Topics in Computational Linguistics
— Grammar Engineering —

Dan Flickinger
Stephan Oepen

Center for the Study of Language and Information
Stanford University — Winter Quarter 2003

{dan|oe}@csli.stanford.edu
# Our Grammars: Table of Contents

## Type Description Language (TDL)
- `types.tdl`  type definitions: hierarchy of grammatical knowledge;
- `lexicon.tdl`  instances of (lexical) types plus orthography;
- `rules.tdl`  instances of construction types; used by the parser;
- `lrules.tdl`  lexical rules, applied before non-lexical rules;
- `irules.tdl`  lexical rules that require orthographemic variation;
- `roots.tdl`  grammar start symbol(s): ‘selection’ of final results.

## Auxiliary Files (Grammar Configuration for LKB)
- `labels.tdl`  TFS templates abbreviating node labels in trees;
- `globals.lsp, user-fns.lsp`  parameters and interface functions;
- `mrsglobals.lsp`  MRS parameters (path to semantics et al.)
The Format of Grammar Rules in the LKB

\[
\begin{pmatrix}
\text{HEAD} & 1 \\
\text{SPR} & 2 \\
\text{COMPS} & \langle \rangle \\
\cdots
\end{pmatrix}
\rightarrow
\begin{pmatrix}
\text{HEAD} & 1 \\
\text{SPR} & 2 \\
\text{COMPS} & \langle 3 \rangle \\
\end{pmatrix},
\begin{pmatrix}
\text{ARGS} & \langle \rangle \\
\cdots
\end{pmatrix}
\]

\[
\begin{pmatrix}
\text{HEAD} & 1 \\
\text{SPR} & 2 \\
\text{COMPS} & \langle \rangle \\
\cdots
\end{pmatrix}
\rightarrow
\begin{pmatrix}
\text{HEAD} & 1 \\
\text{SPR} & 2 \\
\text{COMPS} & \langle 3 \rangle \\
\end{pmatrix},
\begin{pmatrix}
\text{ARGS} & \langle \rangle \\
\cdots
\end{pmatrix}
\]
Bottom-Up Chart Parsing in the LKB

- Initialize chart: retrieve all *lexical entries* for all words in the input string;
- Parsing: apply all *rules* to all adjacent tuples of edges in the chart;
- Add a new *edge* for each successful instantiation of a grammar rule.
- Test spanning edges against *start symbol*: select ‘sentential’ results.

```
girls with hats from France
```

Grammar Engineering (26)
Lexical Variation: Lexical Rules

- Dative shift, passivization, et al. are systematic processes in the lexicon;
- **lexical rules** are unary grammar rules operating ‘within the lexicon’;
- take as input a lexical sign (*syn-struc*) and output a derived lexical sign.

Rough Approximation of Passive Lexical Rule

```
\[
\begin{array}{c}
\text{HEAD} \quad 1 \\
\text{SPR} \quad \langle 2 \rangle \\
\text{COMPS} \quad 3 \\
\text{ARGS} \quad \langle \\
\quad \text{HEAD} \quad 1 \\
\quad \text{SPR} \quad \langle \text{phrase} \rangle \\
\quad \text{COMPS} \quad \langle \\
\quad \quad \text{FIRST} \quad 2 \\
\quad \quad \text{REST} \quad 3 \\
\end{array}
\]
```

Grammar Engineering (27)
Orthographemic Variation: ‘Inflectional’ Rules

\%(letter-set (!s abcdefghijklmnopqrstuvwxyz))

noun-non-3sing_irule :=
%suffix (!s !ss) (!ss !ssses) (ss sses)
word &
[ HEAD [ AGR non-3sing ],
  ARGs < noun-1xm > ].

noun-3sing_irule :=
word &
[ ORTH #1,
  HEAD [ AGR 3sing ],
  ARGs < noun-1xm & [ ORTH #1 ] > ].
Types vs. Named Feature Structures (‘Instances’)

‘types.tdl’

verb-word := word &
[ HEAD verb,
  SPR < phrase & [ HEAD noun,
    SPR < >,
    COMPS < > ] > ].

verb-word-3sing := verb-word &
[ SPR < [ HEAD [ AGR 3sing ] ] > ].

‘lexicon.tdl’

barks := verb-word-3sing &
[ ORTH "barks" ].

word

verb-word

phrase

verb-word-3sing

[ ORTH “barks” ]
Structured Categories in a Unification Grammar

- All (constituent) categories in the grammar are typed feature structures;
- specific TFS configurations may correspond to ‘traditional’ categories;
- labels like ‘S’ or ‘NP’ are mere abbreviations, not elements of the theory.

<table>
<thead>
<tr>
<th>Category</th>
<th>Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>word 'N'</td>
<td>lexical’</td>
</tr>
<tr>
<td>phrase 'S’</td>
<td>‘intermediate’</td>
</tr>
<tr>
<td>phrase 'VP’</td>
<td>‘maximal’</td>
</tr>
</tbody>
</table>
Parse Trees and Node Labeling

- Derivation trees are constructed from lexical items and grammar rules;
- node labels (so far) are instance names from ‘lexicon.tdl’ and ‘rules.tdl’.
Decorating our Trees: Abbreviatory Node Labels

'labels.tdl'

s-label := label &
[ HEAD verb, SPR <> , COMPS <> ,
   LABEL-NAME "S" ].

vp-label := label &
[ HEAD verb, SPR < [ ] > , COMPS <> ,
   LABEL-NAME "VP" ].

v-label := label &
[ HEAD verb ,
   LABEL-NAME "V" ].

np-label := label &
[ HEAD noun, SPR <> , COMPS <> ,
   LABEL-NAME "NP" ].

Grammar Engineering (32)