Mehrdeutige Grammatiken

Wir haben gesehen, dass es auch mehr als eine Linksableitung, d.h. mehr als einen Syntaxbaum geben kann, um das selbe Terminalwort zu erzeugen.

Eine Grammatik, die für mindestens ein Wort zwei Syntaxbäume hat, heißt *mehrdeutige Grammatik.* Wenn es für jedes erzeugte Wort eine einzige (eindeutig bestimmte) Linksableitung, d.h. also auch nur einen einzigen Syntaxbaum gibt, nennen wir die Grammatik *eindeutig.*

Auf Folie 6.4 haben wir bereits eine interessante mehrdeutige Grammatik eingeführt. Man kann prüfen, dass die Sprache, die von dieser Grammatik erzeugt wird, die folgende ist:

$$L = \{a^i b^j c^k \mid i = j \text{ oder } j = k\}$$

Mehrdeutige Sprachen

Eine Typ-2 Sprache nennen wir *mehrdeutig*, wenn jede Typ-2 Grammatik, die diese Sprache erzeugt, mehrdeutig ist.

Man sagt auch: Die Sprache ist *inhärent mehrdeutig*.

Die Sprache L von der vorigen Folie ist inhärent mehrdeutig. Es ist allerdings ziemlich schwierig (und im allgemeinen sogar unmöglich), die Mehrdeutigkeit nachzuweisen bzw. zu prüfen.

Wir stellen noch einmal klar:

Sobald es eine eindeutige Typ-2 Grammatik für die Sprache L gibt, ist L eine eindeutige Sprache.
Gibt es mehrdeutige Typ-2 Grammatiken für L, aber keine einzige eindeutige Typ-2 Grammatik, dann ist L inhärent mehrdeutig.

Backus-Naur-Form

Backus und Naur schrieben Typ-2 Grammatiken auf, indem sie nur die Produktionen, also die Elemente von P, angaben. Dabei wurden mehrere Regeln mit gleicher linker Seite als eine Regel mit Alternativen dargestellt:

Statt $A \to \beta_1, \ldots, A \to \beta_n$ schreibt man

$$A \to \beta_1 \mid \cdots \mid \beta_n$$

Hierbei haben wir wieder die Notation $u \to v$ für $(u, v) \in P$ verwendet. Backus und Naur verwendeten eine dritte Notation, nämlich $u ::= v$ anstelle von (u, v) oder $u \to v$.

Wir werden künftig vorwiegend die Notation $u \to v$ verwenden, seltener auch mal (u, v), aber nie $u ::= v$.

EBNF

Die EBNF (Erweiterte Backus-Naur-Form) sieht noch zwei weitere Kurzschreibweisen vor:

Die Regel $A \to \alpha[\beta]\gamma$ ersetzt die
zwei Regeln $A \to \alpha\gamma$ und $A \to \alpha\beta\gamma$. (β kann da sein, muss aber nicht)

Die Regel $A \to \alpha\{\beta\}\gamma$ ersetzt die
vier Regeln $A \to \alpha\gamma$, $A \to \alpha B\gamma$, $B \to \beta$ und $B \to \beta B$.
(β kann beliebig oft da sein - einschl. null mal)

Man kann recht leicht nachprüfen, dass die Menge der Typ-2 Sprachen identisch ist mit der Menge der durch EBNF beschreibbaren Sprachen; und ebenso mit der Menge der durch BNF beschreibbaren Sprachen!

Zusammenfassung

Abschnitt 1.1 Allgemeines:

- Begriffe: Alphabet, (formale) Sprache
- Formale Definition, es gibt überabzählbar viele Sprachen

Abschnitt 1.1.1 Grammatiken:

- Grammatik G als 4-Tupel
- Die Relation $\Rightarrow_G$ und die Sprache $L(G)$

Abschnitt 1.1.2 Chomsky-Hierarchie:

- Hierarchie der Grammatiken: Typ-0 bis Typ-3
- Übertragung der Hierarchie auf Sprachklassen
- ε-Sonderregel

Abschnitt 1.1.3 Wortproblem:

- Entscheidbarkeit, Definition des Wortproblems
- Beweis der Entscheidbarkeit für Typ-1 (und Typ-2/Typ-3)

Zusammenfassung (Forts.) + Ausblick

Abschnitt 1.1.4 Syntaxbäume:

- Definition des Syntaxbaums zu einer Ableitung
- Linksableitungen entsprechen Syntaxbäumen

Abschnitt 1.1.5 Backus-Naur-Form:

- Definition von BNF und EBNF
- Beide gleichwertig mit Typ-2

In der nächsten Einheit beginnt Abschnitt 1.2: Reguläre Sprachen, Unterabschnitt 1.2.1 Endliche Automaten.

Schauen Sie VOR der nächsten Vorlesung schon Abschnitt 1.2.1 im Buch genau an! Versuchen Sie, selbst einen endlichen Automaten für folgende Sprache (graphisch) zu entwerfen:

$$L = \{w \in \{a, b\}^* : \ |w|_a + 2 \cdot |w|_b \equiv 1 \bmod 7\}$$

Hierbei bezeichnet $|w|_a$ die Anzahl der a's im Wort w, $|w|_b$ entsprechend die der b's.