Grundlagen Theoretischer Informatik 2

WiSe 2009/10 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

Grundlagen Theoretischer Informatik 2 Gesamtübersicht

- Organisatorisches; Einführung
- Ersetzungsverfahren: Grammatiken und Automaten
- Rekursionstheorie-Einführung
- Komplexitätstheorie-Einführung

Eine Anwendung von KfG:
Beschreibung der Syntax (grammatisches Gerüst) natürlicher Sprachen

Satz $\rightarrow$ NP VP
NP $\rightarrow$ Artikel Nomen
VP $\rightarrow$ Verb
Artikel $\rightarrow$ “der”
Artikel $\rightarrow$ “die”
Nomen $\rightarrow$ “Hund”
Nomen $\rightarrow$ “Hunde”
Nomen $\rightarrow$ “Katze”
Verb $\rightarrow$ “beißen”

Satz $\Rightarrow$ NP VP $\Rightarrow^2$ Artikel Nomen Verb $\Rightarrow^3$ “die” “Hunde” “beißen”
Satz $\Rightarrow$ NP VP $\Rightarrow^2$ Artikel Nomen Verb $\Rightarrow^3$ “die” “Katze” “beißen”
Satz $\Rightarrow$ NP VP $\Rightarrow^2$ Artikel Nomen Verb $\Rightarrow^3$ “der” “Katze” “beißen”

Eine Anwendung von KfG: **Beschreibung arithmetischer Ausdrcke**

$\Sigma = \{v, -, +, *, /, (,)\}$, $N = \{E\}$.
Die Regeln seien die folgenden:

$E \to (E)$
$E \to -(E)$
$E \to (E + E)$
$E \to (E - E)$
$E \to (E * E)$
$E \to (E/E)$
$E \to v$

Beispielableitung:

$E \Rightarrow -(E)$
$\Rightarrow -(E + E)$
$\Rightarrow -((E * E) + E)$
$\Rightarrow -((-(E) * E) + E)$
$\Rightarrow -((-(E) * E) + (E - E))$
$\overset{*}{\Rightarrow} -((-(v) * v) + (v - v))$

Hinweis: Der *Scanner*, ein endlicher Automat, produziert idealerweise als Ausgabe die Eingabe für den *Parser* zwecks *syntaktischer Analyse* eines Programmtextes.

“Nebensächlichkeiten” wie Zahlen oder Zahlenvariablen werden in einen Statthalter v bersetzt.

Ein Wort hat viele Ableitungen

Linksableitung:

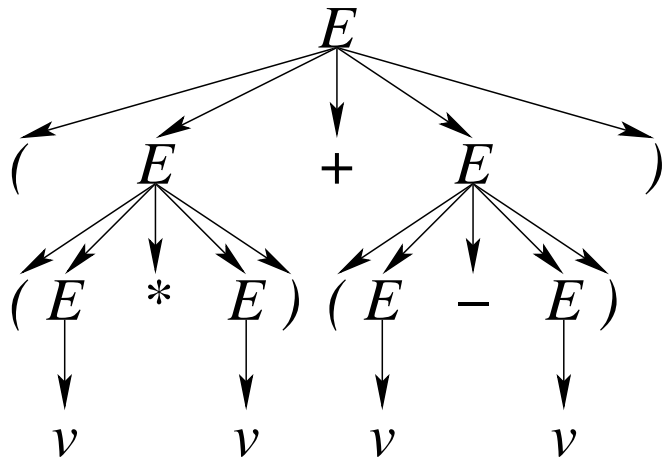
$$
\begin{aligned}
& E \Rightarrow -(E) \\
& \Rightarrow -(E + E) \\
& \Rightarrow -((E * E) + E) \\
& \Rightarrow -((-(E) * E) + E) \\
& \Rightarrow -((-(v) * E) + E) \\
& \Rightarrow -((-(v) * v) + E) \\
& \Rightarrow -((-(v) * v) + (E - E)) \\
& \Rightarrow -((-(v) * v) + (v - E)) \\
& \Rightarrow -((-(v) * v) + (v - v))
\end{aligned}
$$

Rechtsableitung:

$$
\begin{aligned}
& E \Rightarrow -(E) \\
& \Rightarrow -(E + E) \\
& \Rightarrow -(E + (E - E)) \\
& \Rightarrow -(E + (E - v)) \\
& \Rightarrow -(E + (v - v)) \\
& \Rightarrow -((E * E) + (v - v)) \\
& \Rightarrow -((E * v) + (v - v)) \\
& \Rightarrow -((-(E) * v) + (v - v)) \\
& \Rightarrow -((-(v) * v) + (v - v))
\end{aligned}
$$

Ein Syntaxbaum (oder *Ableitungsbaum*) für

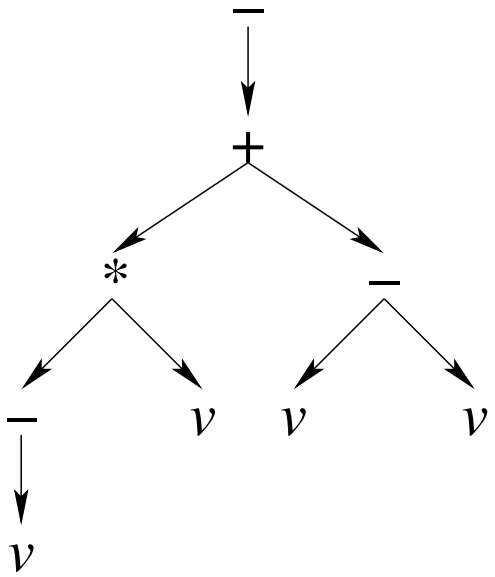
$$E \Rightarrow (E + E) \Rightarrow ((E * E) + E) \Rightarrow ((E * E) + (E - E)) \overset{*}{\Rightarrow} ((v * v) + (v - v))$$



Linksableitung: Tiefensuche mit Linksabstieg durch Syntaxbaum

Rechtsableitung: Tiefensuche mit Rechtsabstieg durch Syntaxbaum

Operatorbaum: ein kompakter Syntaxbaum für das ursprüngliche Beispiel



Ein **Kellerautomat**, engl. Pushdown automaton, ist ein Sextupel $A = (Q, \Sigma, \Gamma, q_0, \Delta, F)$:

- Q ist das *Zustandsalphabet*,
- Σ ist das *Eingabealphabet*,
- Γ ist das *Kelleralphabet*,
- $q_0 \in Q$ ist der *Startzustand (Anfangszustand)*,
- $\Delta \subset (Q \times (\Sigma \cup \{\lambda\}) \times \Gamma^*) \times (Q \times \Gamma^*)$ ist die endl. *Übergangsrelation*,
- $F \subseteq Q$ ist die *Endzustandsmenge*.

Konfigurationsübergänge

Konfiguration: $C \in (Q \times \Sigma^* \times \Gamma^*)$
Die ersten beiden Komponenten entsprechen einer NEA-Konfiguration, die dritte Komponente modelliert den Kellerinhalt.
Für zwei Konfigurationen C_1 und C_2 mit $C_i = (q_i, w_i, \gamma_i)$ definieren wir $C_1 \vdash C_2$ *(Einschrittkonfigurationsübergang)* gdw. es gibt Transitionen $((q_1, a, \alpha_1), (q_2, \alpha_2))$ mit $w_1 = aw_2$, $\gamma_i = \alpha_i\beta$ für ein $\beta \in \Gamma^*$.
Beachte: $a \in \Sigma \cup \{\lambda\}$

$$L(A) = \{w \in \Sigma^* \mid \exists f \in F : (q_0, w, \lambda) \vdash^* (f, \lambda, \lambda)\}$$

Ein Beispiel Betrachte

$$A = (\{q_0, q, f\}, \{a, b\}, \{a\}, \Delta, q_0, \{q_0, f\})$$

mit folgenden Übergängen:

- $((q_0, a, \lambda), (q, a))$
- $((q, a, \lambda), (q, a))$
- $((q, b, a), (f, \lambda))$
- $((f, b, a), (f, \lambda))$

Lemma: $L(A) = \{a^n b^n \mid n \geq 0\} =: L$.

Beweis: Induktion über $k > 0$ zeigt $a^k b^k \in L(A)$: Betrachte hierzu

$$\begin{aligned}(q_0, a^k, \lambda) &\vdash (q, a^{k-1}, a) \vdash^{k-1} (q, \lambda, a^k) \text{ und} \\ (q, b^k, a^k) &\vdash (f, b^{k-1}, a^{k-1}) \vdash^{k-1} (f, \lambda, \lambda)\end{aligned}$$

q_0 und f akzeptieren $\rightsquigarrow L \subseteq L(A)$.

Betrachte umgekehrt $w \in L(A)$.
Induktion über die Länge k einer Ableitung liefert:
Beh. 1: $(q, w, \lambda) \vdash^* (q, \lambda, x) \Rightarrow w = x \in \{a\}^*$
Bew.: $k = 0$: $(q, w, \lambda) \vdash^0 (q, \lambda, x) \rightsquigarrow w = x = \lambda$ ✓
$k > 0$: $(q, w, \lambda)(\vdash \circ \vdash^{k-1})(q, \lambda, x)$ gdw. $(q, w, \lambda) \vdash (q', w', x') \vdash^{k-1} (q, \lambda, x)$.
Inspektion der Übergänge ergibt: $q' = q$ oder $q' = f$.
Im zweiten Fall gibt es keinen Rückweg nach q. $\rightsquigarrow q' = q$ und $aw' = w$, $x' = a$.
IH liefert: $(q, w', a) \vdash^{k-1} (q, \lambda, x) \Rightarrow w' \in \{a\}^*$ und $x = aw'$. $\rightsquigarrow x = w \in \{a\}^*$.

Ähnlich sieht man:
Beh. 2: $(f, w, x) \vdash^* (f, \lambda, y) \Rightarrow \exists k\, w = b^k$ und $x = a^k y$.

Diskutiere $w \in L(A)$. Falls $w \neq \lambda$, so $w = aw'$ und

$$(q_0, aw', \lambda) \vdash (q, w', a)$$

(durch Betrachten der möglichen Übergänge).
Eine weitere Betrachtung offenbart: $w \in L(A)$ heißt $w' = ubv$ mit

$$(q, w', a) \vdash^* (q, bv, au) \vdash (f, v, u) \vdash^* (f, \lambda, \lambda)$$

Gemäß Beh. 1 bedeutet dies $u = a^k$ (für irgendein $k \geq 0$) und
gemäß Beh. 2 $v = b^k$.
$\rightsquigarrow w = a^{k+1}b^{k+1} \rightsquigarrow w \in L$.

Palindrome Betrachte $A = (\{q, f\}, \{a, b\}, \{a, b\}, \Delta, q, \{f\})$ mit Transitionen

- $((q, a, \lambda), (q, a))$
- $((q, b, \lambda), (q, b))$
- $((q, \lambda, \lambda), (f, \lambda))$
- $((f, a, a), (f, \lambda))$
- $((f, b, b), (f, \lambda))$

Lemma: $L(A) = \{ww^R \mid w \in \{a, b\}^*\}$

Kellerautomaten erzeugen kontextfreie Sprachen

Sei $G = (N, \Sigma, P, S)$ eine kfG. Betrachte folgende Transitionen:

- $((s, \lambda, \lambda), (f, S))$,
- für jede Regel $C \to w$: $((f, \lambda, C), (f, w))$,
- für jedes Terminalzeichen a: $((f, a, a), (f, \lambda))$.

f ist der einzige Endzustand und s der Startzustand.

$\rightsquigarrow$ Satz: Jede kontextfreie Sprache ist PDA-Sprache.

Die Konstruktion am Beispiel

$G = (\{S\}, \{a, b\}, P, S)$ mit den Regeln $r_1 = S \to aSb$ und $r_2 = S \to \lambda$.

Der entsprechende Kellerautomat hat folgende Regeln:
$((s, \lambda, \lambda), (f, S))$, $((f, \lambda, S), (f, aSb))$, $((f, \lambda, S), (f, \lambda))$, $((f, a, a), (f, \lambda))$, $((f, b, b), (f, \lambda))$.

Die Ableitung $S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aabb$ wird wie folgt simuliert:
1. Initialisierungsphase:
$(s, aabb, \lambda) \vdash (f, aabb, S)$
2. Simulieren der kfG und 3. Überprüfen der Eingabe:
$(f, aabb, S) \vdash (f, aabb, aSb) \vdash (f, abb, Sb) \vdash (f, abb, aSbb)$
$\vdash (f, bb, Sbb) \vdash (f, bb, bb) \vdash (f, b, b) \vdash (f, \lambda, \lambda)$

Keller als Ersetzungssysteme

Zu Kellerautomat $A = (Q, \Sigma, \Gamma, q_0, \Delta, F)$ definiere, falls (o.E.) $Q \cap \Sigma = \emptyset$:
Ersetzungssystem $ES_A = (X, P)$ mit $X = Q \cup \Sigma \cup \Gamma$ und
Regeln der Form $aqb \to vq'$ gdw. $((q, b, a), (q', v^r)) \in \Delta$.

ES_A akzeptiert mit Anfangszustand q_0 und Endzuständen F die Sprache
$L_a(ES_A) = \{w \in \Sigma^* \mid \exists q_f \in F : q_0 w \stackrel{*}{\Rightarrow} q_f\}$.

Einer von der Anfangskonfiguration (q_0, w, λ) in k Schritten ableitbare Konfiguration (q, x, γ) von A entspricht nun die Tatsache: $q_0 w \Rightarrow^k \gamma^r q x$.

Die Konstruktionen an einem Beispiel

Welche Sprache akzeptiert der folgende Kellerautomat A mit q_f als Endzustand und s als Startzustand ?

$((s, a, \lambda), (s, A))$
$((s, b, \lambda), (q_f, \lambda))$
$((q_f, a, AAA), (q_f, \lambda))$

$L(A) = \{a^{3n}ba^n \mid n \in \mathbb{N}\}$.

Beispielakzeptierung:
$(s, aaaba, \lambda) \vdash (s, aaba, A) \vdash (s, aba, AA) \vdash (s, ba, AAA) \vdash (q_f, a, AAA) \vdash (q_f, \lambda, \lambda)$.

(Akzeptierendes!) Ersetzungssystem arbeitet wie folgt:
$saaaba \Rightarrow Asaaba \Rightarrow AAsaba \Rightarrow AAAsba \Rightarrow AAAq_fa \Rightarrow q_f$

Von Automaten zu Grammatiken

Schritt 1: (auf dem Weg von Automaten zu Grammatiken)
Durch Einfügen von *Zwischenzuständen* klar:
ES_A-Regeln der Form $aqb \to vq'$, $a \in \Gamma \cup \{\lambda\}$ (!!!), erfüllen o.E.
$|v| = 0$ oder $v = ac$ für ein Kellerzeichen c.
Interpretation: Kellerautomaten machen stets nur ein einzelnes Pop oder Push (*Push-Pop-Normalform*) auf dem Keller. Beachte: $b = \lambda$ möglich!
Satz: Jede PD-Sprache wird durch einen Kellerautomaten in Push-Pop-NF akzeptiert.
Hinweis: Konstruktion ähnelt dem Weg zur Chomsky NF!
Einzelheiten sollten am Beispiel klar werden.

Die Konstruktionen an einem Beispiel:

Problem: Einlesen von ganzen Wörtern vom Keller

Lösung: Führe Zwischenzustände ein, die nicht akzeptieren:

$((s, a, \lambda), (s, A))$

$((s, b, \lambda), (q_f, \lambda))$

$((q_f, \lambda, A), (f_1, \lambda))$

$((f_1, \lambda, A), (f_2, \lambda))$

$((f_2, a, A), (q_f, \lambda))$

Beispielakzeptierung: $(s, aaaba, \lambda) \vdash (s, aaba, A) \vdash (s, aba, AA) \vdash (s, ba, AAA) \vdash (q_f, a, AAA) \vdash (f_1, a, AA) \vdash (f_2, a, A) \vdash (q_f, \lambda, \lambda)$.

Von Automaten zu Grammatiken Schritt 2:

Ein *Kellerbodenzeichen* (Symbol $\triangleleft$) ist ein spezielles Kellerzeichen, das nur einmal während eines Akzeptierungslaufs, nämlich eingangs, als Markierung des Kellerbodens eingeführt werden darf und nur einmal, als letzter Schritt eines Akzeptierungslaufs, wieder gelöscht werden darf. (Zwischenzeitliches "Testen" soll aber möglich sein!)
Konstruktion: Führe als neuen Startzustand S und als neuen (einzigen) Endzustand f ein. Neue Regeln (neben den alten):
$(S, \lambda, \lambda) \rightarrow (q_0, \triangleleft)$, $(q_f, \lambda, \triangleleft) \rightarrow (f, \lambda)$.
Automatisch erfüllt: die letzte Keller-OP ist Pop; der Endzustand wird nicht mehr verlassen.

Satz: Jede PD-Sprache wird durch einen Kellerautomaten mit Kellerbodenzeichen akzeptiert (*KBZ-NF*).

Die Konstruktionen an einem Beispiel:

Problem: Sonderrolle von Start- und Endzuständen

$(S, \lambda, \lambda) \rightarrow (s, \triangleleft)$

$((s, a, \lambda), (s, A))$

$((s, b, \lambda), (q_f, \lambda))$

$((q_f, \lambda, A), (f_1, \lambda))$

$((f_1, \lambda, A), (f_2, \lambda))$

$((f_2, a, A), (q_f, \lambda))$

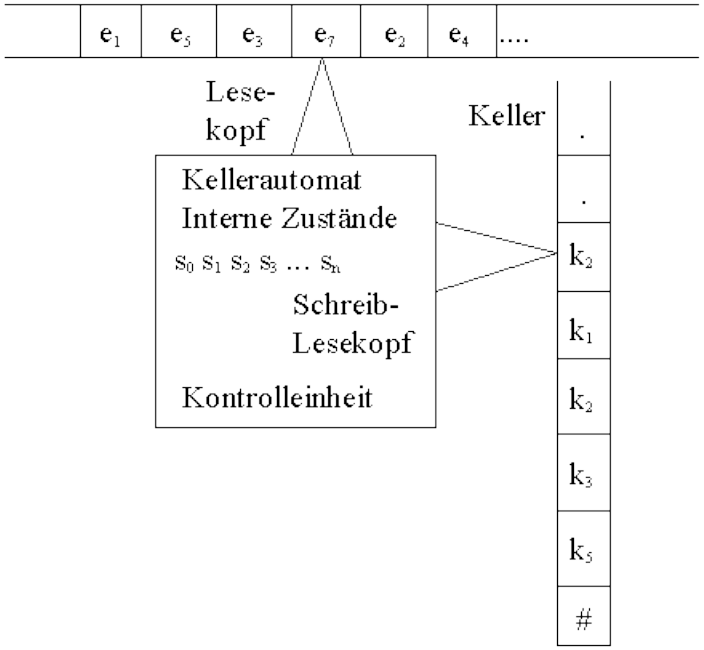
$(q_f, \lambda, \triangleleft) \rightarrow (f, \lambda)$

Beispielakzeptierung: $(S, aaaba, \lambda) \vdash (s, aaaba, \triangleleft) \vdash$
$(s, aaba, A\triangleleft) \vdash (s, aba, AA\triangleleft) \vdash (s, ba, AAA\triangleleft) \vdash (q_f, a, AAA\triangleleft) \vdash$
$(f_1, a, AA\triangleleft) \vdash (f_2, a, A\triangleleft) \vdash (q_f, \lambda, \triangleleft) \vdash (f, \lambda, \lambda).$

Von Automaten zu Grammatiken Ein weiteres technisches Detail:
Wir können nun davon ausgehen, dass außer beim allerersten Male bei jedem Befehl immer das oberste Kellerzeichen “angeschaut” wird. Evtl. ist das eben das Kellerbodenzeichen.
Das ändert nicht die Konfigurationsfolge einer Beispielakzeptierung, wohl aber formal den Automaten:
$(S, \lambda, \lambda) \to (s, \triangleleft)$
$((s, a, x), (s, Ax))$ für $x \in \Gamma$
$((s, b, x), (q_f, x))$ für $x \in \Gamma$
$((q_f, \lambda, A), (f_1, \lambda))$
$((f_1, \lambda, A), (f_2, \lambda))$
$((f_2, a, A), (q_f, \lambda))$
$(q_f, \lambda, \triangleleft) \to (f, \lambda)$

Mehr zur KBZ-NF

Diese entspricht der wohl häufigsten Definition von Kellerautomaten in der Literatur. Dazu passt folgendes Bild zur Anschauung (mit KBZ #):



Leerkellerakzeptanz

Diese ist allgemein bei Kellerautomaten A wie folgt definiert:

$$L_{es}(A) = \{w \in \Sigma^* \mid \exists q \in Q : (q_0, w, \lambda) \vdash^+ (q, \lambda, \lambda)\}$$

Klar: Jede solche Sprache $L_{es}(A)$ ist PDA-Sprache.

Es gilt aber auch die Umkehrung!
Beobachte dazu Kellerautomat in KBZ-NF:
Sein Keller ist nur ganz am Anfang leer und dann erst wieder nach Beendigung einer erfolgreichen Akzeptanz.

Satz: Die PDA-Sprachen sind genau die Sprachen, die von Kellerautomaten mit Leerkellerakzeptanz akzeptiert werden.

Deterministische Kellerautomaten

Ein Kellerautomat in KBZ-NF heißt *deterministisch*, wenn in jedem Zustand q und für jedes Eingabezeichen b und jedes Kellerzeichen a gilt:

$$|(\{q\} \times \{b, \lambda\} \times \{a, \lambda\}) \times (Q \times (\Gamma \cup \{\lambda\})) \cap \Delta| \leq 1$$

Es gibt also höchstens eine Möglichkeit für den Kellerautomaten, in einer vorliegenden Konfiguration fortzufahren.

Nach Definition gilt offenbar [?!] (durch Simulation eines Kellerautomaten):
Satz: Das Wortproblem für deterministische PDA kann in Linearzeit entschieden werden.

Dies macht DPDA für Anwendungen (z.B. Compilerbau) sehr interessant.
Hinweis: Nicht alle PDA-Sprachen lassen sich durch DPDAs akzeptieren.

Von Automaten zu Grammatiken

Schritt 3: *Tripelkonstruktion* (Produktmenge als neues Teilalphabet).
Idee: Simuliere NEA-Komponente eines Kellerautomatens auf dem Keller durch **Raten geeigneter Zwischenzustände**. Ausgangspunkt: KBZ-NF.
Betrachte neues Alphabet $\Gamma' = \Gamma \cup (Q \times \Gamma \times Q)$, $X' = \Gamma' \cup \Sigma \cup \{S\}$.
Neues Ersetzungssystem $ES'_A = (X', P')$ mit Regeln
(1) $S \to (q_0, \triangleleft, f)$
(2) $(q, a, \hat{q})b \to (q'', a, \hat{q})(q', c, q'')$ für $q'', \hat{q}$ bel., mit $((q, b, a), (q', ca)) \in \Delta$ (Push)
(3) $(q, a, q')b \to \lambda$ für beliebige $a \in \Gamma \cup \{\lambda\}$ falls $((q, b, a), (q', \lambda)) \in \Delta$ (Pop)
Das gilt insbesondere für Regeln der Form $((q_f, \lambda, \triangleleft), (f, \lambda))$.
Jetzt sei $L'_a(ES'_A) = \{w \in \Sigma^* \mid Sw \stackrel{*}{\Rightarrow} \lambda\}$.
Hieraus wird (wieder) leicht eine Grammatik G:
Ersetzungsregeln (1) $S \to ((q, \triangleleft, f)b)^r$, (2) $(q, a, \hat{q}) \to ((q'', a, \hat{q})(q', c, q'')b)^r$,
(3) $(q, a, q') \to b$ ersetzen die soeben gelisteten.
$\rightsquigarrow L(A) = L(G)$.

Die Konstruktionen an einem Beispiel

Regeln des neuen Ersetzungssystems:

$S \to (s, \triangleleft, f)$

$(s, x, \hat{q})a \to (q', x, \hat{q})(s, A, q')$ für $x \in \Gamma, q', \hat{q} \in Q$

$(s, x, \hat{q})b \to (q_f, x, \hat{q})$ für $x \in \Gamma, \hat{q} \in Q$

$(q_f, A, f_1) \to \lambda$

$(f_1, A, f_2) \to \lambda$

$(f_2, A, q_f)a \to \lambda$

$(q_f, \triangleleft, f) \to \lambda$

Beispielableitung: $Saaaba \Rightarrow (s, \triangleleft, f)aaaba$

$\Rightarrow (q_f, \triangleleft, f)(s, A, q_f)aaba \Rightarrow (q_f, \triangleleft, f)(f_2, A, q_f)(s, A, f_2)aba$

$\Rightarrow (q_f, \triangleleft, f)(f_2, A, q_f)(f_1, A, f_2)(s, A, f_1)ba$

$\Rightarrow (q_f, \triangleleft, f)(f_2, A, q_f)(f_1, A, f_2)(q_f, A, f_1)a$

$\Rightarrow (q_f, \triangleleft, f)(f_2, A, q_f)(f_1, A, f_2)a$

$\Rightarrow (q_f, \triangleleft, f)(f_2, A, q_f) \Rightarrow (q_f, \triangleleft, f) \Rightarrow \lambda$

Die Konstruktionen an einem Beispiel

Regeln der kontextfreien Grammatik:

$S \to (s, \triangleleft, f)$

$(s, x, \hat{q}) \to a(s, A, q')(q', x, \hat{q})$ für $x \in \Gamma, q', \hat{q} \in Q$

$(s, x, \hat{q}) \to b(q_f, x, \hat{q})$ für $x \in \Gamma, \hat{q} \in Q$

$(q_f, A, f_1) \to \lambda$

$(f_1, A, f_2) \to \lambda$

$(f_2, A, q_f) \to a$

$(q_f, \triangleleft, f) \to \lambda$

Beispielableitung (Linksableitung!): $S \Rightarrow (s, \triangleleft, f)$

$\Rightarrow a(s, A, q_f)(q_f, \triangleleft, f) \Rightarrow aa(s, A, f_2)(f_2, A, q_f)(q_f, \triangleleft, f)$

$\Rightarrow aaa(s, A, f_1)(f_1, A, f_2)(f_2, A, q_f)(q_f, \triangleleft, f)$

$\Rightarrow aaab(f_2, A, q_f)(f_1, A, f_2)(q_f, A, f_1)(q_f, \triangleleft, f)$

$\Rightarrow aaaba(f_1, A, f_2)(q_f, A, f_1)(q_f, \triangleleft, f)$

$\Rightarrow aaaba(q_f, A, f_1)(q_f, \triangleleft, f) \Rightarrow aaaba(q_f, \triangleleft, f) \Rightarrow aaaba$

Zusammenfassung

Satz: Die Typ-2 Sprachen sind genau die PDA-Sprachen.

Wir haben also eine Automaten-Kennzeichnung von $\mathcal{L}_2$ gefunden.