Computational Tools for Logic-Based Grammar Formalisms

*Minimalist Grammar*

Willemijn Vermaat – UiL-OTS, Utrecht

4th August 2002
Contents

1 The problem 3

2 Example 6

3 Minimalist Grammar 8
   3.1 Declarative sentence ........................................ 10
   3.2 Wh-phrase ....................................................... 12
1. The problem

Syntax

Generative syntax seeks answers to linguistic phenomena.

Provide an abstract theory that:

- captures the data descriptively
- can be applied cross-linguistically
- generalizes to similar phenomena in one language
- can be build in one bigger framework

Formal frameworks

Formal frameworks, such as Type Logical Grammar and Minimalist Grammar, might provide the basis for such an abstract theory.
Similarities

The basic machinery of the three frameworks are similar:

- basic operations: Merge and Move,
- lexicon
- important role played by features (properties of words)

Differences

The implementation and level of formality differs.
Plan of action

- analyze empirical data
- implement data in both formal frameworks
- compare the three analysis:
  - enhance the two formal reasoning systems
  - provide answers for generative syntax

Parsers

Parsers such as Grail and MGCKY help:

- to proofcheck the analysis that you made
- to look into the computational complexity
2. Example

Wh-movement in English

(1) Willem loves Maxima
(2) Does Willem love Maxima?
(3) Who does Willem love?

Data shows:

- The object of the sentence, ‘Maxima’ is base-generated as a complement of the verb ‘love’
- English needs do-support for negative sentences, yes/no-questions and wh-phrases
- The wh-object, in wh-phrases, is base-generated in object position and then moved to the front of the sentence to precede the verb phrase.

Puzzle: How is the wh-phrase moved to the front of the sentence?
Syntactic analysis

\[
\begin{align*}
\text{CP} & \quad \text{IP} \\
\text{DP}(0) & \quad \epsilon \\
D' & \quad I' \\
D & \quad [\text{does}] \\
[\text{who}] & \quad \text{VP} \\
V' & \quad \text{DP} \\
V & \quad [\text{love}] \\
\text{D} & \quad [\text{willem}] \\
\text{t}(0) & \quad [\text{willem}] \\
\end{align*}
\]
3. Minimalist Grammar

A minimalist grammar \( MG = (\Sigma, F, Types, Lex, F) \)

Features \( F \):

- base \( B \) = \{v, n, np, case, wh, \ldots\}
- selectors \( S \) = \{=f|f \in B\}
- licensees \( M \) = \{-f|f \in B\}
- licensors \( N \) = \{+f|f \in B\}
- features \( F \) = \( B \cup S \cup M \cup N \)

Grammar

- Lexical types \( LT = \Sigma^* :: F^* \)
- Derived types \( DT = \Sigma^* : F^* \)
  \( \cdot \in \{::, :\} \)
- Lexicon \( Lex \subset LT^+ \)
- Minimalist grammar \( G = Lex \)
Operations

**Merge** \: \((E \times E) \rightarrow E\)

where \(t = (t_s t_h t_c)\)

[r1] if \(s\) is lexical, and \(t\) has one \([f]\)

\[
\begin{align*}
s :: &= f \gamma \quad t_s, t_h, t_c \cdot f \\
\epsilon, s, t &: \gamma
\end{align*}
\]

\[r_1\]

[r2] if \(s\) is derived, and \(t\) has one \([f]\)

\[
\begin{align*}
s_s, s_h, s_c :: &= f \gamma \quad t_s, t_h, t_c \cdot f \\
ts_s, s_h, s_c &: \gamma
\end{align*}
\]

\[r_2\]

[r3] if \(s\) is lexical or derived, and \(t\) has one \([f]\) and a set of (licensee) features \(\delta\)

\[
\begin{align*}
s_s, s_h, s_c :: &= f \gamma \quad t_s, t_h, t_c \cdot f \delta \\
&s_s, s_h, s_c &: \gamma, t : \delta
\end{align*}
\]

\[r_3\]
3.1. Declarative sentence

Lexicon:

<table>
<thead>
<tr>
<th>Lexical:</th>
<th>Functional:</th>
</tr>
</thead>
<tbody>
<tr>
<td>willem :: d</td>
<td>ϵ :: =vp c</td>
</tr>
<tr>
<td>maxima :: d</td>
<td></td>
</tr>
<tr>
<td>loves :: =d vp</td>
<td></td>
</tr>
</tbody>
</table>

willem loves maxima : c
Move : $E \rightarrow E$

[m1] if $s$ is derived, and $t$ in the chain is the only element (SMC) with one $[-f]$

\[
\frac{s_s, s_h, s_c : +f\gamma, \Gamma[t_s, t_h, t_c : -f]}{t s_s, s_h, s_c : \gamma, \Gamma} \quad m1
\]

[m2] if $s$ is derived, and $t$ in the chain is the only element (SMC) with a $[-f]$ followed by a non-empty set of features $\delta$

\[
\frac{s_s, s_h, s_c : +f\gamma, \Gamma[t_s, t_h, t_c : -f\delta]}{s_s, s_h, s_c : \gamma, \Gamma[t_s, t_h, t_c : \delta]} \quad m2
\]
3.2. Wh-phrase

Lexicon:

<table>
<thead>
<tr>
<th>Lexical:</th>
<th>Functional:</th>
</tr>
</thead>
<tbody>
<tr>
<td>willem :: d</td>
<td>e :: =vp c</td>
</tr>
<tr>
<td>maxima :: d</td>
<td>e :: =i c</td>
</tr>
<tr>
<td>loves :: =d =d vp</td>
<td>does :: =v i</td>
</tr>
<tr>
<td>love :: =d =d V</td>
<td></td>
</tr>
</tbody>
</table>

Question:

| who :: ?       | ε :: ? |

Grammar: wh.pl
who does willem love :c