

Einführung in die Computerlinguistik – Chomskyhierarchie

Dozentin: Wiebke Petersen

1.7.2010

Wiederholung: Formale Grammatik

Definition

Eine *formale Grammatik* ist ein 4-Tupel $G = (N, T, S, P)$ aus

- einem Alphabet von Terminalsymbolen T (häufig auch Σ)
- einem Alphabet von Nichtterminalsymbolen N mit $N \cap T = \emptyset$
- einem Startsymbol $S \in N$
- einer Menge von Regeln/Produktionen
 $P \subseteq \{ \langle \alpha, \beta \rangle \mid \alpha, \beta \in (N \cup T)^* \text{ und } \alpha \notin T^* \}$.

Für eine Regel $\langle \alpha, \beta \rangle$ schreiben wir auch $\alpha \rightarrow \beta$.

Formale Grammatiken werden auch *Typ0-* oder *allgemeine Regelgrammatiken* genannt.

Chomskyhierarchie

- Wenn man die Form der Regeln einschränkt erhält man Teilmengen der Menge aller durch eine Grammatik erzeugten Sprachen.
- Die Chomskyhierarchie ist eine Hierarchie über die Regelbedingungen (den verschiedenen Sprachklassen entsprechen Einschränkungen über die rechten und linken Regelseiten).
- Die Chomskyhierarchie reflektiert eine spezielle Form der Komplexität, andere Kriterien sind denkbar und führen zu anderen Hierarchien.
- Die Sprachklassen der Chomskyhierarchie sind in der Informatik intensiv untersucht worden (Berechnungskomplexität, effektive Parser).
- Für Linguisten ist die Chomsky Hierarchie besonders interessant, da sie die Form der Regeln zentral stellt, und somit Aussagen über Grammatikformalisten zulässt.

Noam Chomsky



Noam Chomsky

(* 7.12.1928, Philadelphia)

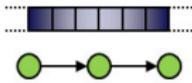
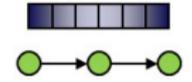
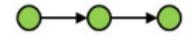
Noam Chomsky, *Three Models for the Description of Language*,
IRE Transactions on Information Theory (1956).

Chomsky-Hierarchie (grober Überblick)

| | | | |
|--|------------|---|---|
| allgemeine Regelsprachen recursively enumerable languages | Typ 0, RE | $\alpha \rightarrow \beta$ | |
| kontextsensitive Sprachen context-sensitive languages | Typ 1, CS | $\alpha A \nu \rightarrow \alpha \beta \nu$ | $a^n b^n c^n$, ww , $a^n b^m c^n d^m$ |
| kontextfreie Sprachen (context-free languages) | Typ 2, CF | $A \rightarrow \beta$ | $a^n b^n$, $w^R w$ |
| reguläre Sprachen (regular languages) | Typ 3, REG | $A \rightarrow bA$ $A \rightarrow a$ | $a^* b^*$ |

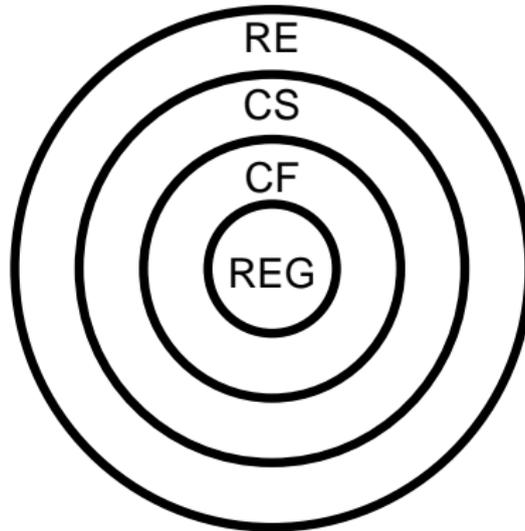
$a \in T$, $A \in N$, $\alpha, \beta, \dots \in (N \cup T)^*$, S Startsymbol

Chomsky-Hierarchie & Automaten

| <i>Sprache</i> | <i>Automat</i> | <i>Grammatik</i> | <i>Erkennung</i> | <i>Abhängigkeit</i> |
|------------------------|--|---|---|---|
| rekursiv aufzählbar | Turing Maschine  | unbeschränkt $Baa \rightarrow \varepsilon$ | unentscheidbar | beliebig |
| kontext- sensitiv | linear gebunden  | kontext- sensitiv $\gamma A \delta \rightarrow \gamma \beta \delta$ | NP-vollständig  | überkreuzt  |
| kontext- frei | Kellerautomat (Stapel)  | kontextfrei $C \rightarrow bABa$ | polynomiell  | eingebettet  |
| regulär | endlicher Automat  | regulär $A \rightarrow bA$ | linear  | strikt lokal  |

Chomskyhierarchie: Hauptsatz

$REG \subset CF \subset CS \subset RE$



Existenz von Sprachen, die nicht kontextfrei sind

- $L_1 = \{a^n b^n c^n\}$
- $L_2 = \{a^n b^m c^n d^m\}$
- $L_3 = \{ww : w \in \{a, b\}^*\}$

Theorem

Der Schnitt einer kontextfreien Sprache mit einer regulären Sprache ist kontextfrei.

$L_3 \cap a^* b^* a^* b^* = a^n b^m a^n b^m$ (L_3 ist nicht kontextfrei)

NL $\not\subseteq$ CF: Shieber 1985

Nebensatzeinbettung im Schweizerdeutschen

- Jan säit das
 mer d'chind em Hans es huus lönd hälfe aastriiche
 wir die Kinder-AKK Hans-DAT das Haus-AKK ließen helfen anstreichen
 NP₁ NP₂ NP₃ VP₁ VP₂ VP₃ "cross serial dependencies"
 
- *mer d'chind de Hans es huus lönd hälfe aastriiche
 wir die Kinder-AKK Hans-AKK das Haus-AKK ließen helfen anstreichen

Nebensatzeinbettung im Deutschen

- weil er die Kinder dem Hans das Haus streichen helfen ließ
 NP₁ NP₂ NP₃ VP₃ VP₂ VP₁ "nested dependencies"
 

Das Schweizerdeutsche ist keine kontextfreie Sprache!

Abschlusseigenschaften formaler Sprachen

| | Typ3 | Typ2 | Typ1 | Typ0 |
|-------------------------------|------|------|------|------|
| Vereinigung | + | + | + | + |
| Schnittmenge | + | - | + | + |
| Komplement | + | - | + | - |
| Konkatenation | + | + | + | + |
| Stern von Kleene | + | + | + | + |
| Schnitt mit regulärer Sprache | + | + | + | + |

Vokabular zur Theorie der Entscheidbarkeit

Algorithmus: Eine aus endlich vielen Schritten bestehende Verarbeitungsvorschrift, die, mechanisch angewandt zur Lösung eines Problems führt.

Entscheidbarkeit: Ein Problem ist entscheidbar, wenn ein Algorithmus existiert, der bei Eingabe einer Instanziierung des Problems nach endlich vielen Schritten angibt, ob dieses lösbar ist oder nicht.

Entscheidbarkeitsprobleme

Gegeben: Grammatiken $G = (N, \Sigma, S, P)$, $G' = (N', \Sigma, S', P')$, Wort $w \in \Sigma^*$

Wortproblem Ist w in G ableitbar?

Leerheitsproblem Erzeugt G eine nichtleere Sprache?

Äquivalenzproblem Erzeugen G und G' die gleichen Sprachen
($L(G) = L(G')$)?

Ergebnisse zu Entscheidungsproblemen

| | Typ3 | Typ2 | Typ1 | Typ0 |
|-------------------|------|------|------|------|
| Wortproblem | E | E | E | U |
| Leerheitsproblem | E | E | U | U |
| Äquivalenzproblem | E | U | U | U |

E steht für entscheidbar

U steht für unentscheidbar

Übung

Überlegen Sie sich, warum das Wort- und das Leerheitsproblem für reguläre Sprachen entscheidbar ist (argumentieren Sie mit endlichen Automaten).

Hausaufgabe (Abgabe 8.7.2010, BN: nur Aufgabe 2)

- 1 Geben Sie zu der kontextfreien Grammatik G einen Kellerautomaten an, der die von der Grammatik erzeugte Sprache akzeptiert. Erklären Sie in Einzelschritten, wie der Automat das Wort acb verarbeitet.
 $G = (\{S, C\}, \{a, b, c\}, S, \{S \rightarrow aSb, S \rightarrow C, C \rightarrow cC, C \rightarrow \epsilon\})$
- 2 Arbeiten Sie das Modul Turingmaschinen von der Seite <http://www.xplora.org/downloads/Knoppix/MathePrisma/Start/Module/Turing/index.htm> bis einschließlich dem Abschnitt über Programme (Seite 10/17) durch.