

Topics in Computational Linguistics — Grammar Engineering —

Dan Flickinger

CSLI Stanford & Saarland University

`danf@csli.stanford.edu`

Stephan Oepen

Universitetet i Oslo & CSLI Stanford

`oe@csli.stanford.edu`

<http://lingo.stanford.edu/courses/05/ge/>

Review: Context-Free Grammars

- Formally, a *context-free grammar* (CFG) is a quadruple: $\langle C, \Sigma, P, S \rangle$
- C is the set of categories (aka *non-terminals*), e.g. $\{S, NP, VP, V\}$;
- Σ is the vocabulary set (aka *terminals*), e.g. $\{\text{kim, snow, adores}\}$;
- P is a set of category rewrite rules (aka *productions*), e.g.

$S \rightarrow NP VP$
 $VP \rightarrow V NP$
 $NP \rightarrow \text{kim}$
 $NP \rightarrow \text{snow}$
 $V \rightarrow \text{adores}$

- $S \in C$ is the *start symbol*, a filter on complete (aka ‘sentential’) results;
- for each rule ‘ $\alpha \rightarrow \beta_1\beta_2 \dots \beta_n$ ’ $\in P$: $\alpha \in C$ and $\beta_i \in C \cup \Sigma$; $1 \leq i \leq n$.



Review: The Chomsky Hierarchy of Languages

- (Formal) Languages vary in ‘degree of structural complexity’ exhibited;
- traditionally: a^* (iteration) vs. $a^n b^n$ (nesting) vs. $a^n b^n c^n$ (‘cross-serial’);
- Chomsky Hierarchy: inclusion classes of formal languages; Type 0 – 3.

0	unrestricted	$\beta_1 \rightarrow \beta_2$	Turing Machine
1	context-sensitive	$\beta_1 \alpha \beta_2 \rightarrow \beta_1 \gamma \beta_2$	linearly-bounded automaton
2	context-free	$\alpha \rightarrow \beta$	push-down automaton
3	regular	$\alpha \rightarrow \delta \alpha \mid \alpha \delta$	finite-state automaton
$\alpha \in C, \beta_i \in (C \cup \Sigma)^*, \gamma \in (C \cup \Sigma)^+, \delta \in \Sigma^+$			

What is the Formal Complexity of Natural Languages?

- Minimally context-free (center self-embedding, e.g. in relative clauses);
- (Culy; Shieber, 1985): *not* context-free (Bambara, Swiss German);
- (Joshi, 1985): extra class of *mildly* context-sensitive languages (TAG).

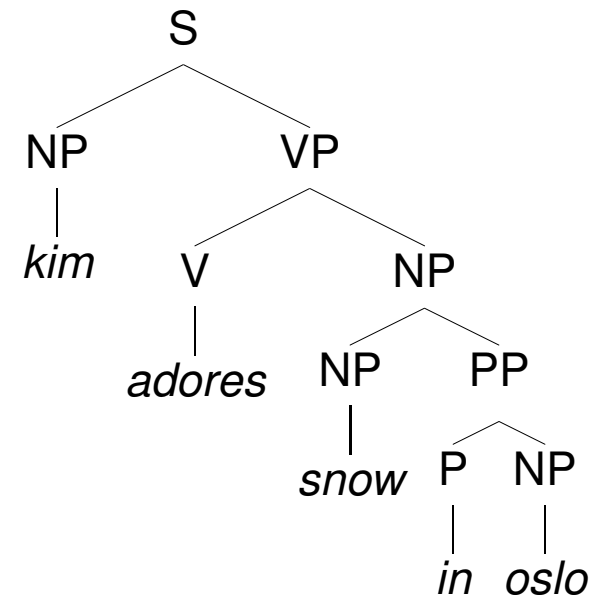
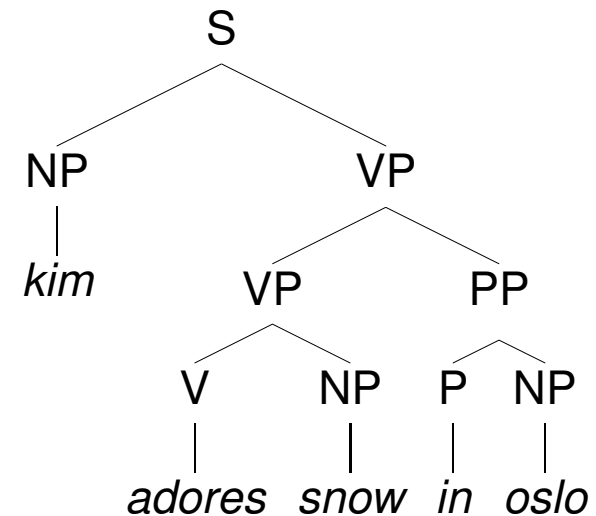


Recognizing the Language of a Grammar

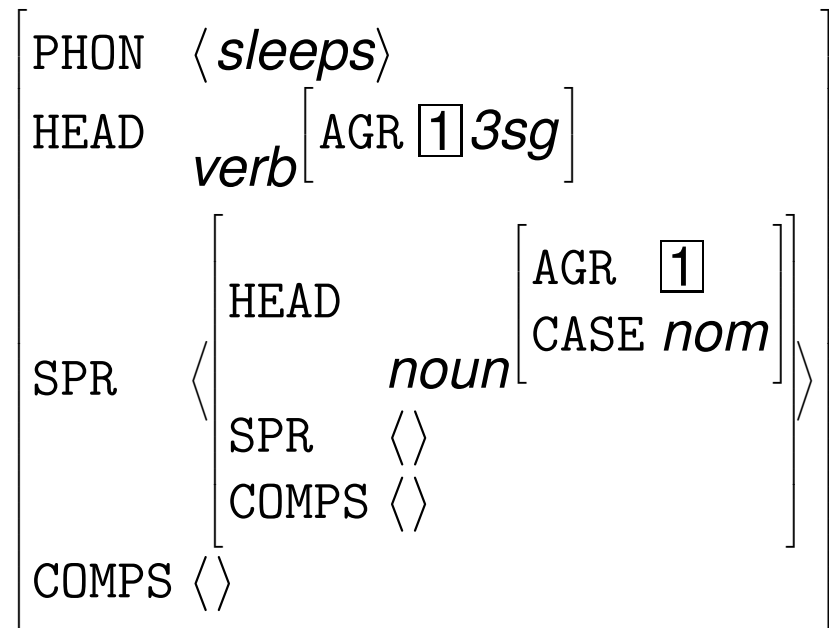
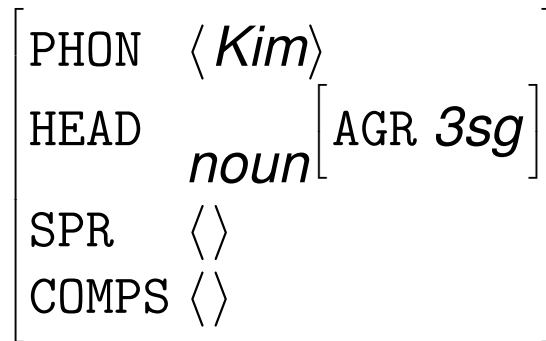
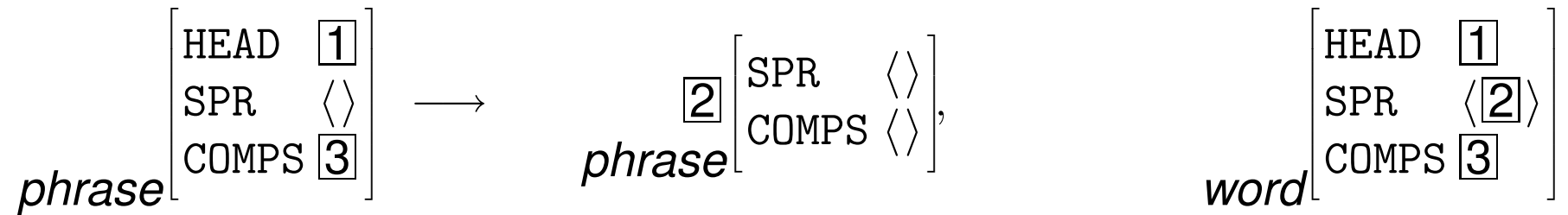
$S \rightarrow NP VP$
 $VP \rightarrow V NP$
 $VP \rightarrow VP PP$
 $NP \rightarrow NP PP$
 $PP \rightarrow P NP$
 $NP \rightarrow kim \mid snow \mid oslo$
 $V \rightarrow snores \mid adores$
 $P \rightarrow in$

All Complete Derivations

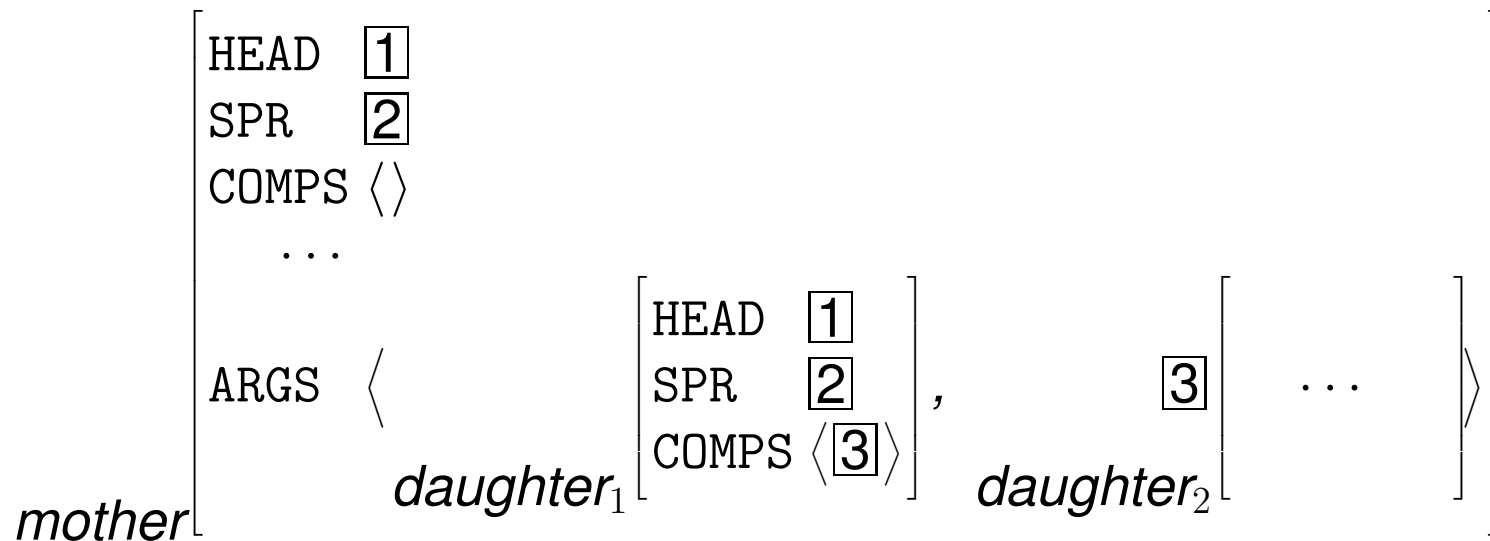
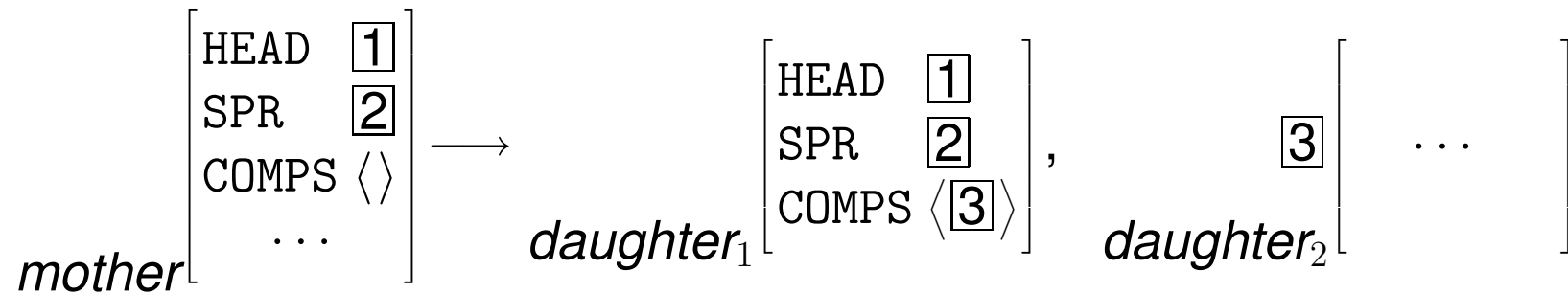
- are rooted in the start symbol S ;
- label internal nodes with categories $\in C$, leafs with words $\in \Sigma$;
- instantiate a grammar rule $\in P$ at each local subtree of depth one.



Interaction of Lexicon and Phrase Structure Schemata



The Format of Grammar Rules in the LKB



The Start Symbol in our First Grammar — Parsing

```
root := phrase &  
[ HEAD verb,  
  SPR < >,  
  COMPS < > ].
```

Bottom-Up Chart Parsing

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



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.

word $\left[\begin{array}{l} \text{HEAD } \textit{noun} \\ \text{SPR } \langle \langle \rangle \rangle \\ \text{COMPS } \langle \rangle \end{array} \right]$

‘N’

‘lexical’

phrase $\left[\begin{array}{l} \text{HEAD } \textit{verb} \\ \text{SPR } \langle \rangle \\ \text{COMPS } \langle \rangle \end{array} \right]$

‘S’

‘maximal’

phrase $\left[\begin{array}{l} \text{HEAD } \textit{verb} \\ \text{SPR } \langle \langle \rangle \rangle \\ \text{COMPS } \langle \rangle \end{array} \right]$

‘VP’

‘intermediate’



Morpho-Syntactic Categories (1 of 2)

Number — Person — Case — Gender

*That dog barks. — Those dogs bark.
I bark. — You bark. — They bark. — Sam shaved himself.
We bark. — You bark. — Those dogs bark.
I saw her. — She saw me. — My dog barked.*

...

How many distinct verb forms according to number and person?

Tense — Aspect — Mood

*The dog barks. — The dog barked — The dog will bark.
The dog has barked. — The dog is barking.
If I were a carpenter, ...*

...



Morpho-Syntactic Categories (2 of 2)

Parts of Speech (PoS)

cat, dog, neighbours, ...	noun (N)
barks, chased, was, ...	verb (V)
fierce, angry, black, young, ...	adjective (Adj)
quickly, probably, not, ...	adverb (Adv)
a, the, my, that, ...	determiner (D)
of, by, on, at, under, ...	preposition (P)
she, mine, those, what, ...	pronoun (Pro)
and, neither ... nor, because, ...	conjunction (C)

- **Paradigm** set of word forms ('units'), e.g. {*bark, barks, barked*};
- **Unit Categories** dimensions structuring a paradigm *internally*;
- **Paradigm Categories** properties *common* to all paradigm units.



Grammatical Functions

Licensing — Government — Agreement

*The dog barks. — *The dog a cat barks — *The dog barks a cat.*

*Kim depends on Sandy — *Kim depends in Sandy*

The class meets on Thursday in Meyer-183 at 2:15.

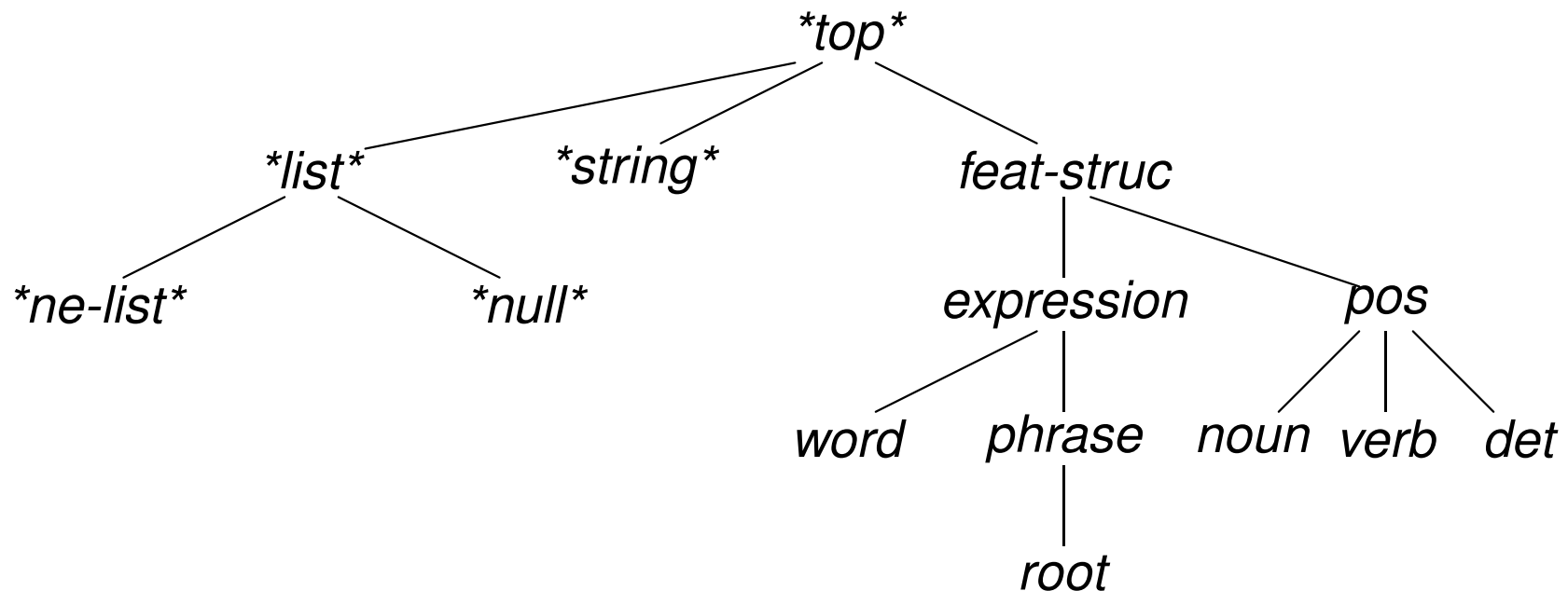
...

- **Head** licenses additional constituents and can govern their form;
- **Specifier** precedes head, singleton, agreement, nominative case;
- **Complement** post-head, governed in form, order constraints;
- **Adjunct** ‘free’ modifier, optional, may iterate, designated position;
- **Government** *paradigm* category of c_1 determines the form of c_2 ;
- **Agreement** bi-directional: concord of *unit* categories of c_1 and c_1 .



Our Initial Type Hierarchy

- Linguistic and non-linguistic objects organized below ‘top’ type (**top**);
- fundamental division of constituents: *word* (lexicon) vs. *phrase* (rules);
- *root* is the ‘start symbol’ — parts of speech as separate sub-hierarchy.



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.

Auxiliary Files (Grammar Configuration for LKB)

- `globals.lsp.` Parameter settings (e.g. path to orthography et al.);
- `user-fns.lsp` (small number) of LKB interface functions;
- `mrsglobals.lsp` MRS parameters (path to semantics et al.)

