Axiomatisierung von Stringkonkatenation

Alphabet: endliche, nicht-leere Menge von Symbolen.

String, Zeichenkette: Folge von Symbolen

Konkatenation: Aneinanderreihung, zweistellige Operation “,” auf einer Menge A von Zeichenketten.

ϵ: Zeichenkette der Länge 0.

- Axiomatisierung ohne leeren String (Halbgruppe)

  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$

  Beispiel für ein Modell dafür: $A = \{z \mid z$ besteht aus einer geraden Anzahl von 'a's und / oder 'b's$\}$.

- Axiomatisierung mit leerem String (Monoid)

  1. Abgeschlossenheit:
     $\forall x \forall y((x \in A \wedge y \in A) \rightarrow x, y \in A)$
  2. Assoziativität:
     $\forall x \forall y \forall z((x, y), z = x, (y, z))$
  3. neutrales Element:
     $\exists e \forall x(x, e = e, x = x)$

Schreibweise: statt x, y auch $x \frown y$ oder nur xy.

Wörter über Alphabeten

Sei Σ ein Alphabet, dann ist Σ^* die Menge aller endlichen *Wörter*, die sich durch Aneinanderreihung der Elemente von Σ bilden lassen, die Menge der Wörter über Σ.

Dazu gehört auch das *leere Wort*, ϵ.

$\Sigma^+ := \Sigma^* \backslash \{\epsilon\}$ ist die Menge der nicht-leeren Wörter über Σ.

Bsp: $\Sigma = \{a, b\}$. $\Sigma^* = \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, ...\}$.

Die *Länge* eines Wortes w, $\mid w \mid$, ist die Anzahl der Zeichen, aus denen es besteht.

Sei a ein Element eines Alphabets. a^n bezeichnet das Wort der Länge n aus lauter as.

Eine (formale) *Sprache* ist eine Menge von Wörtern.
Bsp: $L = \{a^n b^n \mid n \geq 1\}$

Grammatik

<u>Definition</u>:

Eine (*Phrasenstruktur-*)*Grammatik* G ist ein 4-Tupel,

$$(V, \Sigma, P, S)$$

mit folgenden Eigenschaften:

V ist eine endliche Menge, die *Variablen*, oder Nicht-Terminalsymbole.

Σ ist ein Alphabet, das *Terminalalphabet*, die Terminalsymbole. $V \cap \Sigma = \emptyset$.

P ist eine endliche Menge von *Regeln, Produktionen* der Form $u \to v$ mit $u \in ((\Sigma \cup V)^* \times V \times (\Sigma \cup V)^*)$ und $v \in (\Sigma \cup V)^*$, d.h. ein Wort, das mindestens aus einem Nicht-Terminalsymbol besteht, kann durch ein beliebiges Wort ersetzt werden.

$S \in V$ ist das *Startsymbol*, ein besonderes Element von V.

Eine Grammatik ist ein deduktives System: Axiome und Inferenzregeln. Das einzige Axiom einer Grammatik ist das Startsymbol, die Regeln sind die Produktionen.

formale Sprache

$\Rightarrow_G$ ist die Relation in $(V \cup \Sigma)^*$ mit:

$u \Rightarrow_G v$ gdw. $u = xyz, v = xy'z$ mit $x, z, y, y' \in (V \cup \Sigma)^*$ und $y \rightarrow y' \in P$.

"u geht unter G unmittelbar in v über "

$\Rightarrow_G^*$ ist die *reflexive und transitive Hülle* von $\Rightarrow_G$.

$x \Rightarrow_G^* y$, wenn gilt:

$x = y$, $x \Rightarrow_G y$ oder es gibt ein z mit $x \Rightarrow_G^* z$ und $z \Rightarrow_G y$.

Eine Grammatik G *erzeugt* ein Wort w, wenn gilt:

$$S \Rightarrow_G^* w$$

Die Sprache $L(G)$ ist die Menge der Wörter, die von der Grammatik G erzeugt werden.

$$L(G) = \{w \in \Sigma^* \mid S \Rightarrow_G^* w\}$$

Eine Folge von Wörtern $(w_0, w_1, ..., w_n)$ mit $w_0 = S$, $w_n \in \Sigma^*$ und $w_0 \Rightarrow w_1 \Rightarrow ... \Rightarrow w_n$ heisst Ableitung von w_n.

Beispiel

$G = (V, \Sigma, P, S)$

V = { S, NP, VP, T-V, I-V, D, N }

Σ = { [the], [cat], [dog], [sleeps], [chases] }

P = {
S $\rightarrow$ NP VP,
NP $\rightarrow$ D N,
VP $\rightarrow$ T-V NP,
VP $\rightarrow$ I-V,
N $\rightarrow$ [dog],
N $\rightarrow$ [cat],
I-V $\rightarrow$ [sleeps],
T-V $\rightarrow$ [chases],
D $\rightarrow$ [the],
}

S = S.

Erkennungsproblem

Problem

Ist ein gegebenes Objekt Element einer Menge?

Ein Problem ist *entscheidbar*, wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe zur Menge gehört oder nicht.

Ein Problem ist *semi-entscheidbar*, wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe zur Menge gehört, oder wenn in einer endlichen Anzahl von Berechnungsschritten festgestellt werden kann, ob eine Eingabe nicht zur Menge gehört!

Erkennungsproblem oder *Wortproblem*

Ist ein Wort w Element einer Sprache L?

Bäume

Ein Baum besteht aus *Knoten*, $n \in N$, und *Kanten*, $e \in E, E \subset N \times N$, wobei jeder Knoten maximal einen Vorgänger und beliebig, aber endlich viele Nachfolger hat:

Für alle $n \in N$: es gibt maximal ein $m \in N$ mit $\langle m, n \rangle \in E$.

Ausserdem ist E irreflexiv und intransitiv, d.h. es darf keine zirkuläre Folge von Paaren in E geben.

Ein Knoten ohne Vorgänger heisst *Wurzel*, ohne Nachfolger *Blatt* des Baumes.

Ableitungen in Grammatiken, bei deren Regeln immer nur ein einzelnes Nichtterminalsymbol auf der linken Seite steht, kann man einen *Syntaxbaum* zuordnen.

Die Wurzel ist dann mit dem Startsymbol, die inneren Knoten sind mit den verwendeten Nichtterminalsymbolen, die Blätter mit den Terminalsymbolen beschriftet.

Beispiel

(Links-)Ableitung (d.h., das jeweils am weitesten links stehende Nichtterminal wird ersetzt):

S ⇒ NP VP ⇒ D N VP ⇒ [the] N VP ⇒

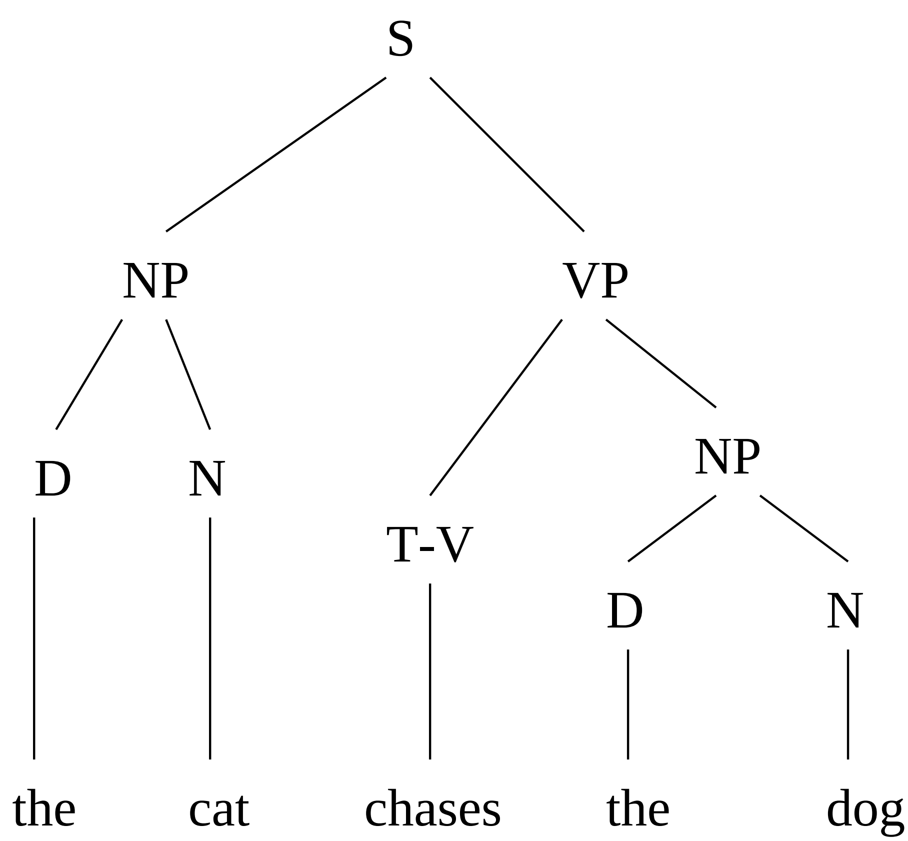
[the] [cat] VP ⇒ [the] [cat] T-V NP ⇒

[the] [cat] [chases] NP ⇒

[the] [cat] [chases] D N ⇒

[the] [cat] [chases] [the] N ⇒

[the] [cat] [chases] [the] [dog]



Syntaxbäume

Auch eine andere Ableitung (z.B. mit NP VP $\Rightarrow$ NP T-V NP) hätte denselben Syntxbaum ergeben...

Es gilt:

$x \in L(G)$,

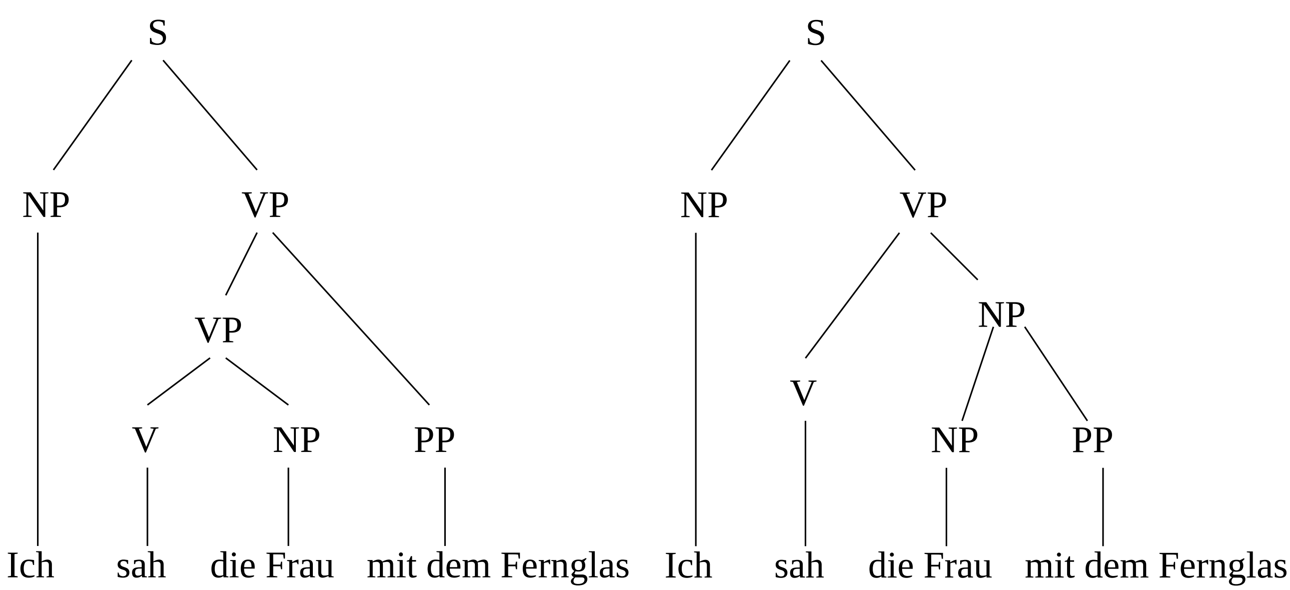
gdw. es gibt einen Syntaxbaum mit x an den Blättern,

gdw. es gibt eine Linksableitung für x

mehrdeutige Grammatik: für dasselbe Wort gibt es mehrere Syntaxbäume.

Beispiel:

G mit $P = \{$ S $\rightarrow$NP VP, NP $\rightarrow$NP PP, NP $\rightarrow$D N, VP $\rightarrow$VP PP, VP $\rightarrow$V NP $\}$.



Chomsky-Hierarchie

Je nach Form der Regeln lassen sich Grammatiken einem (maximalen) *Typ* zuordnen:

Typ 0:

Allgemeine Phrasenstrukturgrammatik, keinerlei Beschränkungen

Typ 1:

kontextsensitive Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt: $|w_1| \leq |w_2|$.

Typ 2

kontextfreie Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt, dass w_1 ein einzelnes Nichtterminal ist, d.h. $w_1 \in V$.

Typ 3

reguläre Grammatik:

Für alle Regeln $w_1 \to w_2$ in P gilt, dass w_2 ein einziges Terminal oder ein Terminal, gefolgt von einem Nichtterminal ist, d.h., $w_2 \in \Sigma \cup (\Sigma \times V)$

Eine Sprache ist vom Typ X, wenn sie von einer Grammatik vom Typ X erzeugt wird.

Es gilt: Typ 3 $\subset$ Typ 2 $\subset$ Typ 1 $\subset$ Typ 0.

Beispiele

Typ 1

$$L = \{a^n b^n c^n \mid n \leq 1\}$$

Typ 2

$$L = \{a^n b^n \mid n \leq 1\}$$

Typ 3

$$L = \{a^n \mid n > 1\}$$

Beispiel: arithmetische Ausdrücke:

$\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, \times, (,)\}$ ...

Grammatik dazu:

P = $\{$ S $\rightarrow$T, T $\rightarrow$(T + T), T $\rightarrow$T $\times$ T,

T $\rightarrow$Z, Z $\rightarrow$$Z_1$ $\mid$ Z Z_0, Z_0 $\rightarrow$$Z_1$ $\mid$ 0,

Z_1 $\rightarrow$1 $\mid$ 2 $\mid$ 3 $\mid$ 4 $\mid$ 5 $\mid$ 6 $\mid$ 7 $\mid$ 8 $\mid$ 9$\}$.

Automaten

Ein Automat ist eine abstrakte Berechnungsmaschine, mit einer Menge von Eingaben und einem Verhalten dazu.

Eingaben: diskret (nicht kontinuierlich), Folgen von Zeichen eines Alphabets.

Innere Struktur: endliche Menge von *Zuständen*.

Ausgabe: Annahme oder Ablehnung einer Eingabe.

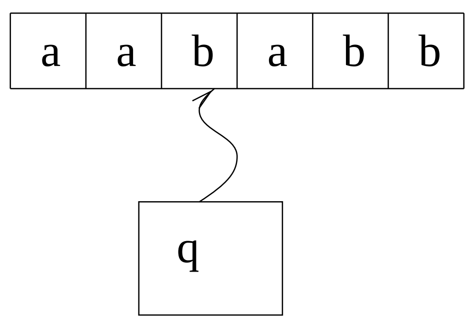
Verhältnis von Automaten und formalen Sprachen:

Automat kann als Akzeptant einer Sprache (und als ”Ablehner” ihres Komplements) betrachtet werden.

Endliche Automaten, DFAs

Ein *deterministischer endlicher Automat* (*deterministic finite Automaton*, DFA) liest eine endliche Folge von Zeichen nacheinander von links nach rechts ein, hält nach dem letzten Zeichen an und gibt ”akzeptiert” oder ”abgelehnt” aus.

Zu jedem Zeitpunkt befindet sich der Automat in einem von einer endlichen Anzahl von Zuständen. Es gibt einen Startzustand und eine Menge von Endzuständen.



Berechnungsschritte: wenn in einem bestimmten Zustand ein bestimmtes Zeichen gelesen wird, wird in einen bestimmten Zustand übergegangen.

Bsp: (q_1, a, q_2)

Endliche Automaten

Definition:

Ein *deterministischer endlicher Automat* ist ein 5-Tupel $(K, \Sigma, \delta, q_0, F)$ mit:

K eine endliche Menge von Zuständen,

Σ ein Alphabet,

δ eine Funktion von $K \times \Sigma$ nach K, die Zustandsübergangsfunktion,

$q_0 \in K$ der Startzustand,

$F \subseteq K$ die Endzustände.

Beispiel:

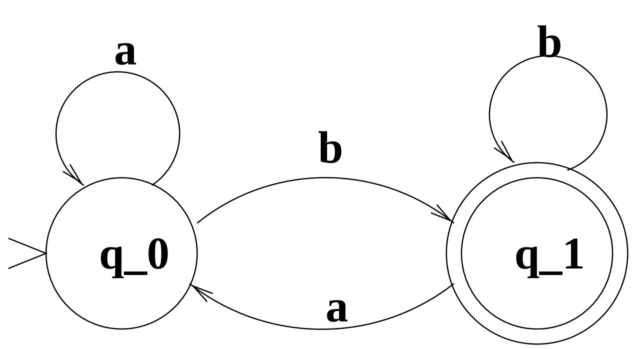
DFA mit Zuständen $\{q_0, q_1\}$,

Zustandsübergangsfunktion

$\delta(q_0, a) = q_0, \delta(q_0, b) = q_1, \delta(q_1, a) = q_0, \delta(q_1, b) = q_1$,

Startzustand q_0, Endzustand q_1.

Zustandsübergangsdiagramm:



Endliche Automaten

Situation eines Automaten: ”Schnappschuss”,

Position des Einlesekopfes auf der Eingabe und Zustand des Automatens.

Sei $M = (K, \Sigma, \delta, q_0, F)$. Eine Situation ist ein Tripel (x, q, y), wobei $q \in K$ und $x, y \in \Sigma^*$

So definierte Situationen müssen nicht erreichbar sein...

Relation ”erzeugt in einem Schritt” zwischen Situationen, $\vdash$:

$(x, q, y) \vdash (x', q', y')$ gdw. es gibt ein $\sigma \in \Sigma$ mit $y = \sigma y'$, $x' = x\sigma$ und $\delta(q, \sigma) = q'$.

$\vdash^*$ reflexive, transitive Hülle dazu: $s \vdash^* s'$, wenn s in keinem oder mehr Schritten s' erzeugt.

Nicht-Deterministische Automaten, NFAs:

statt Funktion δ Relation Δ.

Akzeptanz einer Sprache: wenn <u>ein</u> Pfad im Zustandübergangsdiagramm existiert...

Beide Arten endlicher Automaten akzeptieren dieselbe Klasse von Sprachen, nämlich die regulären Sprachen...

reguläre Sprachen

<u>Def.</u>: Seien A und B Mengen von Zeichenketten. Dann heisst AB Konkatenation von A und B, wenn gilt: $AB = \{x, y \mid x \in A$ und $y \in B\}$

<u>Definition</u>: *reguläre Sprache*

Sei Σ ein Alphabet. Dann gilt:

- $\emptyset$ ist eine reguläre Sprache.
- Für alle $x \in \Sigma^*$ ist $\{x\}$ eine reguläre Sprache.
- Wenn A und B reguläre Sprachen sind, dann auch A^*, $A \cup B$ und AB.

reguläre Ausdrücke

Reguläre Ausdrücke sind eine Meta-Notation für Wörter.

<u>Definition</u>: *reguläre Ausdrücke*

Sei Σ ein Alphabet. Dann gilt:

- $\emptyset$ ist ein regulärer Ausdruck.
- ϵ ist ein regulärer Ausdruck.
- Für alle $a \in \Sigma^*$ ist a ein regulärer Ausdruck.
- Wenn α und β reguläre Ausdrücke sind, dann auch $(\alpha)^*$, $(\alpha \mid \beta)$ und $\alpha\beta$.

$(\alpha \mid \beta)$ entspricht: " entweder ein Ausdruck α oder ein Ausdruck β".

Bsp: die Sprache, die der Beispiel-DFA akzeptiert: $L = (a \mid b)^* b$

Es gilt: Die Sprachen, die durch

- eine Typ-3-Grammatik beschrieben,
- einen endlichen Automaten (deterministisch oder nicht) akzeptiert
- und einen regulären Ausdruck beschrieben

werden, gehören derselben Klasse an.

Pumping-Lema für reguläre Sprachen

Sei L eine reguläre Sprache. Dann gibt es eine Zahl n, so dass gilt: alle Wörter $x \in L$ mit $|x| > 0$ lassen sich zerlegen in $x = uvw$ mit:

- $|v| \geq 1$
- $|uv| \leq n$
- für alle $i = 0, 1, 2, ...$ gilt: $uv^iw \in L$.

<u>Beweis</u>:

Es gibt einen Automaten M, der L akzeptiert.

Sei n die Anzahl der Zustände in M.

Bei Verarbeitung von x mit $|x| \geq n$ durchläuft M $|x| + 1$ Zustände.

Diese können nicht alle verschieden sein (es gibt nur n Zustände).

Also durchläuft M eine Schleife.

Wähle $x = uvw$ so, dass der Zustand nach u und uv derselbe ist.

Wenn also $uvw \in L$, dann auch $uw = uv^0w$, uv^1w,uv^2w etc.

Anwendung:

Das Pumping-Lemma ist zu gebrauchen, um zu zeigen, dass eine Sprache nicht regulär ist:

Bsp.: $L = \{a^m b^m \mid m \geq 1\}$.

Annahme: L sei regulär.

Es gibt n mit den Eigenschaften wie oben.

Man nehme das Wort $x = a^n b^n$ der Länge $2n$.

Die Zerlegung uvw muss erfüllen, dass v nicht leer ist, und nur aus as besteht.

Dann müsste sowohl v aus dem Wort x entfernt werden können, als auch beliebig oft eingefügt werden.

Es gilt aber: $m + \mid v \mid \neq m$.

Also war die Annahme falsch.

Eigenschaften regulärer Sprachen

Die Klasse ist *abgeschlossen* unter

- Vereinigung
- Komplement
- Schnitt
- Konkatenation
- Stern (Kleene)

Natürliche Sprachen $\subseteq$ Reguläre Sprachen?

Pumpinglemma...