

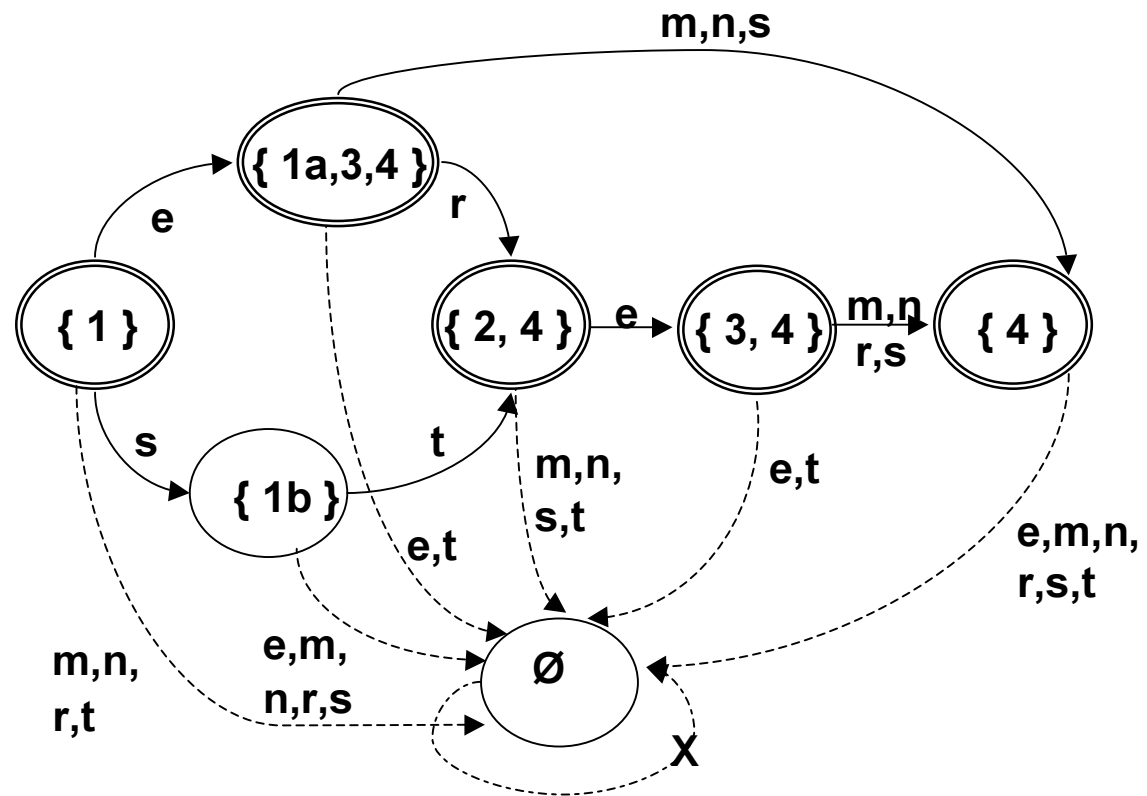
Beispiel: DEA für Adjektiv-Endungen

- Grundlage: der buchstabierende Automat
 $A = \langle \{1, 1a, 1b, 2, 3, 4\}, \{e, m, n, r, s, t\}, \Delta, 1, \{1, 4\} \rangle$,
 Δ wie im Diagramm Folie 42
- Potenzautomat ist $A' = \langle K', \Sigma, \delta, s', F' \rangle$
mit
 $K' = \wp(K)$
 $s' = \{s\}$
 $F' = \{q' \in K' \mid 1 \in q' \text{ oder } 4 \in q'\}$
 δ s. Übergangstabelle nächste Folie

DEA für Adjektiv-Endungen, Übergangstabelle

$\delta:$	e	m	n	r	s	t
{1}	{1a,3,4}	\emptyset	\emptyset	\emptyset	{1b}	\emptyset
{1a,3,4}	\emptyset	{4}	{4}	{2,4}	{4}	\emptyset
{1b}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	{2,4}
{2,4}	{3,4}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
{3,4}	\emptyset	{4}	{4}	{4}	{4}	\emptyset
{4}	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset

Das Diagramm



Noch einmal: Der DEA für Adjektiv-Endungen

- Die Zustandsmenge des Potenzautomaten A' ist eigentlich $K' = \wp(K)$, er hat in unserem Beispiel also $2^6 = 64$ Zustände. Wie die Übergangstabelle zeigt, sind vom Startzustand $\{1\}$ aus aber nur 7, ohne den „trap state“ \emptyset 6 echte Zustände erreichbar. Die übrigen Zustände können ignoriert werden.

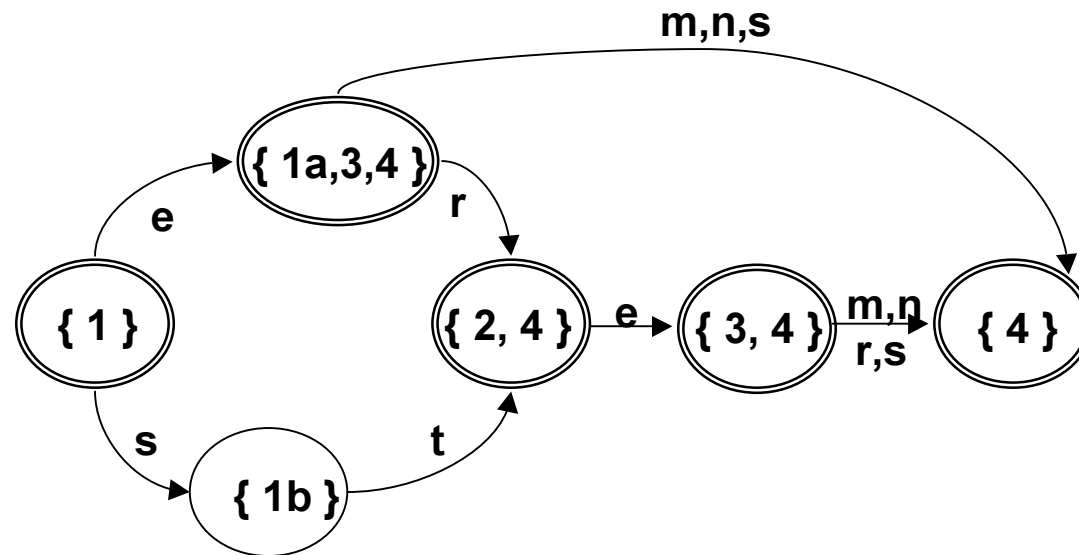
Wir können also, wie im Diagramm, ausgehen von:

$$K' = \{\{1\}, \{1a,3,4\}, \{1b\}, \{2,4\}, \{3,4\}, \{4\}, \emptyset\} \text{ und}$$

$$F' = \{\{1\}, \{1a,3,4\}, \{2,4\}, \{3,4\}, \{4\}\}$$

- Im Diagramm können wir außerdem noch, per Konvention, den Zustand \emptyset und alle hinführenden Kanten unterschlagen, und erhalten dann das vereinfachte Diagramm auf der folgenden Folie.

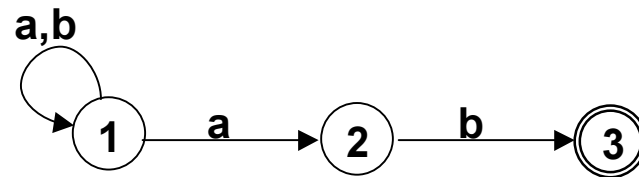
Das Diagramm, vereinfacht



Potenzautomatenkonstruktion, Beispiel 2

NEA $A = \langle \{1,2,3\}, \{a,b\}, \Delta, 1, \{3\} \rangle$

Δ gegeben durch:



DEA

$A' = \langle \wp(\{1,2,3\}), \{a,b\}, \delta, \{1\}, F' \rangle$

$F' = \{\{3\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}$

Potenzautomatenkonstruktion, Beispiel 2: Vollständige Übergangstabelle

q	$\delta(q, a)$	$\delta(q, b)$
{1}	{1,2}	{1}
{2}	\emptyset	{3}
{3}	\emptyset	\emptyset
{1,2}	{1,2}	{1,3}
{1,3}	{1,2}	{1}
{2,3}	\emptyset	{3}
{1,2,3}	{1,2}	{1,3}
\emptyset	\emptyset	\emptyset

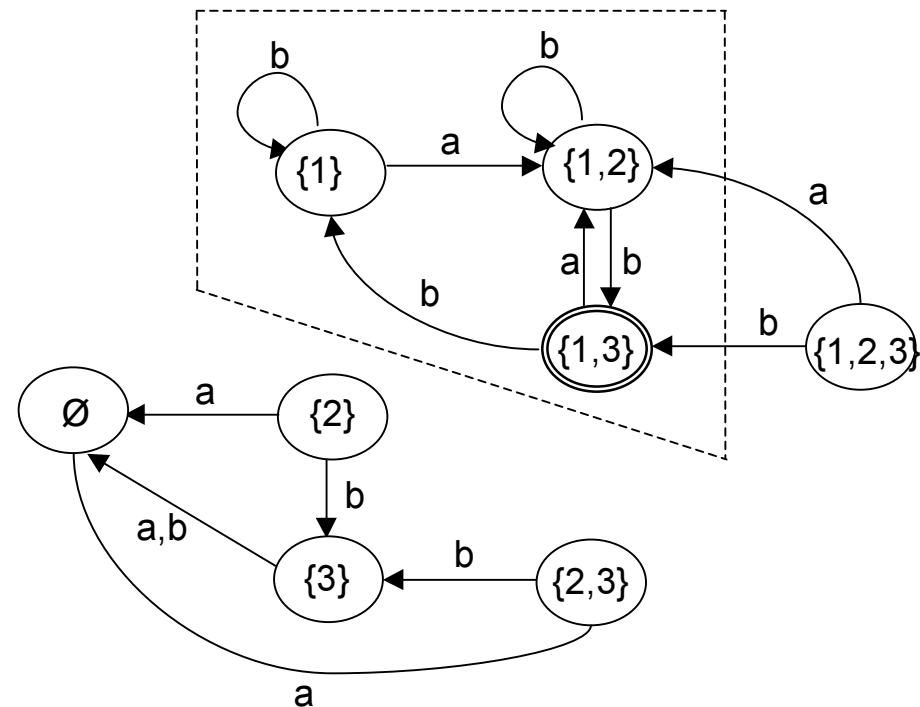
Potenzautomatenkonstruktion, Beispiel 2: Erreichbare Zustände/relevante Übergänge

q	$\delta(q, a)$	$\delta(q, b)$
{1}	{1,2}	{1}
{2}	\emptyset	{3}
{3}	\emptyset	\emptyset
{1,2}	{1,2}	{1,3}
{1,3}	{1,2}	{1}
{2,3}	\emptyset	{3}
{1,2,3}	{1,2}	{1,3}
\emptyset	\emptyset	\emptyset

Potenzautomatenkonstruktion, Beispiel 2: Das Zustandsdiagramm

Nur ein Teil der
Zustände ist vom
Startzustand aus
erreichbar.

Die übrigen
Zustände sind
funktionslos.



Ein dritter Formalismus: Reguläre Ausdrücke

- Reguläre Ausdrücke sind, neben NEA und DEA, ein dritter Formalismus, um Sprachen über einem Alphabet zu beschreiben.
- Reguläre Ausdrücke zu einem Alphabet Σ werden aus den Symbolen von Σ und Operatoren für Konkatination, Alternative/Disjunktion und beliebig häufige Wiederholung/Iteration gebildet.
- Formale Definition: Die Menge der regulären Ausdrücke zu einem Alphabet Σ ist die kleinste Menge, für die gilt:
 - \emptyset ist regulärer Ausdruck
 - Jedes $a \in \Sigma$ ist regulärer Ausdruck
 - Wenn α, β reguläre Ausdrücke sind, so auch
 - $\alpha + \beta$ (Alternative)
 - $\alpha \circ \beta$ (Konkatination)
 - α^* (Iteration)

Reguläre Ausdrücke: Beispiele

- Reguläre Ausdrücke erlauben, anders als endliche Automaten, die komfortable Darstellung von Sprachen als lineare Zeichenketten, die sich gut für die Programmierung eignen (die Sprache PERL basiert auf regulären Ausdrücken).
- Beispiele:
 - $\text{ARToADJA}^*\text{oNN}$ bzw. $\text{ARTo}(\text{GPRT}^*\text{oADJA})^*\text{oNN}$ charakterisieren die Sprachen, die von den früher vorgestellten Wortart-Muster-Automaten für Nominalausdrücke akzeptiert werden.
 - Die Sprache über dem Alphabet $\{a,b\}$, die alle Worte mit Suffix ab enthält, wird durch $(a+b)^*\text{oaob}$ charakterisiert.

Reguläre Ausdrücke: Interpretation

- Die vom regulären Ausdruck Φ über Σ beschriebene Sprache nennen wir $L(\Phi)$. $L(\Phi)$ wird für beliebige reguläre Ausdrücke in der folgenden Weise rekursiv definiert:
 - $L(\emptyset) = \emptyset$
 - $L(a) = \{a\}$ für $a \in \Sigma$
 - $L(\alpha + \beta) = L(\alpha) \cup L(\beta)$
 - $L(\alpha \circ \beta) = \{ww' \mid w \in L(\alpha) \text{ und } w' \in L(\beta)\}$
 - $L(\alpha^*) = \{w_1 \dots w_n \mid w_1, \dots, w_n \in L(\alpha), n \geq 0\}$
- Anmerkung: Es gilt $L(\emptyset^*) = \{\varepsilon\}$

Reguläre Ausdrücke und endliche Automaten [1]

- Wird eine Sprache durch einen regulären Ausdruck (RA) beschrieben, kann sie auch durch einen nicht-deterministischen endlichen Automaten dargestellt werden: Zu jedem regulären Ausdruck Φ gibt es einen NEA A mit $L(A) = L(\Phi)$.
 - Beweisidee: Konstruktiver Beweis durch Induktion über den Aufbau der regulären Ausdrücke.
 - Basis: $L(a) = \{a\}$ und $L(\emptyset) = \emptyset$ werden akzeptiert durch:



- Induktionsschritt: Wenn $L(\alpha)$ durch den NEA A_1 und $L(\beta)$ durch den NEA A_2 akzeptiert wird, wird $L(\alpha + \beta)$ akzeptiert durch den Automaten, den man aus A_1 und A_2 erhält, indem man einen neuen Startzustand einführt und diesen mittels ε -Kanten mit den Startzuständen von A_1 und A_2 verbindet. Entsprechende einfache Konstruktionen können für $L(\alpha \circ \beta)$ und $L(\alpha^*)$ durchgeführt werden.

Reguläre Ausdrücke und endliche Automaten [2]

- Es folgt: Zu jedem RA gibt es einen äquivalenten NEA.
- Da jeder NEA in einen äquivalenten DEA überführt werden kann (Potenzautomatenkonstruktion), gilt auch: Jede reguläre Sprache wird von einem DEA akzeptiert. Zu jedem DEA A gibt es wiederum einen RA Φ mit $L(\Phi) = L(A)$. (Beweis wird hier ausgelassen)
- Es folgt: NEA, DEA und reguläre Ausdrücke sind Formalismen mit äquivalenter Beschreibungsstärke. Die Sprachen, die sich durch sie beschreiben lassen, heißen reguläre Sprachen.
- Die Zugehörigkeit von Worten zu regulären Sprachen kann in linearer Zeit getestet werden.

Einige Anwendungen von endlichen Automaten

- Morphologie:
 - Unter anderem: Flexionsmorphologie, Lemmatisierung/Stemming
- Suche in Textdokumenten
 - Z.B. Korpusuche in der Lexikografie mit regulären Ausdrücken
 - Suche und Informationszugriff in Archiven und Internet
- Syntax:
 - z.B. Erkennung von Wortartmustern in Phrasen/Satzteilen
 - Identifikation von Schlüsselwörtern /-phrasen in Dialogsystemen
- Dialogstruktur
 - Beschreibung von Dialogmustern mit Automaten