a, b\}^* \mid \#_a(x) - \#_b(x) \equiv_3 1\}$$

weder endlich noch co-endlich ist, haben wir damit allerdings noch nicht alle regulären Sprachen erfasst.

Wie umfangreich ist REG?

Nächstes Ziel

Zeige, dass REG unter Produktbildung und Sternhülle abgeschlossen ist.

Problem

Bei der Konstruktion eines DFA für das Produkt $L_1 L_2$ bereitet es Schwierigkeiten, den richtigen Zeitpunkt für den Übergang von (der Simulation von) M_1 zu M_2 zu finden.

Lösungsidee

Ein nichtdeterministischer Automat (NFA) kann den richtigen Zeitpunkt für den Übergang „raten“.

Verbleibendes Problem

Zeige, dass auch NFAs nur reguläre Sprachen erkennen.

Nichtdeterministische endliche Automaten

Definition

- Ein nichtdet. endl. Automat (kurz: NFA; *Nondet. Finite Automaton*)

  $$N = (Z, \Sigma, \Delta, Q_0, E)$$

  ist genau so aufgebaut wie ein DFA, nur dass er
  - eine Menge $Q_0 \subseteq Z$ von Startzuständen hat und
  - die Überführungsfunktion folgende Form hat:

    $$\Delta : Z \times \Sigma \to \mathcal{P}(Z).$$

  - Hierbei bezeichnet $\mathcal{P}(Z)$ die Potenzmenge (also die Menge aller Teilmengen) von Z. Diese wird auch oft mit 2^Z bezeichnet.
- Die von einem NFA N akzeptierte oder erkannte Sprache ist

  $$L(N) = \left\{ x_1 \dots x_n \in \Sigma^* \;\middle|\; \begin{array}{c} \exists\, q_0 \in Q_0, q_1, \dots, q_{n-1} \in Z, q_n \in E: \\ q_{i+1} \in \Delta(q_i, x_{i+1}) \text{ für } i = 0, \dots, n-1 \end{array} \right\}.$$

- Eine Zustandsfolge $q_0, q_1, \dots, q_n$ heißt Rechnung von $N(x_1 \dots x_n)$, falls $q_0 \in Q_0$ und $q_{i+1} \in \Delta(q_i, x_{i+1})$ für $i = 0, \dots, n-1$ gilt.

Eigenschaften von NFAs

- Ein NFA N kann bei einer Eingabe x also nicht nur eine, sondern mehrere verschiedene Rechnungen parallel ausführen.
- Ein Wort x gehört genau dann zu $L(N)$, wenn $N(x)$ mindestens eine akzeptierende Rechnung hat.
- Im Gegensatz zu einem DFA, der jede Eingabe zu Ende liest, kann ein NFA N „stecken bleiben“.
- Dieser Fall tritt ein, wenn N in einen Zustand q gelangt, in dem er das nächste Eingabezeichen x_i wegen

  $$\Delta(q, x_i) = \varnothing$$

  nicht verarbeiten kann.

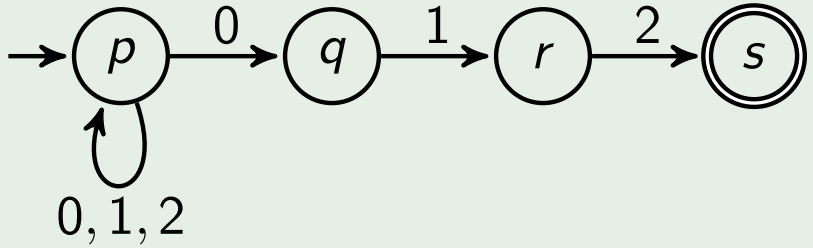
Eigenschaften von NFAs

Beispiel

- Betrachte den NFA $N = (Z, \Sigma, \Delta, Q_0, E)$ mit $Z = \{p, q, r, s\}$, $\Sigma = \{0, 1, 2\}$, $Q_0 = \{p\}$, $E = \{s\}$ und der Überführungsfunktion

Δ	p	q	r	s
0	$\{p, q\}$	$\varnothing$	$\varnothing$	$\varnothing$
1	$\{p\}$	$\{r\}$	$\varnothing$	$\varnothing$
2	$\{p\}$	$\varnothing$	$\{s\}$	$\varnothing$

Graphische Darstellung:



- Dann ist $L(M) = \{x012 \mid x \in \Sigma^*\}$ die Sprache aller Wörter, die mit dem Suffix 012 enden. ◁

Eigenschaften von NFAs

Beobachtung 5

Seien $N_i = (Z_i, \Sigma, \Delta_i, Q_i, E_i)$ NFAs mit $L(N_i) = L_i$ für $i = 1, 2$. Dann wird auch das Produkt $L_1 L_2$ von einem NFA erkannt.

Beweis

- Wir können $Z_1 \cap Z_2 = \varnothing$ annehmen.
- Dann gilt $L(N) = L_1 L_2$ für den NFA $N = (Z_1 \cup Z_2, \Sigma, \Delta, Q_1, E)$ mit

$$\Delta(p,a) = \begin{cases} \Delta_1(p,a), & p \in Z_1 \setminus E_1, \\ \Delta_1(p,a) \cup \bigcup_{q \in Q_2} \Delta_2(q,a), & p \in E_1, \\ \Delta_2(p,a), & \text{sonst} \end{cases}$$

und

$$E = \begin{cases} E_2, & Q_2 \cap E_2 = \varnothing, \\ E_1 \cup E_2, & \text{sonst.} \end{cases}$$

Eigenschaften von NFAs

- Dann gilt $L(N) = L_1 L_2$ für den NFA $N = (Z_1 \cup Z_2, \Sigma, \Delta, Q_1, E)$ mit

$$\Delta(p,a) = \begin{cases} \Delta_1(p,a), & p \in Z_1 \setminus E_1, \\ \Delta_1(p,a) \cup \bigcup_{q \in Q_2} \Delta_2(q,a), & p \in E_1, \\ \Delta_2(p,a), & \text{sonst}, \end{cases}$$

  und $E = E_2$, falls $Q_2 \cap E_2 = \varnothing$, bzw. $E = E_1 \cup E_2$ sonst.

Beweis von $L_1 L_2 \subseteq L(N)$:

- Seien $x = x_1 \cdots x_k \in L_1$, $y = y_1 \cdots y_l \in L_2$ und seien $q_0, \ldots, q_k$ und $p_0, \ldots, p_l$ akzeptierende Rechnungen von $N_1(x)$ und $N_2(y)$.
- Dann gilt $q_0 \in Q_1$, $q_k \in E_1$ und $p_0 \in Q_2$, $p_l \in E_2$.
- Im Fall $l \geq 1$ ist zudem $p_1 \in \Delta_2(p_0, y_1)$ und somit $p_1 \in \Delta(q_k, y_1)$.
- Im Fall $l = 0$ ist zudem $p_l \in Q_2 \cap E_2$ und somit $q_k \in E$.
- Also ist $q_0, \ldots, q_k, p_1, \ldots, p_l$ eine akzeptierende Rechnung von $N(xy)$.

Eigenschaften von NFAs

- Dann gilt $L(N) = L_1 L_2$ für den NFA $N = (Z_1 \cup Z_2, \Sigma, \Delta, Q_1, E)$ mit

$$\Delta(p,a) = \begin{cases} \Delta_1(p,a), & p \in Z_1 \setminus E_1, \\ \Delta_1(p,a) \cup \bigcup_{q \in Q_2} \Delta_2(q,a), & p \in E_1, \\ \Delta_2(p,a), & \text{sonst}, \end{cases}$$

und $E = E_2$, falls $Q_2 \cap E_2 = \varnothing$, bzw. $E = E_1 \cup E_2$ sonst.

Beweis von $L(N) \subseteq L_1 L_2$:

- Sei $x = x_1 \cdots x_n \in L(N)$ und sei $q_0, \ldots, q_n$ eine akz. Rechnung von $N(x)$.
- Dann gilt $q_0 \in Q_1$, $q_n \in E$, $q_0, \ldots, q_i \in Z_1$ und $q_{i+1}, \ldots, q_n \in Z_2$ für ein i.
- Im Fall $i = n$ ist $q_n \in E_1$ (d.h. $x \in L_1$) und $Q_2 \cap E_2 \neq \varnothing$ (d.h. $\varepsilon \in L_2$).
- Im Fall $i < n$ impliziert der Übergang $q_{i+1} \in \Delta(q_i, x_{i+1})$, dass $q_i \in E_1$ und $q_{i+1} \in \Delta_2(q, x_{i+1})$ für ein $q \in Q_2$ ist.
- Also ist $q_0, \ldots, q_i$ eine akz. Rechnung von $N_1(x_1 \cdots x_i)$ und $q, q_{i+1}, \ldots, q_n$ eine akz. Rechnung von $N_2(x_{i+1} \cdots x_n)$, d.h. $x \in L_1 L_2$. □

Eigenschaften von NFAs

Beobachtung 6

Ist $N = (Z, \Sigma, \Delta, Q_0, E)$ ein NFA, so wird auch die Sprache $L(N)^*$ von einem NFA erkannt.

Beweis

Die Sprache $L(N)^*$ wird von dem NFA

$$N' = (Z \cup \{q_{neu}\}, \Sigma, \Delta', Q_0 \cup \{q_{neu}\}, E \cup \{q_{neu}\})$$

mit

$$\Delta'(p,a) = \begin{cases} \Delta(p,a), & p \in Z \setminus E, \\ \Delta(p,a) \cup \bigcup_{q \in Q_0} \Delta(q,a), & p \in E, \\ \varnothing, & p = q_{neu} \end{cases}$$

erkannt.

□

Überblick

Ziel

Zeige, dass REG unter Produktbildung und Sternhülle abgeschlossen ist.

Problem

Bei der Konstruktion eines DFA für das Produkt $L_1 L_2$ bereitet es Schwierigkeiten, den richtigen Zeitpunkt für den Übergang von (der Simulation von) M_1 zu M_2 zu finden.

Lösungsidee (bereits umgesetzt)

Ein nichtdeterministischer Automat (NFA) kann den richtigen Zeitpunkt für den Übergang „raten“.

Noch zu zeigen

NFAs erkennen genau die regulären Sprachen.

NFAs erkennen genau die regulären Sprachen

Satz (Rabin und Scott)

REG = $\{L(N) \mid N \text{ ist ein NFA}\}$.

Beweis von REG $\subseteq \{L(N) \mid N \text{ ist ein NFA}\}$

Diese Inklusion ist klar, da jeder DFA $M = (Z, \Sigma, \delta, q_0, E)$ leicht in einen äquivalenten NFA

$$N = (Z, \Sigma, \Delta, Q_0, E)$$

transformiert werden kann, indem wir $\Delta(q, a) = \{\delta(q, a)\}$ und $Q_0 = \{q_0\}$ setzen. □

Für die umgekehrte Inklusion ist das Erreichbarkeitsproblem für NFAs von zentraler Bedeutung.

Das Erreichbarkeitsproblem für NFAs

Frage

Sei $N = (Z, \Sigma, \Delta, Q_0, E)$ ein NFA und sei $x = x_1 \dots x_n$ eine Eingabe. Welche Zustände sind in i Schritten erreichbar?

Antwort

- in 0 Schritten: alle Zustände in Q_0.
- in einem Schritt: alle Zustände in

$$Q_1 = \bigcup_{q \in Q_0} \Delta(q, x_1).$$

- in i Schritten: alle Zustände in

$$Q_i = \bigcup_{q \in Q_{i-1}} \Delta(q, x_i).$$

Simulation von NFAs durch DFAs

Idee

- Wir können einen NFA $N = (Z, \Sigma, \Delta, Q_0, E)$ durch einen DFA $M = (Z', \Sigma, \delta, q_0', E')$ simulieren, der in seinem Zustand die Information speichert, in welchen Zuständen sich N momentan befinden könnte.
- Die Zustände von M sind also Teilmengen Q von Z (d.h. $Z' = \mathcal{P}(Z)$) mit Q_0 als Startzustand (d.h. $q_0' = Q_0$) und der Endzustandsmenge

  $$E' = \{Q \subseteq Z \mid Q \cap E \neq \varnothing\}.$$

- Die Überführungsfunktion $\delta : \mathcal{P}(Z) \times \Sigma \to \mathcal{P}(Z)$ von M berechnet dann für einen Zustand $Q \subseteq Z$ und ein Zeichen $a \in \Sigma$ die Menge

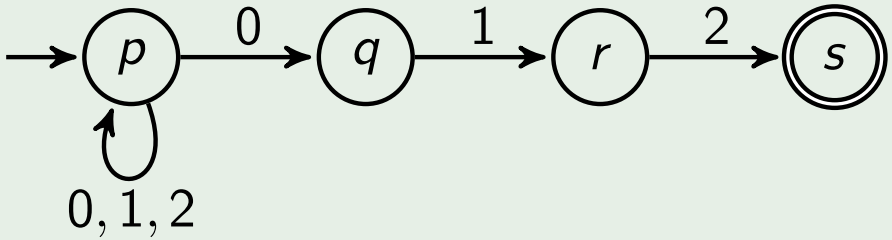
  $$\delta(Q, a) = \bigcup_{q \in Q} \Delta(q, a)$$

  aller Zustände, in die N gelangen kann, wenn N ausgehend von einem beliebigen Zustand $q \in Q$ das Zeichen a liest.
- M wird auch als der zu N gehörige Potenzmengenautomat bezeichnet.

Simulation von NFAs durch DFAs

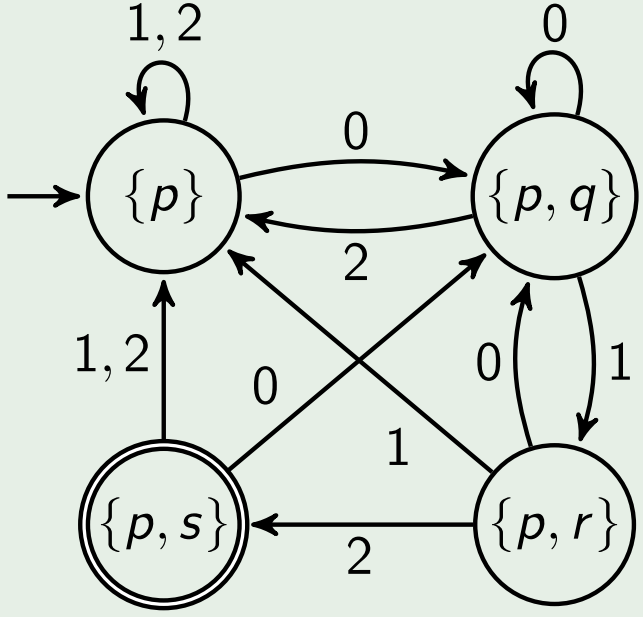
Beispiel

- Betrachte den NFA N



- Ausgehend von $Q_0 = \{p\}$ liefert δ dann die folgenden Werte:

δ	0	1	2
$\{p\}$	$\{p, q\}$	$\{p\}$	$\{p\}$
$\{p, q\}$	$\{p, q\}$	$\{p, r\}$	$\{p\}$
$\{p, r\}$	$\{p, q\}$	$\{p\}$	$\{p, s\}$
$\{p, s\}$	$\{p, q\}$	$\{p\}$	$\{p\}$



Simulation von NFAs durch DFAs

Bemerkung

- Im obigen Beispiel werden für die Konstruktion des Potenzmengenautomaten nur 4 der insgesamt

$$\|\mathcal{P}(Z)\| = 2^{\|Z\|} = 2^4 = 16$$

  Zustände benötigt, da die übrigen 12 Zustände nicht erreichbar sind.
- Es gibt jedoch Beispiele, bei denen alle $2^{\|Z\|}$ Zustände benötigt werden (siehe Übungen).

NFAs erkennen genau die regulären Sprachen

Beweis von $\{L(N) \mid N \text{ ist ein NFA}\} \subseteq \text{REG}$

- Sei $N = (Z, \Sigma, \Delta, Q_0, E)$ ein NFA und sei $M = (\mathcal{P}(Z), \Sigma, \delta, Q_0, E')$ der zugehörige Potenzmengenautomat mit $\delta(Q, a) = \bigcup_{q \in Q} \Delta(q, a)$ und $E' = \{Q \subseteq Z \mid Q \cap E \neq \varnothing\}$.
- Dann folgt die Korrektheit von M leicht mittels folgender Behauptung, die wir auf der nächsten Folie beweisen.

Behauptung

$\hat{\delta}(Q_0, x)$ enthält genau die von N nach Lesen von x erreichbaren Zustände.

- Für alle Wörter $x \in \Sigma^*$ gilt

$$\begin{aligned} x \in L(N) &\Leftrightarrow N \text{ kann nach Lesen von } x \text{ einen Endzustand erreichen} \\ &\overset{\text{Beh.}}{\Leftrightarrow} \hat{\delta}(Q_0, x) \cap E \neq \varnothing \\ &\Leftrightarrow \hat{\delta}(Q_0, x) \in E' \\ &\Leftrightarrow x \in L(M). \end{aligned}$$

□

Beweis der Behauptung

Behauptung

$\hat{\delta}(Q_0, x)$ enthält genau die von N nach Lesen von x erreichbaren Zustände.

Beweis durch Induktion über die Länge n von x

$n = 0$: klar, da $\hat{\delta}(Q_0, \varepsilon) = Q_0$ ist.

$n - 1 \rightsquigarrow n$: Sei $x = x_1 \dots x_n$ gegeben. Nach IV enthält

$$Q_{n-1} = \hat{\delta}(Q_0, x_1 \dots x_{n-1})$$

die Zustände, die N nach Lesen von $x_1 \dots x_{n-1}$ erreichen kann. Wegen

$$\hat{\delta}(Q_0, x) = \delta(Q_{n-1}, x_n) = \bigcup_{q \in Q_{n-1}} \Delta(q, x_n)$$

enthält dann aber $\hat{\delta}(Q_0, x)$ die Zustände, die N nach Lesen von x erreichen kann. □

Abschlusseigenschaften der Klasse REG

Korollar

Die Klasse REG der regulären Sprachen ist unter folgenden Operationen abgeschlossen:

- Komplement,
- Schnitt,
- Vereinigung,
- Produkt,
- Sternhülle.