Automaten und Formale Sprachen

SoSe 2017 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

4. Mai 2017

Automaten und Formale Sprachen Gesamtübersicht

- Organisatorisches
- Einführung
- **Endliche Automaten und reguläre Sprachen**
- Kontextfreie Grammatiken und kontextfreie Sprachen
- Chomsky-Hierarchie

Endliche Automaten und reguläre Sprachen

1. Deterministische endliche Automaten

2. **Nichtdeterministische endliche Automaten**

3. Reguläre Ausdrücke

4. Nichtreguläre Sprachen

5. Algorithmen mit / für endliche Automaten

Operationen auf Sprachen

Erinnerung: Eine Sprache ist eine *Menge* von Wörtern.

Also: Vereinigung, Durchschnitt, Komplement, ... von Sprachen liefern wieder Sprachen, sind also *Operationen auf Sprachen*.

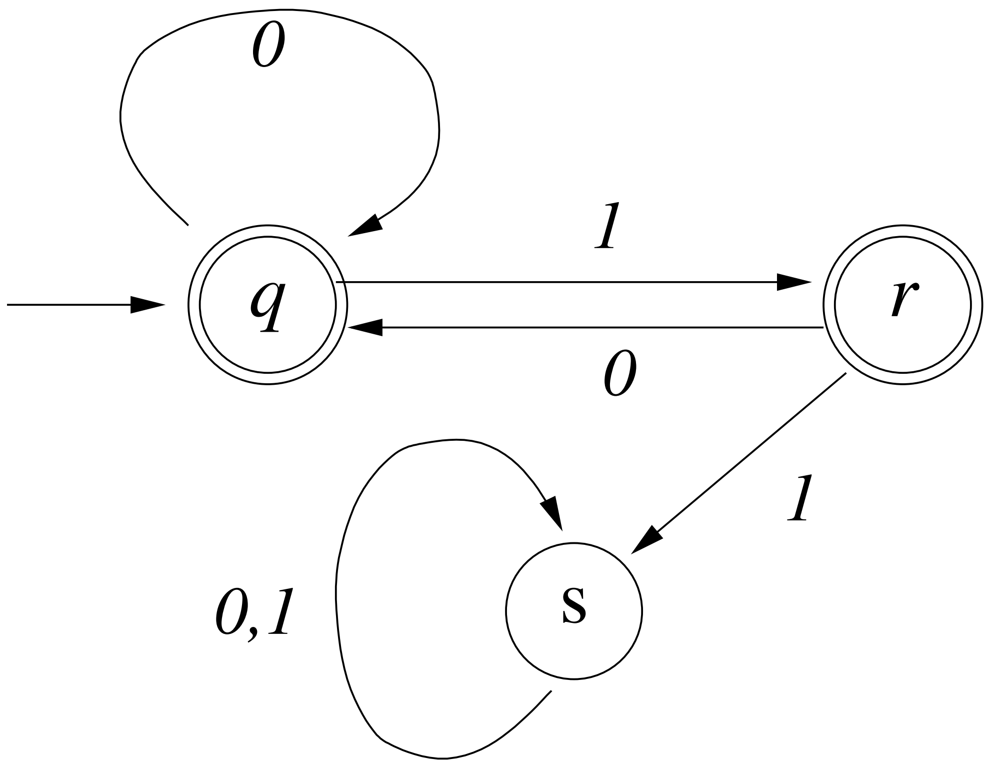
Eine Menge von Sprachen heißt auch *Sprachfamilie*.

Wir haben bislang die Familie der regulären Sprachen **REG** kennengelernt.

Gleichermaßen könnten wir die Familie **DEA** der DEA-akzeptierbaren Sprachen betrachten.

Ist f eine k-stellige Operation auf Sprachen und ist $\mathcal{L}$ eine Sprachfamilie, so heißt $\mathcal{L}$ *abgeschlossen gegen* f gdw. für alle $L_1, \ldots, L_k \in \mathcal{L}$ gilt: $f(L_1, \ldots, L_k) \in \mathcal{L}$.

Ein Beispiel Was beschreibt folgender Automat ?



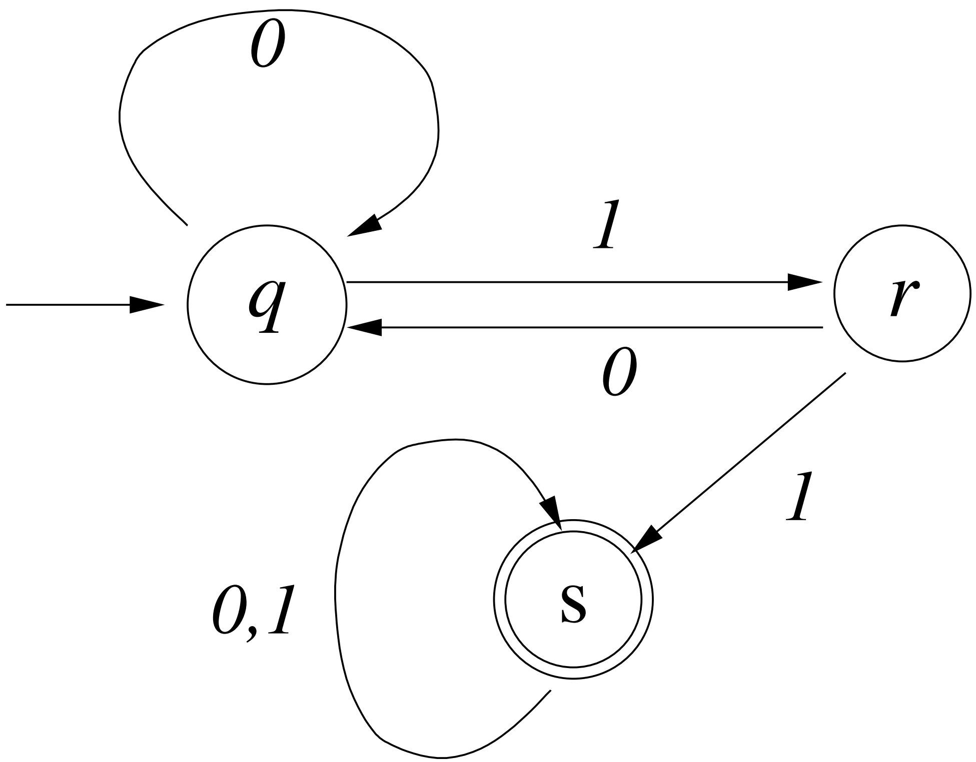
$$
\begin{aligned}
L(A) &= \{w \in \{0,1\}^* \mid w \text{ enthält } \underline{\text{nicht}} \text{ das Teilwort } 11\} \\
&= \{w \in \{0,1\}^* \mid \text{ Auf jedes Vorkommen von } 1 \\
&\qquad \text{vor der letzten Stelle in } w \text{ folgt } 0. \}
\end{aligned}
$$

Satz: **DEA**-Sprachen sind komplementabgeschlossen.

Beweis: Sprachkomplement entspricht Endzustandsmengenkomplement.

Hinweis: Dies ist streng genommen gar kein Beweis, sondern die bloße Angabe einer Konstruktion, die zu vorgelegtem DEA A einen DEA A' bastelt, für den zu zeigen wäre, dass $L(A)$ das Komplement von $L(A')$ akzeptiert. Man hat also zu zeigen: $w \in L(A) \to w \notin L(A')$ und $w \in L(A') \to w \notin L(A)$. Die formale Durchführung ist Ihnen zur Übung überlassen.

Das Beispiel



Schwieriger: Vereinigungsbildung

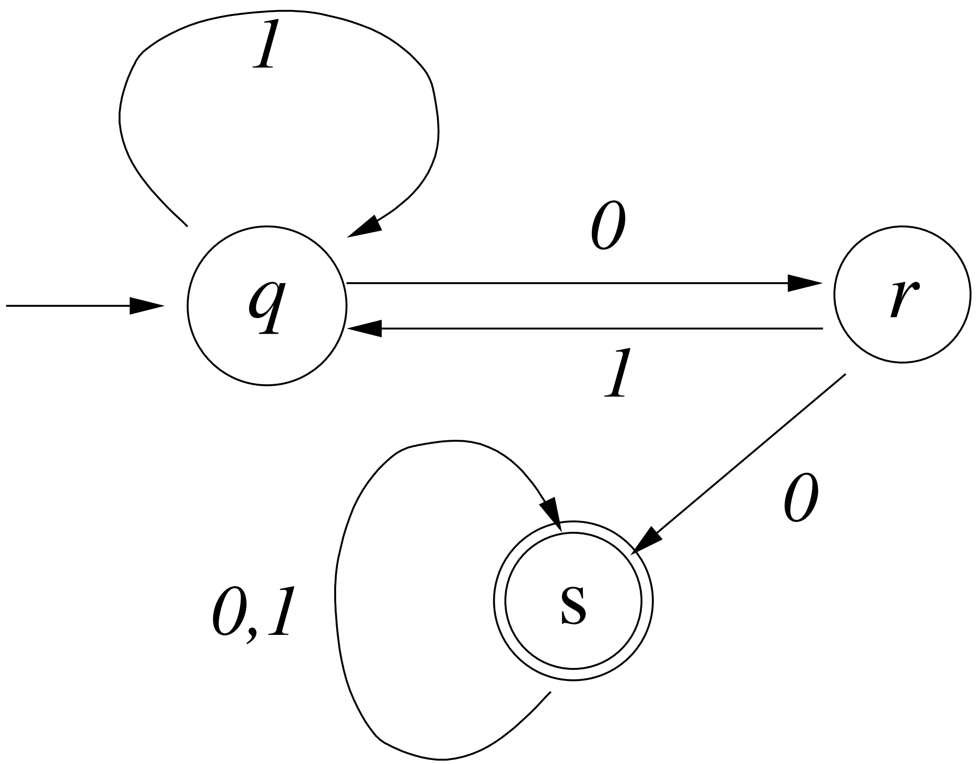
Wie kann man aus DEA-Beschreibungen für

$$\begin{aligned} L_1 &= \{w \in \{0,1\}^* \mid w \text{ enthält das Teilwort } 11\}, \\ L_2 &= \{w \in \{0,1\}^* \mid w \text{ enthält das Teilwort } 00\}, \end{aligned}$$

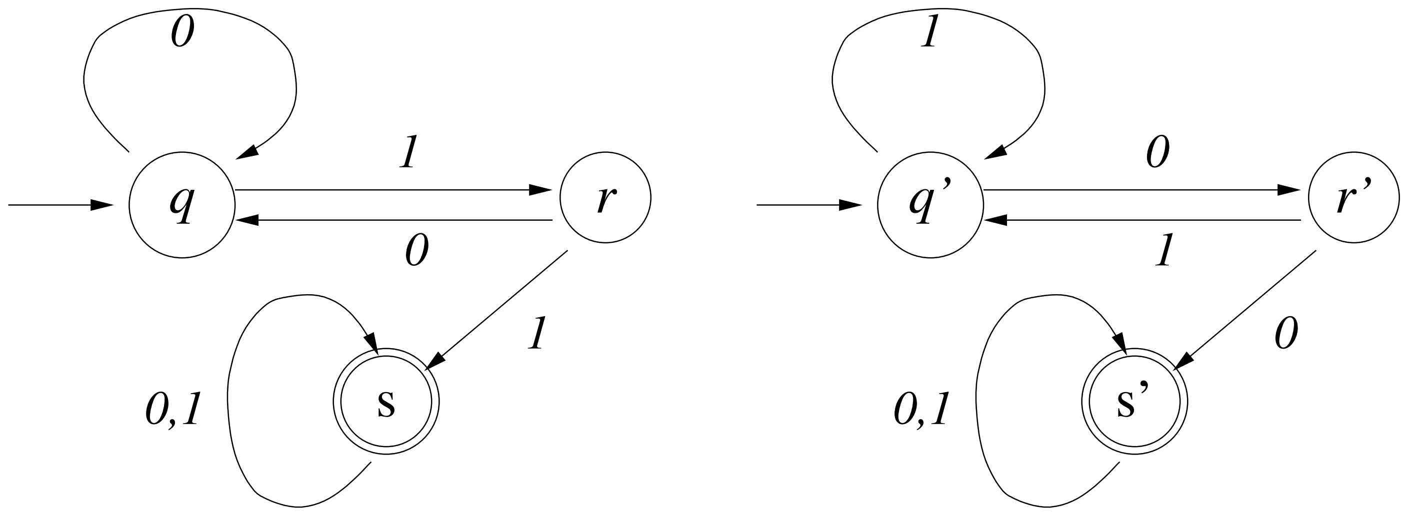
einen DEA für $L_1 \cup L_2$ basteln ?

<u>Idee</u>: Verwende "irgendwie" Automaten für L_1 und für L_2.
Kann man EAs aus "Teilprogrammen" zusammensetzen ?

Ein DEA für L_2 sähe wie folgt aus:



Vereinigung durch mehrere Anfangszustände ??



Ein **nichtdeterministischer endlicher Automat** oder NEA
wird beschrieben durch ein Quintupel

$$A = (Q, \Sigma, \delta, Q_0, F)$$

wobei gilt:

Q: endliche Menge von *Zuständen* (Zustandsalphabet)

Σ: endliche Menge von *Eingabezeichen* (Eingabealphabet)

$\delta \subseteq Q \times \Sigma \times Q$: *Überführungsrelation*

$Q_0 \subseteq Q$: *Anfangszustände*

$F \subseteq Q$: *Endzustände*

Sprachfamilie: **NEA**.

Überführungstafel

Ein endlicher Automat kann vollständig durch seine *Überführungstafel* beschrieben werden.

Beispiel: Betrachte:

δ	0	1
$\rightarrow q$	q	r
$r \rightarrow$	q	s
s	s	s
$\rightarrow q'$	r'	q'
$r' \rightarrow$	s'	q'
s'	s'	s'

Die **von einem NEA $A = (Q, \Sigma, \delta, Q_0, F)$ akzeptierte Sprache** kann man formal wie folgt beschreiben.

Ein Element aus $C = Q \times \Sigma^*$ heißt *Konfiguration* von A.
Definiere eine Binärrelation $\vdash_A$ auf C durch $(q, w) \vdash_A (q', w')$ gdw. $\exists a \in \Sigma$: $w = aw'$ und $\underline{\delta \ni (q, a, q')}$.
Die zweite Komponente einer Konfiguration ist die Resteingabe.
$\vdash_A$ beschreibt einen möglichen Konfigurationsübergang in einem Schritt.
Entsprechend beschreibt $\vdash_A^n$ n Schritte von A.
Daher können wir definieren:

$$L(A) = \{w \in \Sigma^* \mid \underline{\exists q_0 \in Q_0}, q \in F : (q_0, w) \vdash_A^* (q, \lambda)\}.$$

NEAs zur Spezifikation I

Beispiel: Gesucht: NEA, der in der Reihenfolge (aber nicht unbedingt nebeneinander) die Zeichen 0, 1, 1 enthält.

Lösung:

δ	0	1
$\rightarrow q$	q, r	q
r	r	r, s
s	s	s, t
$t \rightarrow$	t	t

Dazu DEA ?!

Idee: “erstes Vorkommen” zu prüfen genügt !

δ	0	1
$\rightarrow q$	r	q
r	r	s
s	s	t
$t \rightarrow$	t	t

Hinweis: Pattern Matching !

NEAs zur Spezifikation II

Satz: Jede endliche Sprache ist **NEA**-Sprache.

Beweis: Sei $L = \{w_1, \dots, w_M\} \subseteq \Sigma^*$ mit

$$w_i = a_{i,1} \cdots a_{i,\ell(w_i)}, \ 1 \leq i \leq M.$$

Betrachte den folgendenmaßen spezifizierten *Skelettautomaten*:

$$Q = \{(i,j) \mid 1 \leq i \leq M, 0 \leq j \leq \ell(w_i)\},$$

$$Q_0 = \{(i,0) \mid 1 \leq i \leq M\} \quad \text{und} \quad F = \{(i, \ell(w_i)) \mid 1 \leq i \leq M\}$$

$$\delta = \{((i,j), a, (i, j+1)) | 1 \leq i \leq M, 0 \leq j < \ell(w_i), a_{i,j+1} = a\}.$$

Unterschiede DEA / NEA Spezifikation

anhand der Überführungstafel; wie DEA, ABER:

- Einträge dürfen leer sein, d.h. der Automat ist *unvollständig*.
  Manchmal auch bei DEAs zugelassen... (partielle DEAs)

- Es gibt mehr als einen Eintrag (das heißt ja Nichtdeterminismus!).

- Es gibt eine Anfangszustandsmenge (auch dies ist Nichtdeterminismus).

- Manchmal werden neben Buchstaben auch Wörter als Spaltenüberschrift zugelassen, insbesondere das leere Wort: *NEA mit λ-Übergängen*.

Vereinigung leicht gemacht

Satz: **NEA** ist unter Vereinigung abgeschlossen.

Beweis: Es seien $A_i = (Q_i, \Sigma, \delta_i, Q_{i,0}, F_i)$ für $i = 1, 2$ zwei NEAs.

Durch Umbenennen können wir $Q_1 \cap Q_2 = \emptyset$ voraussetzen.

Dann wird $L(A_1) \cup L(A_2)$ beschrieben durch:

$$(Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, Q_{1,0} \cup Q_{2,0}, F_1 \cup F_2).$$

Formaler Beweis der Korrektheit der Konstruktion ... $\square$

NEA=DEA

Beweis: $\supseteq$ ✓. Für $\subseteq$ betrachte NEA $A = (Q, \Sigma, \delta, Q_0, F)$.
Die Relation δ lässt sich als Abbildung

$$\delta_f : Q \times \Sigma \longrightarrow 2^Q \text{ auffassen mit } (q, a) \mapsto \{r \in Q \mid (q, a, r) \in \delta\}.$$

Dies liefert auch eine Abbildung $\delta_f : 2^Q \times \Sigma \longrightarrow 2^Q$, $(P, a) \mapsto \bigcup_{p \in P} \delta_f(p, a)$.
Setze $F' = \{P \in 2^Q \mid P \cap F \neq \emptyset\}$ und betrachte DEA

$$A' = (2^Q, \Sigma, \delta_f, Q_0, F').$$

Behauptung: $L(A) = L(A')$. $\square$

Hinweis: Zustandsexplosion !

Satz: NEAs kennzeichnen die regulären Sprachen.

Zustandsexplosion: ein böses Beispiel

$$L_k = \{x \in \{0,1\}^* \mid \ell(x) \geq k,\ \text{das } k\text{-letzte Zeichen von } x \text{ ist } 0\ \}$$

Lemma: L_k kann durch einen NEA mit $k+1$ Zuständen erkannt werden, aber durch keinen DEA mit weniger als 2^k Zuständen.

<u>Beweis:</u> Der NEA ist durch folgende Tafel beschrieben:

	0	1	
$\rightarrow q_0$	q_0, q_1	q_0	
q_i	q_{i+1}	q_{i+1}	für $0 < i < k$
$q_k \rightarrow$			

Warum ist das Beispiel “böse” ?

$$L_k = \{x \in \{0,1\}^* \mid \ell(x) \geq k, \text{ das } k\text{-letzte Zeichen von } x \text{ ist 0 }\}$$

Gäbe es einen DEA A mit $< 2^k$ Zuständen für L_k, so muss es (Dirichletsches Schubfachprinzip) zwei verschiedene Wörter $y_1, y_2 \in \{0,1\}^k$ und einen Zustand q geben mit $(q_0, y_i) \vdash_A^k (q, \lambda)$ für $i = 1, 2$.
Wähle j, sodass es ein Wort $u \in \{0,1\}^{j-1}$ gibt mit $y_1 = ua_1v_1$ und $y_2 = ua_2v_2$ mit $\{a_1, a_2\} = \{0,1\}$. u ist längster gemeinsamer Präfix von y_1, y_2.
Da $\ell(v_1) = \ell(v_2) = k - j$, liegt genau eines der Wörter y_1u bzw. y_2u in L_k.
Andererseits gibt es ein q', sodass:

$$(q_0, y_iu) \vdash_A^k (q, u) \vdash_A^{j-1} (q', \lambda) \quad \text{für } i = 1, 2,$$

d.h., y_1u und y_2u werden gemeinsam akzeptiert oder verworfen. ↯ $\square$

Zustandsexplosion vermeidbar ?

Manchmal ja: durch “lazy evaluation” (Bereitstellen erst wenn nötig!)

Beispiel: Betrachte den Skelettautomaten zu $L = \{a, aa, ab, abb\}$ mit Zustandsmenge $\{(1,0),(1,1);(2,0),(2,1),(2,2);(3,0),(3,1),(3,2);(4,0),(4,1),(4,2),(4,3)\}$.

Beginnen wir mit $Q_0 = \{(1,0),(2,0),(3,0),(4,0)\}$, dem Startzustand des DEA.

$\delta_f(Q_0, a) = \{(1,1),(2,1),(3,1),(4,1)\} =: Q_1$

$\delta_f(Q_0, b) = \emptyset$

$\delta_f(Q_1, a) = \{(2,2)\} =: Q_2$

$\delta_f(Q_1, b) = \{(3,2),(4,2)\} =: Q_3$

$\delta_f(Q_3, a) = \emptyset$

$\delta_f(Q_3, b) = \{(4,3)\} =: Q_4$

$\delta_f(\emptyset, x) = \delta_f(Q_2, x) = \delta_f(Q_4, x) = \emptyset$ für $x = a, b$.

Endzustände sind Q_1, Q_2, Q_3, Q_4.

Anstelle von 2^{12} hat unser DEA nur 6 Zustände, weniger als Ausgangs-NEA!

Der so aus Skelettautomaten gewonnene DEA heißt *Präfixbaumakzeptor*.

λ-Übergänge

Manchmal: NEAs mit λ-Übergängen (kurz: λ-NEA), d.h., für die Überführungsrelation δ gilt: $\delta \subseteq Q \times (\Sigma \cup \{\lambda\}) \times Q$.
Betrachte λ-NEA $A = (Q, \Sigma, \delta, Q_0, F)$ und die Relation

$$R_\lambda = \{(q_1, q_2) \mid (q_1, \lambda) \vdash_A^* (q_2, \lambda)\} \subseteq Q \times Q.$$

Für $A' = (Q, \Sigma, \delta', Q_0', F)$ gilt $L(A) = L(A')$, wobei
$Q_0' = \{q \in Q \mid \exists q_0 \in Q_0 : (q_0, q) \in R_\lambda\}$
$\delta' = \{(p, a, q') \in Q \times \Sigma \times Q \mid \exists q \in Q : (p, a, q) \in \delta, (q, q') \in R_\lambda\}$.
Es gilt für $w = a_0 \dots a_{n-1} \in \Sigma^n$: $(q_0, w) \vdash_A^* (q_f, \lambda)$, $q_f \in F$, gdw. (mit $q_n' = q_f$) $\exists q_0', q_1, q_1', \dots, q_n \in Q : \forall i = 0, \dots, n-1 : (q_i, q_i') \in R_\lambda, (q_i', a_i, q_{i+1}) \in \delta$, $(q_n, q_n') \in R_\lambda$ gdw. $\forall i = 0, \dots, n-1 : (q_i', a_i, q_{i+1}') \in \delta'$. $\square$
Die Konstruktion zeigt:
Satz: Zu jedem λ-NEA gibt es einen *äquivalenten* NEA.

Zur Berechnung der transitiven Hülle I

Motivation: R_λ ist die reflexiv-transitive Hülle der Relation $\delta \cap Q \times \{\lambda\} \times Q$.

Betrachte allgemein $X = \{1, \ldots, n\}$ und eine Binärrelation R auf X.
Frage: Wie berechnet man die transitive Hülle R^+?
(Beachte: $R^* = (R \cup R^0)^+$.)

Die Definition von R^+ liefert direkt einen "$O(n^4)$" Algorithmus:
Berechne R^2, R^3, ..., R^n und dann $R^+ = \bigcup_{i=1}^n R^i$.
Die "Verdopplungstechnik" liefert Ergebnis in Zeit $O(n^3 \log(n))$:
Berechne $R_1 := R^2 \cup R$, $R_2 = (R_1)^2 \cup R$, R_3, ..., $R^+ = R_{\log n}$.

Zur Berechnung der transitiven Hülle II (Wiederholung DS)

Der Algorithmus von Warshall/Floyd liefert kubischen Algorithmus.

Erinnerung: Binärrelationen kann man sich als Graphen veranschaulichen mit Grundelementen als Knoten und Deutung der Relation als Kantenrelation.

Berechnung der transitiven Hülle entspricht Berechnung aller Wege.

$R[i, j, k]$: Gibt es einen “Pfad” von i nach j,
der nur die Knoten $1, \ldots, k$ als Zwischenpunkte benutzt ?

$R[i, j, 0]$ gdw. $(i, j) \in R$, oder kurz: $R[i, j, 0] := [(i, j) \in R]$.

$R[i, j, k] := R[i, j, k-1] \vee (R[i, k, k-1] \wedge R[k, j, k-1])$ für $k > 0$.

Offenbar (?!) ist: $[(i, j) \in R^+] = R[i, j, n]$ für alle i, j.

Zur Berechnung der transitiven Hülle III (Wiederholung DS)

Wieso ist dies tatsächlich ein Algorithmus (entweder *rekursiv* oder *iterativ*) ?

$R[1..n, 1..n, 0..n]$ ist 3-dimensionales Boolesches Array (Feld, Matrix), d.h., die Einträge lauten **wahr** oder **falsch** (1 oder 0).

Für $i := 1$ bis n tue:
    Für $j := 1$ bis n tue:
        $R[i, j, 0] := [(i, j) \in R]$.

Für $k := 1$ bis n tue:
    Für $i := 1$ bis n tue:
        Für $j := 1$ bis n tue:
            $R[i, j, k] := R[i, j, k-1] \vee (R[i, k, k-1] \wedge R[k, j, k-1])$

Damit klar: kubische Komplexität, i.Z.: $O(n^3)$.

Warum NEAs mit λ-Übergängen ?

Lemma: Zu jedem NEA (mit λ-Übergängen) gibt es einen äquivalenten NEA mit λ-Übergängen, der nur einen Anfangs- und nur einen Endzustand besitzt; der Anfangszustand hat nur ausgehende Kanten und der Endzustand nur eingehende Kanten.

Beweis: Führe neuen Anfangszustand q_0 und neuen Endzustand q_f ein.
Verbinde q_0 zu allen "alten" Anfangszuständen mit λ-Übergängen.
Führe λ-beschriftete Kanten ein von allen "alten" Endzuständen zu q_f. $\square$

Beispiel: Man veranschauliche sich die Konstruktion beim Skelettautomaten!

Zusammenfassung und Ausblick

- Wir haben das Konzept des Nichtdeterminismus am Beispiel nichtdeterministischer endlicher Automaten kennengelernt.
- Kurz gesagt: NEAs dürfen bei der Erkennung von Wörtern “richtig raten”.
- Dieses Konzept ist zentral für das Verständnis eines der wichtigsten Probleme der (Theoretischen) Informatik, dem “P-NP-Problem” (siehe Vorlesung B+KT aus demselben Modul)
- Die Zustandsexplosion ist hierbei wiederum wichtig für das Verständnis.