

Script generated by TTT

Title: Seidl: Theoretische_Informatik
(13.05.2013)

Date: Mon May 13 10:22:47 CEST 2013

Duration: 83:25 min

Pages: 94

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L :=$$

Es gilt sogar (Übung!):

Fakt 2.60

Alle Quotientenautomaten A/\equiv_A für die gleiche Sprache $L(A)$ haben die gleiche Struktur, d.h. sie unterscheiden sich nur durch eine Umbenennung der Zustände.

Daher beschriften wir die Zustände des kanonischen Minimalautomaten für eine Sprache L mit \equiv_L Äquivalenzklassen.

Beispiel 2.61

Sei $L := \{w \in \{0,1\}^* \mid w \text{ endet mit } 00\}$.

Die einzigen drei \equiv_L Äquivalenzklassen sind:

$$[\epsilon]_{\equiv_L} = \{w \mid w \text{ endet nicht mit } 0\}$$

$$[0]_{\equiv_L} = \{w \mid w \text{ endet mit } 0, \text{ aber nicht mit } 00\}$$

$$[00]_{\equiv_L} = \{w \mid w \text{ endet mit } 00\}$$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^* / \equiv_L)$$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma,$$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert

$$\begin{array}{l} w'a \equiv_L wa \\ \uparrow \\ w \equiv_L w \end{array}$$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.
Dann gilt offensichtlich $L(M_L) = L$.

$$w \in L \Rightarrow \hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L} \in F_L$$

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.
Dann gilt offensichtlich $L(M_L) = L$.

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.
Dann gilt offensichtlich $L(M_L) = L$.

Satz 2.63 (Myhill-Nerode)

Eine Sprache $L \subseteq \Sigma^$ ist genau dann regulär, wenn \equiv_L endlich viele Äquivalenzklassen hat.*

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.
Dann gilt offensichtlich $L(M_L) = L$.

Satz 2.63 (Myhill-Nerode)

Eine Sprache $L \subseteq \Sigma^$ ist genau dann regulär, wenn \equiv_L endlich viele Äquivalenzklassen hat.*

Beweis:

„ \implies “: Ist L regulär, so wird L von einem DFA A akzeptiert.

Wie sieht M_L aus wenn L nicht regulär ist?

Definition 2.62 (Kanonischer Minimalautomat)

$$M_L := (\Sigma^*/\equiv_L, \Sigma, \delta_L, [\epsilon]_{\equiv_L}, F_L)$$

mit $\delta_L([w]_{\equiv_L}, a) := [wa]_{\equiv_L}$ und $F_L := \{[w]_{\equiv_L} \mid w \in L\}$.

Man sieht: δ_L ist wohldefiniert und $\hat{\delta}_L([\epsilon]_{\equiv_L}, w) = [w]_{\equiv_L}$.
Dann gilt offensichtlich $L(M_L) = L$.

Satz 2.63 (Myhill-Nerode)

Eine Sprache $L \subseteq \Sigma^$ ist genau dann regulär, wenn \equiv_L endlich viele Äquivalenzklassen hat.*

Beweis:

„ \implies “: Ist L regulär, so wird L von einem DFA A akzeptiert.
Daher hat \equiv_L so viele Äquivalenzklassen wie A/\equiv_A Zustände.
„ \impliedby “: Hat \equiv_L endlich viele Äquivalenzklasse,

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

$$\rightarrow [\epsilon] \xrightarrow{0} [0]$$

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

$$\begin{array}{cccccccc} \rightarrow [\epsilon] & \xrightarrow{0} & [0] & \xrightarrow{0} & [0^2] & \xrightarrow{0} \dots \xrightarrow{0} & [0^i] & \xrightarrow{0} \dots \\ & & \downarrow 1 & & \downarrow 1 & & & \\ & & [01] & & [0^2 1] & & & \end{array}$$

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

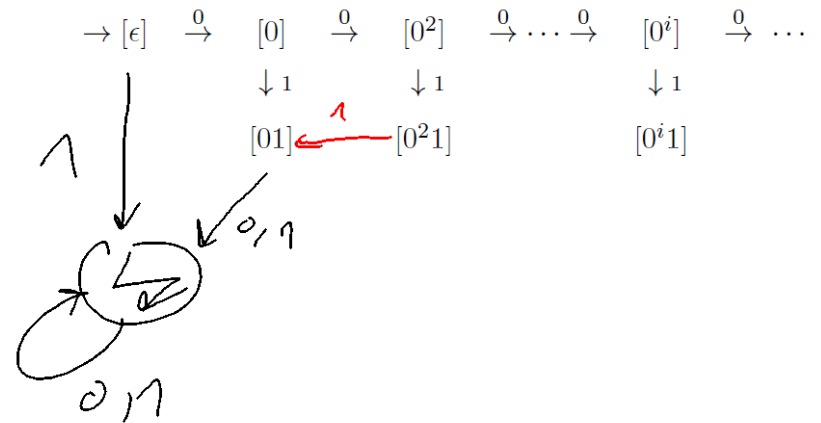
$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

$$\begin{array}{cccccccc} \rightarrow [\epsilon] & \xrightarrow{0} & [0] & \xrightarrow{0} & [0^2] & \xrightarrow{0} \dots \xrightarrow{0} & [0^i] & \xrightarrow{0} \dots \\ & & \downarrow 1 & & & & & \\ & & [01] & & & & & \end{array}$$

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

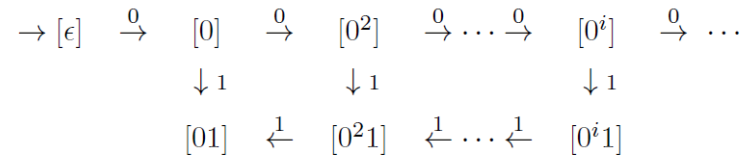
$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$



Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

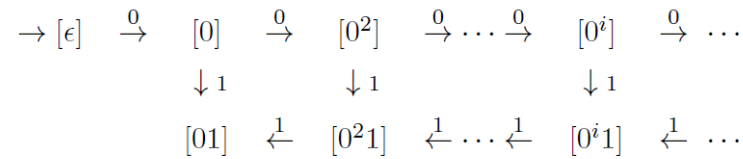
$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$



Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

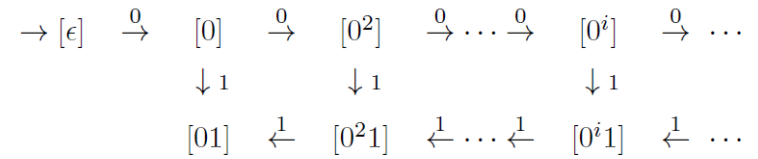


Warum $0^i \not\equiv_L 0^j$ falls $i \neq j$?

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

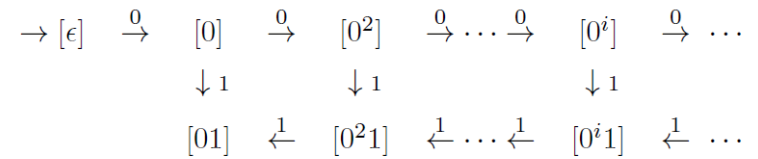
$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$



Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$



Warum $0^i \not\equiv_L 0^j$ falls $i \neq j$?

Was fehlt noch?

Vollständige Methode um Nichtregularität von L zu zeigen:

Wie sieht M_L aus wenn L nicht regulär ist?

Beispiel 2.64

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

$$\begin{array}{cccccccc} \rightarrow [\epsilon] & \xrightarrow{0} & [0] & \xrightarrow{0} & [0^2] & \xrightarrow{0} \dots \xrightarrow{0} & [0^i] & \xrightarrow{0} \dots \\ & & \downarrow 1 & & \downarrow 1 & & \downarrow 1 & \\ & & [01] & \xleftarrow{1} & [0^2 1] & \xleftarrow{1} \dots \xleftarrow{1} & [0^i 1] & \xleftarrow{1} \dots \end{array}$$

Warum $0^i \not\equiv_L 0^j$ falls $i \neq j$?

Was fehlt noch?

Vollständige Methode um **Nichtregularität** von L zu zeigen:

Gib **unendliche** Menge w_1, w_2, \dots an mit $w_i \not\equiv_L w_j$ falls $i \neq j$.

Knobelaufgabe:

Nach Def. gilt $p \not\equiv_A q$ gdw $\exists w. \hat{\delta}(p, w) \in F \not\equiv \hat{\delta}(q, w) \in F$

Man zeige: Falls $p \not\equiv_A q$, dann gibt es ein w der Länge $< |Q|$, das p und q unterscheidet, d.h. $\hat{\delta}(p, w) \in F \not\equiv \hat{\delta}(q, w) \in F$.

Bemerkung

Eindeutigkeit des minimalen Automaten (modulo Umbenennung der Zustände) gilt nur bei DFAs, nicht bei NFAs!

Knobelaufgabe:

Nach Def. gilt $p \not\equiv_A q$ gdw $\exists w. \hat{\delta}(p, w) \in F \not\equiv \hat{\delta}(q, w) \in F$

Man zeige: Falls $p \not\equiv_A q$, dann gibt es ein w der Länge $< |Q|$, das p und q unterscheidet, d.h. $\hat{\delta}(p, w) \in F \not\equiv \hat{\delta}(q, w) \in F$.

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.



Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:
 - $\cup, \cap, \dots, R, \dots$
 - Für welche Darstellung wie teuer? (Komplement, Produkt, ...)

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:
 - $\cup, \cap, \dots, R, \dots$
 - Für welche Darstellung wie teuer? (Komplement, Produkt, ...)
 - NFA kompakt aber oft teuer; DFA oft billiger
- Entscheidungsprobleme auf Automaten und REs:

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:
 - $\cup, \cap, \dots, R, \dots$
 - Für welche Darstellung wie teuer? (Komplement, Produkt, ...)
 - NFA kompakt aber oft teuer; DFA oft billiger

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:
 - $\cup, \cap, \dots, R, \dots$
 - Für welche Darstellung wie teuer? (Komplement, Produkt, ...)
 - NFA kompakt aber oft teuer; DFA oft billiger
- Entscheidungsprobleme auf Automaten und REs:
 - Direkt entscheidbar?
Auf Automat? (Oft mit Erreichbarkeit) Auf RE?

Rückblick reguläre Sprachen

- DFA, NFA, ϵ -NFA und RE definieren die gleiche Sprachklasse.
 - Potenzmengenkonstruktion (exponentiell)
 - Strukturelle Übersetzung von RE nach ϵ -NFA
 - Übersetzung NFA nach RE, zB über Gleichungssysteme (exponentiell)
- Regularitätserhaltende Konstruktionen auf Sprachen bzw Automaten:
 - $\cup, \cap, \dots, R, \dots$
 - Für welche Darstellung wie teuer? (Komplement, Produkt, ...)
 - NFA kompakt aber oft teuer; DFA oft billiger
- Entscheidungsprobleme auf Automaten und REs:
 - Direkt entscheidbar?
Auf Automat? (Oft mit Erreichbarkeit) Auf RE?
 - Reduzierbar auf anderes Problem?
ZB $L(A) \subseteq L(B)$ auf $L(A) \cap \overline{L(B)} = \emptyset$

- Pumping-Lemma
- Äquivalenzen \equiv zw REs:
 - Standardregeln: Kommutativität etc, Beweis mit $L(\cdot)$

- Pumping-Lemma

- Pumping-Lemma
- Äquivalenzen \equiv zw REs:
 - Standardregeln: Kommutativität etc, Beweis mit $L(\cdot)$
 - $\alpha \equiv \beta$ entscheidbar da $L(\alpha) = L(\beta)$ über Automaten entscheidbar
 - Auch für REs mit Variablen entscheidbar: betrachte Variablen als Konstanten
 - Gleichungssysteme lösbar durch Variablenelimination mit Ardens Lemma

$$x \equiv \alpha_1 \gamma_1 \mid \dots \mid \alpha_n \gamma_n$$

- Pumping-Lemma
- Äquivalenzen \equiv zw REs:
 - Standardregeln: Kommutativität etc, Beweis mit $L(\cdot)$
 - $\alpha \equiv \beta$ entscheidbar da $L(\alpha) = L(\beta)$ über Automaten entscheidbar
 - Auch für REs mit Variablen entscheidbar: betrachte Variablen als Konstanten
 - Gleichungssysteme lösbar durch Variablenelimination mit Ardens Lemma
- Minimierung

- Pumping-Lemma
- Äquivalenzen \equiv zw REs:
 - Standardregeln: Kommutativität etc, Beweis mit $L(\cdot)$
 - $\alpha \equiv \beta$ entscheidbar da $L(\alpha) = L(\beta)$ über Automaten entscheidbar
 - Auch für REs mit Variablen entscheidbar: betrachte Variablen als Konstanten
 - Gleichungssysteme lösbar durch Variablenelimination mit Ardens Lemma
- Minimierung
 - Algorithmen zur Kollabierung
 - Relationen \equiv_A und \equiv_L
 - Quotientenautomat A/\equiv_A minimal, da gleichgroß wie kanonischer minimaler Automat $M_{L(A)}$
 - L genau dann regulär wenn \equiv_L endlich viele Klassen hat, dh wenn M_L endlich ist (Myhill-Nerode).

- Pumping-Lemma
- Äquivalenzen \equiv zw REs:
 - Standardregeln: Kommutativität etc, Beweis mit $L(\cdot)$
 - $\alpha \equiv \beta$ entscheidbar da $L(\alpha) = L(\beta)$ über Automaten entscheidbar
 - Auch für REs mit Variablen entscheidbar: betrachte Variablen als Konstanten
 - Gleichungssysteme lösbar durch Variablenelimination mit Ardens Lemma
- Minimierung
 - Algorithmen zur Kollabierung
 - Relationen \equiv_A und \equiv_L

3. Kontextfreie Sprachen

3. Kontextfreie Sprachen

3.1 Kontextfreie Grammatiken

3. Kontextfreie Sprachen

3.1 Kontextfreie Grammatiken

Beispiel 3.1 (Arithmetische Ausdrücke)

- $\langle \text{Expr} \rangle \rightarrow \langle \text{Term} \rangle$
- $\langle \text{Expr} \rangle \rightarrow \langle \text{Expr} \rangle + \langle \text{Term} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Factor} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Term} \rangle * \langle \text{Factor} \rangle$
- $\langle \text{Factor} \rangle \rightarrow a$
- $\langle \text{Factor} \rangle \rightarrow (\langle \text{Expr} \rangle)$

3. Kontextfreie Sprachen

3.1 Kontextfreie Grammatiken

Beispiel 3.1 (Arithmetische Ausdrücke)

- $\langle \text{Expr} \rangle \rightarrow \langle \text{Term} \rangle$
- $\langle \text{Expr} \rangle \rightarrow \langle \text{Expr} \rangle + \langle \text{Term} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Factor} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Term} \rangle * \langle \text{Factor} \rangle$
- $\langle \text{Factor} \rangle \rightarrow a$
- $\langle \text{Factor} \rangle \rightarrow (\langle \text{Expr} \rangle)$

Eine (Links)Ableitung:

$\langle \text{Expr} \rangle \rightarrow$  $\rightarrow a * (a + a)$

3. Kontextfreie Sprachen

3.1 Kontextfreie Grammatiken

Beispiel 3.1 (Arithmetische Ausdrücke)

- $\langle \text{Expr} \rangle \rightarrow \langle \text{Term} \rangle$
- $\langle \text{Expr} \rangle \rightarrow \langle \text{Expr} \rangle + \langle \text{Term} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Factor} \rangle$
- $\langle \text{Term} \rangle \rightarrow \langle \text{Term} \rangle * \langle \text{Factor} \rangle$
- $\langle \text{Factor} \rangle \rightarrow a$
- $\langle \text{Factor} \rangle \rightarrow (\langle \text{Expr} \rangle)$

Eine (Links)Ableitung:

$\langle \text{Expr} \rangle \rightarrow \langle \text{Term} \rangle$ $\rightarrow a * (a + a)$

Bemerkungen

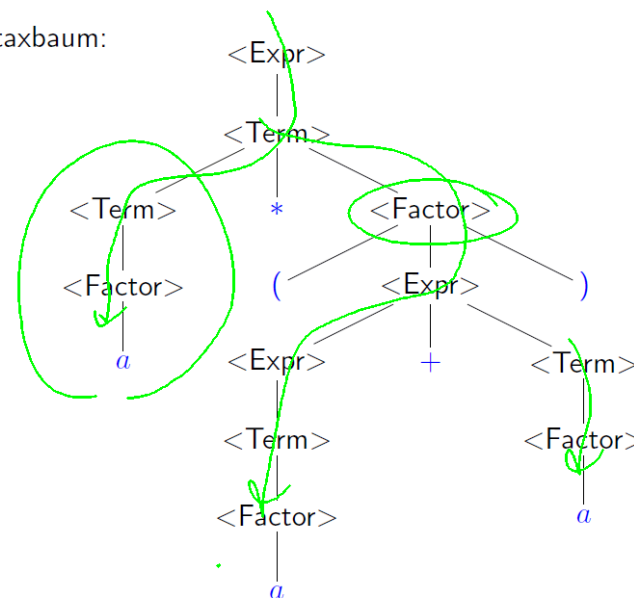
- Der vollständige Syntaxbaum enthält die gesamte Information über die Ableitung, bis auf die (irrelevante) Reihenfolge des Aufbaus.

Bemerkungen

- Der vollständige Syntaxbaum enthält die gesamte Information über die Ableitung, bis auf die (irrelevante) Reihenfolge des Aufbaus.
- Kontextfreie Grammatiken dienen zur Spezifikation von Sprachen. Ihre Implementierung ist das Parsen:

$$a * (a + a)$$

Der Syntaxbaum:



Die Blätter des Baums, von links nach rechts gelesen, ergeben das abgeleitete Wort

Bemerkungen

- Der vollständige Syntaxbaum enthält die gesamte Information über die Ableitung, bis auf die (irrelevante) Reihenfolge des Aufbaus.
- Kontextfreie Grammatiken dienen zur Spezifikation von Sprachen. Ihre Implementierung ist das Parsen:
 - Die Überprüfung, ob ein Wort von einer Grammatik abgeleitet werden kann, bzw

Bemerkungen

- Der vollständige Syntaxbaum enthält die gesamte Information über die Ableitung, bis auf die (irrelevante) Reihenfolge des Aufbaus.
- Kontextfreie Grammatiken dienen zur Spezifikation von Sprachen. Ihre Implementierung ist das **Parsen**:
 - Die Überprüfung, ob ein Wort von einer Grammatik abgeleitet werden kann, bzw
 - Die Erzeugung des Syntaxbaums (*parse tree*).

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

Bemerkungen

- Der vollständige Syntaxbaum enthält die gesamte Information über die Ableitung, bis auf die (irrelevante) Reihenfolge des Aufbaus.
- Kontextfreie Grammatiken dienen zur Spezifikation von Sprachen. Ihre Implementierung ist das **Parsen**:
 - Die Überprüfung, ob ein Wort von einer Grammatik abgeleitet werden kann, bzw
 - Die Erzeugung des Syntaxbaums (*parse tree*).

Parsen ist die Transformation eines Worts in einen Syntaxbaum.

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,

$P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und

(A, aAb)

$A \rightarrow aAb$

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,

$P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,

$P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und

$S \in V$ ist das Startsymbol.

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

- V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),
- Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,
- $P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und
- $S \in V$ ist das Startsymbol.

Konventionen:

- A, B, C, \dots sind Nichtterminale,

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

- V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),
- Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,
- $P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und
- $S \in V$ ist das Startsymbol.

Konventionen:

- A, B, C, \dots sind Nichtterminale,
- a, b, c, \dots (und Sonderzeichen wie $+, *, \dots$) sind Terminale,
- $\alpha, \beta, \gamma, \dots \in (V \cup \Sigma)^*$

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

- V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),
- Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,
- $P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und
- $S \in V$ ist das Startsymbol.

Konventionen:

- A, B, C, \dots sind Nichtterminale,
- a, b, c, \dots (und Sonderzeichen wie $+, *, \dots$) sind Terminale,

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

- V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),
- Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,
- $P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und
- $S \in V$ ist das Startsymbol.

Konventionen:

- A, B, C, \dots sind Nichtterminale,
- a, b, c, \dots (und Sonderzeichen wie $+, *, \dots$) sind Terminale,
- $\alpha, \beta, \gamma, \dots \in (V \cup \Sigma)^*$
- Produktionen schreiben wir $A \rightarrow \alpha$ statt $(A, \alpha) \in P$.

$u, v, w, x, y, z \in \Sigma^*$

Definition 3.2

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ ist ein 4-Tupel:

V ist eine endlichen Menge, die Nichtterminalzeichen (oder Variablen),

Σ ist ein Alphabet, die Terminalzeichen, disjunkt von V ,

$P \subseteq V \times (V \cup \Sigma)^*$ eine endlichen Menge, die Produktionen, und

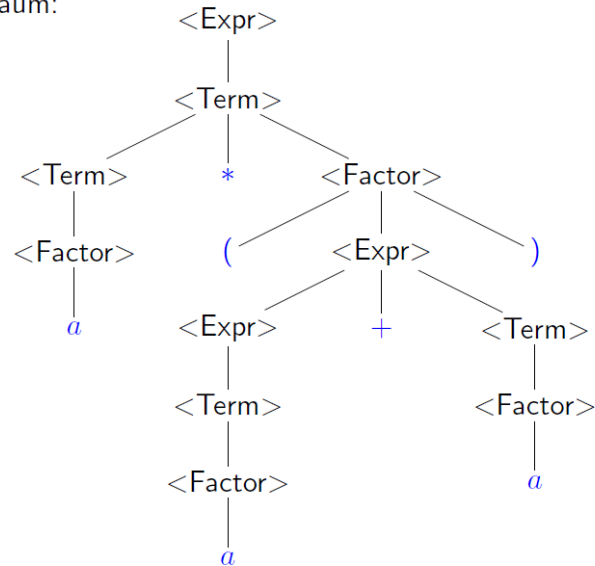
$S \in V$ ist das Startsymbol.

Konventionen:

- A, B, C, \dots sind Nichtterminale,
- a, b, c, \dots (und Sonderzeichen wie $+, *, \dots$) sind Terminale,
- $\alpha, \beta, \gamma, \dots \in (V \cup \Sigma)^*$
- Produktionen schreiben wir $A \rightarrow \alpha$ statt $(A, \alpha) \in P$.
- Statt $A \rightarrow \alpha_1, A \rightarrow \alpha_2, A \rightarrow \alpha_3$ schreiben wir einfach

$$A \rightarrow (\alpha_1 \mid \alpha_2 \mid \alpha_3)^*$$

Der Syntaxbaum:



Die Blätter des Baums, von links nach rechts gelesen, ergeben das abgeleitete Wort

Beispiel 3.3 (Arithmetische Ausdrücke)

$$V = \{E, T, F\}$$

Beispiel 3.3 (Arithmetische Ausdrücke)

$$V = \{E, T, F\}$$

$$\Sigma = \{a, +, *, (,)\}$$

$$P =$$

$$\left\{ \begin{array}{l} E \rightarrow T \mid E + T \\ T \rightarrow F \mid T * F \\ F \rightarrow a \mid (E) \end{array} \right\}$$

Definition 3.4

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ induziert eine **Ableitungsrelation** \rightarrow_G auf Wörtern über $V \cup \Sigma$:

$$\alpha \rightarrow_G \beta$$

Definition 3.4

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ induziert eine **Ableitungsrelation** \rightarrow_G auf Wörtern über $V \cup \Sigma$:

$$\alpha \rightarrow_G \beta$$

gdw es eine Regel $A \rightarrow \gamma$ in P gibt, und Wörter α_1, α_2 , so dass

$$\alpha = \alpha_1 A \alpha_2$$

Definition 3.4

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ induziert eine **Ableitungsrelation** \rightarrow_G auf Wörtern über $V \cup \Sigma$:

$$\alpha \rightarrow_G \beta$$

gdw es eine Regel $A \rightarrow \gamma$ in P gibt, und Wörter α_1, α_2 , so dass

$$\alpha = \alpha_1 A \alpha_2 \quad \text{und} \quad \beta = \alpha_1 \gamma \alpha_2$$

Definition 3.4

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ induziert eine **Ableitungsrelation** \rightarrow_G auf Wörtern über $V \cup \Sigma$:

$$\alpha \rightarrow_G \beta$$

gdw es eine Regel $A \rightarrow \gamma$ in P gibt, und Wörter α_1, α_2 , so dass

$$\alpha = \alpha_1 A \alpha_2 \quad \text{und} \quad \beta = \alpha_1 \gamma \alpha_2$$

Beispiel:

$$a + T + a \rightarrow_G a + T * F + a$$

Definition 3.5 (Reflexive transitive Hülle)

$$\alpha \rightarrow_G^0 \alpha$$

Definition 3.5 (Reflexive transitive Hülle)

$$\alpha \rightarrow_G^0 \alpha$$

$$\alpha \rightarrow_G^{n+1} \gamma \Leftrightarrow \exists \beta. \alpha \rightarrow_G^n \beta \rightarrow_G \gamma$$

Definition 3.5 (Reflexive transitive Hülle)

$$\alpha \rightarrow_G^0 \alpha$$

$$\alpha \rightarrow_G^{n+1} \gamma \Leftrightarrow \exists \beta. \alpha \rightarrow_G^n \beta \rightarrow_G \gamma$$

$$\alpha \rightarrow_G^* \beta \Leftrightarrow \exists n. \alpha \rightarrow_G^n \beta$$

Definition 3.5 (Reflexive transitive Hülle)

$$\alpha \rightarrow_G^0 \alpha$$

$$\alpha \rightarrow_G^{n+1} \gamma \Leftrightarrow \exists \beta. \alpha \rightarrow_G^n \beta \rightarrow_G \gamma$$

$$\alpha \rightarrow_G^* \beta \Leftrightarrow \exists n. \alpha \rightarrow_G^n \beta$$

$$\alpha \rightarrow_G^+ \beta \Leftrightarrow \exists n > 0. \alpha \rightarrow_G^n \beta$$

Definition 3.5 (Reflexive transitive Hülle)

$$\begin{aligned}\alpha &\rightarrow_G^0 \alpha \\ \alpha \rightarrow_G^{n+1} \gamma &:\Leftrightarrow \exists \beta. \alpha \rightarrow_G^n \beta \rightarrow_G \gamma \\ \alpha \rightarrow_G^* \beta &:\Leftrightarrow \exists n. \alpha \rightarrow_G^n \beta \\ \alpha \rightarrow_G^+ \beta &:\Leftrightarrow \exists n > 0. \alpha \rightarrow_G^n \beta\end{aligned}$$

Beispiel: $E \rightarrow_G^{11} a * (a + a)$ und daher $E \rightarrow_G^* a * (a + a)$.

Definition 3.6 (Kontextfreie Sprache)

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ erzeugt die Sprache

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

Definition 3.5 (Reflexive transitive Hülle)

$$\begin{aligned}\alpha &\rightarrow_G^0 \alpha \\ \alpha \rightarrow_G^{n+1} \gamma &:\Leftrightarrow \exists \beta. \alpha \rightarrow_G^n \beta \rightarrow_G \gamma \\ \alpha \rightarrow_G^* \beta &:\Leftrightarrow \exists n. \alpha \rightarrow_G^n \beta \\ \alpha \rightarrow_G^+ \beta &:\Leftrightarrow \exists n > 0. \alpha \rightarrow_G^n \beta\end{aligned}$$

Beispiel: $E \rightarrow_G^{11} a * (a + a)$ und daher $E \rightarrow_G^* a * (a + a)$.

Wir nennen

$$\alpha_1 \rightarrow_G \alpha_2 \rightarrow_G \cdots \rightarrow_G \alpha_n$$

eine **Linksableitung** gdw in jedem Schritt das linkeste Nichtterminal in α_i ersetzt wird.

Definition 3.6 (Kontextfreie Sprache)

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ erzeugt die Sprache

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

Eine Sprache $L \subseteq \Sigma^*$ heißt **kontextfrei** gdw es eine kontextfreie Grammatik G gibt mit $L = L(G)$.

Definition 3.6 (Kontextfreie Sprache)

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ erzeugt die Sprache


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

Eine Sprache $L \subseteq \Sigma^*$ heißt **kontextfrei** gdw es eine kontextfreie Grammatik G gibt mit $L = L(G)$.

Abkürzungen:

CFG Kontextfreie Grammatik (*context-free grammar*)

CFL Kontextfreie Sprache (*context-free language*)



Beispiel 3.7

Die nicht-reguläre Sprache $L = \{a^n b^n \mid n \in \mathbb{N}\}$ ist kontextfrei, da sie von der CFG

$$S \rightarrow aSb \mid \epsilon$$

erzeugt wird.

Definition 3.6 (Kontextfreie Sprache)

Eine kontextfreie Grammatik $G = (V, \Sigma, P, S)$ erzeugt die Sprache

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

Eine Sprache $L \subseteq \Sigma^*$ heißt **kontextfrei** gdw es eine kontextfreie Grammatik G gibt mit $L = L(G)$.

Abkürzungen:

CFG Kontextfreie Grammatik (*context-free grammar*)

CFL Kontextfreie Sprache (*context-free language*)

Konvention:

Ist G aus dem Kontext eindeutig ersichtlich, so schreibt man auch nur $\alpha \rightarrow \beta$ statt $\alpha \rightarrow_G \beta$.