Die Anwendung von Regeln $t_1 \to t_1'$, $t_2 \to t_2'$
in einem Wort $\alpha, \beta, \gamma \in (T \cup N)^*$

$$\alpha \circ t_1 \circ \beta \circ t_2 \circ \gamma \longrightarrow\!\!>$$
$$[\alpha \circ t_1' \circ \beta \circ t_2 \circ \gamma \longrightarrow\!\!>] \quad \alpha \circ t_1 \circ \beta \circ t_2' \circ \gamma \longrightarrow\!\!>$$
$$\alpha \circ t_1' \circ \beta \circ t_2' \circ \gamma$$

beide Regeln anwendbar, Reihenfolge egal

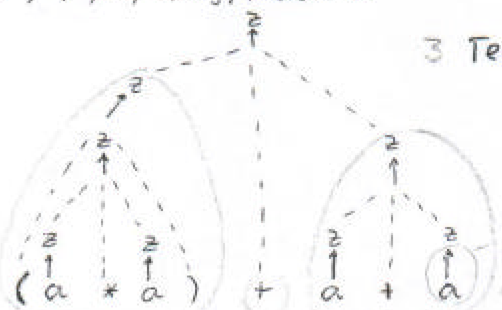
heißt vertauschbar.

Bemerkung: Formal ist jede Ableitung ein Wort aus $(T \cup N \cup \{\longrightarrow\!\!>\})^*$
Vertauschbarkeit definiert eine Relation auf der Menge der Ableitungen.
Wir können zu dieser Relation die transitive, symmetrische, reflexive Hülle bilden. ist logischerweise Äquivalenzrelation

Stehen zwei Ableitungen in dieser Relation, dann heißen sie strukturell äquivalent.

Ch-2-Grammatiken (kontextfreie Grammatiken) können wir aufgrund der einfachen Struktur ihrer Regeln jeder Ableitung auf das Axiom einen Strukturbaum zuordnen.

Beispiel: Nichtterminalzeichen $N = \{z\}$
Terminalzeichen $T = \{"(", ")", +, *, a\}$, Axiom z
Regeln: $z + z \longrightarrow z$
$z * z \longrightarrow z$
$a \longrightarrow z$
$(z) \longrightarrow z$



Ableitungsbaum, Strukturbaum

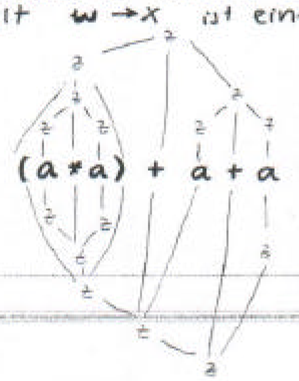
Strukturbäume geben genau die Struktur der Ableitungen (Ch-2 und Ch-3)

Wichtig: Zwei Ableitungen sind genau dann strukturell äquivalent, wenn sie den gleichen Strukturbaum darstellen.
d.h. Strukturbäume sind Normalform in der Klasse der strukturell äquivalenten Ableitungen.

Ein Strukturbaum für eine Ch-2-Grammatik ist ein endlich verzweigender Baum (geordneter Baum):
(1) Die Wurzel ist das Axiom.
(2) Jeder Teilbaum hat folgende Eigenschaften: (entweder a oder b)
(a) Er ist genau ein Terminalzeichen und hat keine Teilbäume
(b) Er hat als Wurzel ein Nichtterminalzeichen^x; dann gilt: die Wurzel seiner Teilbäume können zu dem Wort w konkateniert werden und es gilt $w \to x$ ist eine Regel der Grammatik

Anmerkung zum Bsp:

Grammatik hat zwei Strukturbäume!



für jedes Wort genau ein Strukturbaum → eindeutige Grammatik

Eine Grammatik heißt <u>eindeutig</u>, wenn für jedes Wort $w \in T^*$ mit $w \twoheadrightarrow^* z$ (z Wurzel) alle Ableitungen auf z strukturell äquivalent sind.
Dann existiert genau ein Strukturbaum für jedes Wort im Sprachschatz.
<u>Feststellung</u>: Die Grammatik aus obigem Beispiel ist nicht eindeutig.

<u>1.2.4. Sackgassen, unendliche Ableitungen</u>

Ch-2-G werden praktisch eingesetzt, um
(1) eine formale Sprache zu definieren
(2) jedem Wort im Sprachschatz einen Strukturbaum (am besten eindeutig) zuzuordnen.

Ein Wort w heißt „im Sprachschatz", bzw. reduzierbar oder auch von der Grammatik akzeptiert, wenn gilt $w \twoheadrightarrow^* z$.

Aufgaben für eine gegebene Grammatik:
(1) Stelle fest, ob ein gegebenes Wort im Sprachschatz ist.
(2) Ermittle (falls es im Sprachschatz ist) den Strukturbaum.

Naive Idee: Wende auf das gegebene Wort Regeln an. Wir erhalten eine Ableitung $w \to w_1 \to w_2 \to w_3 \to \dots$ Fragestellung: ist w im Sprachschatz?

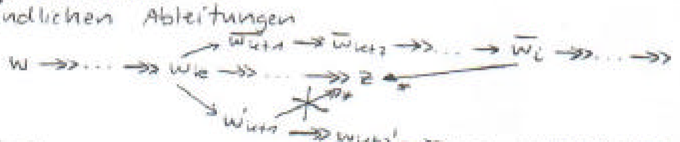
Wir erhalten zwei Möglichkeiten:
(1) Die Ableitung führt zu einem Wort, auf das keine Regel mehr anwendbar ist.
(2) Die Ableitung bricht nicht ab und kann unendlich fortgesetzt werden.

Falls (1) gilt und das Wort, auf das keine Regel mehr anwendbar ist, das Axiom ist, so ist w im Sprachschatz.
Falls (1) gilt und das Wort, auf das keine Regel mehr anwendbar ist, nicht das Axiom ist, dann können wir daraus nicht schließen, dass w <u>nicht</u> im Sprachschatz ist. Wir sprechen von einer <u>Sackgasse</u>.

$w \twoheadrightarrow \dots \twoheadrightarrow w_k \twoheadrightarrow \dots \twoheadrightarrow z$ (w ist im Sprachschatz)
$w_k \twoheadrightarrow w'_{k+1}$ — w'_{k+1} nicht auf Wurzel reduzierbar; Sackgasse.

Eine unendliche Ableitung (Fall (2)) sagt auch nicht darüber aus, ob das Wort im Sprachschatz ist.

Zwei Arten von unendlichen Ableitungen



unendliche Sackgasse

todsicherer, sehr aufwendiger Algorithmus zur Feststellung, ob ein Wort im Sprachschatz ist:

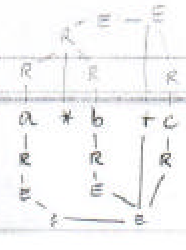
für Ch-1, Ch-2, Ch-3
wenn Wort nicht im Sprachschatz: nach Schritten Suche abbrechen (da wortlängenmonoton)
Für Ch-0 gibt es keinen Algorithmus, der feststellen kann, daß ein Wort <u>nicht</u> im Sprachschatz ist.

<u>Beispiel</u>: Einfache arithmetische Ausdrücke als Ch-2-G. $T = \{a, \dots, z, +, *, (,)\}$
$N = \{R, E\}$ Wurzel E

Regeln:
$a \to R$
$z \to R$
$R*R \to R$
$E+R \to E$
$R \to E$
$(E) \to R$

BNF: <R> ::= a | .. | z
| <R*R> | (<E>)
<E> ::= <E> + <R> | <R>

Strukturbaum:



1. Frage: Hat die Grammatik Sackgassen? Ja! Sackgasse

2. Frage: Ist die Grammatik eindeutig? nein, nicht eindeutig!

Allgemeine Form eines Ausdrucks: $a_1 * a_2 * a_3 * a_4 \dots a_n + b_1 * b_2 \dots +$

R

R ..

Lösung: Vorgegebene Klammerung bei *
(entspr. bei +)

<u>Neue Grammatik</u>, T wie vorher $N = \{F, R, A, E\}$

Faktor Term Additiver Ausdruck Ausdruck (Expression)

$a \to F$

$\vdots$

$z \to F$

$F * R \to R$

$R + A \to A$

$F \to R$

$R \to A$

$A \to E$

$(E) \to F$

Die Grammatik ist eindeutig, aber nicht sackgassenfrei.
Ziel erreicht:
Strukturgraphen eindeutig zugeordnet.
⇒ dadurch werden die + / * - Regeln festgelegt, auch "* vor +"

Sackgassen sind nicht schlimm solange sie vermeidbar sind („Warnungsschild Sackgasse im Straßenverkehr")

$w_0 \twoheadrightarrow w_1 \twoheadrightarrow w_2 \twoheadrightarrow \boxed{\dots} \twoheadrightarrow \dots \twoheadrightarrow w_n$

$w_0 \twoheadrightarrow w_1'$

$w_0 \twoheadrightarrow w_1' \twoheadrightarrow \dots \boxed{w_i' \twoheadrightarrow w_{i+1} \twoheadrightarrow w_{i+2}} \boxed{\dots} \twoheadrightarrow w_n$

$w_0 \twoheadrightarrow w_1' \twoheadrightarrow \dots \boxed{w_i' \twoheadrightarrow w_{i+1}' \twoheadrightarrow w_{i+2}} \dots \twoheadrightarrow w_n$

Jeweils Regeln vertauscht, kommt aber zum gleichen Erg. Wenn so ein „Weg" existiert, dann sind die einzelnen Zeilen strukturell äquivalent

In den Ableitungen für ein Wort über einer Grammatik können wir eine sogenannte Linksableitung (analog Rechtsableitung) auszeichnen. Wir erhalten eine Linksableitung.
Dies ist eine Ableitung, in der jeweils soweit links wie möglich eine Regel angewendet wird. (Linksreduktion)

Achtung: (1) I. d. R. sind Linksableitungen nicht eindeutig.
(2) In der Menge der strukturell äquivalenten Ableitungen sind Linksableitungen eindeutig.

Eine Linksableitung führt für ein Wort im Sprachschatz nicht unbedingt zum Axiom.

Begriff: Akzeptierende Linksableitung: Regeln werden so weit links wie möglich angewendet, Regeln die nicht zum Axiom führen (in eine Sackgasse führen) werden übersprungen.

1.3. Chomsky-3-Sprachen, endl. Automaten, reguläre Ausdrücke

25.4.03 (Fr)

Wir betrachten 3 Beschreibungsmittel für formale Sprachen:
- reguläre Ausdrücke
- endliche Automaten
- Ch-3-G

Wir werden zeigen: Alle 3 Beschreibungsmittel haben die gleiche Beschreibungsmächtigkeit. Sie beschreiben die gleiche Klasse formaler Sprachen, genannt reguläre Sprachen.

1.3.1. Reguläre Ausdrücke (RA)

Gegeben Zeichenmenge T. Die Syntax und Semantik der regulären Ausdrücke (RA) ist wie folgt gegeben:

(1) Einfache reg. Ausdrücke

x mit $x \in T$ ist RA mit Sprache $\{\langle x\rangle\}$
ε —— " —— $\{\varepsilon\}$
$\{\}$ —— " —— $\emptyset$

(2) Zusammengesetzte RA (Seien X und Y RA)

$[X|Y]$ ist RA mit Sprache $L_X \cup L_Y$ (Vereinigung d. Sprachen)
$[XY]$ —— " —— $\{x \cdot y : x \in L_X \wedge y \in L_Y\}$
X^* —— " —— $\{x_1 \cdot \ldots \cdot x_n : n \in \mathbb{N} \wedge x_1, \ldots, x_n \in L_X\}$

wobei L_X die formale Sprache zu RA X sei und L_Y die formale Sprache zu RA Y.

Bsp. $a[ab]^* = \{\langle a\rangle, \langle aab\rangle, \ldots\}$ formale Sprache
$aa[ba]^* = \{\langle aa\rangle, \langle aaba\rangle, \ldots\}$ formale Sprache
Die beiden formalen Sprachen sind unterschiedlich.
$X^+ = XX^*$

Gesetze der RA:

$$
\begin{aligned}
[[X|Y]|Z] &= [X|[Y|Z]] \\
[[XY]Z] &= [X[YZ]] \\
[X|Y] &= [Y|X] && \text{„Kommutativität"} \\
[[X|Y]Z] &= [[XZ]|[YZ]] && \text{„Distributivität"} \\
[Z[X|Y]] &= [[ZX]|[ZY]] \\
\{\}^* &= \varepsilon && \varepsilon \text{ Einselement} \\
X\varepsilon &= X && \{\} \text{ Nullelement} \\
X\{\} &= \{\} \\
X^* &= XX^* | \varepsilon \\
X^* &= [X|\varepsilon]^*
\end{aligned}
$$

1.3.2. Endliche Automaten (EA)

Endlicher Automat $A = (S, T, s_0, S_Z, \delta)$

S endliche Menge von Zuständen (beliebige Menge → auch unendl. Automaten)
T endliche Menge von Eingangszeichen
$s_0 \in S$ Anfangszustand
$S_Z \in S$ Menge der Endzustände
$\delta : S \times (T \cup \{\varepsilon\}) \rightarrow \mathcal{P}(S)$ ← Potenzmenge zu S: Menge der Teilmengen von S δ Übergangsfunktion
Übergänge ohne Eingangszeichen

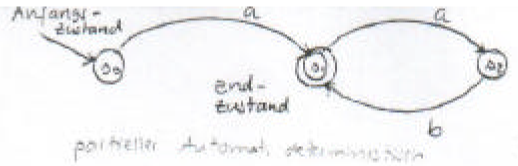
$\delta(s,t)$ ist die Menge der Nachfolgezustände zu $s \in S$, $t \in T \cup \{\varepsilon\}$

Deterministischer Automat: $\delta(s, \varepsilon) \subseteq \{s\}$ d.h. ε-Übergänge gibt es nicht
$|\delta(s,t)| \leq 1$ für alle $s \in S$, $t \in T$
d.h. es gibt höchstens einen Nachfolgezustand.

Partieller Automat: $\exists\, t \in T, s \in S : \delta(s,t) = \emptyset$ [illegible]

ε-freier Automat: $\delta(s, \varepsilon) \subseteq \{s\}$ für alle $s \in S$ d.h. keine spontanen Übergänge

Beispiel: graph. Darstellung



partieller Automat, deterministisch

Übergangstabelle für Zustandsautomaten

δ	ε	a	b
s_0	$\emptyset$	$\{s_1\}$	$\emptyset$
s_1	$\emptyset$	$\{s_2\}$	$\emptyset$
s_2	$\emptyset$	$\emptyset$	$\{s_1\}$

! Notation: wir schreiben $s_1 \xrightarrow{a} s_2$ falls $s_2 \in \delta(s_1, a)$

jedem Wort $w \in T^*$ ordnen wir eine Relation $\xrightarrow{w}{}^*$ auf den Zuständen zu:

$$\xrightarrow{\varepsilon}{}^* = (\xrightarrow{\varepsilon})^*$$

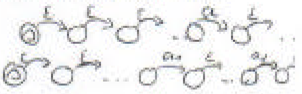
$s \xrightarrow{\varepsilon}{}^* s'$

d.h. es ex. ein ε-Weg nach s', alle Markierungen der Pfeile sind ε

$$\xrightarrow{\langle a \rangle}{}^* = \xrightarrow{\varepsilon}{}^* \circ \xrightarrow{a} \circ \xrightarrow{\varepsilon}{}^*$$

$s \xrightarrow{\langle a \rangle} s'$

$s \xrightarrow{\langle a_1 \dots a_n \rangle} s'$



$$\xrightarrow{w_1 \circ w_2}{}^* = \xrightarrow{w_1}{}^* \circ \xrightarrow{w_2}{}^* \qquad w_1, w_2 \in T^*$$

wir schreiben $\delta^*(s, w)$ für $\{s' : s \xrightarrow{w}{}^* s'\}$

Damit können wir die Sprache charakterisieren, die durch einen endl. Automaten A beschrieben wird:

$$L(A) = \{w \in T^* : \exists z \in S_Z : s_0 \xrightarrow{w}{}^* z\}$$

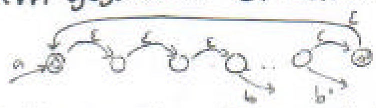
1.3.3. Äquivalenz der Darstellungsformen

Zunächst beantworten wir die Frage, ob Automaten mit spontanen Übergängen beschreibungsmächtiger sind als Automaten ohne spontane Übergänge.

Satz: Zu jedem endl. Automaten existiert ein ε-freier endl. Automat, der die gleiche Sprache akzeptiert

Beweisidee: Wir konstruieren zu einem gegebenen EA einen ε-freien EA mit gleicher Sprache:

(1) $\xrightarrow{\varepsilon}{}^*$-Zyklen:



Alle Knoten im Zyklus äquivalent

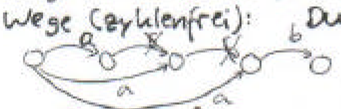
wir fassen Knoten, die durch $\xrightarrow{\varepsilon}{}^*$-Zyklen verbunden sind, zu einem Knoten zus.

Bsp.

Es entsteht ein Automat ohne $\xrightarrow{\varepsilon}{}^*$-Zyklen ($\varepsilon$-Schlingen können weggelassen werden).

(2) $\xrightarrow{\varepsilon}{}^*$-Wege (zyklenfrei): Durchschalten von ε-Übergängen

Bsp.



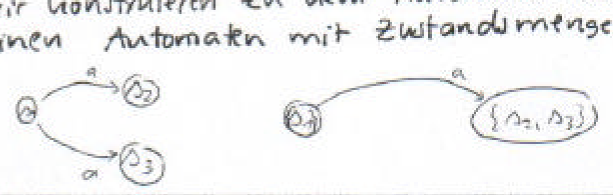
„durchschalten"

Aus $\bigcirc \xrightarrow{a} (s) \xrightarrow{\varepsilon} \bigcirc$ wird

für alle Kanten in den Knoten s.

Satz: Zu jedem ε-freien nichtdet. Automaten existiert ein (ε-freier) det. Automat mit gleichem Sprachschatz

Beweisidee: Wir konstruieren zu dem Automaten mit Zustandsmenge S einen Automaten mit Zustandsmenge $\mathcal{P}(S)$. $\square$



1.3.4. Äquivalenz RA, EA, Ch-3-G

Satz: Zu jedem endl. Automaten existiert ein regulärer Ausdruck, der die gleiche Sprache beschreibt.

Beweis: konstruktiv! Wir geben zu einem gegebenen EA einen RA mit gleicher Sprache.

Zustandsmenge : $S = \{s_1, \dots, s_n\}$

o.B.d.A. sei EA ε-frei.

Wir konstruieren eine Familie formaler Sprachen

$L(i,j,k) \subseteq T^*$

wobei $i,j \in \{1, \dots, n\}$, $k \in \{0, \dots, n\}$

und die dazugehörige RA $E(i,j,k)$ die jeweils die formale Sprache $L(i,j,k)$ beschreiben. Dabei sei

$L(i,j,k) = \{w \in T^* : s_i \xrightarrow[k]{w} s_j\}$.

Dabei betrachten wir in $\xrightarrow[j]{w}$ nur Pfade im Übergangsgraphen mit Zwischenzuständen s_l mit $l \leq k$.

$$L(i,j,0) = \{\langle a \rangle : s_j \in \delta(s_i, a)\}$$

$$E(i,j,0) = \underbrace{a_1 | \dots | a_z}_{\text{Menge aller } a \in T \text{ mit } s_i \xrightarrow{a} s_j}$$

Stern-Operator auf Sprache angewendet

$$L(i,j,k+1) = L(i,j,k) \cup \{u \circ v \circ w : u \in L(i,k+1,k) \wedge v \in L(k+1,k+1,k)^* \wedge w \in L(k+1,j,k)\}$$

Konkatenation

$$E(i,j,k) = E(i,j,k) \mid E(i,k+1,k)\, E(k+1,k+1,k)^*\, E(k+1,j,k)$$

Die Sprache des EA sind die Wege von s_1 (Anfangszustand) zu Endzuständen

$$S_Z = \{s_{z_1}, \dots, s_{z_p}\}.$$

Dies entspricht den RA

$$E(1, z_1, n) \mid \dots \mid E(1, z_p, n) \qquad \square$$