Lernalgorithmen

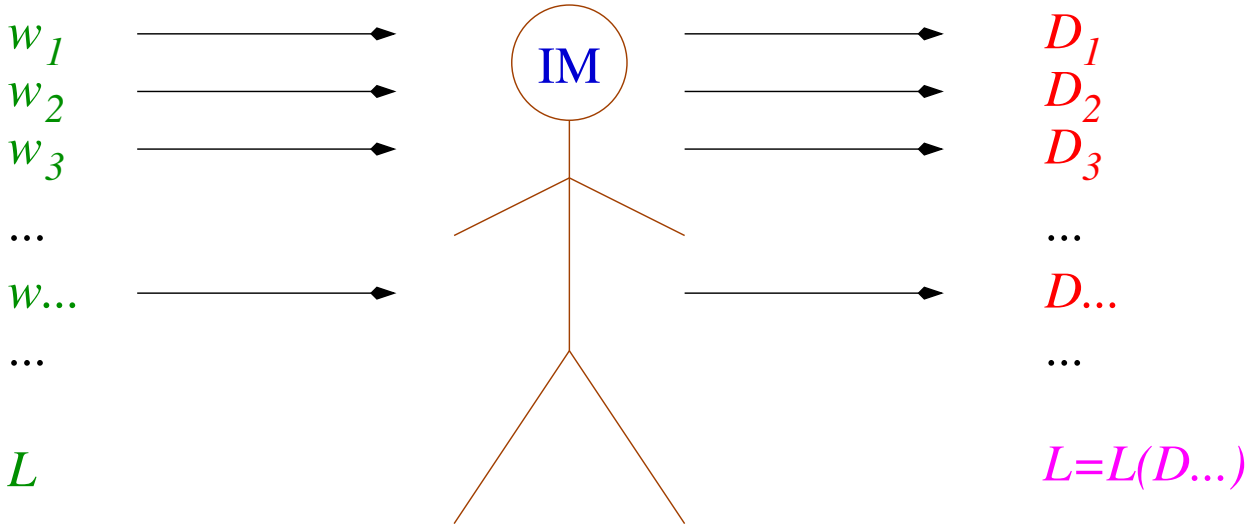
SoSe 2012 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

Lernalgorithmen Gesamtübersicht

0. Einführung
1. Identifikation (aus positiven Beispielen)
2. Zur Identifikation regulärer Sprachen, mit XML-Anwendung
3. HMM — Hidden Markov Models
4. Lernen mittels Anfragen & zur Roboterorientierung
5. Lernen mit negativen Beispielen
6. PAC-Lernen

Golds Lernmodell des Textlernens



Ziel der VL 3–5

- Vorstellung konkreter Lernalgorithmen für reguläre Sprachen
- Superfinitheitsproblem $\leadsto$ Teilklassen regulärer Sprachen
- Unsere Lerner sind insbesondere speicherbeschränkt, konsistent und konservativ
- Die Reihenfolge der Beispielpräsentation spielt keine Rolle
  $\leadsto$ Lerner erhält positive Beispielmenge I_+
- Hier wichtiges allgemeines Ergebnis: Satz von Angluin
  $\leadsto$ Definition geeigneter charakteristischer Mengen

Zur Identifikation regulärer Sprachen

Quellen

Übersichten

J. Gregor. Data-driven inductive inference of finite-state automata. *International Journal of Pattern Recognition and Artificial Intelligence* **8** (1994), pp. 305–322.

P. Dupont, L. Miclet. Inférence grammaticale régulière: Fondements théoriques et principaux algorithmes. INRIA Rennes. Rapport de recherche n 3449, Juillet 1998.

Spezialartikel

D. Angluin. Inference of reversible languages. *Journal of the ACM* **29** (1982), pp. 741–765.

H. Ahonen. Generating Grammars for Structured Documents Using Grammatical Inference Methods. University Helsinki, Department of Computer Science, Report A-1996-4.

P. García, E. Vidal, J. Oncina. Learning locally testable languages in the strict sense. *Proc. ALT'90*, pp. 325–338.

Reguläre Sprachen

lassen sich kennzeichnen durch:

- nichtdeterministische endliche Automaten (NEAs),
- deterministische endliche Automaten (DEAs),
- reguläre Ausdrücke sowie
- rechtslineare (oder auch linkslineare) Grammatiken.

Ein *nichtdeterministischer endlicher Automat* (siehe AFS / GTI 1)

kann spezifiziert werden als Quintupel $A = (Q, \Sigma, \delta, q_0, F)$, wobei

Q die endliche *Zustandsmenge* des Automaten A,
Σ das *Eingabealphabet* von A,
$\delta \subseteq Q \times \Sigma \times Q$ die *Übergangsrelation* von A,
q_0 der *Anfangszustand* von A und
F die *Endzustandsmenge* von A sind.

Die Erweiterung δ^* von δ auf ganze Eingabewörter ist gegeben als kleinste Teilmenge von $Q \times \Sigma^* \times Q$, die folgende zwei Punkte erfüllt:

- $\forall q \in Q : (q, \lambda, q) \in \delta^*$;

- $\forall q, q', q'' \in Q \forall w \in \Sigma^* \forall a \in \Sigma :$
$$((q, w, q') \in \delta^* \wedge (q', a, q'') \in \delta) \Rightarrow (q, wa, q'') \in \delta^*.$$

Die *von A erkannte Sprache* ist

$$L(A) = \{w \in \Sigma^* \mid \exists q_f \in Q_f : (q_0, w, q_f) \in \delta^*\}.$$

Ein *deterministischer endlicher Automat*

A erfüllt $\forall q \in Q \forall a \in \Sigma : |\{ q' \in Q \mid (q, a, q') \in \delta \}| \leq 1$.
Zu jeder regulären Sprache L existiert ein
deterministischer Minimalautomat $A(L)$, der bis auf Isomorphie
(das heißt hier: Zustandsumbenennung) eindeutig bestimmt ist:
$A(L) = (Q, \Sigma, \delta, q_0, F)$ lässt sich wie folgt beschreiben:
$Q = \{u^{-1}L | u \in Pref(L)\}$, wobei
$Pref(L)$ die Menge der Präfixe von L und
$u^{-1}L = \{v | uv \in L\}$ der Quotient von L durch u ist.
$q_0 = \lambda^{-1}L = L$;
$F = \{u^{-1}L | u \in L\}$;
$\delta(u^{-1}L, a) = (ua)^{-1}L$ mit $u, ua \in Pref(L)$, $a \in \Sigma$.

k-testbare Sprachen

Das Quadrupel $Z_k = (\Sigma, I_k, F_k, T_k)$ mit
$I_k, F_k \subseteq \Sigma^{<k}$ und $T_k \subseteq \Sigma^k$ definiert
die reguläre Sprache $L(Z_k) = ((I_k\Sigma^*) \cap (\Sigma^* F_k)) \setminus (\Sigma^* T_k \Sigma^*)$.
Solche Sprachen nennen wir k-*testbar* (k-TLSS).
Wir wollen zeigen, dass k-TLSS identifizierbar ist.
Zu I_+ definiere $Z_k(I_+) = (\Sigma(I_+), I_k(I_+), F_k(I_+), T_k(I_+))$ durch:

$$
\begin{aligned}
\Sigma(I_+) &: \text{alle in } I_+ \text{ vorkommenden Zeichen} \\
I_k(I_+) &= \left(\{u \mid \exists v \in (\Sigma(I_+))^* : uv \in I_+\Sigma^*\} \cap (\Sigma(I_+))^{k-1}\right) \cup \left(I_+ \cap (\Sigma(I_+))^{<k-1}\right) \\
F_k(I_+) &= \left(\{u \mid \exists v \in (\Sigma(I_+))^* : vu \in \Sigma^* I_+\} \cap (\Sigma(I_+))^{k-1}\right) \cup \left(I_+ \cap (\Sigma(I_+))^{<k-1}\right) \\
T_k(I_+) &= (\Sigma(I_+))^* \setminus \{v \in (\Sigma(I_+))^k \mid \exists u, w \in (\Sigma(I_+))^* : uvw \in I_+\}.
\end{aligned}
$$

k-TLSS ist identifizierbar

Satz: $L(Z_k(I_+))$ ist die kleinste k-TLSS, die I_+ umfasst.

Der **Beweis** des letzten Satzes ist ein einfacher Widerspruchsbeweis.

Folgerung (aus Satz von Angluin): k-TLSS ist identifizierbar, denn charakteristische Mengen sind leicht definierbar
$\rightsquigarrow$ Übungsaufgabe!

Eine formalsprachliche Anmerkung

Mitteilung: Das formalsprachliche Interesse an testbaren Sprachen beruht auf $\mathrm{REG} = \mathrm{H}(2 - \mathrm{TLSS})$.

Das bedeutet insbesondere:
Jede reguläre Sprache lässt sich als homomorphes Bild einer 2-testbaren Sprache darstellen.
Ist L regulär, gibt es mithin einen Homomorphismus h und eine 2-testbare Sprache T, sodass gilt:
$w \in L \iff \exists t \in T : w = h(t)$.
Ein *Homomorphismus* ist eine Abbildung $h : X^* \to V^*$ mit $h(\lambda) = \lambda$ und $h(xy) = h(x)h(y)$.
Es handelt sich also um eine strukturerhaltende Abbildung zwischen den Monoiden $(X^*, \cdot, \lambda)$ und $(V^*, \cdot, \lambda)$.

k-TLSS-Inferenzalgorithmus k-TLSSI:

Eingabe: $k \geq 2$, I_+; *Ausgabe*: DEA $A_k = (Q, \Sigma, \delta, q_0, F)$, $Q \subseteq \Sigma^{<k}$

1. Bestimme $Z_k(I_+) =: (\Sigma, I, F, T)$.

2. Setze $Q := \{\lambda\}$, $\delta := \emptyset$, $q_0 := \lambda$.

3. Für alle $a_1 \dots a_m \in I$ tue
   für $j = 1 \dots m$:
   $Q := Q \cup \{a_1 \dots a_j\}$;
   $\delta := \delta \cup \{(a_1 \dots a_{j-1}, a_j, a_1 \dots a_j)\}$

4. Für $a_1 \dots a_k \in \Sigma^k \setminus T$ tue
   $Q := Q \cup \{a_2 \dots a_k\}$;
   $\delta := \delta \cup \{(a_1 \dots a_{k-1}, a_k, a_2 \dots a_k)\}$;

Dies bestimmt $A_k = (Q, \Sigma, \delta, q_0, F)$.

Ein verlockender Fehler

In der erwähnten Originalarbeit sowie auch in früheren Ausgaben dieser Folien stand (leider) folgende Definition:

$$\begin{aligned} I_k(I_+) &= \left(\{u \mid \exists v \in (\Sigma(I_+))^* : uv \in I_+\} \cap (\Sigma(I_+))^{k-1}\right) \cup \left(I_+ \cap (\Sigma(I_+))^{<k-1}\right) \\ F_k(I_+) &= \left(\{u \mid \exists v \in (\Sigma(I_+))^* : vu \in I_+\} \cap (\Sigma(I_+))^{k-1}\right) \cup \left(I_+ \cap (\Sigma(I_+))^{<k-1}\right) \end{aligned}$$

Das ist im Allgemeinen falsch, wie das aufmerksame Publikum bemerkt hat:

Mit $k = 3$ und $I_+ = \{a, baaa, baaaa\}$ sollte die kleinste I_+ umfassende 3-testbare Sprache a^* enthalten, denn insbesondere alle Wörter, die mit a anfangen und aufhören und als Teilwörter der Länge drei von der Form aaa sind, sollten dabei sein.

Der wie vorher beschrieben erzeugte Automat würde aber die nicht 3-testbare Sprache $a \cup baa(a)^*$ akzeptieren.

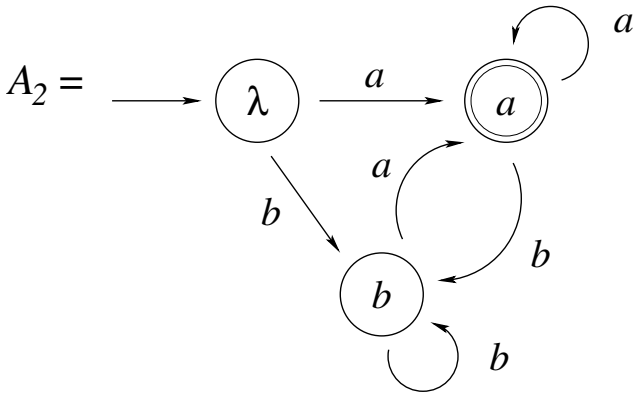
Ein Beispiel für k-TLSSI

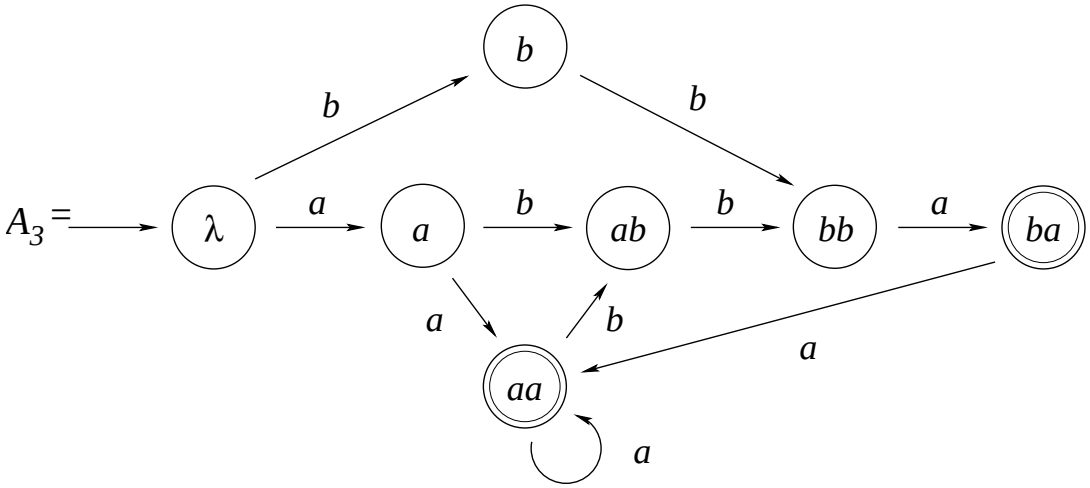
Beispiel: $I_+ = \{abba, aaabba, bbaaa, bba\}$.

$k = 2 \rightsquigarrow Z_2(I_+) = (\{a, b\}, \{a, b\}, \{a\}, \emptyset) \Rightarrow L(Z_2(I_+)) = \{a, b\}^* a$

$k = 3 \rightsquigarrow Z_3(I_+) = (\{a, b\}, \{aa, ab, bb\}, \{aa, ba\}, \{a, b\}^3 \setminus \{aaa, aab, abb, baa, bba\})$

Der Algorithmus liefert als Automaten:





Die Korrektheit von k-TLSSI

Satz: Es sei I_+ eine positive Beispielmenge und A_k sei gemäß k-TLSSI hergeleitet. Dann ist $L(Z_k(I_+)) = L(A_k)$.

BEWEIS.

(a) Für ein Wort $a_1 \dots a_m,\ m < k$ ist zu zeigen:
$a_1 \dots a_m \in I_k(I_+) \cap F_k(I_+)$ genau dann wenn $a_1 \dots a_m \in L(A_k)$.
"$\Rightarrow$": Aus $a_1 \dots a_m \in I_k(I_+)$ folgt nach Konstruktion von δ von A_k:
Für $j = 1, \dots, m$: $\delta(a_1 \dots a_{j-1}, a_j) = a_1 \dots a_j$.
Aus $a_1 \dots a_m \in F_k(I_+)$ folgt $a_1 \dots a_m \in F$ (der Endzustandsmenge von A_k).
Also liegt $a_1 \dots a_m \in L(A_k)$.

(b) Hausaufgabe: Zeige für ein Wort $a_1 \dots a_m,\ m \geq k$:
$a_1 \dots a_m \in L(Z_k(I_+)) \Leftrightarrow a_1 \dots a_m \in L(A_k)$. □

Partitionen / Zerlegungen

Eine *Partition* oder *Zerlegung* $\pi \subseteq 2^S \setminus \emptyset$ einer Menge S erfülle:
(1) $\forall A, B \in \pi : A \cap B = \emptyset$ und (2) $\bigcup_{A \in \pi} A = S$.

Die Elemente einer Zerlegung heißen auch *Blöcke*. Sie entsprechen den Äquivalenzklassen der von π induzierten Äquivalenzrelation.
Bezeichne, für $s \in S$, $B(s, \pi) \in \pi$ den Block von π, der s enthält.
Sind π, π' Partitionen von S, so ist π *feiner* als π', beziehungsweise π' *gröber* als π, in Zeichen $\pi \leq \pi'$, falls jeder Block aus π' eine Vereinigung von Blöcken aus π ist.
$\{\{x\}|x \in S\}$ ist also die feinste Partition und $\{S\}$ die gröbste.

Zerlegungen im formalsprachlichen Bereich

Es sei $L \subseteq \Sigma^*$ eine reguläre Sprache.
Definiere für $u, v \in \Sigma^*$: $u \equiv_L v \iff u^{-1}L = v^{-1}L$.
Dies liefert eine auch als *Rechtskongruenz von Nerode* bekannte Äquivalenzrelation auf Σ^*.
Die zugehörige Zerlegung wird auch mit π_L bezeichnet.
Zu jedem Automaten $A = (Q, \Sigma, \delta, q_0, F)$ mit $L(A) = L$ kann man eine (andere) Äquivalenzrelation auf Σ^* assoziieren vermöge:
$u \equiv_A v \iff \forall q \in Q : ((q_0, u, q) \in \delta^* \iff (q_0, v, q) \in \delta^*)$.
Offenbar gilt: $\pi_L = \pi_{A(L)}$, und für jeden Automaten A mit $L(A) = L$ gilt:
π_A ist eine Verfeinerung von π_L.

Quotientenautomat

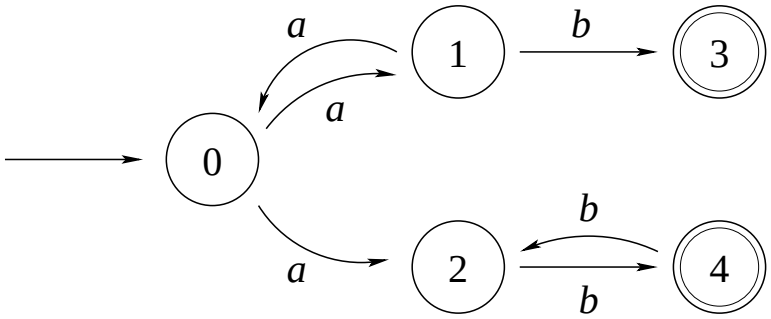
Es sei $A = (Q, \Sigma, \delta, q_0, F)$ ein endlicher Automat und π eine Partition von Q.
Der *Quotientenautomat* $\pi^{-1}A = (Q', \Sigma, \delta', B(q_0, \pi), F')$ ist wie folgt definiert:

$$
\begin{aligned}
Q' &= \pi^{-1}Q = \{B(q, \pi) | q \in Q\} \\
F' &= \{B \in Q' | B \cap F \neq \emptyset\} \\
\delta' &= \{(B, a, B') | B, B' \in Q', a \in \Sigma, \exists q, q' \in Q, q \in B, q' \in B' : (q, a, q') \in \delta\}
\end{aligned}
$$

Die Zustände, die zum selben Block einer Partition gehören, werden so verschmolzen.

Ein Beispiel (regulärer Ausdruck und NEA)

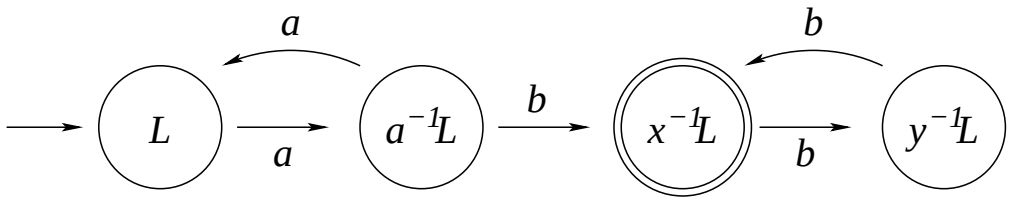
$(aa)^*(ab \cup ab(bb)^*)$ ist ein regulärer Ausdruck für L,
der sich durch den NEA



beschreiben lässt.

Ein Beispiel (deterministischer Minimalautomat)

Der zugehörige deterministische Minimalautomat ist



mit $x = ab$ und $y = abb$.
(Wir haben in diesem Bild auf den “Fehlerzustand” $\emptyset$ verzichtet.
Dies sei hinfort Konvention bei NEAs und DEAs.)

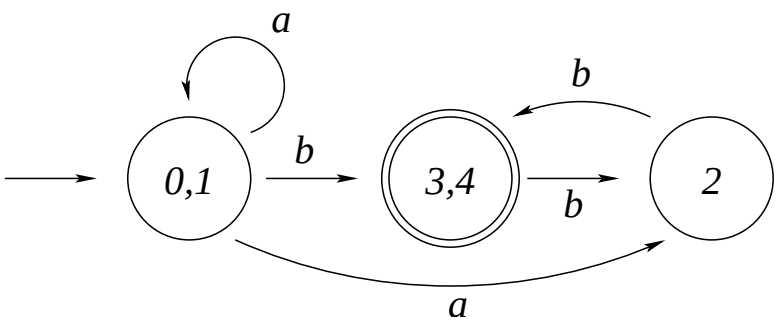
Ein Beispiel (Quotientenautomat)

Betrachten wir jetzt die Partition

$$\pi = \{\{0, 1\}, \{2\}, \{3, 4\}\}$$

der Zustände des obigen NEAs.

Der zugehörige Quotientenautomat ist:



Strukturelle Vollständigkeit

I_+: “Trainingsdatenmenge”, also endliche Teilmenge “Menge positiver Beispiele” der zu identifizierenden Sprache L. I_+ heiße *strukturell vollständig* oder *repräsentativ* bezüglich eines Automaten A, falls

- jede Transition beim Erkennen von A einmal “ausgeführt” wird (jede Transition t wird auf ein Wort aus I_+ abgebildet, bei dessen Akzeptanz t benutzt wird) und
- jeder Endzustand einmal erreicht wird.

Andere Sprechweise: A ist I_+-vollständig.
$\{\mathrm{aaab}, \mathrm{ab}, \mathrm{abbbbb}\}$ ist strukturell vollständig bezüglich jedem der drei Automaten aus obigem Beispiel.

Skelettautomat

Es sei $I_+ = \{u_1, \ldots, u_M\}$, $M = |I_+|$,
$u_i = a_{i,1} \cdots a_{i,|u_i|}$, $1 \leq i \leq M$.
Der *Skelettautomat* (maximal canonical automaton)
$MCA(I_+) = (Q_{MCA}, \Sigma, \delta, q_0, F)$ ist definiert durch:

Σ ist das Alphabet der Zeichen, die in I_+ vorkommen,
$Q_{MCA} = \{(i,j) | 1 \leq j \leq |u_i|, 1 \leq i \leq M\} \cup \{\lambda\}$;
$q_0 = \lambda$;
$F = \{(i, |u_i|) | 1 \leq i \leq M\}$;
$\delta = \{(\lambda, a_{i,1}, (i,1)) | 1 \leq i \leq M\}$
$\quad \cup \{((i,j), a_{i,j+1}, (i,j+1)) | 1 \leq i \leq M, 1 \leq j < |u_i|\}$.

$MCA(I_+)$ ist der I_+-vollständige Automat mit größter Zustandszahl, der keinen Zustand enthält, den man einfach herauslöschen kann, ohne die Sprache zu verändern.

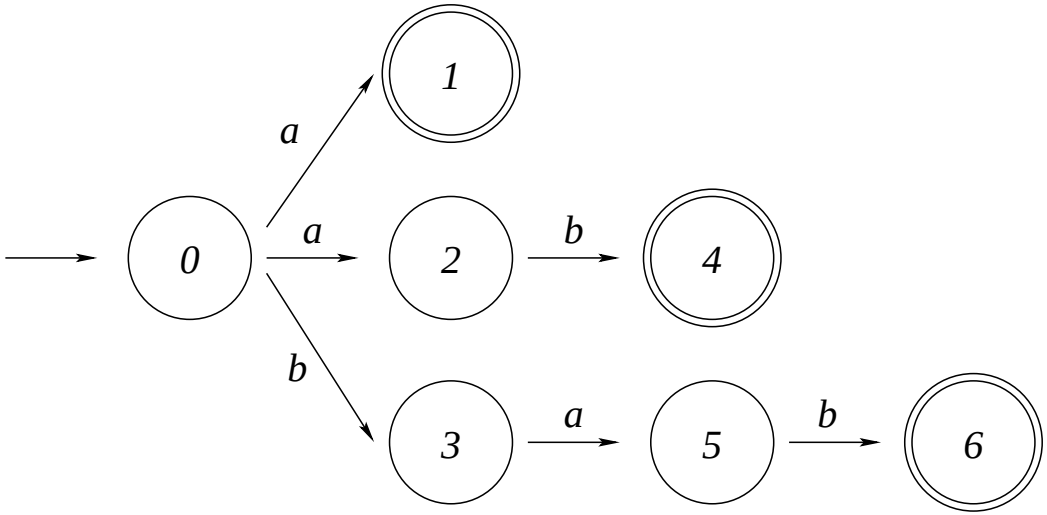
Der Präfixautomat

(prefix tree acceptor) $\mathrm{PTA}(\mathrm{I}_+)$ ist der Quotientenautomat
$\pi^{-1}\mathrm{MCA}(\mathrm{I}_+)$ mit $\mathrm{B}(q,\pi) = \mathrm{B}(q',\pi)$ genau dann, wenn
es einen Präfix $p \in \mathrm{Pref}(\mathrm{I}_+)$ gibt, der sowohl nach q als auch nach q' führt.

Mitteilung: $\mathrm{PTA}(\mathrm{I}_+)$ ist stets deterministisch.

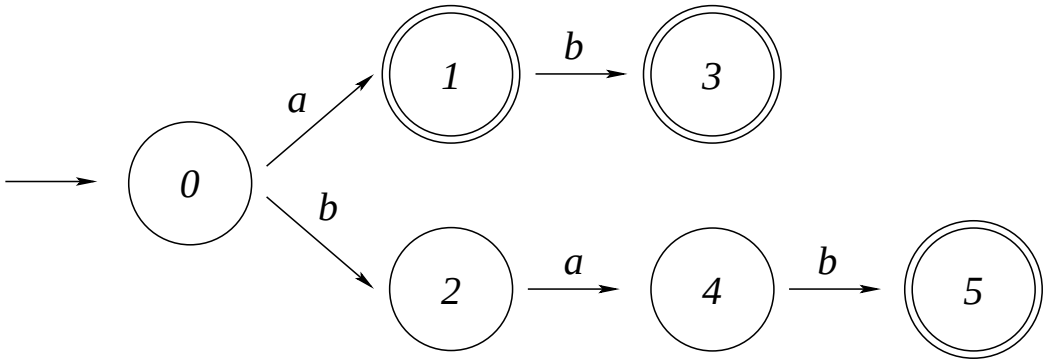
Ein Beispiel (Skelettautomat)

Beispiel: $I_+ = \{a, ab, bab\}$. $MCA(I_+)$:



Ein Beispiel (Präfixautomat)

Beispiel: $I_+ = \{a, ab, bab\}$. $PTA(I_+)$:



Eigenschaften von Automatenverfeinerungen:

1. Sind π_1, π_2 Partitionen der Zustandsmenge Q eines Automaten A mit $\pi_1 \leq \pi_2$, so gilt: $L(\pi_1^{-1}A) \subseteq L(\pi_2^{-1}A)$.
   Ist insbesondere π_1 die feinste Partition von Q, so ist $L(\pi_1^{-1}A) = L(A)$, und ist π_2 die gröbste Partition von Q, so ist $L(\pi_2^{-1}A) = \Sigma^*$, wobei Σ das Alphabet aller Zeichen ist, die in Wörtern von $L(A)$ vorkommen.

   Punkt 1 beweist man am saubersten per Induktion.

   Bem.: (a) π_1 ist eine *direkte Verfeinerung* von π_2 ist, falls

   $\pi_2 = \pi_1 \setminus \{B, B'\} \cup \{B \cup B'\}$ für Blöcke $B, B' \in \pi_1$ gilt.

   $\pi_2^{-1}A$ ergibt sich also aus $\pi_1^{-1}A$ durch Verschmelzung zweier Zustände.

   (b) π_1 verfeinert π_2, falls ein Kette direkter Verfeinerungen von π_1 nach π_2 führt.

   Der Induktionsanfang ist somit trivial (nämlich $\pi_1 = \pi_2$), und der Induktionsschritt eine einfache Übungsaufgabe.

2. Sei $I_+ \subseteq L$ eine endliche Menge positiver Beispiele für die reguläre Sprache $L \subseteq \Sigma^+$. Die Zerlegung π_L von Σ^* ist daher endlich.
Betrachte die Restriktion $\pi = \pi_L|_{Pref(I_+)} = \pi_L \cap (Pref(I_+) \times Pref(I_+))$.
π ist eine Partition der Zustände $Pref(I_+)$ von $PTA(I_+)$, und
$A = \pi^{-1}PTA(I_+)$ ist isomorph zu einem Teilakzeptor von $A(L)$.

BEWEIS. Die Aussage gilt trivial für $I_+ = \emptyset$.
Für $I_+ \neq \emptyset$ ist die Partition π definiert durch $B(w_1, \pi) = B(w_2, \pi)$ genau dann, wenn ${w_1}^{-1}L = {w_2}^{-1}L$ für alle $w_1, w_2 \in Pref(I_+)$.
Die Abbildung h von Zuständen in A in Zustände in $A(L)$ ist durch $h(B(w_1, \pi)) = {w_1}^{-1}L$ wohldefiniert und injektiv.
Der Anfangszustand $B(\lambda, \pi)$ von A wird auf $\lambda^{-1}L$ abgebildet.
Ist B_1 Endzustand, so ist $B_1 = B(w, \pi)$ für ein $w \in I_+$, und da $I_+ \subseteq L$, ist $h(B)$ Endzustand.
Ist (B_1, a, B_2) ein Zustandsübergang in A, so gibt es ein $w \in Pref(I_+)$ mit $B_1 = B(w, \pi)$ und $B_2 = (wa, \pi)$, $wa \in Pref(I_+)$. Daher ist $(h(B_1), a, h(B_2))$ ein Zustandsübergang in $A(L)$. $\square$

Folgerung: (aus 1.) Identifikationsalgorithmen, die auf Zustandsverschmelzungen basieren und von $MCA(\mathrm{I}_+)$ oder $\mathrm{PTA}(\mathrm{I}_+)$ ausgehen, sind konsistent.

BEWEIS. $\mathrm{I}_+ = \mathrm{L}(MCA(\mathrm{I}_+)) = \mathrm{L}(\mathrm{PTA}(\mathrm{I}_+))$. □

Folgerung: (aus 2.) Identifikationsalgorithmen, die stets Verfeinerungen von $\pi_{\mathrm{L}}|_{\mathrm{Pref}(\mathrm{I}_+)}{}^{-1}\mathrm{PTA}(\mathrm{I}_+)$ ergeben, liefern Teilsprachen von L.

Induktion als Suchaufgabe I

Suchräume: $\mathrm{LAT}(\mathrm{MCA}(\mathrm{I}_+))$ und $\mathrm{LAT}(\mathrm{PTA}(\mathrm{I}_+))$ (LAT: Lattice (engl. Verband))
"Dahinterliegend" ist der durch Inklusion geordnete Sprachenverband $\mathcal{L}(\mathrm{LAT}(\mathrm{MCA}(\mathrm{I}_+)))$
Wegen Eigenschaft 1 ist $\mathrm{L} : A \mapsto \mathrm{L}(A)$ ein Verbandsmorphismus.

Satz: Es sei A irgendein I_+-vollständiger Automat. $\rightsquigarrow \mathrm{L}(A) \in \mathcal{L}(\mathrm{LAT}(\mathrm{MCA}(\mathrm{I}_+)))$.

BEWEIS. Wir konstruieren eine Partition π der Zustände von $\mathrm{MCA}(\mathrm{I}_+)$, sodass
$\pi^{-1}\mathrm{MCA}(\mathrm{I}_+)$ $\mathrm{L}(A)$, $A = (Q, \Sigma, \delta, q_0, F)$, akzeptiert.
Betrachte also $\mathrm{I}_+ = \{u_1, \ldots, u_M\}$, $u_i = a_{i,1} \cdots a_{i,|u_i|}$
und den zugehörigen Skelettautomaten $\mathrm{MCA}(\mathrm{I}_+)$.
Die Abb. $\varphi : Q_{MCA} \to 2^Q$, $(i, j) \mapsto \{q \in Q | (q_0, a_{i1} \ldots a_{ij}, q) \in \delta^*\}$
definiert durch $B(\nu, \pi) = B(\nu', \pi) \Leftrightarrow \varphi(\nu) = \varphi(\nu')$ ein Partition.
$\mathrm{L}(A) \subseteq \mathrm{L}(\pi^{-1}\mathrm{MCA}(\mathrm{I}_+))$ ist klar, und die I_+-Vollständigkeit von A liefert die andere Inklusion. $\rightsquigarrow$
eine hübsche Übungsaufgabe! □

Induktion als Suchaufgabe II

Umgekehrt gilt:
Satz: Jeder Automat in $\mathrm{LAT}(\mathcal{MCA}(\mathrm{I}_+))$ ist I_+-vollständig.

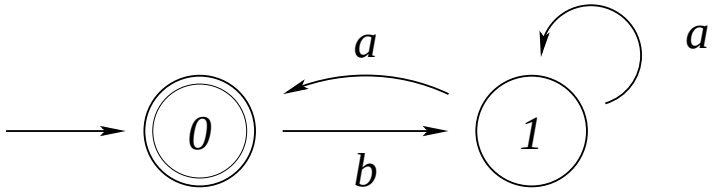
BEWEIS. Übungsaufgabe! □

Satz: Ist $\mathcal{A}(\mathrm{L})$ I_+-vollständig, so gehört $\mathcal{A}(\mathrm{L})$ zu $\mathrm{LAT}(\mathrm{PTA}(\mathrm{I}_+))$.

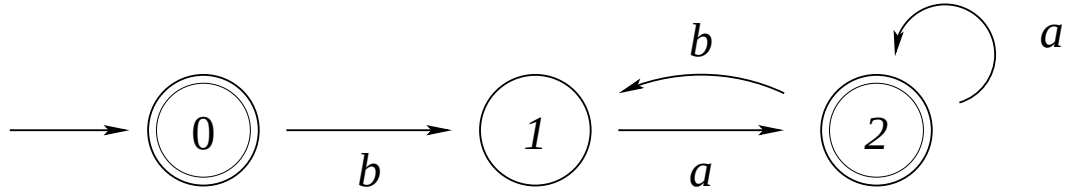
BEWEIS. Ergibt sich sofort aus Eigenschaft 2. □

Bemerkung: In der Praxis Suchraumeinschränkung durch gewisse Heuristiken (Verschmelzungsregeln)!

Induktion als Suchaufgabe: Beispiele

$A =$ 

ist $\{ba, baa\}$-vollständig, also in $LAT(MCA(\{ba, baa\}))$ enthalten, aber

A(L(A))= 

ist nicht $\{ba, baa\}$-vollständig $\rightsquigarrow LAT(PTA(\{ba, baa\})) \subsetneq LAT(MCA(\{ba, baa\}))$.
Durch Analyse des Verbandes $LAT(PTA(\{ba, baa\}))$ kann man zeigen, dass $L(A) = \langle (ba^*a)^* \rangle$ nicht von irgendeinem Automaten des Verbandes erzeugt wird $\rightsquigarrow$ Übungsaufgabe!

k-reversible Sprachen k-REV

Eine reguläre Sprache heißt k-*reversibel* genau dann, wenn

$$\forall u_1 v w, u_2 v w \in L,\ |v| = k \Rightarrow (u_1 v)^{-1} L = (u_2 v)^{-1} L.$$

Hinweis: Für $u_1, u_2, v \in \Sigma^*$ und $L \subseteq \Sigma^*$ gilt:

$(u_1 v)^{-1} L = (u_2 v)^{-1} L$ gdw. $\forall z \in \Sigma^* : u_1 v z \in L \iff u_2 v z \in L$.

Zusammenhang TLSS und REV

Satz: Ist L $(k+1)$-TLSS, so ist L k-reversibel.

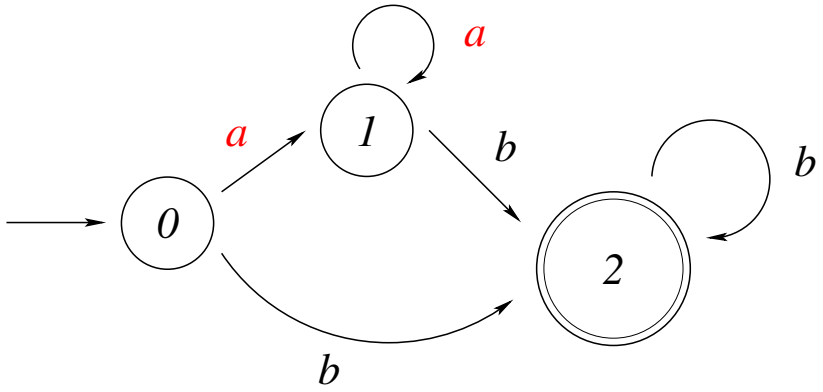
BEWEIS. Es sei L $(k+1)$-TLSS. Damit ist L regulär.
L sei beschrieben durch (Σ, I, F, T).
Betrachte $u_1vw, u_2vw \in L$, $|v| = k$.
Der Präfix der Länge k von u_ivw ist gleich dem Präfix der Länge k von u_iv und liegt in I für $i = 1, 2$.
Entsprechend liegt der Suffix der Länge k von vw in F.
Zudem ist keines der Teilworte der Länge $(k+1)$ von u_ivw verboten, das heißt, alle Teilworte der Länge $(k+1)$ von 1. u_1v, 2. u_2v und 3. vw sind zulässig.
Ist nun für irgendein $z \in \Sigma^*$ $u_1vz \in L$, so ist dies ($u_1vw \in L$ und $u_2vw \in L$ vorausgesetzt) gleichwertig damit, dass
(a) der Suffix der Länge k von vz in F liegt und
(b) alle Teilworte der Länge $(k+1)$ von vz zulässig sind,
was wiederum gleichwertig damit ist, dass $u_2vz \in L$ gilt. □

Ein paar weitere automatentheoretische Begriffe:

Ist $A = (Q, \Sigma, \delta, q_0, F)$ EA, so ist $A^r = (Q, \Sigma, \delta^r, F, q_0)$ mit
$(q, a, q') \in \delta$ gdw. $(q', a, q) \in \delta^r$
der zugehörige *Spiegelautomat*, der die Spiegelsprache
$L(A^r) = (L(A))^r = \{w^r | w \in L(A)\}$ (mit $\lambda^r = \lambda$, $(av)^r = v^r a$) akzeptiert.

$u \in \Sigma^k$ heißt k-*Nachfolger* von $q \in Q$ gdw. $\delta^*(q, u) \neq \emptyset$.
Ein EA $A = (Q, \Sigma, \delta, I, F)$ [Anfangszustandsmenge I !] heiße *deterministisch mit k-Vorschau* genau dann, wenn für alle verschiedenen $q_1, q_2 \in Q$, für die gilt
(a) $q_1, q_2 \in I$ oder
(b) $\exists q_3 \in Q,\ a \in \Sigma : q_1, q_2 \in \delta(q_3, a)$,
der Schnitt der Mengen der k-Nachfolger von q_1 und q_2 leer ist.
(k-Vorschau klärt möglichen Nichtdeterminismus!)
Ein Automat A heißt k-*reversibel* genau dann, wenn A deterministisch und A^r deterministisch mit k-Vorschau ist.

Ein Beispiel für einen 1-reversiblen Automaten



Machen Sie sich klar: Insbesondere die eingezeichneten beiden roten Kantenbeschriftungen stellen keinen Widerspruch zum geforderten Rückwärtsdeterminismus mit “1-Rückschau” dar. Allerdings ist der Automat daher nicht 0-reversibel.

Charakterisierungssatz für k-REV

Satz: Die folgenden Aussagen sind für reguläre Sprachen L äquivalent:

(i) L ist k-reversibel.

(ii) $A(\mathrm{L})$ ist k-reversibel.

(iii) $\exists \mathrm{EA}\ A : \mathrm{L} = \mathrm{L}(A) \wedge A$ ist k-reversibel.

BEWEIS. Die Implikation (ii) nach (iii) ist trivial.

(iii)$\rightarrow$(ii): Es sei $A = (Q, \Sigma, \delta, q_0, F)$ ein DEA, der L akzeptiert. Die durch die Blöcke

$$B(w_1, \pi_A) = B(w_2, \pi_A) \iff \delta^*(q_0, w_1) = \delta^*(q_0, w_2)$$

definierte Partition π_A verfeinert $\pi_L (= \pi_{A(L)})$ (Hinweis: Zustandsminimierungsalgorithmus).

Wir müssen zeigen: $(A(L))^r$ ist deterministisch mit k-Vorschau.

Betrachte zwei Zustände $w_1^{-1}L$ und $w_2^{-1}L$ in $(A(L))^r$.

Annahme: v wäre k-Nachfolger von $w_1^{-1}L$ und $w_2^{-1}L$ in $(A(L))^r$.

Dann gibt es u_1, u_2 mit $(u_i v^r)^{-1}L = w_i^{-1}L$, $i = 1, 2$.

Sind, in Fall (a), $w_1^{-1}L$ und $w_2^{-1}L$ Anfangszustände in $(A(L))^r$ oder gibt es, in Fall (b), für ein $a \in \Sigma$ einen Zustand $w_3^{-1}L$ von $A(L)$ mit $(w_1 a)^{-1}L = w_3^{-1}L = (w_2 a)^{-1}L$, dann ex. ein Wort w, sodass $A(L)$ sowohl $u_1 v^r w$ als auch $u_2 v^r w$ akzeptiert.

Daher akzeptiert auch A beide diese Wörter.

Da A k-reversibel, gibt es einen Zustand $\delta^*(q_0, u_1 v^r) = \delta^*(q_0, u_2 v^r)$ in A, d.h., $u_1 v^r$ und $u_2 v^r$ gehören zum selben Block von π_A und mithin von $\pi_{A(L)}$, d.h., $(u_1 v^r)^{-1}L = (u_2 v^r)^{-1}L$, also $w_1^{-1}L = w_2^{-1}L$. Widerspruch! Also ist $A(L)$ k-reversibel.

(i)$\rightarrow$(ii): Da $A(L)$ deterministisch ist, hat man nur nachzurechnen, dass $(A(L))^r$ deterministisch mit k-Vorschau ist.

Betrachte (für Punkt (a)) zwei Endzustände von $A(L)$, nämlich $(u_1v)^{-1}L$ und $(u_2v)^{-1}L$ mit $|v| = k$. D.h., $u_1v, u_2v \in L$.

Da L k-reversibel, heißt dies $(u_1v)^{-1}L = (u_2v)^{-1}L$.

Daher kann es keine zwei verschiedenen Endzustände von $A(L)$ mit gleichem "k-Vorgänger" geben.

In (b) ist $q_1 \neq q_2$ zu betrachten mit $q_3 = \delta(q_1, a) = \delta(q_2, a)$. Dann sind sozusagen die k-Vorgänger von q_1 und q_2 zu diskutieren. Wir nehmen in einem Widerspruchsargument an, v sei sowohl k-Vorgänger von q_1 als auch von q_2. Also ist $q_i = (u_iv)^{-1}L$ für ein geeignetes u_i, $i = 1, 2$. Ferner sei w' so, daß q_3 bei Eingabe von w' in einen Endzustand führt. D.h., $u_ivaw' \in L$ für $i = 1, 2$. Aus der k-Reversibilität von L folgt nun sofort $(u_1v)^{-1}L = (u_2v)^{-1}L$, m.a.W., $q_1 = q_2$.

(ii)$\rightarrow$(i) Betrachte $u_1vw, u_2vw \in L$, $v \in \Sigma^k$. Wäre $(u_1v)^{-1}L \neq (u_2v)^{-1}L$, so wäre v k-Vorgänger jener beiden als verschieden angenommenen Zustände von $A(L)$. $\square$

Mitteilung: Für 0-Reversibilität gilt sogar:
Jeder 0-reversible Automat, bei dem kein Zustand “nutzlos” ist (d.h., jeder Zustand ist vom Anfangszustand aus erreichbar und führt irgendwie in einen Endzustand), ist minimal.
Darüber hinaus sieht man leicht, dass jeder 0-reversible Automat nur (höchstens) einen Endzustand besitzt.

Charakteristische Teilmengen für k-REV

Satz: Jede k-reversible Sprache hat eine charakteristische Teilmenge.

BEWEIS. Wir zeigen hier nur den Fall $k = 0$, der allgemeinere Fall folgt analog.
Ist $L = \emptyset$, so ist $I_+ = \emptyset$ eine charakteristische Teilmenge für L.
Sei nun $L \neq \emptyset$. Betrachte den (gemäß dem Charakterisierungssatz) 0-reversiblen Automaten $A(L) = (Q, \Sigma, q_0, \{q_f\}, \delta)$.
Für $q \in Q$ bezeichne $u(q)$ und $v(q)$ Wörter minimaler Länge mit $\delta^*(q_0, u(q)) = q$ und $\delta^*(q, v(q)) = q_f$. Definiere

$$\begin{aligned} \chi(L) &= \{u(q)v(q) \mid q \in Q\} \\ &\cup \{u(q)av(\delta(q,a)) \mid q \in Q, a \in \Sigma\} \end{aligned}$$

Wir zeigen: $\chi(L)$ ist charakteristische Teilmenge von L.

Betrachte eine beliebige 0-reversible, $\chi(L)$ umfassende Sprache L'. Wir werden beweisen:
(*) $w^{-1}L' = (u(q))^{-1}L'$ mit $q = \delta^*(q_0, w)$ für alle $w \in Pref(L)$.
Speziell für $w \in L$ folgt dann aus (*): $w^{-1}L' = (u(q_f))^{-1}L'$.
Wegen $u(q_f) \in \chi(L) \subseteq L'$ ist $(u(q_f))^{-1}L'$ akzeptierender Zustand des Minimalautomaten von L', d.h., $w \in L'$.
Also gilt: $L \subseteq L'$, d.h.,
L ist die kleinste $\chi(L)$ umfassende 0-reversible Sprache.

Wir beweisen (*) per Induktion über die Länge des Präfixes w:
Wegen $u(q_0) = \lambda$ gilt (*) für $|w| = 0$, d.i., $w = \lambda$.
Wir nehmen an, (*) gelte für alle $w \in \Sigma^{\leq n}$, $n \geq 0$.
Diskutiere den Fall $wa \in \Sigma^{n+1}$, $w \in \Sigma^n$, $a \in \Sigma$, $wa \in Pref(L)$.
Da $w \in Pref(L)$, gilt nach Induktionsannahme:
$w^{-1}L' = (u(q))^{-1}L'$ mit $q = \delta^*(q_0, w)$.
Also ist $(wa)^{-1}L' = (u(q)a)^{-1}L'$.
Für $q' = \delta(q, a) = \delta^*(q_0, wa)$ sind sowohl $u(q')v(q')$ als auch $u(q)av(q')$ in $\chi(L) \subseteq L'$. Nach dem Charakterisierungssatz ist L' 0-reversibel, d.h. es folgt $(u(q'))^{-1}L' = (u(q)a)^{-1}L'$.
Zusammen liefert dies $(wa)^{-1}L' = (u(\delta^*(q_0, wa)))^{-1}L'$.

Für $k > 0$ geben wir nur die charakteristische Teilmenge an:

$$\begin{aligned}\chi(L) &= (\Sigma^{<k} \cap L) \\ &\cup \{u(q,x)xv(q) \mid q \in Q, x \in L_q\} \\ &\cup \{u(q,x)xav(\delta(q,a)) \mid q \in Q, x \in L_q, a \in \Sigma\}\end{aligned}$$

Dabei wird wieder von einem Minimalautomaten mit Überführungsfunktion δ ausgegangen.

L_q bezeichnet die Sprache der k-Vorgänger von q in $A(L)$.

Für $x \in L_q$ sei $u(q,x)$ ein Wort minimaler Länge mit

$\delta^*(q_0, u(q,x)x) = q$.

$v(q)$ sei ein Wort minimaler Länge mit $\delta^*(q, v(q)) \in F$. □

Die angegebenen charakteristischen Teilmengen sind strukturell vollständig für $A(L)$.

Algorithmus k-RI zur Inferenz von k-REV (informell)

Ausgehend vom Präfixakzeptor $\mathrm{PTA}(\mathrm{I}_+)$ verschmilzt der Algorithmus k-RI solange Zustände, bis eine Partition π der Zustände von $\mathrm{PTA}(\mathrm{I}_+)$ gefunden wurde derart, dass $\pi^{-1}\mathrm{PTA}(\mathrm{I}_+)$ k-reversibel ist.
Genauer sei π eingangs die feinste Zustandspartition von $\mathrm{PTA}(\mathrm{I}_+)$,
d.i., $\pi^{-1}\mathrm{PTA}(\mathrm{I}_+) = \mathrm{PTA}(\mathrm{I}_+)$.
Danach bestimmt man rekursiv im Quotientenautomaten $\pi^{-1}\mathrm{PTA}(\mathrm{I}_+)$ zwei Zustände, d.h., zwei verschiedene Blöcke B_1 und B_2, die wenigstens eine der folgenden Verschmelzungsbedingungen erfüllen.
Sobald zwei solche Blöcke gefunden wurden, wird ein neuer Automat durch Verschmelzen dieser beiden Blöcke gebildet.
Die Reihenfolge der Anwendung dieser Verschmelzungsregeln ist beliebig.

Verschmelzungsbedingungen des Algorithmus k-RI

1. $\exists a \in \Sigma \exists B \in \pi : (B, a, B_1) \in \delta \wedge (B, a, B_2) \in \delta$ oder

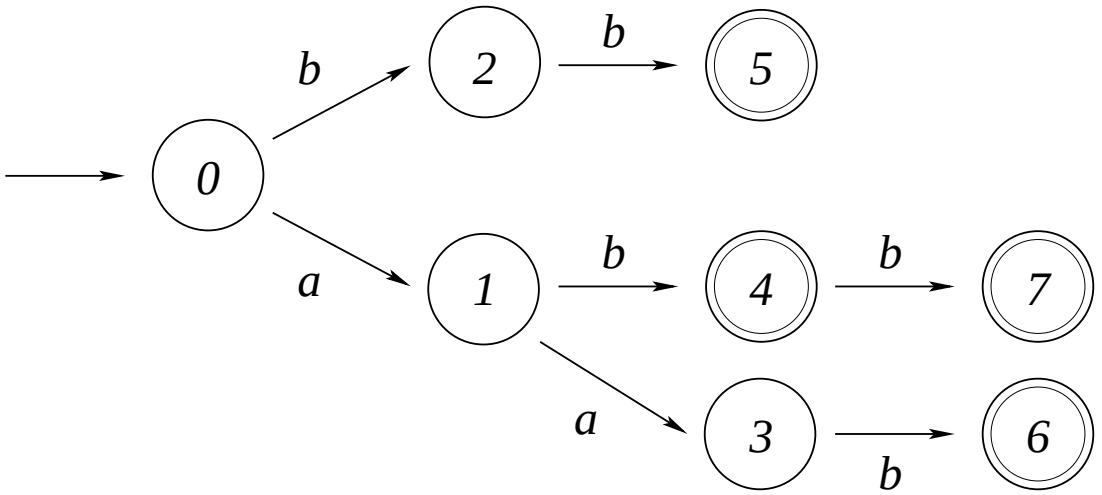
2. $\exists v, u_1, u_2 \in \Sigma^* : |v| = k \wedge (q_0, u_1 v, B_1) \in \delta^* \wedge (q_0, u_2 v, B_2) \in \delta^*$ sowie entweder

   (a) B_1 und B_2 sind akzeptierende Zustände in $\pi^{-1}\mathrm{PTA}(I_+)$ oder

   (b) $\exists a \in \Sigma \exists B \in \pi : (B_1, a, B) \in \delta \wedge (B_2, a, B) \in \delta$.

Algorithmus k-RI zur Inferenz von k-REV für $k = 1$ (Bsp. I)

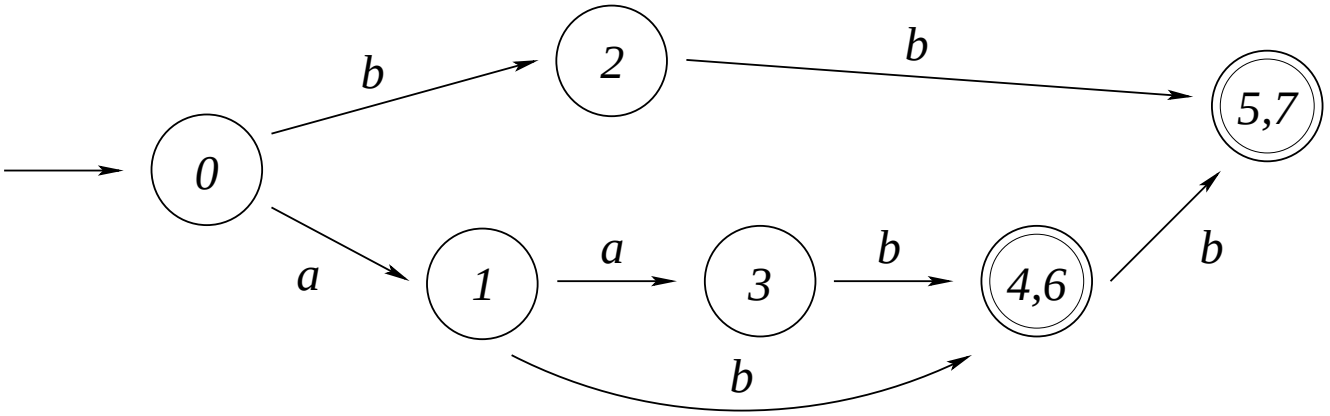
$\mathrm{PTA}(I_+)$ mit $I_+ = \{ab, bb, aab, abb\}$:



Bedingung 2a. ist zweimal verletzt:
b führt von 5 bzw. 7 aus rückwärts in zwei verschiedene Zustände;
b führt im Spiegelautomaten von 4 bzw. 6 aus in zwei verschiedene Zustände

Algorithmus k-RI zur Inferenz von k-REV (Bsp. II)

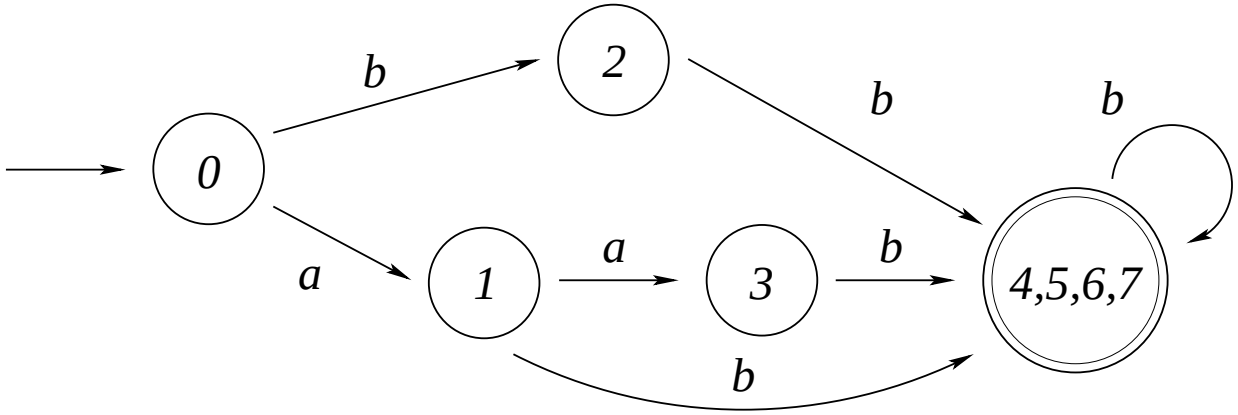
$\pi^{-1}\mathrm{PTA}(I_+)$ mit $\pi = \{\{0\},\{1\},\{2\},\{3\},\{4,6\},\{5,7\}\}$:



(Bedingung 2a.: wiederum “b-1-Rückschau”, angewendet auf $\{4,6\}$ und $\{5,7\}$)

Algorithmus k-RI zur Inferenz von k-REV (Bsp. III)

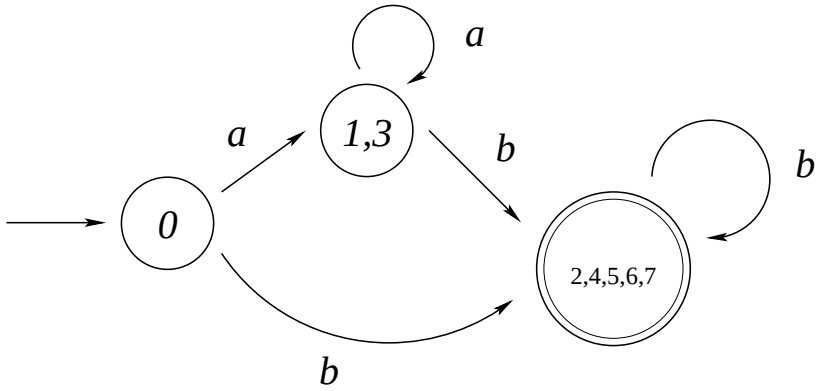
$\pi^{-1}\mathrm{PTA}(I_+)$ mit $\pi = \{\{0\},\{1\},\{2\},\{3\},\{4,5,6,7\}\}$:



Bedingung 2b. ist verletzt; das bedeutet intuitiv:
Im Spiegelautomaten kann man im Zustand $\{4,5,6,7\}$ bei Eingabe von ba nicht entscheiden, ob man im Zustand 0 oder im Zustand 1 landet, weshalb die beiden durch b erreichten Zustände zu $\{1,3\}$ verschmolzen werden.

Algorithmus k-RI zur Inferenz von k-REV (Bsp. IV)

$\pi^{-1}\mathrm{PTA}(\mathrm{I}_+)$ mit $\pi = \{\{0\},\{1,3\},\{2,4,5,6,7\}\}$:



Dieser Automat ist 1-reversibel.

Übung: Was hätte der Algorithmus 2-RI bei derselben Eingabe geliefert ?

k-reversibel-Inferenzalgorithmus k-RI

Eingabe: $k \geq 1$, I_+;
Ausgabe: DEA $A_k = (Q, \Sigma, \delta, q_0, F)$

1. Setze $A := \mathrm{PTA}(I_+)$ (mit N Zuständen);

2. Setze $\pi := \{\{0\}, \{1\}, \ldots, \{N-1\}\}$;

3. Solange $((B_1, B_2) :=$ nicht_reversibel$(\pi^{-1}A, \pi, k)) \neq (\emptyset, \emptyset)$ tue:
   $\pi := \pi \setminus \{B_1, B_2\} \cup \{B_1 \cup B_2\}$

4. Setze $A_k := A(L(A))$

Bemerkungen zu k-RI

Die Funktion nicht_reversibel liefert ein Paar von Blöcken $(\mathrm{B}_1, \mathrm{B}_2)$ zurück, falls B_1 und B_2 einer der Verschmelzungsbedingungen genügen, und $(\emptyset, \emptyset)$ wird geliefert, falls kein solches Blockpaar existiert.

Beweis der Korrektheit des Algorithmus $\rightsquigarrow$ siehe unten.

Mitteilung: k-RI liefert stets einen deterministischen Minimalautomaten, der die kleinste I_+ umfassende k-reversible Sprache erzeugt.

Allgemeines Schema für Korrektheitsbeweise von Zustandsverschmelzungsalgorithmen

für eine reguläre Sprachklasse $\mathcal{L}$, definiert durch eine Automatenklasse $\mathcal{A}$:

1. "Teilautomaten" von Automaten aus $\mathcal{A}$ liegen in $\mathcal{A}$.

2. Definiere *geeignete kanonische Objekte* aus $\mathcal{A}$, die $\mathcal{L}$ kennzeichnen.

3. Konstruiere *charakteristische Teilmengen*.

4. Stelle den naheliegenden *Verschmelzungsalgorithmus* auf, ausgehend von $\mathrm{PTA}(\mathrm{I}_+)$ (bzw. $\mathrm{MCA}(\mathrm{I}_+)$).

5. Der Algorithmus liefert eine (nicht eindeutig bestimmte) *Automatenkette* $A_0, \ldots, A_f$. Dabei ist $A_0 = \mathrm{PTA}(I_+)$ und $A_i = \pi_i^{-1} A_0$ für eine geeignete Zustandspartition π_i von A_0. Da $\pi_i \leq \pi_{i+1}$ (Zustandsverschmelzung!) gilt:

$$L(A_0) \subseteq L(A_1) \subseteq \cdots \subseteq L(A_f).$$

6. Für π_f mit $A_f = \pi_f^{-1} A_0$ gilt: π_f ist die feinste Partition von Zuständen von A_0, sodass $\pi_f^{-1} A_0$ in $\mathcal{A}$ liegt.

7. $L(A_f)$ ist die kleinste I_+ umfassende Sprache in $\mathcal{L}$.

8. Der Verschmelzungsalgorithmus aus Punkt 3. konvergiert bei Aufzählung einer Sprache aus $\mathcal{L}$ und liefert stets ein korrektes Ergebnis.

Satz: $L(A_f)$ ist die kleinste I_+ umfassende Sprache in $\mathcal{L}$.

BEWEIS. Jedenfalls ist $L(A_f) \in \mathcal{L}$ und $I_+ = L(A_0) \subseteq L(A_f)$ wegen Punkt 5.
Betrachte beliebige Sprache $L \in \mathcal{L}$ mit $I_+ \subseteq L$. Definiere $\pi = \pi_L|_{Pref(I_+)}$.
Nach der zweiten Eigenschaft von Automatenverfeinerungen ist $\pi^{-1}A_0$ isomorph zu einem Teilautomaten von $A(L)$.
Wegen Punkt 2. gilt $A(L) \in \mathcal{A}$.
Wegen Punkt 1. ist daher $\pi^{-1}A_0 \in \mathcal{A}$.
Wegen Punkt 6. gilt $\pi_f \leq \pi$.
Die erste Eigenschaft von Automatenverfeinerungen liefert:
$L(A_f) = L(\pi_f^{-1}A_0) \subseteq L(\pi^{-1}A_0) \subseteq L(A(L)) = L$. □