
Formale Sprachen und Automaten

Reguläre Ausdrücke und Endliche Automaten

Avant Propos

Sie [die Theorie der formalen Sprachen] ist ein Musterbeispiel einer informatischen Theorie, weil es ihr gelingt, einen großen Bestand an Einsichten und Zusammenhängen aus wenigen Grundbegriffen und Schlussweisen heraus methodisch zu entwickeln und systematisch darzustellen.”

(Loos 1993, 1)

“Sie [die Sprache] muß daher von endlichen Mitteln einen unendlichen Gebrauch machen [...]”

(Humboldt 1836, 122)

Alphabet und Wort

- Ein **Alphabet** ist eine endliche Menge von **Symbolen**.

Römisches Alphabet = $\{a, b, c, \dots, z\}$

Griechisches Alphabet = $\{\alpha, \beta, \gamma, \dots, \omega\}$

Binäres Alphabet = $\{0, 1\}$

- Ein **Wort** ist eine endliche Folge von Symbolen über einem Alphabet.

(o, d, e, s, s, a) ist ein Wort über $\{a, b, c, \dots, z\}$

$(0, 1, 0, 1)$ ist ein Wort über $\{0, 1\}$

- Konventionen:

1. Statt (o, d, e, s, s, a) schreiben wir *odessa*.
2. w, x, y, z sind Variablen für Ketten.

- Das **leere Wort** ϵ enthält kein Symbol.
- Σ^* ist die Menge aller Wörter, ϵ inklusive, über Σ .

Operationen auf Wörtern 1

- Die **Länge** eines Wortes w , $|w|$, ist die Zahl der Symbole, die w enthält.

$|odessa| = 6$

$|0110101| = 7$

$|\epsilon| = 0$

- Die **Verkettung** zweier Wörter x und y , geschrieben als $x \circ y$ oder xy , ist die Kette x gefolgt von der Kette y . Es gilt:

1. $w = x \circ y \Rightarrow |w| = |x| + |y|$

2. $((x \circ y) \circ z) = (x \circ (y \circ z))$

3. $x \circ \epsilon = \epsilon \circ x$

- v ist

1. ein **Teilwort** von w , gdw. es x, y gibt, so dass $w = xvy$.
2. ein **Präfix** von w , gdw. es y gibt, so dass $w = vy$.
3. ein **Suffix** von w , gdw. es x gibt, so dass $w = xv$.

Operationen auf Wörtern 2

- **Rekursive Verkettung:** das Wort w^i ist definiert als
 1. $w^0 = \epsilon$
 2. $w^{i+1} = w^i \circ w$ für $i \geq 0$
- Die **Umkehrung** eines Wortes w , w^R , ist w rückwärts geschrieben.
 $neger^R = regen$
- Formale Definition der Umkehrung:
 1. $w^R = w$, wenn $|w| = 0$
 2. $w^R = au^R$, für ein $a \in \Sigma$, wenn $|w| \geq 1$ und $w = ua$.
- Für beliebige Wörter w, x ist $(wx)^R = x^R w^R$ (Beweis durch Induktion über die Länge von x).

Sprachen

- Eine beliebige Menge von Wörtern über Σ ist eine **Sprache** L ($L \subseteq \Sigma^*$).
- Beispiele: $\Sigma, \Sigma^*, \emptyset$ sind Sprachen.
- Endliche Sprachen können aufgelistet werden. Unendliche werden durch Abstraktion dargestellt.
 $L = \{w \in \Sigma^* | w \text{ hat die Eigenschaft } P\}$
Palindrome = $\{w \in \Sigma^* | w = w^R\}$
Endet-auf-1 = $\{w \in \Sigma^* | w \text{ endet auf } 1\}$

Operationen auf Sprachen

- Sprachen sind Mengen. Alle Mengenoperationen sind für Sprachen definiert.
- Das Komplement von A , $\bar{A} = \Sigma^* - A$.
- Die **Verkettung von Sprachen** L_1 und L_2 wird geschrieben als L_1L_2 .

$L_1L_2 = \{w \mid \text{Es gibt } x \in L_1 \text{ und } y \in L_2, \text{ so dass } w = xy\}$

- Der **Kleene-Stern** (Stephen C. Kleene, 1909-1994; amerikanischer Logiker) von L wird geschrieben als L^* .

$L^* = \{w \in \Sigma^* \mid \text{Es gibt } w_1, \dots, w_k \in L \text{ und } w = w_1 \circ \dots \circ w_k, \text{ für beliebige } k \geq 0\}$

$L^* =$ Menge der Wörter, erzeugt durch beliebig häufige Verkettung beliebiger Elemente von L .

Reguläre Ausdrücke

- **Reguläre Ausdrücke** (RAs) über einem Alphabet Σ sind alle Ketten (Wörter) über dem Alphabet $\Sigma \cup \{(\,), \phi, \cup, \star\}$, die auf folgende Weise generiert werden:
 1. ϕ und $x \in \Sigma$ sind RAs.
 2. Wenn α und β RAs sind, dann ist auch $(\alpha\beta)$ ein RA.
 3. Wenn α und β RAs sind, dann ist auch $(\alpha \cup \beta)$ ein RA.
 4. Wenn α ein RA ist, dann ist auch α^* ein RA.
 5. Nichts anderes ist ein RA.

Reguläre Ausdrücke 2

Reguläre Sprachen

- Welche Sprache (Menge von Wörtern) ein RA definiert, ist festgelegt durch die Funktion \mathcal{L} :

1. $\mathcal{L}(\phi) = \emptyset$ und $\mathcal{L}(x) = \{x\}$
2. $\mathcal{L}(\alpha\beta) = \mathcal{L}(\alpha)\mathcal{L}(\beta)$
3. $\mathcal{L}(\alpha \cup \beta) = \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$
4. $\mathcal{L}(\alpha^*) = \mathcal{L}(\alpha)^*$

- So ist jeder RA mit einer Sprache assoziiert.
- Beispiel: Sei $\Sigma = \{a, b, c\}$. Ein RA über Σ ist $((ab)^* \cup (b(ca)^*))$. Dieser RA denotiert die Sprache

1. $\mathcal{L}(((ab)^* \cup (b(ca)^*))) =$
2. $\{ab\}^* \cup \{b\}\{ca\}^* =$
3. $\{b, bca, bcaca, bcacaca, \dots, ab, abab, ababab, \dots\}$

- Die **Regulären Sprachen** über Σ sind die Menge aller Sprachen L , so dass ein RA α über Σ existiert, mit $\mathcal{L}(\alpha) = L$.
- Man kann auch eine Charakterisierung der regulären Sprachen geben, die vom Begriff des reflexiv transitiven Abschlusses Gebrauch macht.
- Die Menge der regulären Sprachen über Σ ist der reflexiv transitive Abschluss der Menge $\{\{\sigma\} \mid \sigma \in \Sigma\} \cup \emptyset$ bezüglich Verkettung, Vereinigung und Kleene-Stern.

Reguläre Sprachen 2

- Theorem: Es gibt Sprachen über einem Alphabet Σ , die nicht regulär sind.
- Begründung:
 1. Jede reguläre Sprache über Σ ist mit einem RA assoziiert.
 2. Für jeden RA α über $\Sigma \cup \{(\,, \phi, \cup, \star\}$ ($= \Sigma_{RA}$) gilt: $\alpha \in \Sigma_{RA}^*$.
 3. $|\Sigma_{RA}^*|$ ist abzählbar (die Elemente von Σ_{RA}^* können systematisch aufgezählt werden, so dass jedes einmal dran kommt).
 4. Für jede Sprache L über Σ gilt, dass $L \subseteq \Sigma^*$.
 5. Es gibt $|2^{\Sigma^*}|$ verschiedene Sprachen über Σ . Das sind überabzählbar viele.
 6. Es gibt also mehr Sprachen über Σ als RAs. Dann muss es Sprachen über Σ geben, die nicht regulär sind.

Erkennung und Generierung

- Eine Sprache L kann definiert werden, indem man einen **Spracherkenner** angibt, der für jedes beliebige Wort $w \in \Sigma^*$ sagt, ob $w \in L$.
- Eine Sprache L kann definiert werden, indem man einen **Sprachgenerator** angibt, der die Wörter von L aufzählt.
- Spracherkenner sind **Automaten**.
- Sprachgeneratoren sind **Grammatiken**.

Deterministische Endliche Automaten

- Ein DEA besteht aus einem **Leseband**, welches die Symbole des Eingabewortes w enthält und das von einem **Lesekopf** von links nach rechts eingelesen wird. Der Kopf kann nicht zurücksetzen.
- Der DEA ist zu Beginn in einem ausgezeichneten **Anfangszustand**. In einem Schritt liest der Kopf ein Symbol σ und wechselt in einen neuen Zustand q' , der ausschließlich vom aktuellen Zustand q und von σ abhängig ist.
- Danach bewegt sich der Kopf einen Schritt weiter nach rechts und liest das nächste Symbol.
- Wenn das letzte Symbol gelesen ist, und wenn sich der DEA in einem **finalen** Zustand befindet, dann hat der DEA w **akzeptiert**.
- Die Sprache, die der DEA akzeptiert, ist die Menge aller Wörter, die der DEA akzeptiert.

Formale Definition von DEA

- Ein DEA ist ein 5-Tupel $M = \{K, \Sigma, \delta, s, F\}$, wobei
 1. K die Menge der Zustände ist
 2. Σ das Eingabealphabet ist
 3. δ die **Übergangsfunktion** von $K \times \Sigma$ nach K ist
 4. s der Anfangszustand ist
 5. $F \subseteq K$ die Menge der finalen Zustände ist

Konfigurationen und Relationen zwischen ihnen

- Eine **Konfiguration** eines DEA ist $\in K \times \Sigma^*$, besteht also aus dem aktuellen Zustand des DEA und dem ungelesenen Teil der Eingabe.
- \vdash_M ist die Relation, die zwischen zwei Konfigurationen eines DEA M gilt, gdw. M in einem Schritt von der einen Konfiguration in die andere gelangen kann.
- Wenn (q, w) und (q', w') zwei Konfigurationen eines DEA M sind, dann gilt $(q, w) \vdash_M (q', w')$ gdw.
 1. es ein $a \in \Sigma$ gibt, so dass $w = aw'$ und
 2. $\delta(q, a) = w'$.
- \vdash_M ist eine Funktion $K \times \Sigma^+ \mapsto K \times \Sigma^*$. Wenn M in der Konfiguration (q, ϵ) ist, dann ist die Eingabe komplett gelesen.

Konfigurationen und Relationen zwischen ihnen 2

- \vdash_M^* ist der reflexiv transitive Abschluss von \vdash_M . $(q, w) \vdash_M^* (q', w')$ bedeutet, dass M von (q, w) aus (q', w') in beliebig vielen Schritten (inklusive 0 Schritten) erreicht.
- Ein Wort w wird von M akzeptiert, gdw. $(s, w) \vdash_M^* (q, \epsilon)$, mit $q \in F$.
- $L(M)$ ist die Sprache, die von M akzeptiert wird ($L(M) = \{w \mid w \text{ wird von } M \text{ akzeptiert}\}$).

Die Konstruktion von DEAs

- Ein DEA muss an jedem Punkt in der Lage sein, zu entscheiden, ob die Kette w , die er bereits gelesen hat, in L ist.
- Dazu muss er allerdings immer nur auf endlich viel Information zurückgreifen.
- Die relevante Information wird jeweils in den Zuständen ausgedrückt.
- Der Zustand, der die Information enthält, dass 0 Symbole gelesen wurden, ist der Anfangszustand.
- Zustände, die die Information enthalten, dass $w \in L$, sind die finalen Zustände.

Die Konstruktion von DEAs 2

- Beispiel 1: $L = \{w \mid w \text{ endet auf eine } 1\}$. Man muss wissen, ob
 1. das letzte Symbol eine 1 war
 2. das letzte Symbol keine 1 war
- Beispiel 2: $L = \{w \mid w \text{ enthält eine gerade Anzahl von } 1\text{en und } 0\text{en}\}$. Man muss wissen, ob
 1. Anzahl der 0en gerade ist, der 1en gerade ist
 2. Anzahl an 0en gerade ist, der 1en ungerade ist
 3. Anzahl an 0en ungerade ist, der 1en gerade ist
 4. Anzahl an 0en ungerade ist, der 1en ungerade ist

Die Konstruktion von DEAs 3

- Beispiel 3: $L = \{w | w \text{ enthält } 001 \text{ als Teilwort}\}$.
Man muss wissen, ob man gerade
 1. kein nicht-leeres Präfix von 001 gelesen hat
 2. 0 gelesen hat
 3. 00 gelesen hat
 4. 001 gelesen hat
- Beispiel 4: $L = \mathcal{L}((0 \cup 1)^*(000 \cup 111)(0 \cup 1)^*)$

Nicht-Determinismus

- Bei einem **nicht-deterministischen** endlichen Automaten (NDEA) bestimmen aktueller Zustand und gelesenes Symbol den nächsten Zustand nicht eindeutig.
- Das heißt, es gibt keine Übergangsfunktion sondern eine **Übergangsrelation**.
- Ein Tupel aus (Zustand, Symbol) kann zu verschiedenen Zuständen führen, oder zu überhaupt keinem Zustand.

Formale Definition von NDEA

- Ein NDEA ist ein 5-Tupel $M = \{K, \Sigma, \Delta, s, F\}$, wobei
 1. K die Menge der Zustände ist
 2. Σ das Eingabealphabet ist
 3. die Übergangsrelation Δ eine Teilmenge von $K \times (\Sigma \cup \{\epsilon\}) \times K$ ist
 4. s der Anfangszustand ist
 5. $F \subseteq K$ die Menge der finalen Zustände ist

Konfigurationen von NDEAs

- Jedes Tripel $(q, a, q') \in \Delta$ nennt man einen Übergang eines NDEA M .
- Konfigurationen von M sind Elemente aus $K \times \Sigma^*$.
- Die Relation \vdash_M zwischen zwei Konfigurationen von M ist definiert als $(q, w) \vdash_M (q', w')$ gdw.
 1. es ein $a \in (\Sigma \cup \{\epsilon\})$ gibt, so dass $w = aw'$ und
 2. $(q, a, q') \in \Delta$
- \vdash_M^* ist der reflexiv transitive Abschluss von \vdash_M .
- Ein Wort w wird von M akzeptiert, gdw. $(s, w) \vdash_M^* (q, \epsilon)$ und $q \in F$.
- $L(M) = \{w \mid w \text{ wird von } M \text{ akzeptiert}\}$.

Äquivalenz von Automaten

- Zwei Automaten M_1 und M_2 sind **äquivalent**, gdw. $L(M_1) = L(M_2)$.
- Theorem: Für jeden NDEA gibt es einen äquivalenten DEA.
- Intuition: der DEA simuliert in einem einzigen Zustand die Menge der Zustände, in die der NDEA gelangen würde, wenn er seine möglichen Übergänge bezüglich einer bestimmten Eingabe simultan ausführen würde. Wenn am Ende eine solche Zustandsmenge einen finalen Zustand enthält, dann wird die Eingabe akzeptiert.
- Einen Beweis für diese Äquivalenz findet man z.B. in Hopcroft & Ullman (1990).

Abschlusseigenschaften

- Theorem: Die Klasse der Sprachen, die von endlichen Automaten akzeptiert werden, ist abgeschlossen unter
 1. Vereinigung
 2. Verkettung
 3. Kleene-Stern
 4. Komplementbildung
 5. Schnitt
- Vereinigung: Seien $M_1 = \{K_1, \Sigma, \Delta_1, s_1, F_1\}$ und $M_2 = \{K_2, \Sigma, \Delta_2, s_2, F_2\}$ zwei NDEAs. Dann ist $M = \{K, \Sigma, \Delta, s, F\}$ ein NDEA, so dass $L(M) = L(M_1) \cup L(M_2)$, wobei
 1. $K = K_1 \cup K_2 \cup \{s\}$
 2. $F = F_1 \cup F_2$
 3. $\Delta = \Delta_1 \cup \Delta_2 \cup \{(s, \epsilon, s_1), (s, \epsilon, s_2)\}$

Abschlusseigenschaften 2

- Verkettung: Seien $M_1 = \{K_1, \Sigma, \Delta_1, s_1, F_1\}$ und $M_2 = \{K_2, \Sigma, \Delta_2, s_2, F_2\}$ zwei NDEAs. Dann ist $M = \{K, \Sigma, \Delta, s, F\}$ ein NDEA, so dass $L(M) = L(M_1) \cup L(M_2)$, wobei

1. $K = K_1 \cup K_2 \cup \{s\}$
2. $F = F_2$
3. $s = s_1$
4. $\Delta_3 = \{(f, \epsilon, s_2) \mid f \in F_1\}$
5. $\Delta = \Delta_1 \cup \Delta_2 \cup \Delta_3$

- Kleene-Stern: Sei $M_1 = \{K_1, \Sigma, \Delta_1, s_1, F_1\}$ ein NDEA. Dann ist $M = \{K, \Sigma, \Delta, s, F\}$ ein NDEA, so dass $L(M) = L(M_1)^*$, wobei

1. $K = K_1 \cup \{s\}$
2. $F = F_1 \cup \{s\}$
3. $s = s_1$
4. $\Delta_2 = \{(f, \epsilon, s_1) \mid f \in F\}$
5. $\Delta = \Delta_1 \cup \Delta_2$

Abschlusseigenschaften 3

- Komplementbildung: Sei $M = \{K, \Sigma, \delta, s, F\}$ ein **deterministischer** EA. Dann ist $\overline{M} = \{K, \Sigma, \delta, s, K - F\}$ ein DEA, so dass $L(\overline{M}) = \Sigma^* - L(M)$.

- Schnitt: $L_1 \cap L_2 = \Sigma^* - ((\Sigma^* - L_1) \cup (\Sigma^* - L_2)) = \overline{\overline{L_1} \cup \overline{L_2}}$. Mit anderen Worten: da die Sprachen, die durch endliche Automaten akzeptiert werden unter Komplementbildung und Vereinigung abgeschlossen sind, sind sie auch unter Schnitt abgeschlossen.

Endliche Automaten und reguläre Ausdrücke

- Theorem (Kleene): Eine Sprache ist regulär, gdw. sie von einem endlichen Automaten akzeptiert wird.
- Beweis (\Leftarrow): Es wird gezeigt, dass die Menge der Sprachen, die von einem endlichen Automaten akzeptiert werden, Teilmenge der Menge von Sprachen ist, die durch einen RA repräsentiert werden können.
 1. Die regulären Sprachen über Σ sind der reflexiv transitive Abschluss der Menge $\{\{\sigma\} | \sigma \in \Sigma\} \cup \emptyset$ bezüglich Verkettung, Vereinigung und Kleene-Stern.
 2. Man kann Automaten konstruieren, die \emptyset und jeweils die Elemente aus $\{\{\sigma\} | \sigma \in \Sigma\}$ akzeptieren.

Endliche Automaten und reguläre Ausdrücke 2

- Fortsetzung Beweis (\Leftarrow):
 3. Wir wissen, dass die Sprachen, die von endlichen Automaten akzeptiert werden, abgeschlossen sind bezüglich Vereinigung, Verkettung und Kleene-Stern.
 4. Aber dann sind die regulären Sprachen eine Teilmenge der von endlichen Automaten akzeptierten Sprachen.
- Beweis (\Rightarrow): zu komplex, aber siehe z.B. Hopcroft & Ullman (1990) oder Schöning (1992).

Reguläre und nicht-reguläre Sprachen

- Alle endlichen Sprachen sind regulär.
- Nicht alle unendlichen Sprachen sind regulär. Es gibt also Sprachen, die nicht durch einen RA repräsentiert werden können und die nicht von einem DEA/NDEA akzeptiert werden.
- Beispiele für nicht-reguläre Sprachen:
 1. $L_1 = \{0^n 1^n \mid n \geq 0\}$
 2. $L_2 = \{w \mid w \text{ enthält gleichviele 0en wie 1en}\}$
 3. $L_3 = \{w \mid w = w^R\}$ (Sprache der Palindrome)
 4. $L_4 = \{ww \mid w \in \{0, 1\}^*\}$ (die Kopiersprache)
 5. $L_5 = \{0^n \mid n \text{ ist eine Primzahl}\}$
 6. $L_6 = \{0^{n^2} \mid n \geq 0\}$
 7. $L_7 = \{0^i 1^j \mid i > j\}$

Das Pumping-Lemma für reguläre Sprachen

- Das **Pumping-Lemma** für reguläre Sprachen wird meist benutzt, um zu zeigen, dass eine Sprache L nicht regulär ist.
- Intuition: wenn eine reguläre Sprache L unendlich ist, dann muss der Automat, der L akzeptiert, eine **Schleife** haben, die man beliebig oft (auch 0-mal) durchlaufen kann (**aufpumpen** kann). Wenn so eine Schleife nicht existiert, dann ist L nicht regulär.
- Pumping-Lemma: Wenn A eine reguläre Sprache ist, dann gibt es eine Zahl p (die **Pumpzahl**), so dass für jedes Wort $w \in A$, mit $|w| \geq p$ folgendes gilt. Man kann w aufteilen in $w = xyz$, so dass die folgenden Bedingungen erfüllt sind.
 1. $|xy| \leq p$ (Bedingung 1)
 2. $|y| > 0$ (Bedingung 2)
 3. für jedes $i \geq 0 : xy^i z \in A$ (Bedingung 3)

Beweis des Pumping-Lemmas

- Sei $M = \{K, \Sigma, \delta, q_0, F\}$ ein DEA, der A akzeptiert und sei p die Anzahl seiner Zustände.
- Sei $w = s_1 s_2 \dots s_n \in A$ ein Wort der Länge n , wobei $n \geq p$.
- Sei schließlich $(r_1 r_2 \dots r_{n+1})$ die Folge der Zustände, die M beim akzeptieren von w durchläuft ($r_1 = q_0, r_{i+1} = \delta(r_i, s_i)$).
- Da dies mehr Zustände sind, als M besitzt, müssen nach dem Schubfachprinzip unter den ersten $p + 1$ Zuständen der Folge mindestens zwei identisch sein. Nenne den ersten der beiden r_j und den zweiten r_l (man beachte, dass daraus folgt, dass $l \leq p + 1$).

Beweis des Pumping-Lemmas 2

- Sei nun $x = s_1 \dots s_{j-1}, y = s_j \dots s_{l-1}$ und $z = s_l \dots s_n$. Dann gilt:
 1. x führt M von r_1 nach r_j .
 2. y führt M von r_j nach $r_j (= r_l)$.
 3. z führt M von r_j nach r_{n+1} .
- Aus $l \leq p + 1$ folgt $|xy| \leq p$.
- Wir wissen, dass $j \neq l$, daher ist $|y| > 0$.
- Da $r_j = r_l$, erreicht M bei Eingabe von $w = xz$ ($= xy^0 z$) denselben Zustand wie nach Eingabe von $w = xyz$ ($= xy^1 z$). Dasselbe gilt für $w = xy^2 z$ etc., also für $w = xy^i z$ für $i \geq 0$.
- Also sind alle drei Bedingungen des Pumping-Lemmas erfüllt.

Die Anwendung des Pumping-Lemmas

- $L_1 = \{0^n 1^n \mid n \geq 0\}$ ist nicht regulär.
- Beweis durch Widerspruch: Angenommen, L_1 sei regulär. Sei p die Pumpzahl und $w = 0^p 1^p$ ($|w| > p$). Man zeigt nun, dass es kein y geben kann, mit $|y| > 0$, $w = xyz$ und $xy^i z \in L_1$, für alle $i \geq 0$.
 1. Fall 1: Angenommen, y enthält nur 0en. Dann führte das Aufpumpen von y zu mehr 0en als 1en in w (dann aber wäre $w \notin L_1$).
 2. Fall 2: Angenommen, y enthält nur 1en. Dann führte das Aufpumpen von y zu mehr 1en als 0en in w (dann aber wäre $w \notin L_1$).
 3. Fall 3: Angenommen, y besteht aus 0en und 1en. Dann führte das Aufpumpen von y zu wiederholt abwechselndem Auftauchen von 0en als 1en in w (dann gilt wieder: $w \notin L_1$).
 4. y kann also nicht existieren. Das ist aber ein Widerspruch zum Pumping-Lemma. Daher kann L_1 nicht regulär sein.

Die Anwendung des Pumping-Lemmas 2

- Das letzte Beispiel zeigte, dass kein w , das länger ist als p , aufgepumpt werden kann. Es reicht aber schon, zu zeigen, dass dies für **ein** Wort nicht gilt.
- $L_2 = \{w \mid w \text{ enthält gleichviele 0en wie 1en}\}$ ist nicht regulär.
- Beweis durch Widerspruch: Angenommen, L_2 sei regulär. Sei p die Pumpzahl und $w = 0^p 1^p$ ($|w| > p$). Man zeigt, dass es keine x, y, z geben kann, mit $|y| > 0$ (Bed. 2), $|xy| \leq p$ (Bed. 1) und $xy^i z \in L_2$, für alle $i \geq 0$ (Bed. 3).
 1. Nach der Wahl von w und wegen Bedingung 1 muss y ausschließlich aus 0en bestehen.
 2. Dann führt das Aufpumpen von y aber zu Wörtern, die nicht in L_2 sind.
 3. y kann also nicht aufgepumpt werden, im Widerspruch zum Pumping-Lemma.
 4. Dann kann L_2 aber nicht regulär sein.

Anwendung des Pumping-Lemmas 3

- Beim letzten Beweis war Bedingung 1 wichtig: sonst hätte man $x = 0^{p-1}$, $y = 01$ und $z = 1^{p-1}$ wählen können, was y aufpumpbar macht.
- Die Wahl von w war ebenfalls wichtig. Man hätte auch $w = (01)^p$ wählen können, was dann mit $x = \epsilon$, $y = 01$ und $z = 1^{p-1}$ zur Pumpbarkeit von y geführt hätte.

Anwendung des Pumping-Lemmas 4

- $L_3 = \{w | w = w^R\}$ (Sprache der Palindrome) ist nicht regulär.
- Beweis: Angenommen L_3 ist regulär. Sei p die Pumpzahl und sei $w = 0^p 1 0^p$.
 1. Dann muss y nach Bedingung 1 ausschließlich aus 0en bestehen.
 2. Dann entsteht durch Aufpumpen von y ein Wort, welches nicht in L_3 ist.
 3. Dann ist y aber nicht pumpbar, in Widerspruch zum Pumping-Lemma.
 4. Also ist L_3 nicht regulär.

Anwendung des Pumping-Lemmas 5

- $L_4 = \{ww \mid w \in \{0,1\}^*\}$ (die Kopiersprache) ist nicht regulär.
- Beweis: Angenommen L_4 ist regulär. Sei p die Pumpzahl und sei $w = 0^p 1 0^p 1$.
 1. Dann muss y nach Bedingung 1 ausschließlich aus 0en bestehen.
 2. Dann entsteht durch Aufpumpen von y ein Wort, welches nicht in L_4 ist.
 3. Dann ist y aber nicht pumpbar, in Widerspruch zum Pumping-Lemma.
 4. Also ist L_4 nicht regulär.
- Die Wahl von w ist wichtig: bei $w = 0^p 0^p$ ist y pumpbar.

Abwärts pumpen

- Die Beweise bisher haben immer davon Gebrauch gemacht, dass y aufgepumpt wurde und dabei ein Wort entstand, welches nicht in der Sprache liegt.
- Bedingung 3 sagt: für jedes $i \geq 0 : xy^i z \in A$. Das heißt, dass man auch den Fall $i = 0$ benutzen kann. In diesem Fall pumpt man abwärts.
- $L_7 = \{0^i 1^j \mid i > j\}$ ist nicht regulär.
- Beweis: Angenommen, L_7 sei regulär. Sei p die Pumpzahl und $w = 0^{p+1} 1^p$. Dann muss $w = xyz$ die drei Bedingungen des Pumping-Lemmas erfüllen:
$$\begin{array}{ll} |xy| \leq p & \text{(Bedingung 1)} \\ |y| > 0 & \text{(Bedingung 2)} \\ \text{für jedes } i \geq 0 : xy^i z \in A & \text{(Bedingung 3)} \end{array}$$

Abwärts pumpen 2

- Fortsetzung Beweis
 1. Es folgt aus Bedingung 1, dass y nur aus 0en besteht.
 2. Man beachte, dass für alle $i > 0$ gilt: $xy^iz \in L_7$.
 3. Aber da $|y| > 0$ und da y nur aus 0en besteht, gilt für $i = 0$, dass $xz \notin L_7$.
 4. Daher kann $w = 0^{p+1}1^p$ nicht abgepumpt werden, was ein Widerspruch zum Pumping-Lemma ist.
 5. Deswegen ist L_7 nicht regulär.

Eine Alternative zum Pumping-Lemma

- Ein anderer Weg, zu zeigen, dass eine Sprache nicht regulär ist, involviert die Abschlusseigenschaften der regulären Sprachen.
- Beispiel: Reguläre Sprachen sind abgeschlossen unter Schnittbildung.
 1. $L_1 = \{0^n 1^n \mid n \geq 0\}$ ist nicht regulär.
 2. $L' = \mathcal{L}(0^* 1^*)$ ist regulär.
 3. $L_2 = \{w \mid w \text{ enthält gleichviele 0en wie 1en}\}$
 4. $L_1 = L_2 \cap L'$
 5. Daher kann L_2 nicht regulär sein.

Entscheidbarkeit

- Eine Menge $A \subseteq \Sigma^*$ heißt **entscheidbar**, falls es ein **mechanisches** Verfahren gibt, welches für eine beliebige Eingabe $w \in \Sigma^*$ nach endlich vielen Schritten anhält und entscheidet, ob $w \in A$ oder $w \notin A$.
- Eine Menge $A \subseteq \Sigma^*$ heißt **semi-entscheidbar**, falls es ein mechanisches Verfahren gibt, welches für eine beliebige Eingabe $w \in \Sigma^*$ nach endlich vielen Schritten anhält, wenn $w \in A$.
- Der Begriff der Entscheidbarkeit kann ausgeweitet werden auf andere Fragen als die Elementbeziehung.
- So kann man fragen, ob es entscheidbar ist, ob eine Menge leer oder unendlich ist.

Entscheidbarkeit 2

- Theorem: Die Sprache L , die von einem endlichen Automat M mit n Zuständen akzeptiert wird, ist
 1. **nicht leer** gdw. M ein Wort w akzeptiert, so dass $|w| < n$.
 2. **unendlich** gdw. M ein Wort w der Länge l akzeptiert, wobei $n \leq l < 2n$.
- Aus diesem Theorem folgt automatisch, dass es entscheidbar ist, (1) ob $L(M)$ leer ist und (2) ob $L(M)$ unendlich ist.
- Um (1) zu entscheiden, prüft man alle $w, |w| < n$, ob $w \in L(M)$. Das ist entscheidbar, da es nur endlich viele solche w s gibt.
- Um (2) zu entscheiden, prüft man alle $w, n \leq |w| < 2n$, ob $w \in L(M)$. Das ist auch entscheidbar, weil diese Menge ebenfalls endlich ist.

Entscheidbarkeit 3

- Beweis des ersten Theoremteils.
- \Leftarrow : klar. Wenn es ein $w \in L(M)$ gibt, dann ist $L(M) \neq \emptyset$.
- \Rightarrow : Sei $L(M) \neq \emptyset$ und sei w ein Wort in $L(M)$ mit minimaler Länge. Angenommen $|w| \geq n$.
 1. Dann gibt es nach dem Pumping-Lemma eine Zerlegung von $w = xyz$, so dass auch $xy^0z = xz \in L(M)$.
 2. Das widerspricht aber der Minimalität von w .
 3. Also ist $|w| < n$.

Entscheidbarkeit 4

- Beweis des zweiten Theoremteils.
- \Leftarrow : Sei $w \in L(M)$ mit $n \leq |w| < 2n$.
 1. Wegen des Pumping-Lemmas lässt sich w zerlegen in xyz , so dass $\{xy^i z | i \geq 0\} \subseteq L(M)$.
 2. Dann ist $L(M)$ unendlich.
- \Rightarrow : Sei $L(M)$ unendlich und sei per Annahme das kürzeste Wort $w \in L(M)$, mit $|w| \geq n$, so, dass tatsächlich $|w| \geq 2n$.
 1. Wegen des Pumping-Lemmas lässt sich w zerlegen in xyz , so dass auch $xy^0z = xz \in L(M)$.
 2. Da $|y| \leq |xy|$ und da $|xy| \leq n$ (Bedingung 1), folgt dass $|xz| \geq n$ (und $|xz| < 2n$, denn $|y| \leq n$ und $|xyz| > 2n$).
 3. Das widerspricht aber der Minimalität von w .
 4. Also gibt es ein Wort w' $n \leq |w'| < 2n$.

References

Hopcroft, John E. & Jeffrey D. Ullman (1990):
Einführung in die Automatentheorie, formale Sprachen und Komplexitätstheorie. Addison-Wesley.

Humboldt, Wilhelm von (1836): *Über die Verschiedenheit des menschlichen Sprachbaues und ihren Einfluss auf die geistige Entwicklung des Menschengeschlechts*. Königlich Preussische Akademie der Wissenschaften, Berlin.

Loos, Rüdiger (1993): *Formale Sprachen – Theorie und Anwendung*. Vorlesungsmanuskript, Universität Tübingen.

Schöning, Uwe (1992): *Theoretische Informatik kurz gefasst*. Wissenschaftsverlag, Mannheim.