

Combinatory Categorical Grammar

Fabienne Eigner

Seminar
“Formal Grammars”
22. März 2007

Inhalt

1 Motivation

Inhalt

- 1 Motivation
- 2 Was sind CCG's?

Inhalt

- 1 Motivation
- 2 Was sind CCG's?
- 3 Copy-Sprache

Inhalt

- 1 Motivation
- 2 Was sind CCG's?
- 3 Copy-Sprache
- 4 Parsing

Inhalt

- 1 Motivation
- 2 Was sind CCG's?
- 3 Copy-Sprache
- 4 Parsing
- 5 Fazit

Motivation

Wofür CCG?

- natürliche Sprache nicht kontextfrei
- Beispiele für nicht kontextfreie Phänomene:
 - “Copy-Sprache” im Niederländischen
 - “Koordination”
- Lösung: “schwach kontextsensitive Grammatiken”
- Grammatiken dieser Art: TAG's, LIG's und **CCG's**

CCG

- In den 80ern entwickelt von Mark Steedman
- “The Syntactic Process” 2000 von Steedman
- Tutorial “Combinatory Categorical Grammar” 2003 von Steedman und Baldridge

Grammatik

Grammatik $G = (V_T, V_N, f, S, R)$:

- V_T : Menge der Terminale
- V_N : Menge der Nichtterminale (atomare Kategorien)
- Funktion f , bildet Terminale auf Mengen von Kategorien ab
- einziges Startsymbol S
- $n \geq 1, |i \in \{\backslash, / \}$: kombinatorische Regeln R der Form:
 - $x/y \quad y \rightarrow x \quad (>)$
 - $y \quad x \backslash y \rightarrow x \quad (<)$
 - $x/y \quad y |_1 z_1 |_2 \dots |_n z_n \rightarrow x |_1 z_1 |_2 \dots |_n z_n \quad (B_{>}^{(*)})$
 - $y |_1 z_1 |_2 \dots |_n z_n \quad x \backslash y \rightarrow x |_1 z_1 |_2 \dots |_n z_n \quad (B_{<}^{(*)})$

Copy-Sprache

Grammatik

Copy-Sprache: $L = \{ww \mid w \in \{a, b\}^+\}$

Grammatik $G = (V_T, V_N, f, S, R)$:

- $V_T = \{a, b\}$
- $V_N = \{S, A, B\}$
- Funktion f :
 - $f(a) = \{A, S \setminus A, S \setminus A / S\}$
 - $f(b) = \{B, S \setminus B, S \setminus B / S\}$
- Regeln R :
 - $y \quad (x \setminus y) \rightarrow x$
 - $(x / y) \quad (y \setminus z) \rightarrow (x \setminus z)$

Copy-Sprache

Beispiel Parsebaum

a b b a b b

Copy-Sprache

Beispiel Parsebaum

a b b a b b
 \overline{A} \overline{B} \overline{B} $\overline{S \setminus A / S}$ $\overline{S \setminus B / S}$ $\overline{S \setminus B}$

Copy-Sprache

Beispiel Parsebaum

$$\begin{array}{ccccccc} a & b & b & a & b & b & \\ \bar{A} & \bar{B} & \bar{B} & S \backslash A / S & S \backslash B / S & S \backslash B & \\ & & & \hline & & & S \backslash A \backslash B / S & & & \end{array} \mathbf{B}^* >$$

Copy-Sprache

Beispiel Parsebaum

$$\begin{array}{ccccccc}
 a & b & b & a & b & b & \\
 \bar{A} & \bar{B} & \bar{B} & S\backslash A/S & S\backslash B/S & S\backslash B & \\
 & & & \hline & \mathbf{B}^* & & \\
 & & & S\backslash A\backslash B/S & & & \\
 & & & \hline & \mathbf{B}^* & & \\
 & & & S\backslash A\backslash B\backslash B & & &
 \end{array}$$

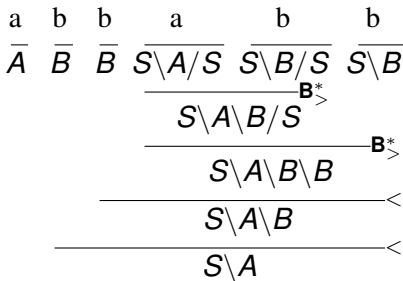
Copy-Sprache

Beispiel Parsebaum

$$\begin{array}{ccccccc}
 a & b & b & a & b & b & \\
 \bar{A} & \bar{B} & \bar{B} & S \backslash A / S & S \backslash B / S & S \backslash B & \\
 & & & \hline & \mathbf{B}^* & & \\
 & & & S \backslash A \backslash B / S & & & \\
 & & & \hline & \mathbf{B}^* & & \\
 & & & S \backslash A \backslash B \backslash B & & & \\
 & & & \hline & & & \mathbf{B}^* \\
 & & & S \backslash A \backslash B & & & <
 \end{array}$$

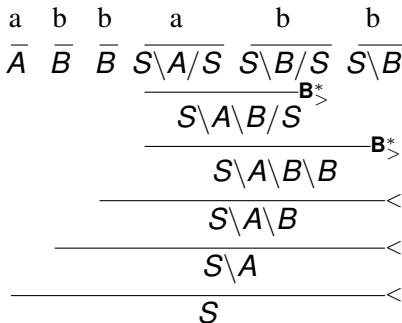
Copy-Sprache

Beispiel Parsebaum



Copy-Sprache

Beispiel Parsebaum



Parsing nach Vijay-Shanker und Weir

Überblick

Drei Schritte:

- Erkennungsalgorithmus
- Parsingalgorithmus
- Effiziente Darstellung der Parsebäume mittels Parsewaldes

Parsing

Idee und Probleme

- im Prinzip Bottom-Up Parsing mittels CKY
- Problem: innere Knoten des Parsebaums Art Stacks (LIFO)
Knoten-Stacks können polynomiell in der Länge des zu erkennenden Strings wachsen
- \Rightarrow Erkennungsalgorithmus möglicherweise exponentielle Laufzeit

Parsing

Lösungsansatz

- Betrachte:
 - $A/X_1/\dots/X_l/B \quad B/C$
 - $A/Y_1/\dots/Y_m/B \quad B/C$
- Anwendung einer Regel hängt nur von A (Target), B (Top) und B/C ab
- Idee: Sharing: statt komplette Kategorie zu speichern, speichere Target, Top (die maximal c obersten Stacksymbole) und Link auf den Rest (Tail)

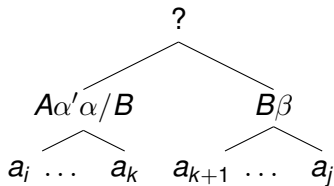
Der Erkennungsalgorithmus

Notationen

- 4-dimensionales Array $L[i, j][p, q]$ zum Speichern der Einträge
- Einträge der Form:
 - $((A, \alpha), -) \in L[i, j][0, 0]$
 - $((A, \alpha), T) \in L[i, j][p, q]$
- $x \in L[i, j][p, q] \Rightarrow$ aus x lässt sich der String $a_i \dots a_j$ ableiten

Inferenzregeln

Idee



Der Erkennungsalgorithmus

Inferenzregeln

Initialisierung:

$$\frac{c \in f(a_i) \quad c = A\alpha}{((A, \alpha), -) \in L[i, i][0, 0]}$$

Der Erkennungsalgorithmus

Inferenzregeln

Fall 1:

$$\frac{((A, \alpha/B), T) \in L[i, k][p, q] \quad ((B, \beta), -) \in L[k + 1, j][0, 0]}{((A, \alpha\beta), T) \in L[i, j][p, q]}$$

falls:

- $T = -$ und $|\alpha\beta| < c$
- $T \neq -$ und $|\beta| > 1$ und $\alpha = \epsilon$
- $T \neq -$ und $|\beta| = 1$
- $T \neq -$ und $|\beta| = 0$ und $\alpha \neq \epsilon$

Der Erkennungsalgorithmus

Inferenzregeln

Fall 2:

$$\frac{((A, \alpha/B), T) \in L[i, k][p, q] \quad ((B, \beta), -) \in L[k + 1, j][0, 0]}{((A, \beta), /B) \in L[i, j][i, k]}$$

falls:

- $T = -$ und $|\alpha\beta| \geq c$
- $T \neq -$ und $|\beta| > 1$ und $\alpha \neq \epsilon$

Der Erkennungsalgorithmus

Inferenzregeln

Fall 3:

$$\frac{\begin{array}{l} ((A, /B), \gamma) \in L[i, k][p, q] \\ ((B, \epsilon), -) \in L[k + 1, j][0, 0] \\ ((A, \beta'\gamma), T') \in L[p, q][r, s] \end{array}}{((A, \beta'), T') \in L[i, j][r, s]}$$

falls:

- $\gamma = T \neq -$

Der Erkennungsalgorithmus

Abbruchbedingung

Wenn $((S, \epsilon), -) \in L[1, n][0, 0]$

\Rightarrow Wort $a_1 \dots a_n$ erkannt

Der Erkennungsalgorithmus

Laufzeit

- Betrachte letzte Inferenzregel:

$$\frac{\begin{array}{l} ((A, /B), \gamma) \in L[i, k][p, q] \\ ((B, \epsilon), -) \in L[k + 1, j][0, 0] \\ ((A, \beta'\gamma), T') \in L[p, q][r, s] \end{array}}{((A, \beta'), T') \in L[i, j][r, s]}$$

- \Rightarrow Sieben unterschiedliche Variablen zwischen 1 und n
- \Rightarrow Laufzeit in $O(n^7)$
- durch geschickte Implementierung $O(n^6)$ erreichbar

Fazit

- Gewisse linguistische Phänomene lassen sich nicht in CFG's ausdrücken
- Stattdessen “schwach kontextsensitive Grammatiken”
- CCG Beispiel für solche Grammatik
- CCG lässt sich in $O(n^6)$ parsen

Literatur I



Mark Steedman und Jason Baldridge

Tutorial: Combinatory Categorical Grammar.

<http://groups.inf.ed.ac.uk/ccg/publications.html>
2003.



K. Vijay-Shanker und David J. Weir

Polynomial Time Parsing of Combinatory Categorical Grammars.

<http://acl.ldc.upenn.edu/P/P90/P90-1001.pdf>
1990.

Literatur II



K. Vijay-Shanker und David J. Weir

Parsing Some Constrained Grammar Formalisms.

<http://acl.ldc.upenn.edu//J/J93/J93-4002.pdf>

1993

**Vielen Dank für die
Aufmerksamkeit!**