Logics in Constraint-Based Lexicalized Grammar (CBLG). Linguistically Significant Generalizations

Roussanka Loukanova
Logics for Linguistics, Autumn, 2014 (Handout 2)

5 November, 2014

Abstract

The purpose of this overview of CBLG is to introduce major linguistic concepts and how CBLG captures linguistically significant generalizations. We present the technical architecture of CBLG, in a way that can be a basis for syntax-semantics interfaces by using various semantic approaches.

1 Some questions that we address

- Why do we need formal grammar?
- What is the difference between a formal system and a computational system?
- Why do we need computational grammar?

Intuitively, we need computational grammar

1. To make correct predictions of well-formed and meaningful expressions
2. To extract the meanings of well-formed expressions
3. To express meanings, which are given in one or other way, e.g., by a language (a formal or HL), into other languages
4. To express linguistic information consistently (without contradictions)
5. To cover *productivity* of HLs:
   (a) Syntactic Compositionality
   (b) Semantic Compositionality
6. To use computer systems doing the above and other related HLP, notably translation between formal and natural languages

7. Methods of *The Theory of Computability* can be used to get a computational grammar from a formal grammar.

The most distinctive phenomenon of HLs is their syntactic and semantic productivity.

Syntactic productivity (recursion is essential) : with a finite lexicon, and a finite set of syntactic rules, a grammar can license infinitely many structural descriptions as well formed.

Chomsky (1957, 1965) initiated the formal and rigorous approaches to HL syntax by introducing recursive CFGs to represent the HL which is potentially infinite phenomena with finite recursive mathematical methods.

- Recursive CFGs: efficient parsing, but not sufficiently adequate as linguistic theory. Why? (We provide some answers in what follows.)
- Transformational Grammars are descriptively adequate, but are equivalent to Turing Machines, i.e., too powerful. What does this mean?
- Therefore, there emerged need of other approach to grammar theory, which is both sufficiently adequate and computational.

Semantic productivity — Frege's Principle of Compositionality The meaning of a compound expression depends on

- the meanings of its parts, and
- the *way the parts are put together*

Frege 1892

See Frege [2, 3].
Therefore, a grammar theory has to define, as a minimum

- the basic language units
- the semantic representations of the basic language units
- the lexical rules
- syntactic and semantic rules that compose "smaller" parts into larger ones.

2 Contemporary approaches to formal grammar

Interdisciplinary efforts from Mathematics, Mathematical Logic, Linguistics, Computer Science, Artificial Intelligence, Cognitive Science, and in recent years, Neuroscience.

The target is adequate, formal grammar theory that covers lexicon, syntax, and semantics from computational perspective, i.e., provides computational grammar.

The roots of formal grammar:

Gottfried Wilhelm von Leibniz (1646-1716)

Friedrich Ludwig Gottlob Frege (1848-1925)

2.1 Context Free Grammars (CFGs) Attempted as Grammar of HL

CFGs specify cooccurrence properties of expressions by using nonterminal symbols S, NP, VP, NOM, etc., which are called syntactic categories, and stipulating the possible combinations between the categories with CFG rules.

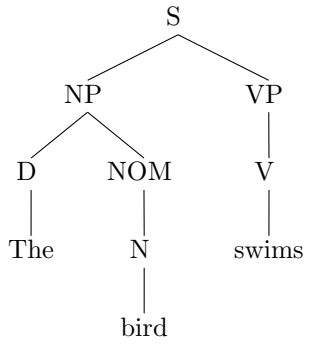
Here is a toy CFG grammar, with which we demonstrate some of the problems in attempts to use bare CFR for computational grammar of HL:

S	→	NP VP	VP	→	V
NP	→	D NOM	VP	→	V NP
NOM	→	N	VP	→	V NP PP
NOM	→	Adj NOM	VP	→	V NP NP
NOM	→	NOM PP	PP	→	P NP
D	→	the	V	→	swims
N	→	bird	V	→	swim

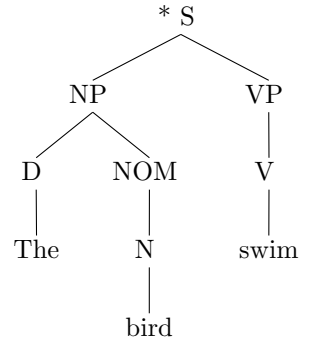
...

The above rules explicitly show that they fail to represent agreement and verb complement requirements:

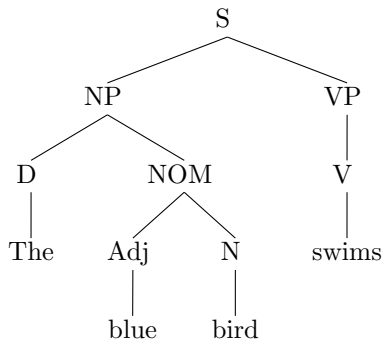
(1)



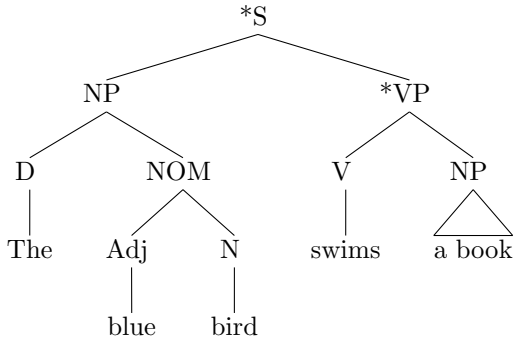
(2)



(3)



(4)



To represent co-occurrence requirements on agreement within a CFG, one may introduce proliferated syntactic categories and respective rules as follows:

$$
\begin{array}{lcl}
\text{S} & \rightarrow & \text{NP-SG VP-SG} \\
\text{S} & \rightarrow & \text{NP-PL VP-PL} \\
\text{NP-SG} & \rightarrow & \text{D-SG N-SG} \\
\text{NP-PL} & \rightarrow & \text{D-PL N-PL} \\
\text{VP-SG} & \rightarrow & \text{V-SG NP-SG} \\
\ldots &&
\end{array}
$$

$$
\begin{array}{lcl}
\text{D-SG} & \rightarrow & \text{the} \\
\text{N-SG} & \rightarrow & \text{bird} \\
\text{V-SG} & \rightarrow & \text{swims} \\
\text{Adj} & \rightarrow & \text{blue}
\end{array}
\qquad
\begin{array}{lcl}
\text{D-PL} & \rightarrow & \text{the} \\
\text{N-PL} & \rightarrow & \text{birds} \\
\text{V-PL} & \rightarrow & \text{swim}
\end{array}
$$

2.2 CFG subcategorization:

```
vp --> [walks].

vp --> tv_np, np.          tv_np --> [writes].

vp --> tv_pp, pp.          tv_pp --> [relies].

vp --> tv_np_pp, np, pp.   tv_np_pp --> [put].

np --> [Terry].            np --> [Pat].
```

These rules determine the following (non)well-formedness :

(1) a. Terry walks.

b. *Terry walks Kim.

c. Pat relies on Kim.

d. *Pat relies.

e. Pat put the toy on the table.

f. *Pat put the toy.

From the perspective of computational linguistics, a grammar theory of HL has to satisfy adequately at least the following requirements:

1. linguistic generalizations
2. computational efficiency
3. correctness of the linguistic information (grammaticality in a generalized sense, with respect to syntax-semantics interface, e.g., semantic relations between active and passive forms of sentences)

CFGs do not meet adequately any of the above requirements, by having to introduce enormous number of rules to meet syntactic correctness, with respect to to limiting under-generation and over-generation.

2.3 Some Conclusions about CFGs

• Positive features of CFGs:

1. finite representation of infiniteness by recursion
2. efficient parsing based on CFGs — $O(n^3)$
3. CFGs associate syntactic structure to expressions
4. Why do we need parse trees?
5. the difference between context-free parsing for programming languages and context-free parsing for HL¡.

• Negative features of CFGs:

1. syntactic redundancy:
   (a) spurious parse trees
   (b) redundancy in constraining the selectional co-occurrence restriction (for example, w.r. to complements and agreement requirements), and by this:
   (c) lack of linguistic generalizations
   (d) CFGs fail to represent the important property of headedness.
   What is a head of a grammatical structure? Intuitively, the head of an expression is the unit which contributes the core grammatical properties of the expression.
   What is the head of the expressions represented by the tree structures that we have considered?
   (e) diverging into non-efficient set of rules
2. semantic deficiency: CFGs are not enough powerful for representing all possible ambiguities, in particular, they are inadequate for quantificational and intentional scoping, etc.
3. descriptively adequate, but explanatory inadequate

3 The constraint-based, lexicalist grammar (CBLG)

The variety of CBLG approaches Currently, the constraint-based, lexicalist (CBL) approaches to formal grammar, encompass grammar theories such as:

- Categorial Grammar
- Construction Grammar
- Dependency Grammar
- Generalized Phrase Structure Grammar
- Lexical Functional Grammar
- Head-Driven Phrase Structure Grammar
- Grammatical Framework GF

  GF pertains to or has potentials for some of the characteristics of CBLG. In the same time it exceeds the above approaches in its theoretical foundations and its applications to HLP.

Characteristics of the architecture of CBLG In this section, we summarise some of the major characteristics of the above approaches to formal grammar, without GF, into a generalised approach, which we call Constraint-Based Lexicalist Grammar (CBLG). The relation of GF with these characteristics is a subject of research.

The characteristics of CBLG cover major linguistic concepts.

- What are the grammar components of CBLG?
  - principles
  - rules
  - lexicon
  - type hierarchy
  - constraints
- How do they work together?

  The components of the grammar are defined to work mutually in an unified way via the formal definitions of:
  - well-formed feature structures
  - well formed tree structures
- How does agreement work in CBLG?
- The notion of a *head* facilitates representing grammatical agreement.

Here we give an introduction to major linguistic notions by using the formal approach to syntax in Sag et al. [12].

4 Mathematics of CBLG

Existing formalizations of CBLG Here we present a generalization of some of the distinctive characteristics of a class of approaches to formal syntax, encompassing

- *GPSG:* Gazdar et al. 1979-87 to the present
- *LFG:* 1979 to the present
- *HPSG:* 1984 to the present
- *GF:* 1998, to the present

A detailed overview of these approaches to formal grammar with references to literature is given by the Appendix B in Sag et al. [12]. We mention only the most complete and prominent formalizations of HPSG as an approach to large-scale grammar closest to this overview.

- The most complete works on formalization of HPSG by mathematical logic:
  - Mark Johnson, 1988 [4]
  - Paul J. King, 1989 [5]
  - Bob Carpenter, 1992 [1]
  - Gerald B. Penn, 2000 [8]
  - Frank Richter, 2004 [11]
- The most complete works on formalization of GF:
  - Aarne Ranta, 1994 [9, 10]

  Note: I would classify GF as a new direction in (a new kind of) CBLG, under active development. While GF may not be classified as lexicalist, in the currently typical classification, GF has characteristics and module for lexicon, for which we consider it lexicalized.

The formal syntax in CBLG, by Sag et al., 2003 [12], is based on models of typed functions, see:

- Ch.9 [12]
- a manuscript by Roussanka Loukanova [7]

5 CBLG as Formal Grammar System

Informally

- Linguistic information of various kinds (lexical, syntactic and semantic) is represented by *typed feature structure descriptions* and *typed feature structures*.
- The feature structure descriptions are recursively determined sets of feature-value pairs represented by feature-value matrices, or attribute-value matrices (AVM).

  AVM are popular descriptions in linguistics, esp., in computational linguistics.
- The values of the features in a feature structure description can be either atomic or more complex feature structure descriptions.
- The formal system of CBLG, as introduced in Sag at al. [12],see Ch.9, and Loukanova [6, 7], consists of a formal language and its functional models.

The formal CBLG language The language of CBLG represents linguistic concepts in a hierarchically organized, typed system of expressions of a *formal feature-value language*.

- *Feature structure descriptions* are well-typed, well-formed expressions of this language.
- The well-formed expressions satisfy *grammar constraints*, also expressed in this language.
- A constraint of a given CBLG grammar is a partial representation of a class of well-formed entities. Typically, a constraint is a partial representation of linguistic information.

Models of CBLG Te model domains consist of typed, feature-value functions, called *feature structures*. *Feature structures* are recursive, total functions defined over atomic features. The values of these functions are also feature structures. The constant functions represent atomic feature structures. Typed, well-formed feature structure descriptions denote feature structures. A feature-value function represents

- linguistic information
- the grammatical structure of a well-formed HL expression (e.g., of a sentence)
- Each feature-value function is totally defined over its domain, and thus, represents well-formed total information.

For more details, see Sag at al. [12], see Ch.6–9, and the Appendices.

5.1 Examples

(2) a. $\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{verb} & \\ \text{AGR} & \begin{bmatrix} \text{NUM} & \text{sg} \\ \text{PER} & \text{3rd} \end{bmatrix} \end{bmatrix} \end{bmatrix}$

b. $\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \begin{bmatrix} \text{NUM} & \text{sg} \\ \text{PER} & \text{3rd} \end{bmatrix} \end{bmatrix} \end{bmatrix}$

c. $\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \begin{bmatrix} \text{NUM} & X \\ \text{PER} & Y \end{bmatrix} \end{bmatrix} \end{bmatrix}$

d. $\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \begin{bmatrix} \text{NUM} & \text{pl} \end{bmatrix} \end{bmatrix} \end{bmatrix}$

e. $\begin{bmatrix} \text{HEAD} & \textit{noun} \end{bmatrix}$

f. $\begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{pos} & \\ \text{AGR} & \textit{agr-cat} \end{bmatrix} \end{bmatrix}$

g. $\begin{bmatrix} \text{HEAD} & \textit{pos} \end{bmatrix}$

6 CBLG Type Hierarchy

6.1 The Feature-structures

- Feature structures are total functions over atomic features, by the functional approach in Sag et al. [12], using feature structure descriptions as denoting typed functions.
- Each CBLG type introduces some features and constraints over the appropriate values of those features.
- Feature structures are associated with types in a hierarchical type system. The features and values of each feature structure are determined by its type.
- *Feature structure descriptions* are partial pieces of representations.

A constraint associated with a type τ is given in a formal language, the expressions of which are called feature structure descriptions.

Feature-structure descriptions are commonly denoted with attribute-value matrices.

$$\begin{bmatrix} \tau & \\ \text{FEATURE}_1 & \mathit{description}_1 \\ \dots & \\ \text{FEATURE}_n & \mathit{description}_n \end{bmatrix}$$

An examples for a feature-structure descriptions as a simple constraint over a type

$$\begin{bmatrix} \mathit{agr\text{-}cat} & \\ \text{PER} & \{\text{1st, 2nd, 3rd}\} \\ \text{NUM} & \{\text{sg, pl}\} \end{bmatrix}$$

An examples for a feature-structure description The following is a feature structure description of type *agr-cat*, which is an instantiation of the above constraint:

$$\begin{bmatrix} \mathit{agr\text{-}cat} & \\ \text{PER} & \text{3rd} \\ \text{NUM} & \text{pl} \end{bmatrix}$$

6.2 The Lexicon

Lexical entries and expressions can be organized by the following portion of a type hierarchy:

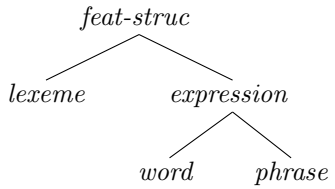


Figure 1: Layout of Lexicon Placement in the Type Hierarchy

1. The feature structures of subtype *lexeme*, intuitively, represent abstract lexical items that are inputs to inflectional and derivational lexical rules for producing specific words, i.e. feature structures of type *word*.

   There are lexical rules that take lexemes as input and "output" other lexemes, which can further undergo lexical manipulations before being used by the grammar rules.

2. The subtype *word* represents the basic lexical objects that are "ready" to be used in well-formed tree-structures.

3. The subtype *phrase* represents linguistic descriptions that are "derived" by grammar rules.

Each type introduces constraints on the feature structures of that type, i.e., each type declares:

- features that are appropriate for the type
- restrictions on the appropriate values for the features.
- Each feature structure of type τ has to be defined for $\textsc{feature}_1, \ldots, \textsc{feature}_n$.
- The values on $\textsc{feature}_i$ have to satisfy $description_i$ $(i = 1, \ldots, n)$.
- $description_i$ can be a set descriptor:
  $\{\textsc{value}_{i,1}, \ldots, \textsc{value}_{i,k_i}\}$ $(i = 1, \ldots, n)$, where each $\textsc{value}_{i,j}$ is a feature structure description.

  The values on $\textsc{feature}_i$ have to satisfy at least one of the descriptions $\textsc{value}_{i,j}$.

6.3 Words

The lexical entries that have gone through the lexical rules of type *i-rule*, for inflection, or *pi-rule*, for post-inflection changes, are of type *word*.

(5) $$\left\langle \text{walks}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

(6) $$\left\langle \text{writes}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle \text{NP} \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

(7) $$\left\langle \text{relies}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle \text{PP} \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

(8) $$\left\langle \text{put}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle \text{NP},\ \text{PP} \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

(9) $$\left\langle \text{eats}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle (\text{NP}) \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

(10) $$\left\langle \text{believes}, \begin{bmatrix} \textit{word} \\ \text{SYN} & \begin{bmatrix} \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \text{NP}\ \rangle \\ \text{COMPS} & \langle \text{NP} \mid \text{S} \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \right\rangle$$

There are three types of lexical rules, as subtypes of the type *l-rule*:

- *i-rule* is the type of the inflection rules, which formalizes a system of morphological inflection, e.g., variation of verb forms with respect to tense, number, person, and noun forms with respect to number, case, etc.

  The rules of type *i-rule* take as input entries of type *lexeme* and output structures of type *word*.

- *d-rule* is the type of the derivational rules.

  The lexicon is productive and the some lexical items are derived from others by lexical rules of type *d-rule*. E.g., these rules use morphological operators for morphological (orthographical) changes of the lexemes, depending on the HL. They take care for the relevant syntactic arguments of the lexemes and their corresponding semantic counterparts. E.g.,

  - Agent Nominalization Lexical Rule derives nominal lexemes, like "driver" from corresponding verbal lexemes, "drive".
  - Present (Past) Participle Lexical Rule
  - Passive Lexical Rule rearranges the syntactical arguments, by preserving their correspondences with the semantic representation.

- *pi-rule* is the type of the post-inflection rules that map lexical entries of type *word* to items of type *word*.

  An instance of a rule of type *pi-rule* is the Extraposition Lexical Rule for the words corresponding to the verbal lexemes in sentences like (11a)–(11b). The rule takes an INPUT entry of type *word* that has a complementizer phrase (CP) as its specifier SPR. I.e., the CP is its subject syntactic argument given in the list ARG-ST (see later).

  It moves the CP to an additional complement of the new, OUTPUT *word*, and adds a dummy pronoun "it" as the subject NP.

(11a) $[[\text{That dogs bark}]_{\text{CP}}]_{\text{SPR}}$ $[\text{annoys}]_{:\textit{pi-rule}[\text{INPUT}]}$ $[\text{people}]_{\text{COMPS}}$

(11b) $[\text{It}]_{\text{SPR}}$ $[\text{annoys}]_{:\textit{pi-rule}[\text{OUTPUT}]}$ $[\text{people } [\text{that dogs bark}]_{\text{CP}}]_{\text{COMPS}}$

The Extraposition Lexical Rule contributes significantly to grammar generalizations. A syntactic analysis that makes use of such a lexical rule is both simple and uses the general rules for phrasal structures. The phrasal structures of sentences like (11a)–(11b) are not generated by any additional phrasal analysis. These phrases use the same phrasal rules, e.g., HSR and HCR (see later). Foe these sentences, the rules HSR and HCR take correspondingly different instances of type *word* that have the same orthographical expression, i.e., "annoys".

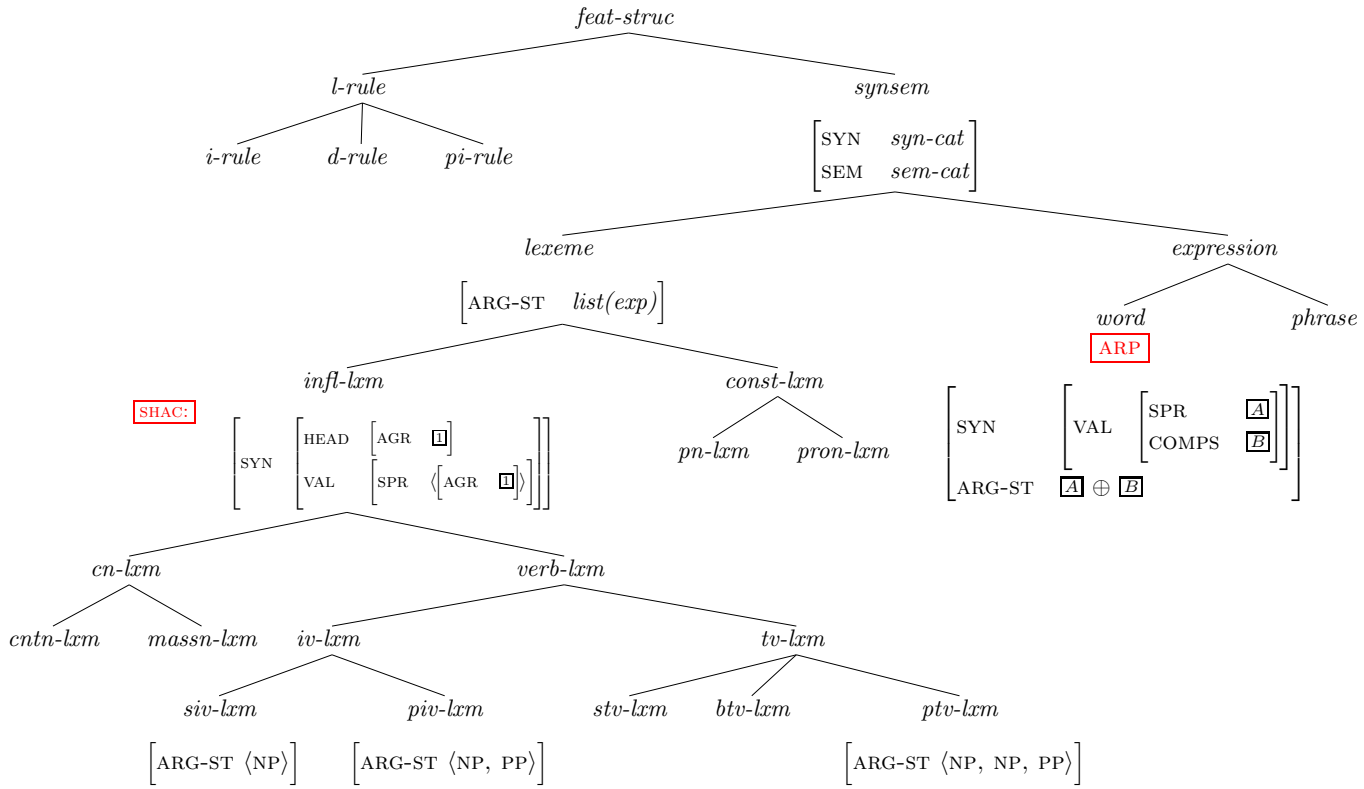


Figure 2: The Subtype Hierarchy of the Lexicon

The syntactic arguments of lexemes are given in the value of the feature ARG-ST, which is required for the types *lexeme* and *word*. E.g., see how ARG-ST determines the syntactic arguments of the lexical items of type *verb-lxm* in Figure 2.

The information about the arguments, given in the value of ARG-ST, is distributed in the VAL feature, via the Argument Realization Principle (ARP):

(12) **Argument Realization Principle (ARP)**

$$\begin{bmatrix} \textit{word} & & \\ \text{SYN} & \text{VAL} & \begin{bmatrix} \text{SPR} & \boxed{A} \\ \text{COMP} & \boxed{B} \end{bmatrix} \\ \text{ARG-ST} & \boxed{A}\ \textit{list(expr)} \oplus \boxed{B}\ \textit{list(expr)} & \end{bmatrix}$$

where $\boxed{A}$ and $\boxed{B}$ are restricted to be of type $\textit{list(expr)} \equiv \textit{list(expression)}$, for lists of feature structure descriptions of type *expression*, and $\oplus$ is the operation of appending lists.

Note that the availability of the feature VAL with a value that is a feature structure of type *val-cat* is a constraint required for the type *syn-cat*. The type *val-cat* is constrained to determine the valance lists SPR, COMPS, and MOD:

(13)
$$\begin{bmatrix} \text{VAL} & \begin{bmatrix} \textit{val-cat} & \\ \text{SPR} & \textit{list(expression)} \\ \text{COMPS} & \textit{list(expression)} \\ \text{MOD} & \textit{list(expression)} \end{bmatrix} \end{bmatrix}$$

6.4 Subject-verb agreement and determiner-noun agreement

Grammatical agreement is guaranteed by the Head Specifier Rule (HSR) (see (17)), the Head Feature Principle (HFP) (see (7.4)), the Head Specifier Agreement Constraint (HSAC) (see Figure 2), and Argument Realization Principle (ARP) (see (12)):

- The HSAC constraint (see Figure 2) propagates to the corresponding entry of type *word* via the ARP (see (12)). See the lexical entries (14), (15).
- Then the HSAC constraint propagates upwards in the phrase via the HFP (see (7.4)).
- By the HSR (see (17)), The VAL feature SPR requirement of the head (e.g., *verb* or *noun*) of a phrase must be satisfied by a matching SPR (NP or Det, respectively).
- The agreement features of the determiner "the" in (16) are underspecified. They will be constrained by the head *noun*, with which it will combine in an NP by the HSR.

(14)

$$\left\langle \text{dog}, \begin{bmatrix} \textit{word} & & \\ \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \boxed{1} \end{bmatrix} & \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \left\langle \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{det} & \\ \text{AGR} & \boxed{1}\ \textit{3sing} \\ \text{COUNT} & + \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \rangle \\ \text{COMP} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \right\rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \right\rangle$$

(15)

$$\left\langle \text{walks}, \begin{bmatrix} \textit{word} & & \\ \text{HEAD} & \begin{bmatrix} \textit{verb} & \\ \text{AGR} & \boxed{1} \end{bmatrix} & \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \left\langle \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \textit{noun} & \\ \text{AGR} & \boxed{1}\ \textit{3sing} \end{bmatrix} \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \rangle \\ \text{COMP} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \right\rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \right\rangle$$

(16)

$$\left\langle \text{the}, \begin{bmatrix} \textit{word} & & \\ \text{HEAD} & \textit{det} & \\ \text{VAL} & \begin{bmatrix} \text{SPR} & \langle\ \rangle \\ \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \right\rangle$$

7 Architecture Components of CBLG

7.1 The Type Organization of CBLG

The type system of CBLG, without the subtype hierarchy *l-rule* for the lexical rules, has a typical layout as in Figure 3.

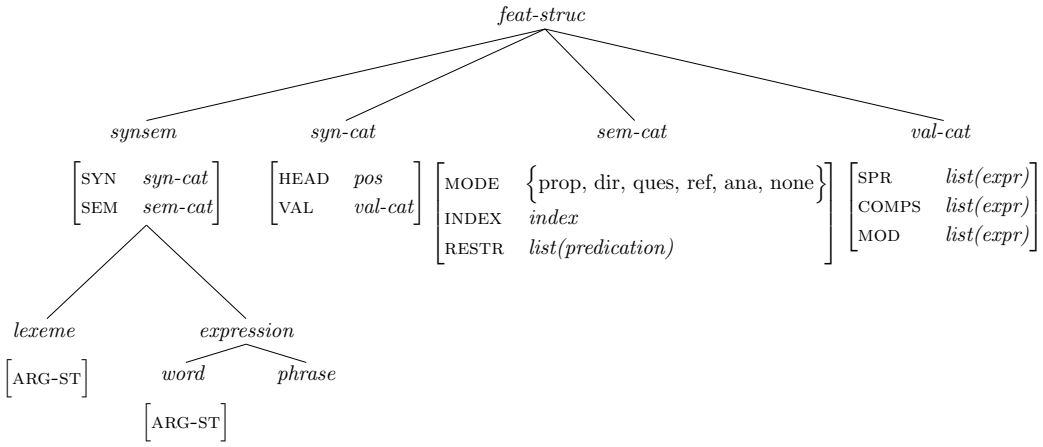


Figure 3: A Layout of Partial Type Organization

7.2 An example of a (small) realistic organization of the type hierarchy in CBLG

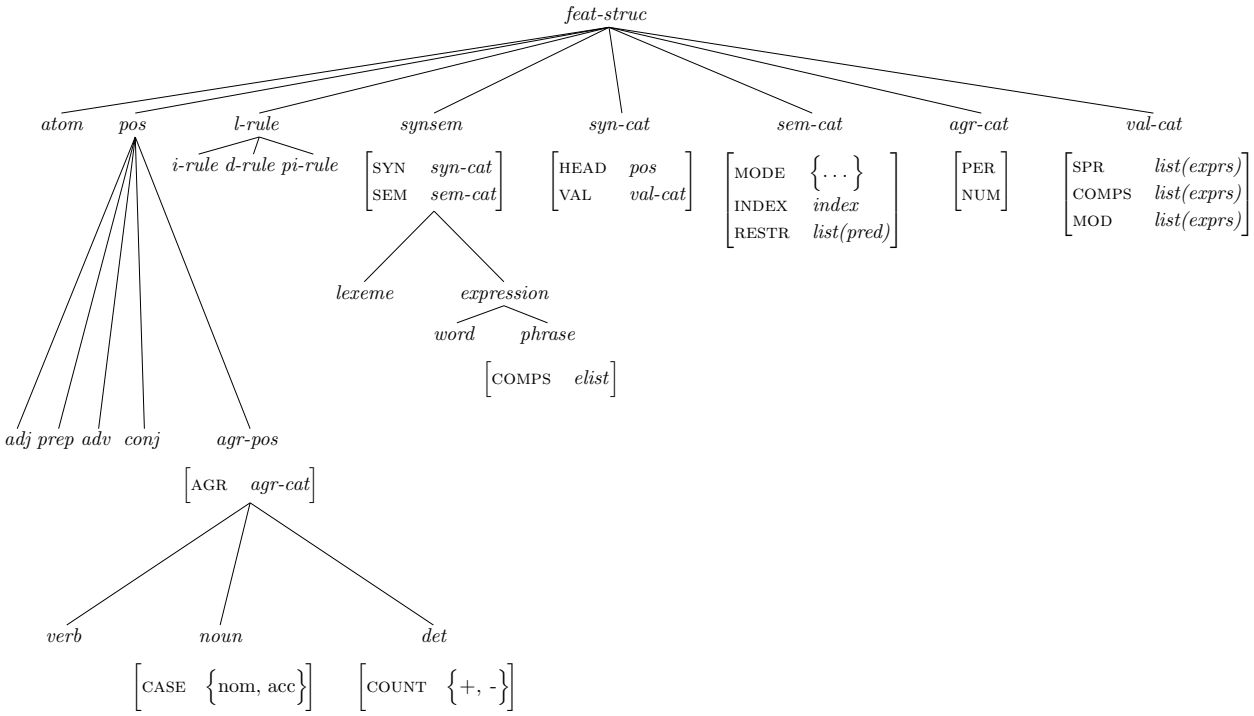


Figure 4: Partial Type Organization

7.3 Phrasal Rules of CBLG

(17) **Head Specifier Rule (HSR)**

A phrasal head preceded by its specifier $\boxed{1}$ forms a phrase:

$$\begin{bmatrix} \textit{phrase} \\ \text{SYN}\left[\text{VAL}\left[\text{SPR}\ \langle\ \rangle\right]\right] \end{bmatrix} \longrightarrow \boxed{1}\ \text{H}\left[\text{SYN}\left[\text{VAL}\begin{bmatrix}\text{SPR} & \langle\ \boxed{1}\ \rangle \\ \text{COMPS} & \langle\ \rangle\end{bmatrix}\right]\right]$$

(18) **Head Complement Rule (HCR)**

A lexical head (i.e., a feature structure of type *word*) followed by all its complements forms a phrase:

$$\begin{bmatrix} \textit{phrase} \\ \text{SYN}\left[\text{VAL}\left[\text{COMPS}\ \langle\ \rangle\right]\right] \end{bmatrix} \longrightarrow \text{H}\left[\begin{bmatrix} \textit{word} \\ \text{SYN}\left[\text{VAL}\left[\text{COMPS}\ \langle\ \boxed{1}, \ldots, \boxed{n}\ \rangle\right]\right] \end{bmatrix}\right] \boxed{1} \ldots \boxed{n}$$

where $n \geq 0$

(19) **Head-Modifier Rule (HMR) (post-modifier)**

A phrase can consist of a (lexical or phrasal) head followed by a compatible modifier:

$$[\textit{phrase}\ ] \longrightarrow \text{H}\boxed{1}\left[\text{SYN}\left[\text{COMPS}\ \langle\ \rangle\right]\right] \left[\text{SYN}\left[\text{VAL}\begin{bmatrix}\text{COMPS} & \langle\ \rangle \\ \text{MOD} & \langle\ \boxed{1}\ \rangle\end{bmatrix}\right]\right]$$

Note that MOD is a VAL feature:

- SPR, COMPS and MOD are similar w.r. to specifying co-occurrence restrictions on the lexical head.
- The way of imposing those restrictions for MOD is different from that of SPR and COMPS (compare with the HSR and the HSR w.r. to what is the head daughter).

(20)

Coordination Rule

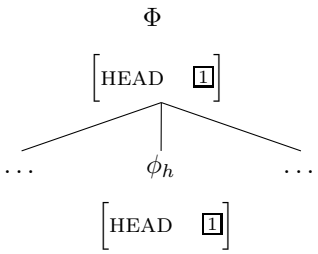
$$\begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{FORM} & \boxed{1} \end{bmatrix} \\ \text{VAL} & \boxed{0} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{IND} & s_0 \end{bmatrix} \end{bmatrix} \longrightarrow \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{FORM} & \boxed{1} \end{bmatrix} \\ \text{VAL} & \boxed{0} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{IND} & s_1 \end{bmatrix} \end{bmatrix} \ \ldots \ \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{conj} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{IND} & s_0 \\ \text{RESTR} & \langle\ [\text{ARGS}\ \langle s_1, s_2 \rangle\ ]\ \rangle \end{bmatrix} \end{bmatrix}$$

$$\begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \begin{bmatrix} \text{FORM} & \boxed{1} \end{bmatrix} \\ \text{VAL} & \boxed{0} \end{bmatrix} \\ \text{SEM} & \begin{bmatrix} \text{IND} & s_2 \end{bmatrix} \end{bmatrix}$$

7.4 Principles: HFP, VPr

Head Feature Principle (HFP) In any headed feature-structure, the HEAD value of the mother is identical to the HEAD value of the head daughter.

- A local tree structure Φ satisfies the HFP with respect to a headed rule ρ iff Φ satisfies:



  where ϕ_h is the head daughter of Φ.

Valence Features: spr, comps, mod The value of the feature MOD, like the values of SPR and COMPS, is "passed up" from the head daughter to the mother:

Valence Principle (VPr) Unless the rule dictates otherwise, the mother's values of the VAL features (SPR, COMPS, MOD) are identical to those of the head daughter.

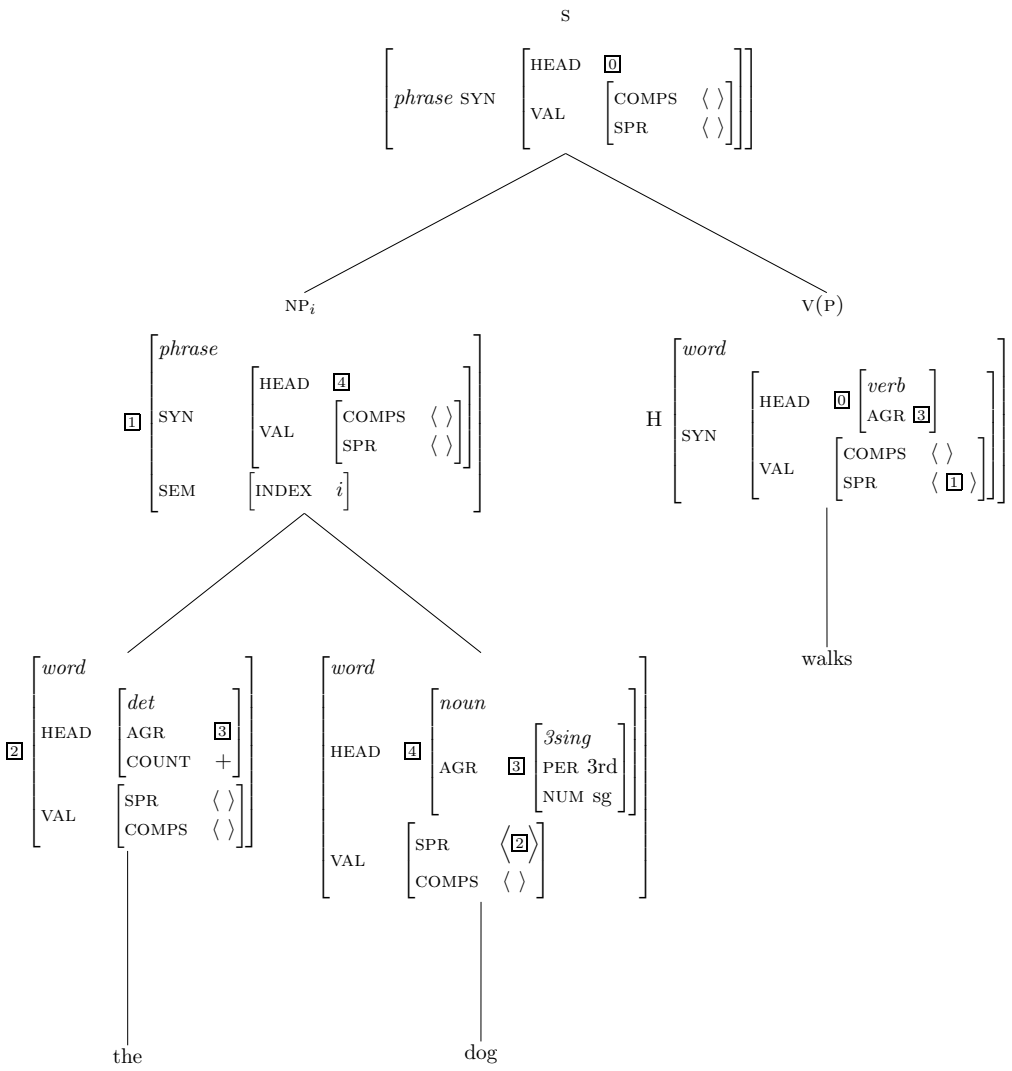
- The kind of expression a modifier can modify is determined by the MOD value of its head. This value is passed up to the modifier phrase by the Valance Principle.

- Packing together MOD, SPR, and COMPS as VAL features facilitates the statement of the Coordination Rule, which implies that:
  - the conjuncts have identical VAL values

  For an example with coordination see (pelle2).
- Unlike SPR and COMPS, no rule contradicts the Valence Principle with respect to the value of MOD.

Note: In the following sentence tree structure licensed by the HSR, the subject NP combines with a *feat-struc* of type *word*. This is possible according to the HSR because the COMPS requirements of the word *walks* are saturated: [VAL [COMPS $< >$]].

(21)


S
phrase SYN HEAD 0 VAL COMPS ⟨ ⟩ SPR ⟨ ⟩
NP$_i$
V(P)
1 phrase SYN HEAD 4 VAL COMPS ⟨ ⟩ SPR ⟨ ⟩ SEM INDEX i
H word SYN HEAD 0 verb AGR 3 VAL COMPS ⟨ ⟩ SPR ⟨ 1 ⟩
walks
2 word HEAD det AGR 3 COUNT + VAL SPR ⟨ ⟩ COMPS ⟨ ⟩
word HEAD 4 noun AGR 3 3sing PER 3rd NUM sg VAL SPR ⟨ 2 ⟩ COMPS ⟨ ⟩
the
dog

8 Generalizations Across Syntactic Categories via Valence Saturation

In CBLG, the symbols N, NOM, Det, DP, NP, V, VP, S, Adj, AdjP, Adv, and AdvP are abbreviations. Note how the phrases are distinctive from *words*, and with respect to saturation of valence features.

$$\textsc{v} = \begin{bmatrix} \textit{word} \\ \textsc{syn} \quad \begin{bmatrix} \textsc{head} \quad \textit{verb} \end{bmatrix} \end{bmatrix} \qquad \textsc{n} = \begin{bmatrix} \textit{word} \\ \textsc{syn} \quad \begin{bmatrix} \textsc{head} \quad \textit{noun} \end{bmatrix} \end{bmatrix}$$

Figure 5: Verb (V) and Noun (N) Feature Structure Descriptions

$$\textsc{Adj} = \begin{bmatrix} \textit{word} \\ \textsc{syn} \quad \begin{bmatrix} \textsc{head} \quad \textit{adj} \end{bmatrix} \end{bmatrix} \qquad \textsc{Det} = \begin{bmatrix} \textit{word} \\ \textsc{syn} \quad \begin{bmatrix} \textsc{head} \quad \textit{det} \end{bmatrix} \end{bmatrix}$$

Figure 6: Adjective (ADJ) and Determiner (DET) Feature Structure Descriptions

$$\textsc{s} = \begin{bmatrix} \textsc{syn} & \begin{bmatrix} \textsc{head} & \begin{bmatrix} \textit{verb} \\ \textsc{form} \quad \text{fin} \end{bmatrix} \\ \textsc{val} & \begin{bmatrix} \textsc{comps} & \langle\ \rangle \\ \textsc{spr} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad \textsc{np} = \begin{bmatrix} \textsc{syn} & \begin{bmatrix} \textsc{head} & \textit{noun} \\ \textsc{val} & \begin{bmatrix} \textsc{comps} & \langle\ \rangle \\ \textsc{spr} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 7: Saturated Phrases: S and NP Feature Structure Descriptions

$$\textsc{vp} = \begin{bmatrix} \textsc{syn} & \begin{bmatrix} \textsc{head} & \textit{verb} \\ \textsc{val} & \begin{bmatrix} \textsc{comps} & \langle\ \rangle \\ \textsc{spr} & \langle\ \textsc{x}\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad \textsc{nom} = \begin{bmatrix} \textsc{syn} & \begin{bmatrix} \textsc{head} & \textit{noun} \\ \textsc{val} & \begin{bmatrix} \textsc{comps} & \langle\ \rangle \\ \textsc{spr} & \langle\ \textsc{x}\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

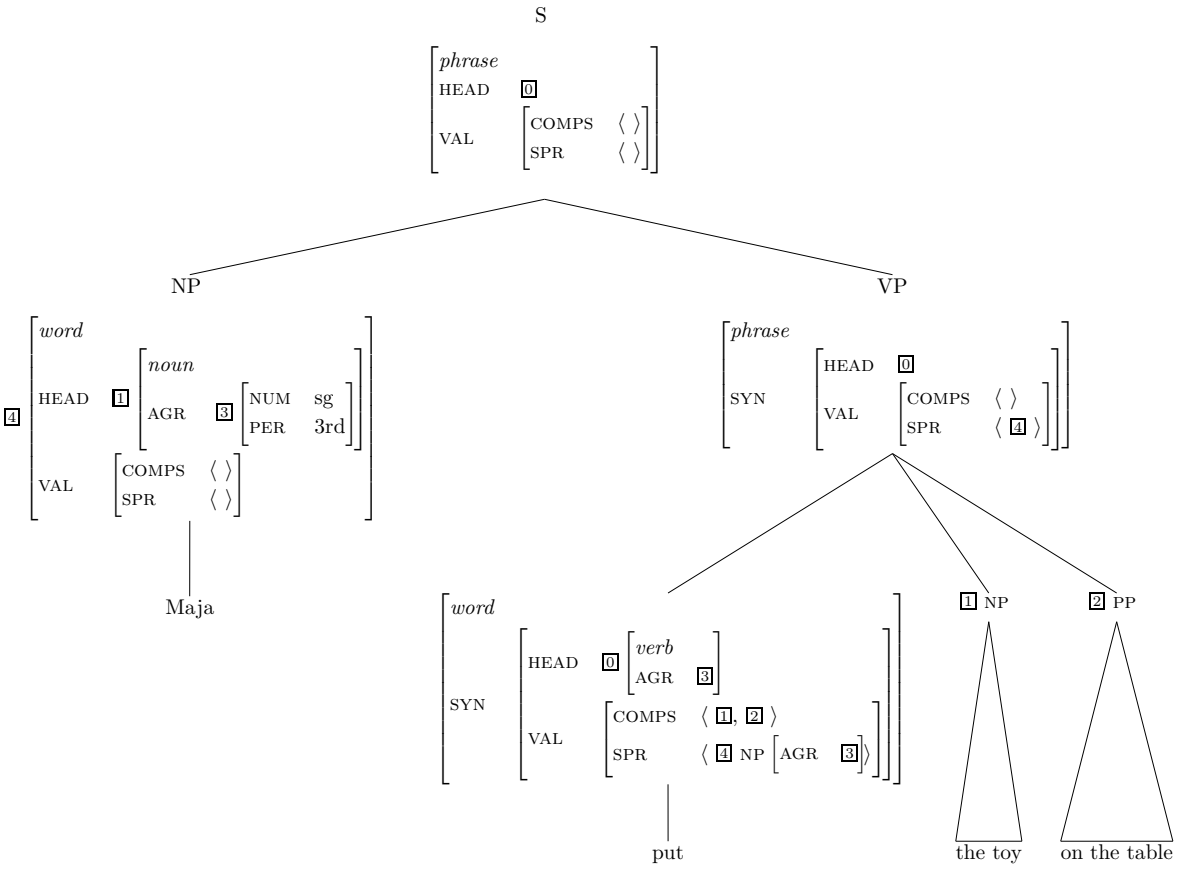
Figure 8: Semi-saturated Phrases: VP and NOM Feature Structure Descriptions

$$\text{PP} = \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{prep} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix} \qquad \text{AdjP} = \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{adj} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

$$\text{DP} = \begin{bmatrix} \text{SYN} & \begin{bmatrix} \text{HEAD} & \textit{det} \\ \text{VAL} & \begin{bmatrix} \text{COMPS} & \langle\ \rangle \\ \text{SPR} & \langle\ \rangle \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 9: Phrases: PP, AdjP and DP Feature Structure Descriptions

(put)



9 Modifiers

Lexical adjectives and adverbs have strong co-occurrence requirements w.r. to what items they modify. (The examples are from Sag et al. [12].)

(3) a. A cat came yesterday.

b. *A cat yesterday came.

(4) a. The person responsible confessed.

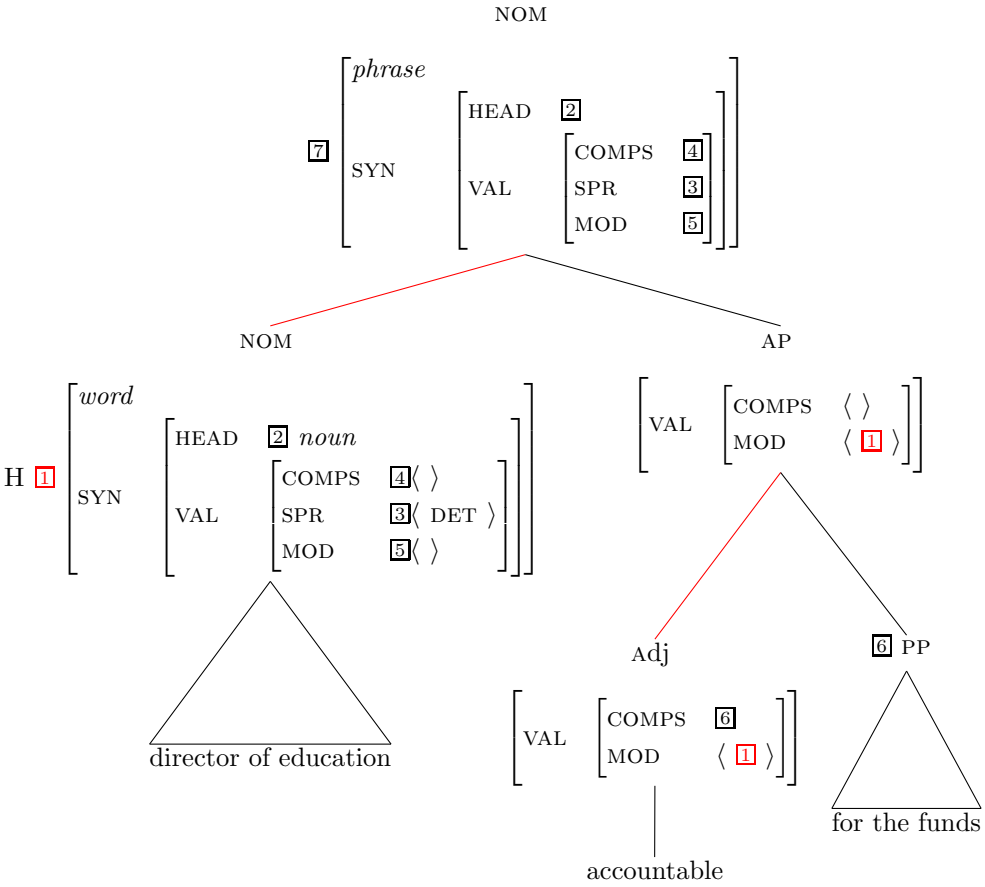
b. *The person confessed responsible.

- MOD is a VAL feature: why?
- The value of MOD is a list of *expressions*.
- The MOD value of a lexical item specifies what kind of expression this item can modify.
  - adjectives are specified as:
    [VAL [MOD ⟨ NOM ⟩]] or [VAL [MOD ⟨ NP ⟩]]
  - adverbs are specified as:
    [VAL [MOD ⟨ VP ⟩]] or [VAL [MOD ⟨ S ⟩]]
  - non-modifier items are specified as:
    [VAL [MOD ⟨ ⟩]]

MOD is a VAL feature

- SPR, COMPS and MOD are similar w.r. to specifying co-occurrence restrictions on the lexical head.
- The way of imposing those restrictions for MOD is different from that of SPR and COMPS (compare the phrasal rules in (17), (18), and (19), with respect to what is the head daughter).
- MOD like SPR and COMPS is "passed up" from the head daughter to the mother, by the Valence Principle (VPr), (7.4).

An Example for a modifier with a complement
(22)



Note:

- HMR does not requite anything about MOD in 7.
- ∴ By the VPr, all VAL values of the moder structure 7 are identical to the VAL values of its head NOM structure 1.
- ∴ The MOD value 1 of the phrasal modifier AP does not get passed up to the top mother structure 7 NOM after the HMR has been applied.

10 Agreement and the notion of a *head*

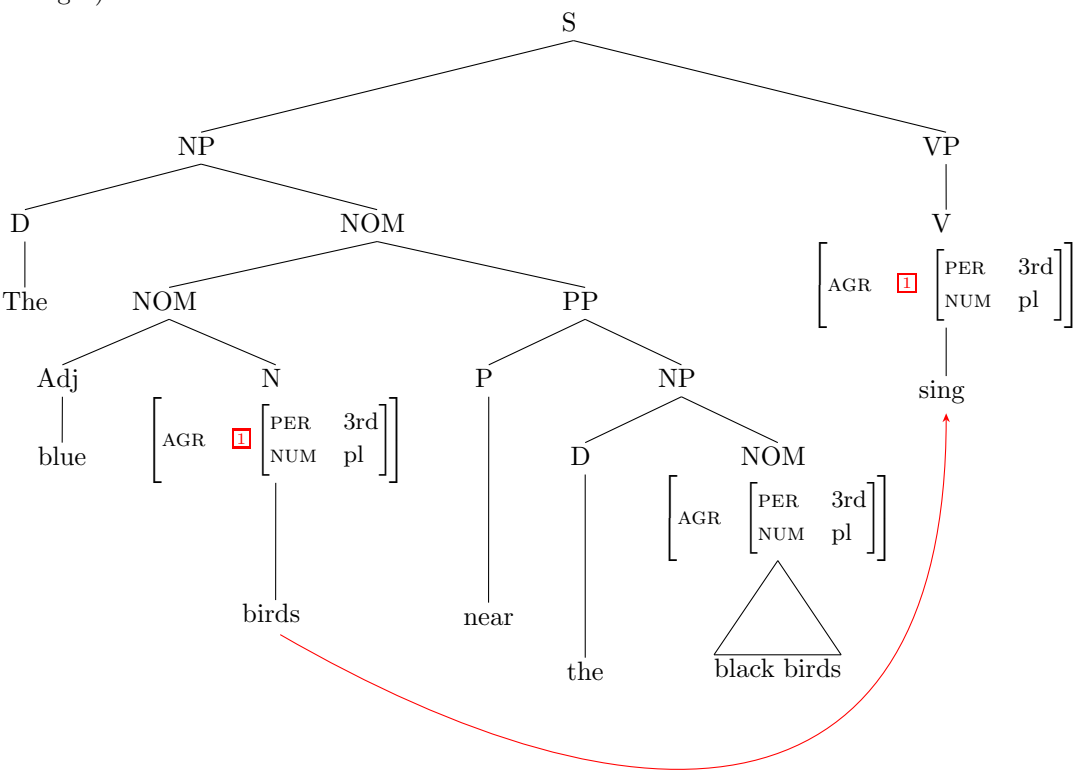
- The notion of a *head*: intuitively, the head of a given expression is the sub-expression, which contributes major linguistically significant information.
- E.g, the head daughter determines the *pos* of the mother (or the other way around):

  The *pos* of a NP is *noun* because its head daughter is a N, or a NOM, the *pos* of which is *noun*.

The head feature-structure is determined by:

- the feature HEAD in the type hierarchy
- the Specifier-Head Agreement Constraint (SHAC)
- Head Specifier Rule (HSR)
- Head Complement Rule (HCR)
- Head Feature Principle (HFP)

(birds-agr0)



11 More about the notion of a *head*

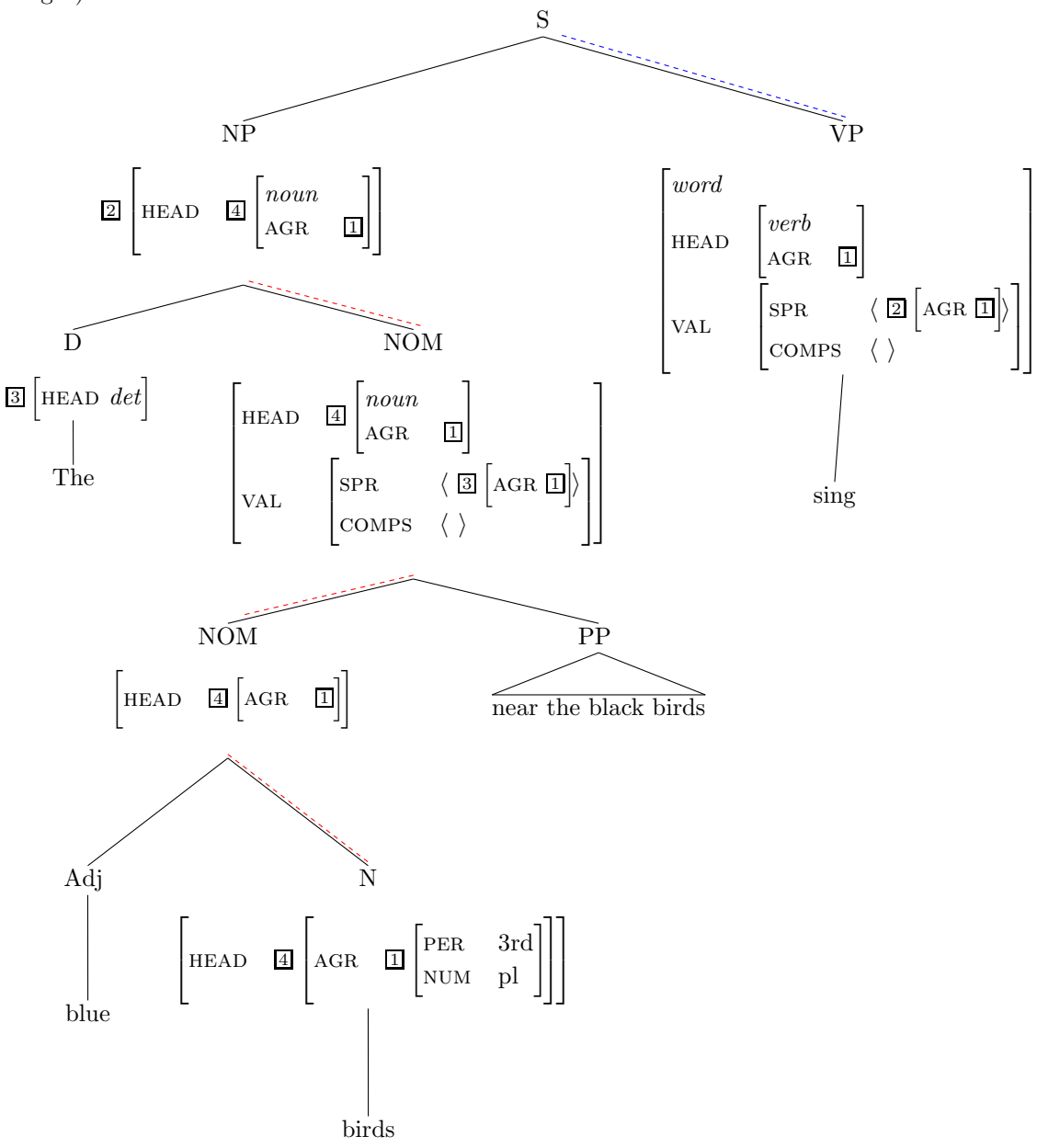
The notion of a head has the following variants:

- a lexical head
- a phrasal head
- a local head in a tree structure.

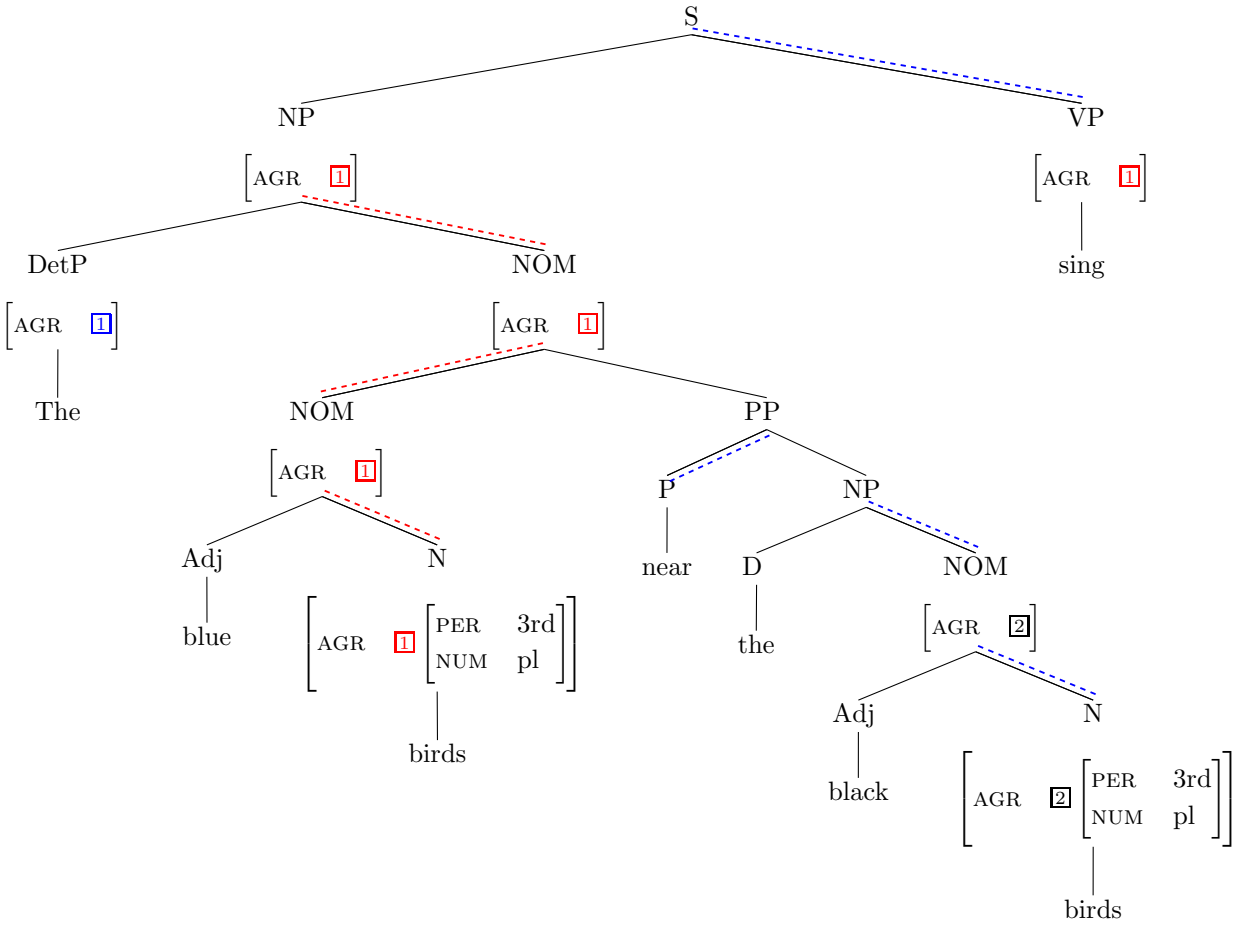
See examples (birds-agr1), (birds-agr:HeadPropagation), (pelle2).

11.1 Propagation of the Head

(birds-agr1)

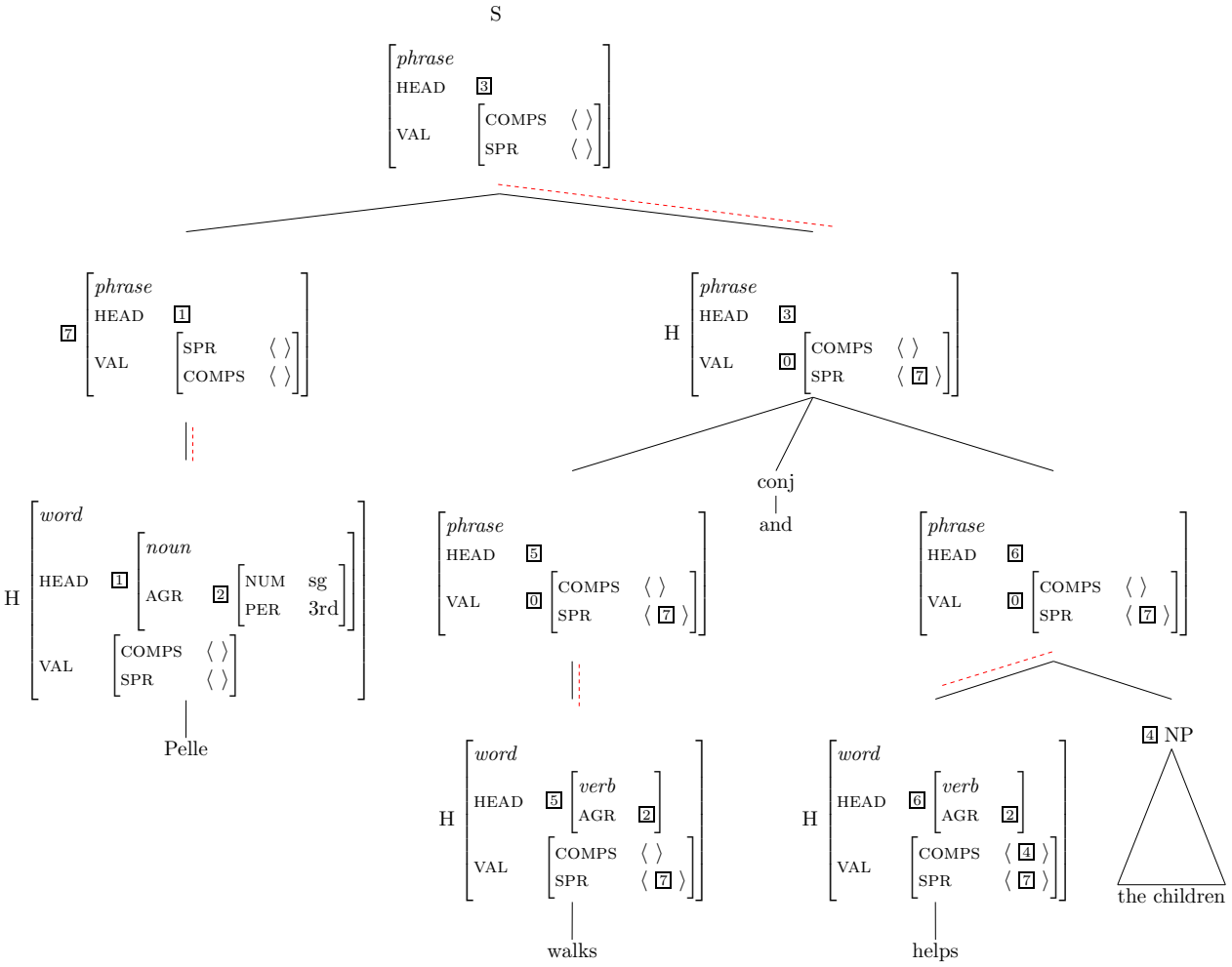


(birds-agr:HeadPropagation)



12 Head Propagation and Coordination: an Example

(pelle2)



13 Factorization of Grammatical Information in CBLG

- The CFGs categories (nonterminals) are elaborated into feature structure descriptions.

  A recursive definition of the set of feature structures: a set of pairs of features and values. The values of the features can be either atomic entities (for example, sg, pl) or other feature structures.

- The grammar and lexical rules have very general forms cutting across grammatical categories: the relevant particularities get filled up during application of the rules for building up well-formed tree structures.

  In this introduction to CBLG, we have only four rules: the HSR, the HCR, a simple HMR for post-modifiers, and the Coordination Rule. These rules are already a major core of the grammar apparatus sufficient for a reasonable coverage of English. However, a CBLG that covers a serious fragment of HL has far more specialized rules.

- The majority of the specific syntactic information about the lexical items is moved into the hierarchical organization of the lexicon. Most of the lexical information is not included directly into the individual lexical entries. It is filtered out by making appropriate generalizations and distributed hierarchically over the lexical types.

- Very minimal, specific syntactic information is left into the lexemes. E.g., the specific co-occurrence requirements are given in the lexical entries via hierarchical inheritance, as in Figure 2.

- The feature structures of the lexemes introduce their semantic representations, which are specific for each individual lexeme.

- The grammar principles facilitate the distribution of information (features and their values) in well-formed trees, e.g., such principles are the HFP, the VP.

References

[1] Bob Carpenter. *The Logic of Typed Feature Structures*, volume 32 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge University Press, 1992.

[2] Gottlob Frege. Über Sinn und Bedeutung. *Zeitschrift für Philosophie und philosophische Kritik*, C:25–50, 1892.

[3] Gottlob Frege. On sense and nominatum. In Herbert Fiegl and Wilfrid Sellars, editors, *Readings in Philosophical Analysis*, pages 85–102. Appleton-Century-Crofts, 1949.

[4] Mark Johnson. *Attribute-Value Logic and the Theory of Grammar*, volume 16 of *CSLI Lecture Notes*. CSLI Publications, Stanford, California, 1988.

[5] Paul John King. *A logical formalism for Head-Driven Phrase Structure Grammar*. PhD thesis, University of Manchester, 1989.

[6] Roussanka Loukanova. An Approach to Formal Models of Constrained Based Lexicalized Grammar. In *Proceedings of International Workshop on Non-classical Formal Languages in Linguistics (ForLing 2007)*. Computer and Automation Research Institute of the Hungarian Academy of Sciences (MTA SZTAKI), Budapest, 2007.

[7] Roussanka Loukanova. An Approach to Functional Formal Models of Constraint-Based Lexicalized Grammar (CBLG). draft, 2010.

[8] Gerald B. Penn. *The Algebraic Structure of Attributed Type Signatures*. PhD thesis, School of Computer Science, Carnegie Mellon University, 2000.

[9] Aarne Ranta. *Type-theoretical Grammar*. Oxford University Press, 1994.

[10] Aarne Ranta. Grammatical framework: A Type-Theoretical Grammar Formalism. *Journal of Functional Programming*, 14(2):145–189, 2004. URL: http://www.cse.chalmers.se/~aarne/publications.html.

[11] Frank Richter. *A mathematical formalism for linguistic theories with an application in Head-driven Phrase Structure Grammar*. PhD thesis, Eberhard Karls Universität Tübingen, 2000.

[12] Ivan A. Sag, Thomas Wasow, and Emily M. Bender. *Syntactic Theory: A Formal Introduction*. CSLI Publications, Stanford, California, 2003.