Chapter 2

Core concepts of LFG

Oleg Belyaev

Lomonosov Moscow State University, Institute of Linguistics of the Russian Academy of Sciences, and Pushkin State Russian Language Institute

This chapter provides an in-depth coverage of the main features of the LFG framework, focusing mainly on its syntactic representations: c- and f-structure. The makeup of each level is discussed in detail. For c-structure, I describe the version of X′ theory used in LFG and the status of lexical integrity as a core principle of the framework. I discuss the notion of f-structure as a function/set of feature-value pairs that is used in the majority of LFG work; attribute value types and well-formedness conditions on f-structure (Uniqueness, Completeness and Coherence) are covered as well. I also describe the metalanguage for defining f-structures and the mapping from c- to f-structures, and note some linguistically relevant consequences of how this mapping is organized. Three proposed extensions of the standard architecture are also discussed: templates (constructions), minimal c-structure, and lexical sharing.

1 Introduction

This chapter provides a detailed survey of the main syntactic levels of LFG, constituent structure (c-structure) and functional structure (f-structure). It complements the more general introduction in Belyaev 2023b [this volume]. In Section 2, I describe the c-structure model used in standard LFG, its understanding of constituency, and the role of X′ theory. In Section 3, the notion of f-structure is discussed, including the metalanguage used for describing f-structures and constraints on possible f-structure. In Section 4, I discuss the mapping from c- to f-structure. Finally, in Section 5 I describe recently proposed modifications to the basic architecture of LFG that have not yet been universally accepted, but which may shape the development of this framework in the future.

Oleg Belyaev. 2023. Core concepts of LFG. in Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 23–96. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185936

2 C-structure

The nature of constituent structure (c-structure) in LFG and its main properties are summarized in Belyaev 2023b [this volume]. Briefly, c-structure is a phrase structure tree; constraints on possible trees are usually described via context-free rules as in (1). Other metalanguages are sometimes used as well.

(1) S $\longrightarrow$ NP VP

An important feature of c-structure in LFG is that empty nodes are not usually employed. This is not a limitation imposed by the framework itself, but a theoretical decision. It is formally possible to define grammars with null terminal nodes in LFG: this is implemented in XLE (Crouch et al. 2011) and was used to capture long-distance dependencies in early versions of LFG (Kaplan & Bresnan 1982). However, since Kaplan & Zaenen (1989b) it has become a universal practice to capture long-distance dependencies through functional uncertainty at f-structure, and the use of empty categories at c-structure has become unnecessary (see Kaplan 2023 [this volume]). For more information on the formal features of c-structure in LFG, see Andrews 2023 [this volume].

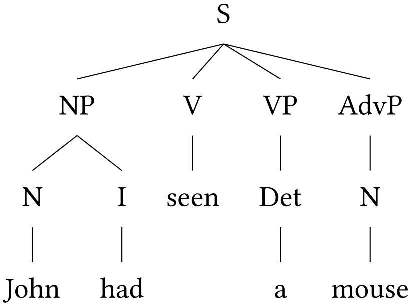
Without additional theoretical restrictions, context-free grammars allow far more possible phrase structure trees than actually attested in natural languages. In this section, I will focus on two main constraints on c-structure in LFG: X-Bar Theory and lexical integrity.

2.1 X′ Theory

Every theory of constituency based on phrase structure grammar faces what Everett (2015), in his review of Adger (2013), called “Lyons’ Problem”. Lyons (1968) famously asked what guarantees that NPs are headed by Ns, VPs are headed by Vs, etc., such that rules like VP → … V … or NP → … N … are allowed, but rules like NP → … V … are not.

Indeed, from the point of view of context-free rules, VP and V are atomic symbols that are not related to each other; labeling one of the daughters of NP as N is merely a convention, and nothing in the formalism excludes a hypothetical language with constituent structures like in (2) – “monsters” in Bresnan et al.’s (2016) terms.

(2)



Intuitively, there are many things that are wrong with this structure: an I head cannot be the daughter of NP; the VP cannot be headed by, or even immediately dominate, a Det; an AdvP cannot be headed by a noun.[1] The principle that prohibits this is called endocentricity; roughly stated, it means that the external distribution of a phrase (e.g. NP) is determined by the category of one and only one of its daughters, the head. Disallowing non-endocentric structures requires a theory of constituent structure labels that limits the range of available configurations. To this end, X-bar (X') theory has been proposed in mainstream generative grammar (Chomsky 1970, Jackendoff 1977).

X' theory enforces endocentricity by introducing the notion of projection and "bar level" and requiring that each non-maximal projection (X^0 and X'; X'', or XP, is usually assumed to be the maximum level of projection) be dominated by a node belonging to the same category, with the bar level either incremented by one or unchanged. The sisters of c-structure heads (complements, specifiers and adjuncts) have to be maximal projections or non-projecting words (on which see below).

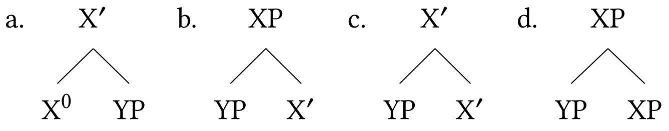
One variant of X' theory has been adopted in LFG from the very early days and continues to be used in most LFG work. An in-depth exposition of X' theory as it is used in LFG, with certain additional theoretical innovations, can be found in Bresnan et al. (2016). The most important features of X' theory as it is practiced in LFG are as follows. First, as in the original formulation, X' theoretical constraints are viewed as constraints on phrase structure *rules*; the later GB view of a kind of universal "X' schema" has not gained acceptance in LFG, primarily because

[1] Curiously, each of the features of this illustration *ad absurdum* has a counterpart in real languages: noun phrases do sometimes mark the tense of their clauses, verbs do mark the definiteness of their arguments, and bare nouns (although probably not nouns like 'mouse') are used adverbially. But there is broad consensus in theoretical lingustics that such phenomena are more exceptions than rules and should *not* be modeled by allowing the theory of phrase structure to license such configurations.

the architecture of the framework is fundamentally based on language-specific rules and does not allow such schemas.

Second, X′ theory in LFG allows for the following positions: complement (3a), specifier (3b), X′ adjunct (3c) and XP adjunct (3d).[2]

(3)



As in all versions of X′ theory, only maximal projections may appear in these positions.

There is some disagreement concerning the possibility of X′ adjunction: While most authors accept both kinds of adjunction, Toivonen (2003) only allows XP-adjunction (and head adjunction, see below) because in her theory only constituents of the same bar level may be adjoined.

The LFG literature also generally allows for multiple complements and specifiers dominated by the same mother node; thus, a sequence of several phrases instead of YP is possible in (3a–c); multiple adjuncts in one position are also usually allowed, even though this creates redundancy since this structure could always be replaced by multiple binary adjunction.

Third, LFG uses the following functional projections: DP for NPs, IP and CP for VPs. Some work also uses additional phrases, such as KP/CaseP for clitic case markers (Broadwell 2008). The number of functional positions is limited compared to mainstream theories, and this is not merely a stipulation: LFG requires all constituency in a given language to be empirically motivated in a way that is more narrow than in frameworks that represent the bulk of syntactic information in phrase structure (such as transformational frameworks). Specifically, heads may only be stipulated if there is actual lexical material that can occupy them; therefore, even the existence of projections such as CP or IP cannot be automatically assumed for all languages. More abstract projections such as TopicP or ForceP are not usually introduced because there are few suitable candidates for the status of heads of these phrases, and little distributional evidence to argue that their specifiers are distinct structural positions.

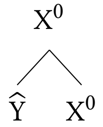
It turns out, in fact, that the set of functional projections listed above is fully adequate for the overwhelming majority of languages. Moreover, some categories, like DP, are not viewed as universal; authors, such as Sells (1994) for Japanese and

[2] The order of constituents is only an illustration; X′ theory itself does not impose any specific order.

Korean, even limit the number of projections to one (X′) instead of the standard two.

Fourth, LFG admits non-projecting words, i.e. lexical items that do not project X′ and XP levels and hence cannot have complements or specifiers; their maximum projection level is 0. The category of non-projecting words is marked as X^0. Toivonen (2003) develops a detailed theory of non-projecting words. Being maximal projections, they can appear at any non-head X′ theoretic positions (i.e. specifier, complement, or adjunct), but the only dependents that they may have are X^0 adjuncts, which must themselves be non-projecting. Thus, an additional type of adjunction – head adjunction – is introduced into X′ theory, illustrated in (4), where X^0 can also be $\widehat{X}$, but, crucially, $\widehat{Y}$ cannot be Y^0, as that would violate the principle that only maximal projections can appear in non-head positions.

(4)



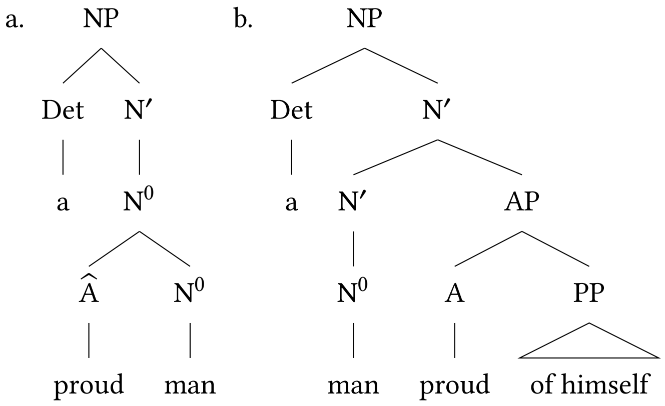
The theory of non-projecting words presented in Toivonen (2003) further requires that only same-level projections are adjoined; this effectively prohibits adjoining non-projecting words at X′ or XP level, as well as any adjunction at X′ level in languages where XP is the maximal projection (because only maximal projections can be adjuncts, as stated above). However, these more restrictive principles are not accepted by all authors who use non-projecting words in their analyses: for example, Spencer (2005) analyzes case markers in Hindi as $\hat{P}$ nodes adjoined to NP. X′ adjunction also remains quite common in LFG analyses.

Sadler & Arnold (1994) use non-projecting words to account for the behaviour of English prenominal adjectives, which cannot have phrasal complements if they are prenominal; consider the contrast between (5a) and (5b), while (5c) is ungrammatical.

(5)
  a. a **proud** man
  b. a man [**proud** of himself]
  c. * a [**proud** of himself] man

Sadler and Arnold argue that this contrast is due to the fact that prenominal adjectives in English are non-projecting words with the category $\widehat{A}$ that are adjoined to N^0, while postnominal adjectives form AP and can therefore have complements. Thus the structure of (5a) is (6a), while the structure of (5b) is (6b).

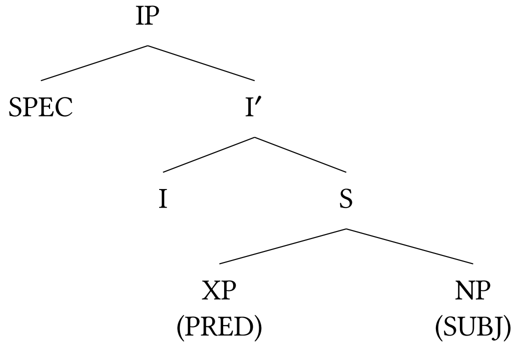
(6)



Finally, X' theoretic principles are not viewed as fully universal in LFG. The most prominent exception is the exocentric category S.[3] This category does not have a "head" in the normal sense: it can be "headed" by a verb, but also by an adjective or another nonverbal predicate; this is why the term S is used instead of, for example, VP. The category S is most extensively used in nonconfigurational languages (see Andrews 2023 [this volume]), but this is not its exclusive role. Many languages have a fairly configurational structure overall but allow predicates of various categories to be embedded under a general "predicative marker", which sits in the I or C node. For example, Kroeger (1993: 119) proposes the phrase structure in (7) for Tagalog. The SPEC position can be optionally occupied by fronted constituents of several types (such as topics); the I node is occupied by an auxiliary or the finite verb; the predicate XP can be a VP in verbal sentences, but can also be AP or NP if the predicate is nonverbal. Hence, the structure is indeed non-endocentric, and the use of the label S is justified.

[3] Bresnan et al. (2016: 112ff.) present the category S and non-projecting words as effectively the *only* exceptions from standard principles of X' theory. This, however, is a theoretical idealization insofar as it applies to actual LFG analyses, which routinely make us of *ad hoc* categories such as CL, CCL (for "clitic", "clitic cluster") in Bögel et al. (2010) and Lowe (2011). Such minor innovations do not seem to influence the overall theory in any meaningful way, since they deal with exceptional cases such as second-position clitics or language-specific, idiosyncratic linear order distributions. It is also conceivable that many of them could be converted to analyses that conform to X' theoretic principles; for example, CCL could be treated as a phrase consisting of multiple $\widehat{\text{D}}$ head adjunction (if the clitics are pronominal).

(7)



Since c-structure is not the only level of representation in LFG and models only a subset of syntactic phenomena (word order, embedding), X′ theory does not do much by itself to limit the range of possible languages. Unlike frameworks such as GB, for which the theory was originally devised, X′ positions are not inherently or uniquely associated with specific syntactic or semantic functions – as a result, X′ theory, understood purely in terms of c-structure, is little else than a system of labeling nodes which allows us to generalize endocentricity at constituent structure level. In order to make it more meaningful, it should be augmented by a set of principles that determine the mapping of X′ positions to f-structure – such a system has been developed in LFG, and will be described in Section 4.3.

2.2 Lexical Integrity

As its name implies, Lexical Functional Grammar was originally conceived as a lexicalist framework, a term that has several meanings. In the most general sense, lexicalism implies that the features of individual syntactic elements (morphemes and wordforms) as well as their subcategorization frames are determined in the lexicon, and cannot be modified in the syntax (such as by promoting the direct object in a passive construction). Lexicalism in this sense requires no additional stipulation and is enforced by the LFG architecture itself: there are no transformations or other means to change the c-structure or f-structure features; syntax can only mutiply define lexical features, but cannot override them.[4]

[4] Kaplan & Wedekind (1993) introduced the restriction operator: $f\backslash_{\text{A}}$ denotes the f-structure f with the attribute A and its value removed. As suggested by an anonymous reviewer, this violates lexical integrity in the weak sense, because here the syntax effectively accesses an f-structure constructed otherwise (possibly by means of morphology) to retrieve some of its information. This operator is not widely employed but was used in several LFG analyses, notably in Asudeh (2012) and Falk (2010).

LFG is also lexicalist in another sense: it subscribes to the idea that the building blocks of syntax are not roots or affixes, but individual words that are constructed from different blocks and according to different rules than syntactic constituents.[5] Thus, the distinction between morphology and syntax in LFG is viewed as fundamental, which is against the views of many recent approaches, both formal (Bruening 2018) and typological (Haspelmath 2011).

This understanding of lexicalism is more formally termed LEXICAL INTEGRITY and has been given two formulations in LFG (8)–(9).

(8) Words are built out of different structural elements and by different principles of composition than syntactic phrases. (Bresnan & Mchombo 1995: 181)

(9) Morphologically complete words are leaves of the c[onstituent]-structure tree and each leaf corresponds to one and only one c[onstituent]-structure node. (Bresnan et al. 2016: 92)

The definition in (8) is rather broad and can be compatible with several different understandings of the morphology–syntax interface, as long as the border between the two levels in maintained in some way. The second definition (9) is more specific and is only compatible with one view of the interaction between morphology and syntax. For example, lexical sharing, which allows one word to correspond to two X^0 nodes (discussed in Section 5.2.2), is compatible with (8) but not with (9).

Interestingly, despite the rather strict definition in (9), much work in LFG uses the concept of "sublexical nodes", like in the rule for Greenlandic nouns in (10), from Bresnan et al. (2016: 368). This is formally incompatible with (9) because the preterminal nodes correspond to morphemes, not morphologically complete nodes.

(10) $N \longrightarrow N_{\text{stem}} \quad N_{\text{aff}}$

In practice, such analyses are rather harmless because in their predictions they are equivalent to analyses that strictly adhere to (9): since all morphology is sublexical, the position of individual affixes cannot have any syntactic relevance, as opposed to approaches like Distributed Morphology (Halle & Marantz 1993), where morphological features often occupy higher-level functional projections

[5] These two understandings of lexicalism are sometimes conflated, but they are actually independent: A framework may be lexicalist in the former sense, but consider the distinction between words and syntactic phrases to be ephemeral.

that can scope over syntactic phrases. However, the use of sublexical representations does raise the issue of how the individual contribution of morphemes to f-structure should be represented – standard LFG does not provide such a way, because words are viewed as complete, unsegmented bundles of morphosyntactic information. These issues are discussed in detail in Asudeh & Siddiqi 2023 [this volume].

3 F-structure

3.1 The notion of f-structure

As described in Belyaev 2023b [this volume], c-structure in LFG is complemented by an additional level of representation called f-structure. F-structure is an attribute-value structure that includes information on valency, grammatical functions, and the features of clauses and their syntactic arguments. An f-structure for the English sentence *John has seen David* is given in (11).

(11)
$${}_{f}\left[\begin{array}{ll} \textsc{pred} & \text{‘}\textsc{see}\langle(f\ \textsc{subj})(f\ \textsc{obj})\rangle\text{’} \\ \textsc{tense} & \textsc{prs} \\ \textsc{aspect} & \textsc{perf} \\ \textsc{subj} & {}_{g}\left[\begin{array}{ll} \textsc{pred} & \text{‘}\textsc{John}\text{’} \\ \textsc{pers} & 3 \\ \textsc{num} & \textsc{sg} \end{array}\right] \\ \textsc{obj} & \left[\begin{array}{ll} \textsc{pred} & \text{‘}\textsc{David}\text{’} \\ \textsc{pers} & 3 \\ \textsc{num} & \textsc{sg} \end{array}\right] \end{array}\right]$$

F-structure is usually thought of as a set of attribute-value pairs, or a function that maps attribute names to their values. This understanding of f-structure has important implications for the architecture of LFG. Specifically, it implies that f-structures are solely and uniquely defined by their set of attribute-value pairs; there is no type system as in Carpenter (1992) or HPSG (Pollard & Sag 1994). Therefore, there is no such thing as two different f-structures having the same set of attributes and values; the notion of an empty f-structure is also problematic, because all empty structures are equivalent to each other.[6] This notion of

[6] Observe that the standard LFG notation does not even have a way to specify empty f-structures, on the tacit assumption that every non-vacuous f-structure would have at least one feature. However, the notion could be useful e.g. for expletive arguments that are not specified for any morphosyntactic features such as pers or case but simply appear to satisfy Completeness.

identity is somewhat mitigated by the uniqueness of PRED values (Section 3.3.4), which ensures that any two independently introduced, semantically interpreted f-structures are formally distinct, even if they have the same lexical predicate and the same set of morphosyntactic features. However, not all f-structures have PRED values; thus, for instance, all expletive subjects of the same form are described by the same f-structure, regardless of the clauses in which they occur. For example, all bundles of agreement features (AGR) with the same set of values are identical to each other. One AGR bundle may be required to be identical to another via agreement sharing (Haug & Nikitina 2015) in the f-description (using an equation such as (↑ AGR)=(↑ SUBJ AGR)), but it will also be identical to all other such bundles elsewhere in the same sentence, if they occur. Counterintuitive though such results may seem, it is not clear whether they can lead to any undesirable effects in practice. The notion of identity of f-structures is important for understanding the concept of STRUCTURE SHARING (Section 3.3.3).

3.2 The metalanguage

3.2.1 Defining equations

The standard notation for describing f-structures are DEFINING EQUATIONS.These utilize the idea that f-structures are functions. For example, the value of the attribute TENSE of f-structure f in (11) can be defined by the equation (f TENSE)=PRS. It is possible to use nested function applications; thus, since (f SUBJ)=g, ((f SUBJ) PERS) is equivalent to (g PERS) and has the value 3. By convention, function application is left associative, thus the parentheses can be omitted and the equation written as (f SUBJ PERS)=3.

Defining equations are grouped into F-DESCRIPTIONS. An f-description describes the *minimal* f-structure that satisfies all the equations included in the description. The default relation between equations forming an f-description is conjunction, but disjunction is also possible; for example, {(↑ SUBJ PERS)=1 | (↑ SUBJ PERS)=2} means that the subject is defined as being either 1st or 2nd person.[7] For more examples and discussion of defining equations, see Belyaev 2023b [this volume].

3.2.2 Constraining equations

The f-structure equations described above are all evaluated to construct the minimal complete and coherent f-structure that satisfies all of them together (if such

[7] Disjunction can be represented by either a vertical line (|) or a logical disjunction sign (∨); both notations are found in the literature.

an f-structure exists). In this sense, they are “constructive”, or defining: informally, a defining equation introduces a feature value, regardless of whether it is the only such equation or the same value is defined elsewhere.

But sometimes it is necessary to check the value of a feature without actually assigning it. For example, a matrix verb might require its complement to have a specific mood value, such as subjunctive. A defining equation like $(\uparrow \textsc{comp mood}){=}\textsc{sbjv}$ also licenses a complement that is not marked for mood, i.e. does not have a lexically defined mood feature (e.g., it is non-finite), which probably leads to an incorrect prediction (unless additional constraints block the use of such forms in this context).

Defining equations also provide no way to capture purely negative requirements, i.e. to ensure that a feature *does not* have a specific value. Clearly, this is not equivalent to the disjunction of other possible values of the feature, since, first, absence of the feature also satisfies the negative condition; second, the disjunction would freely assign any feature value except for the disallowed one, which is definitely not what a negative constraint should do.

The need for such constraints is accounted for in LFG by allowing a special class of equations, constraining equations. These equations are special in that they do not participate in constructing the f-structure of the sentence. In contrast, they are only evaluated once the minimal f-structure satisfying all defining equations has been constructed. Then, violation of a constraining equation leads to ungrammaticality.

The simplest type of constraining equations involve equality relations; these are annotated in the same way as defining equations, but with a subscript *c*, e.g.: $(f\ a) =_c \textsc{x}$. To illustrate how constraining equations work, consider the following f-descriptions and their corresponding f-structures:

(12) a.
$$\begin{array}{lcl} (f\ \textsc{a}) & = & \textsc{x} \\ (f\ \textsc{b}) & = & \textsc{y} \\ (f\ \textsc{a}) & =_c & \textsc{x} \end{array} \rightarrow {}_f\!\begin{bmatrix} \textsc{a} & \textsc{x} \\ \textsc{b} & \textsc{y} \end{bmatrix}$$
(constraining equation satisfied)

b.
$$\begin{array}{lcl} (f\ \textsc{b}) & = & \textsc{y} \\ (f\ \textsc{a}) & =_c & \textsc{x} \end{array} \nrightarrow {}_f\!\begin{bmatrix} \textsc{b} & \textsc{y} \end{bmatrix}$$
(constraining equation not satisfied)

In (12a), the constraining equation is satisfied because the feature value is defined elsewhere. By contrast, in (12b) the constraining equation is not satisfied because a has no value, and a value cannot be assigned by a constraining equation.

Note that constraining equations serve as a good illustration of the LFG principle of separation between description and the object being described. Just as multiple feature definitions are not represented in the f-structure, there is also no trace of constraining equations having been "checked" in (12a). The only thing a constraining equation does is to put constraints on permissible structures; it does not contribute to the structures themselves.

The other two types of constraining equations are existential and negative constraints. Existential equations check that a feature has *any* value rather than testing for a specific value. They are written as simple function applications: $(f\,a)$ means that the f-structure f must have the feature a with any value; the absence of an equality statement indicates that we are dealing with an existential constraint. Negative constraints check that a feature does not have a given value ($(f\ a) \neq x$; this is compatible with the feature having no value) or has no value ($\neg(f\ a)$; this is called a negative existential constraint).

Constraining equations are also implicitly introduced by conditional statements of the form $X \Rightarrow Y$. These are, by definition (Bresnan et al. 2016: 61, Dalrymple et al. 2019: 168), equivalent to a disjunction: $\neg A \vee (A_c \wedge B)$.

Off-path constraints are conceptually similar to conditional statements in that they are used to restrict function application to apply only to f-structures satisfying additional conditions on their features. For example,

$$(f \quad \underset{(\to \textsc{b}) =_c y}{\textsc{a}} \quad \textsc{c}) = x$$

means that the value x is only assigned to the feature $\textsc{c}$ of the f-structure $(f\ \textsc{a})$ if $(f\ \textsc{a})$ has an attribute $\textsc{b}$ with the value y. If only constraining equations are used in such statements (as assumed in some of the literature), they could all in principle be rewritten as conditional statements (provided that local names are used: see Section 3.2.5), but the notation is more cumbersome. This is indeed assumed in some LFG literature, and perhaps most prominently in the XLE implementation (Crouch et al. 2011), where defining equations cannot be used in off-path constraints (see Patejuk & Przepiórkowski 2014: 7 for a discussion). However, in spite of their name, the theoretical literature (Bresnan et al. 2016: 65, fn. 26, Dalrymple et al. 2019: 230) unanimously suggests that off-path constraints can be constructive, and this feature is used in some LFG analyses.[8] Off-path constraints are especially important for Functional Uncertainty expressions (Section 3.2.3), where a path may be a regular expression with many elements and the direct use of conditional statements is impractical.

It is clear from the discussion above that while the concept of constraining equations appears rather simple, it actually introduces some additional complexity into the system. Instead of just evaluating an f-description that consists of a

[8] I am thankful to an anonymous reviewer for drawing my attention to this fact.

set of defining equations, the resolution of a valid f-structure for a sentence must proceed through two steps: (a) evaluation of defining equations; (b) evaluation of constraining equations. The notion of constraining equations has also raised concerns about the metatheoretical status of LFG grammars; in particular, Pullum (2013) and Blackburn & Gardent (1995) have argued that constraining equations introduce a degree of procedurality into the framework, which is incompatible with the notion of model-theoretic syntax. However, the specific implications of this procedurality have never been systematically studied. It is clear that many, perhaps most, grammars that use constraining equations could be rewritten without them, but with more notational complexity: for example, by requiring every f-structure to have certain attributes, introducing "empty" attribute values (i.e. treating "no value" as one of the values for atomic features), and so on. Thus the issue might, in the end, be more of notation rather than substance, as suggested, in fact, in the conclusion to Blackburn & Gardent (1995).

3.2.3 Functional uncertainty

The basic LFG architecture outlined in the preceding sections is adequate to handle most phenomena that are relevant to the local structure of clauses and noun phrases, such as argument selection and realization, modification, and word order. However, it is missing a component that could handle unbounded dependencies of any kind, i.e. those dependencies between elements of a sentence that are not tied to any specific structural position. For example, consider the behaviour of "cyclic" extraction from complement clauses. This process is in principle unbounded: an interrogative might be extracted from the matrix clause (13a), from the complement clause (13b), from the complement of the complement (13c), etc.

(13) a. **Who** does John like _?
     b. **Who** does John think Mary likes _?
     c. **Who** does John believe David thinks Mary likes _?

For (13a), one might write an f-structure equation annotating the extracted NP node such as (↑ OBJ)=↓, and augment it with a disjunction for each other available grammatical function – which, by itself, is not very elegant, but seems to adequately account for the facts. To capture (13b), another set of equations must be added to the disjunction, this time with COMP before OBJ: (↑ COMP OBJ)=↓, etc. This already seems like a rather artificial solution, but when (13c) is considered, yet another disjunction is required: (↑ COMP COMP OBJ)=↓, etc. Clearly, the sequence of COMP's can be arbitrarily large (if memory constraints and other extralinguistic considerations are not taken into account), and any grammatical framework must account for such boundless iteration. LFG, in its basic form described above, clearly cannot do so.

Intuitively, what is required is to allow generalizing over sets of functional equations, specifically, introducing disjunction to allow selecting different GFs, and arbitrary iteration of comp. This is achieved by the notion of functional uncertainty, introduced to LFG in Kaplan & Zaenen (1989b). In a nutshell, functional uncertainty extends the LFG notion of function application by allowing function names – x in a statement like $(f\ x)$ – to be regular expressions. Thus, a single f-structure equation may correspond to a (possibly infinite) set of statements. More formally, functional uncertainty defines function application as in (14).

(14) $(f\ \alpha) = v$ holds if and only if f is an f-structure, α is a set of strings, and for some s in the set of strings α, $(f\ s) = v$.

Thus, the distribution in (13) can be captured by a single equation, such as in the following rule for extracted interrogatives:

(15)
$$\begin{array}{lcc} \text{CP} \longrightarrow & \text{NP} & \text{C}' \\ & (\uparrow \textsc{dis}) = \downarrow & \uparrow=\downarrow \\ & (\uparrow \textsc{comp}^{*}\ \{\textsc{obj} \mid \textsc{obj}_{\theta} \mid \textsc{obl}_{\theta}\}) = \downarrow & \end{array}$$

The disjunction in the NP annotation is typically abbreviated as gf, which stands for "any grammatical function" – but which GFs exactly can appear in a given position is construction-specific; for example, adjuncts may or may not be included in the list of gfs. In general, so-called "island constraints" are typically captured in LFG as constraints on paths in functional uncertainty equations (Kaplan & Zaenen 1989b). This correctly predicts that what counts as an "island" varies across languages and across different constructions within the same language.

3.2.4 Inside-out function application and functional uncertainty

Standard function application in LFG is "outside-in": an expression $(f\ a)$ refers to a feature that belongs to the f-structure f or at any deeper level of embedding. This presupposes a "top-down" style of describing and constraining f-structures. However, it may sometimes be useful to describe constraints on the *external* distribution of an f-structure: for instance, limit the range of attributes it may occupy, or define some features of "sister" f-structures, i.e. f-structures that occupy different attributes in the containing f-structure (e.g., subj constraining attributes of obj). For this, LFG uses an additional mechanism called inside-out expressions. Inside-out expressions use the same parenthetical notation as ordinary LFG notation, but the f-structure now acts as an argument rather than as a function. Formally, inside-out expressions are defined as follows:

(16) $(a\ f) = g$ holds if and only if g is an f-structure, a is a symbol, and the pair $\langle a, f\rangle \in g$.

Informally, this definition means that $(a\ f)$ refers to an f-structure g (or set of f-structures) whose attribute a has f as its value. For example, in (17), $(\textsc{a}\ g) = f$ holds because $(f\ \textsc{a}) = g$ is satisfied.

(17) $_{f}\left[\textsc{a}\ \ _{g}\left[\textsc{b}\ \ x\right]\right]$

Functional uncertainty can also be extended to cover inside-out expressions by replacing a in the definition above by a regular expression α. The formal definition is as follows:

(18) $(\alpha\ f) \equiv g$ if and only if g is an f-structure, α is a set of strings, and for some s in the set of strings α, $(s\ f) = g$.

Inside-out function application is by its nature a rather limited formal device compared to “outside-in” function application. It is mainly used either to constrain the grammatical functions that an f-structure may occupy, or to constrain the features of a higher-level f-structure. Importantly, it cannot actually be used as the main mechanism of constructing f-structures. For example, one may formulate a defining equation such as $((\textsc{a}\ f)\,\textsc{a}) = f$ to force f to appear in grammatical function A. But this definition will produce an “orphaned” f-structure which can only be integrated with other f-structures by additional “outside-in” statements, which, in turn, make such an inside-out statement redundant.

Which grammatical phenomena is inside-out functional uncertainty used to model? Perhaps the simplest is the restriction of certain grammatical forms to certain syntactic positions. For example, if nominative marking in a given language is always associated with the grammatical function SUBJ, one may avoid referring to a feature CASE, instead adding $(\textsc{subj}\ \uparrow)$ to the lexical entries of all nominative nouns. This correctly ensures that nominative nouns are only used in those positions which the grammar defines as being associated with subjects.

Another phenomenon where inside-out function application plays a role is agreement on modifiers. For example, Russian adjectives agree in gender and number with their heads. In standard LFG terms, this means that they are lexically annotated to co-define (together with the head noun) the features CASE and num of the f-structures whose adjunct position they occupy. An adjective like *krasnaja* ‘red’ (fem. sg.) might have the following lexical entry:[9]

(19) *krasnaja* Adj $(\uparrow \textsc{pred}) = \text{‘}\textsc{red}\text{’}$
$((\textsc{adj} \in \uparrow)\ \textsc{num}) = \textsc{sg}$
$((\textsc{adj} \in \uparrow)\ \textsc{gend}) = \textsc{fem}$

A somewhat more exotic phenomenon that inside-out functional uncertainty succeeds at capturing is “case stacking” in Australian languages, where NP-internal dependents are marked not only with the case that indicates their position within this NP, but also for the case that indicates the position of this NP at a higher level. Nordlinger (1998) develops a theory called Constructive Case[10] to account for this behaviour; “stacked” cases are treated as denoting the case values of the f-structures that contain the noun as their complement, via the mechanism of inside-out functional uncertainty.

3.2.5 Local names

F-structures in annotated rules are typically referred to relative to the nodes in the phrase structure rule, i.e. using paths that begin in the metavariables ↑ or ↓. This is sufficient if these paths are uniquely and unambiguously resolved; the relevant reference may just be repeated in all equations that use it. But in some cases functional annotations do not uniquely identify f-structures that should be referred to. This most frequently occurs when functional uncertainty is involved (described in Section 3.2.3), i.e. when paths are regular expressions that can resolve to different f-structures. Another example is when the same set of rules can apply to different f-structures, either in free variation or subject to additional conditions. Consider a hypothetical language where verbal agreement morphology can alternatively define the person and number features of the subject or direct object.[11] In this case, it is of course possible to introduce disjunction of two sets of equations, as in (20), but this clearly misses the crucial generalization that the same features are defined in both disjuncts.

[9] The set membership symbol ∈ may be used in inside-out statements just as well as it can be used in outside-in statements. $(\textsc{a} \in f) = g$ entails that $(g\ \textsc{a} \in) = f$, which is notationally equivalent to $f \in (g\ \textsc{a})$. See Section 3.3.3.

[10] “Constructive” is, in fact, somewhat of a misnomer: as shown above, inside-out statements cannot actually “construct” anything, but can only test for feature values of f-structures that have already been constructed.

[11] This example is actually not so hypothetical: such an analysis of Dargwa is proposed in Belyaev (2013), where person-number agreement can be associated with either subject or object, and the choice is then “filtered” using a set of OT constraints.

(20)
$$\left\{\begin{array}{l} (\uparrow \textsc{subj pers}) = 1 \\ (\uparrow \textsc{subj num}) = \textsc{sg} \\ \mid (\uparrow \textsc{obj pers}) = 1 \\ (\uparrow \textsc{obj num}) = \textsc{sg} \end{array}\right\}$$

A more economical way to formulate this constraint is to introduce a temporary label for the f-structure involved – a LOCAL NAME – and then refer to this name in the two equations assigning person and number features. Normal names in LFG, by convention, are written with an initial % and assigned using the standard equation operators, as in (21):

(21)
$$\begin{array}{l} \left\{ \%\textsc{agr} = (\uparrow \textsc{subj}) \mid \%\textsc{agr} = (\uparrow \textsc{obj}) \right\} \\ (\%\textsc{agr pers}) = 1 \\ (\%\textsc{agr num}) = \textsc{sg} \end{array}$$

While local names are not very frequent in LFG analyses, their use is essential for some phenomena where there is a need to consistently refer to an f-structure whose identity is not uniquely deducible from its path (set members, functional uncertainty, etc.).

3.2.6 F-precedence

The basic architecture of LFG is devised to be modular, such that different linguistic phenomena are accounted for at separate levels. In the interaction between c- and f-structure, c-structure is exclusively concerned with linear order and hierarchical embedding, while f-structures do not reflect linear order or constituent structure in any way. Therefore, linear order is relevant for most morphosyntactic constraints only in a limited way, insofar as it distinguishes between different c- to f-structure mappings (such as, for example, in English, where Spec,IP is mapped to subject and precedes the verb and Comp,VP). Without extensions to the standard LFG notation, there is no way to state a constraint like "the verb agrees in person and number with whatever NP stands to its left", because agreement features are the domain of f-structure, and functional equations can only refer to f-structure functions, not linear or constituent-based positions.

However, in certain cases linear order does seem to play a role in determining constraints on syntactic relations. A well-known example is the availability of discourse anaphora between adverbial clauses and main clauses: If the antecedent precedes the pronoun, coreference is possible regardless of which clauses the two are located in (22), while cataphora (backwards anaphora) is only possible if the cataphor stands in the subordinate clause (23).

(22) a. [When John$_i$ came], I saw him$_i$.
b. I saw John$_i$ [when he$_i$ came].

(23) a. [When he$_i$ came], I saw John$_i$.
b. * I saw him$_i$ [when John$_i$ came].

Such behaviour has been generalized since Langacker (1969) as "precede-and-command".[12] Coindexation is possible if at least one of the following is true: the antecedent c-commands[13] the pronoun; the antecedent precedes the pronoun.

Similar constraints operate in other languages. For example, Mohanan (1982) argues that in Malayalam, pronouns *must* follow their antecedents. In LFG, such constraints can be captured using the relation of F-PRECEDENCE (Kaplan & Zaenen 1989a), which is a way of introducing linear order constraints in f-structure using the inverse projection ϕ^{-1}, which maps f-structures to the corresponding c-structure nodes.

(24) f *f-precedes* g ($f <_f g$) if and only if for all $n_1 \in \phi^{-1}(f)$ and for all $n_2 \in \varphi^{-1}(g)$, n_1 c-precedes n_2.

The formal definition in (24)[14] essentially means that an f-structure f_1 f-precedes f_2 iff all c-structure constituents that map to f_1 linearly precede the constituents that map to f_2. Given this definition, anaphoric constraints such as precede-and-command may be formulated as the requirement that the pronoun's antecedent f-precede the pronun.

Note that f-precedence is a rather straightforward relation if an f-structure corresponds to a single constituent. In more complex situations, such as when discontinuous constituents are involved, or one of the elements does not have a c-structure exponent, its application is not so intuitive. In particular, in the latter case, null elements f-precede and are f-preceded by all other elements in the sentence, because one of the sets n_1, n_2 is empty. This property of f-precedence is used to analyze the behaviour of null anaphora in languages like Malayalam (Mohanan 1982) or Japanese (Kameyama 1985), where null pronouns behave differently from full pronouns. For such languages, the definition in (24), combined

[12] The relevance of linear order has been hotly contested in the literature on anaphora, especially in mainstream transformational grammar; for a recent take on precede-and-command, see Bruening (2014). This is not relevant for our discussion, though, as within LFG no one ever argued against linear-order constraints on anaphora.

[13] In LFG, c-command is replaced by outranking on the grammatical function hierarchy: see Rákosi 2023 [this volume].

[14] C-precedence requires that all daughter nodes of a node precede all daughter nodes of another node – essentially a linear precedence relation for c-structure constituents.

with the generalization in the preceding paragraph, correctly predicts that linear order does not influence the anaphoric requirements of null pronouns (Dalrymple et al. 2019: 257).

An alternative definition of f-precedence, that leads to a different treatment of null pronouns, is proposed in Bresnan et al. (2016: 213):

(25) f f-precedes g if and only if the rightmost node in $\phi^{-1}(f)$ precedes the rightmost node in $\phi^{-1}(g)$.

Under this definition, null pronouns in fact do not f-precede and are not f-preceded by any constituent, because their inverse projections lack a rightmost node. To capture the data of Japanese or Malayalam using this definition, a different, negative formulation of the precedence binding constraint should be used: "The domain of a binder *excludes* any pronominal that f-precedes it" (Bresnan et al. 2016: 213, emphasis mine), i.e. the pronoun *must not* f-precede its antecedent. For more information on f-precedence and linear order constraints on anaphora in general, see Rákosi 2023 [this volume].

Thus, the use of inverse projection does allow a degree of influence of linear order on syntactic constraints, in a limited way (as intended): linear order may serve as an additional constraint on relations formulated in f-structure terms, but does not serve as the only or as the main factor determining these relations.

3.3 Attribute value types

3.3.1 General remarks

The system of attribute values in the core LFG architecture is very straightforward. There are only three types of values: atomic values, semantic forms and other f-structures (of which sets are a special instance).

The simplicity of this system follows from the fact that, as mentioned above, LFG has no type system for f-structures. This means that the list of potential attributes and their values for any given f-structure is defined only by annotated phrase structure rules and lexical entries. Thus, there is nothing in the formal architecture or in any part of an LFG grammar that would prohibit a "clausal" f-structure to have the feature CASE or a "nominal" f-structure to have the feature TENSE; such constraints are only implicit in the way these f-structures are constructed and mapped from c-structure nodes.

Similarly, the attributes themselves are not by default associated with any specific value type: LFG grammars by themselves contain no stipulation of possible attributes and the values they may take. Only grammatical function values are required to be f-structures, and PRED values to be semantic forms due to Completeness and Coherence (see Section 3.4). Nothing prevents the value for CASE or PERS to be an f-structure rather than an atomic value; in fact, the former option has been used in analyses such as Dalrymple & Kaplan (2000).

This simplicity of the type system may be viewed as an advantage, as it simplifies the LFG metalanguage without introducing unnecessary redundancy (see Asudeh & Toivonen 2006: 412ff. for a criticism of the Minimalist feature system). There are few problems that a more complex type system would solve, as the architecture of a well-defined grammar typically prevents f-structures from being assigned incorrect attribute values. Still, sometimes it is necessary to check that an f-structure belongs to a given type – for example, whether it is nominal or clausal. LFG provides several ways to do so: one might directly check the category of the corresponding c-structure node using an inverse projection (Section 4.1), or check for certain characteristic attributes (such as CASE for nominals or TENSE for finite clauses) using constraining equations. The latter method, however, is error-prone, as the grammar writer has to ensure that all relevant f-structures have these attributes. This issue can be partly remedied using templates (Asudeh et al. 2013), but templates are an optional, purely notational device; care must be taken that templates are used consistently.

Another solution has been implemented in XLE, which allows the grammar writer to optionally use FEATURE DECLARATIONS to describe the restrictions on feature values (Crouch & King 2008). This is a robust system which, if employed properly, can provide grammars with a higher degree of generalization while also decreasing the number of accidental errors in feature descriptions. Unfortunately, it is virtually unknown in the LFG theoretical literature, being meant as an engineering solution rather than a theoretical proposal and limited to computational work that uses XLE (see Forst & King 2023 [this volume] for more detail).

3.3.2 Atomic values

The simplest type of attribute value is an atomic value: essentially a token that represents a given value of a grammatical feature (e.g. ACC for the feature CASE, PRESENT for the feature TENSE, etc.). There is no single agreed-upon set of "standard" features and the valid values they might take: in principle, it is the task of the grammar writer or analyst to determine the set of features required to describe a particular language.

In current LFG practice, there is, however, a set of informal conventions on the general inventory of atomic features. These fall into two types. The first type are morphosyntactic features of the same kind as those standardly used in typology and descriptive grammars: features such as CASE, TENSE, ASP, PERS, etc. An overview of the use of features in syntactic and morphological description can be found in Corbett (2012).

The second type are more technical features that are specific to the LFG understanding of specific syntactic phenomena. For example, Dalrymple (2001: 396ff.) uses the feature LDD (for Long-Distance Dependency) to mark whether an f-structure is available for extraction. If (f LDD) $= -$, the f-structure f cannot be in the path that specifies a long-distance dependency. This feature is checked by an off-path constraint (see Section 3.2.2). These and similar constraints are discussed in more detail in Kaplan 2023 [this volume].

Similarly, features such as PRONTYPE or NUCLEAR are used in Dalrymple (1993), Bresnan et al. (2016) to distinguish between different kinds of pronouns to account for the differences in binding constraints. See Rákosi 2023 [this volume] for more detail.

In spite of the theoretical significance and cross-linguistic ubiquity of such features as LDD and PRONTYPE, it is generally assumed that they are also not universal and not part of an innate grammatical blueprint (although, to my knowledge, this question has never been explicitly discussed in the literature). Thus, while Bresnan et al.'s (2016) approach to anaphora relies on grammar-wide constraints and distinguishes pronouns via their features, Dalrymple (1993) rather assumes that all binding constraints are lexically specified by the pronouns themselves. The latter point of view is supported by the cross-linguistic diversity of binding domains. It might be that both approaches are valid, but the efficiency of each depends on the language in question. Hence, like in many other domains, LFG as a framework is agnostic as to whether cross-linguistic similarities are due to innate, universal constraints or are a result of independent, functionally motivated convergence of grammars in the course of their evolution. Particular analyses can strike a balance between these two factors that explain cross-linguistic similarities.

3.3.3 F-structure

As seen in (11), f-structures can themselves serve as attribute values. F-structures are predominantly values of grammatical functions such as SUBJ, OBJ, etc., and discourse functions such as DIS (or TOPIC and FOCUS in earlier approaches; see Belyaev 2023a [this volume]). F-structures are sometimes also used to represent "compound" attribute values; for example, agreement features are sometimes represented as the “bundle” AGR in (26), and PRED values can be viewed as composite (Section 3.3.4).

(26)
$$\begin{bmatrix} \textsc{pred} & \text{‘}\textsc{house}\text{’} \\ \textsc{agr} & \begin{bmatrix} \textsc{pers} & 3 \\ \textsc{num} & \textsc{sg} \end{bmatrix} \end{bmatrix}$$

Just as different atomic-valued attributes can have identical values, one f-structure can also serve as a value for several attributes. This phenomenon is called STRUCTURE SHARING and is the closest LFG counterpart to the notion of “movement” in transformational frameworks; it is discussed in more detail in Belyaev 2023b [this volume]. This configuration can be visually represented in two ways: either the f-structure is fully spelt out in every occurence (27a), or only once – then the other occurences are connected by lines (27b) or coindexed (27c).

(27) a.
$$\begin{bmatrix} \textsc{attr1} & \begin{bmatrix} \textsc{a1} & \textsc{v1} \\ \textsc{a2} & \textsc{v2} \end{bmatrix} \\ \textsc{attr2} & \begin{bmatrix} \textsc{a1} & \textsc{v1} \\ \textsc{a2} & \textsc{v2} \end{bmatrix} \end{bmatrix}$$

b.
$$\begin{bmatrix} \textsc{attr1} & \begin{bmatrix} \textsc{a1} & \textsc{v1} \\ \textsc{a2} & \textsc{v2} \end{bmatrix} \\ \textsc{attr2} & \underline{\hspace{3em}} \end{bmatrix}$$

c.
$$\begin{bmatrix} \textsc{attr1} & {}_{f}\begin{bmatrix} \textsc{a1} & \textsc{v1} \\ \textsc{a2} & \textsc{v2} \end{bmatrix} \\ \textsc{attr2} & f \end{bmatrix}$$

Some grammatical phenomena, in particular coordination, adjunction and feature indeterminacy, are represented in LFG via set-valued attributes, as in (28).

(28)
$$f\colon\begin{bmatrix} \textsc{a} & \left\{ \begin{matrix} \begin{bmatrix} \textsc{distr1} & l \\ \textsc{distr2} & m \end{bmatrix} \\ \begin{bmatrix} \textsc{distr1} & l \\ \textsc{distr2} & n \end{bmatrix} \end{matrix} \right\} \end{bmatrix}$$

At first sight, this may appear to violate the notion of f-structure as a function, and the consequent Uniqueness constraint (Section 3.4.1). However, sets in LFG are not multiple values of a single attribute; they are rather viewed as a special kind of f-structure – a *hybrid object* that has both attributes that pertain to it as a whole and attributes whose value is determined based on the values of the set members. This is based on the distinction between DISTRIBUTIVE and NON-DISTRIBUTIVE features.[15] The value of a *distributive* feature for a set is determined as follows:

(29) If a is a *distributive* feature and s is a set of f-structures, then $(s\ a) = v$ holds if and only if $(f\ a) = v$ for all f-structures f that are members of the set s. (Bresnan et al. 1985, Dalrymple & Kaplan 2000)

A distributive feature for a set is only defined as having a value if it has this value in all f-structures in the set. Thus, for (28), the equation $(f\ \textsc{a distr}1) = \textsc{l}$ is true; conversely, no equation invoking the feature DISTR2 (such as $(f\ \textsc{a distr}2) = \textsc{m}$ or $(f\ \textsc{a distr}2) = \textsc{n}$) can be satisfied, since the set elements differ in the value of this feature. Crucially, there is no requirement that distributive features be the same for all elements of a set unless they have been invoked; the structure in (28) is valid as long as the grammar does not assign any value to $(f\ \textsc{a distr}2)$.

While distributive features are resolved on the basis of their values for individual members of a set, *non-distributive* features apply to sets as a whole:

(30) If a is a *non-distributive* feature, then $(f\ a) = v$ holds if and only if the pair $\langle a, v\rangle \in f$. (Bresnan et al. 1985, Dalrymple & Kaplan 2000)

In (3.3.3), the value of the attribute A illustrates a set with a non-distributive feature.

(31)
$$f{:}\left[\begin{array}{ll}\textsc{a} & \left[\begin{array}{l}\textsc{ndistr}\ \ \textsc{n}\\ \left\{\begin{array}{l}\left[\textsc{ndistr}\ \ \textsc{l}\right]\\ \left[\textsc{ndistr}\ \ \textsc{m}\right]\end{array}\right\}\end{array}\right]\end{array}\right]$$

This notation, standard in LFG work, is meant to represent that, while the feature NDISTR has the values L and M for the individual set members, it has the value N for the whole set. The equation $(f\ \textsc{a ndistr}) = \textsc{n}$ is therefore satisfied regardless of the set members' values of NDISTR.

Distributive and non-distributive features in LFG are used to model different ways in which feature values are resolved and checked in coordination and similar structures. For example, number is typically viewed as non-distributive, because a coordinate NP triggers plural agreement regardless of the number features of its conjuncts. In contrast, case is usually distributive: when a case value is assigned to a coordinate phrase, it must be borne by all its conjuncts. The issue of sets and distributivity with respect to coordination is dealt with in Patejuk 2023 [this volume].

[15] This distinction is normally understood as being grammar-wide, or even universal; some authors have recently proposed treating distributivity as a property of *feature application*, not features as such; the most recent such account seems to be Przepiórkowski & Patejuk (2012), and similar ideas are explored in Belyaev et al. (2015) and Andrews (2018).

3.3.4 Semantic forms

A SEMANTIC FORM is a special type of attribute value that is exclusively assigned to the attribute PRED. Semantic forms consist of the predicate name followed by the list of its syntactic arguments; arguments that are assigned thematic roles are written in angled brackets, while arguments that are not thematic (such as expletive subjects or "raised" subjects and objects) are written outside angled brackets. For example, the PRED value for a transitive verb like 'see' will be 'see ⟨SUBJ OBJ⟩'. A verb like 'rain', which has no thematic arguments but an expletive subject, will have the PRED value 'RAIN⟨⟩SUBJ'. Finally, an "object raising" verb like 'believe' will have the PRED value 'BELIEVE⟨SUBJ⟩OBJ': its subject is assigned a semantic role, while its object is not.

In the preceding paragraph, arguments were represented as mere lists of grammatical function names. This convention, which is followed in much LFG work (see e.g. Dalrymple 2001, Dalrymple et al. 2019), is but a simplification: arguments inside PRED values are usually understood as direct references to the corresponding attribute values. Thus, in the left-hand side of (32), the PRED value is represented as 'see⟨(f SUBJ) (f OBJ)⟩'. As observed in Kuhn (2003: 63), PRED values as used in typical LFG representations can be viewed as shorthands for complex structures such as in the right-hand side of (32);[16] FN is an abbreviation for FUNCTOR; SFID stands for "semantic form identifier", on which see below. Similar structures are used in implemented parsers like the Xerox Grammar Writer's Workbench (Kaplan & Maxwell 1996) and the Xerox Linguistic Environment (XLE, Crouch et al. 2011; see Forst & King 2023 [this volume]).

(32)
$$
{}_{f}\left[\begin{array}{ll}
\textsc{pred} & \text{`}\textsc{see}\langle(f\ \textsc{subj})(f\ \textsc{obj})\rangle\text{'} \\
\textsc{subj} & \left[\begin{array}{ll} \textsc{pred} & \text{`John'} \\ \textsc{num} & \textsc{sg} \\ \textsc{pers} & 3 \end{array}\right] \\
\textsc{obj} & \left[\begin{array}{ll} \textsc{pred} & \text{`David'} \\ \textsc{num} & \textsc{sg} \\ \textsc{pers} & 3 \end{array}\right]
\end{array}\right]
=
\left[\begin{array}{ll}
\textsc{pred} & \left[\begin{array}{ll} \textsc{fn} & \textsc{see} \\ \textsc{argument1} & \\ \textsc{argument2} & \\ \textsc{sfid} & i \end{array}\right] \\
\textsc{subj} & \left[\begin{array}{ll} \textsc{pred} & \left[\begin{array}{ll} \textsc{fn} & \text{John} \\ \textsc{sfid} & j \end{array}\right] \\ \textsc{num} & \textsc{sg} \\ \textsc{pers} & 3 \end{array}\right] \\
\textsc{obj} & \left[\begin{array}{ll} \textsc{pred} & \left[\begin{array}{ll} \textsc{fn} & \text{David} \\ \textsc{sfid} & k \end{array}\right] \\ \textsc{num} & \textsc{sg} \\ \textsc{pers} & 3 \end{array}\right]
\end{array}\right]
$$

[16] I follow the representation used by Kuhn (2003), which does not distinguish between thematic and non-thematic arguments. In XLE, this is implemented by distinguishing between the attributes ARG1, ARG2, ... for thematic arguments and NOTARG1, NOTARG2, ... for non-thematic arguments.

If semantic forms were just a bundle of a functor and one or more argument slots, there would be no need to treat them as a special argument value type. What distinguishes them from any other value is their *uniqueness*: each introduction of a PRED value is treated as unique. That is, whenever an expression like (f PRED)=‘FN’ introduces a new semantic form, it is assigned a unique identifier, even if it is lexically identical to another predicate. Thus the equivalence in (33): each PRED assignment is viewed as also introducing an invisible “index” to distinguish between individual PRED values. Thus, if atomic values can be introduced multiple times, PRED values cannot; different grammatical or discourse functions can have the same PRED value only through structure sharing,[17] when the whole f-structure is constrained to be identical. In XLE and other implemented versions of LFG, this uniqueness effect is achieved by including a special feature SFID in the PRED, that is assigned a unique value each time a PRED is introduced in the f-description.[18]

The uniqueness of PRED values is needed to prevent multiple introduction of arguments and will be discussed in Section 3.4.1.

(33)
$$\left\{ \begin{array}{l} (f\ \textsc{pred}) = \text{‘}\textsc{apple}\text{’} \\ (f\ \textsc{pred}) = \text{‘}\textsc{apple}\text{’} \end{array} \right\} \equiv \left\{ \begin{array}{l} (f\ \textsc{pred}) = \text{‘}\textsc{apple}_1\text{’} \\ (f\ \textsc{pred}) = \text{‘}\textsc{apple}_2\text{’} \end{array} \right\}$$

In current LFG research, PRED values mainly serve only to specify argument lists to satisfy Completeness and Coherence, and to provide unique “labels” for f-structures that have PREDs. Even this limited functionality is contested in the literature, with some authors proposing to abandon f-structures in favour of a purely semantic approach, see Section 3.4.4. Originally, however, PREDs were thought to have a more central role, providing a kind of link from syntax to semantics (Kaplan & Bresnan 1982). It is important to observe that PREDs are no

[17] Or, as an anonymous reviewer observes, through sharing of the PRED value itself, as e.g. in the analysis of adjective coordination in Belyaev et al. (2015).

[18] XLE extends standard LFG by allowing any atomic value to be unique – an *instantiated symbol* notated via a subscript following its name: VAL_. Thus in XLE, semantic forms do not seem to require any special machinery as such. However, an anonymous reviewer observes that if the left-hand side of (32) is indeed the abbreviation of its right-hand side, it should be possible to manipulate argument structure in the syntax via equations such as (↑ PRED ARGUMENT3)=↓. XLE seems to circumvent this by tacitly introducing a negative existential constraint that prevents any additional attributes from appearing in PRED except the ones included at its introduction. This includes both argument features and any other feature names: both the XLE version of the above statement and (↑ PRED FOO)=BAR lead to an existential constraint violation. It is also impossible to “construct” a semantic form using a set of separate statements for the individual features; thus even XLE does technically treat semantic forms as a special value type.

longer viewed in these terms in the LFG literature; the functor names are only arbitrary labels, and all semantic derivation is separate from syntax, being done through Glue Semantics, described in Asudeh 2023 [this volume].[19]

3.4 Well-formedness conditions

There are three conditions that any f-structure must satisfy in order to be treated as valid: Uniqueness (also known as Consistency), Completeness, and Coherence. Any f-structure that violates these conditions cannot be part of a valid analysis of any sentence, regardless of the rules of a particular grammar.

3.4.1 Uniqueness

3.4.1.1 Definition

Uniqueness (Consistency) is the requirement that every attribute in an f-structure must have a single value. Thus, the two equations in (34) do not describe any valid f-structure.

(34) Ill-formed f-structure:

$$
\begin{array}{l}
(f\ \textsc{a}) = \textsc{l}\\
(f\ \textsc{a}) = \textsc{m}
\end{array}
\qquad
{}_{f}\!\left[\begin{array}{ll}\textsc{a} & \begin{array}{l}\boxed{\textsc{l}}\\ \boxed{\textsc{m}}\end{array}\end{array}\right]
$$

It should be noted that Uniqueness is not, in fact, a constraint that needs to be stipulated separately: it follows from the notion of f-structure as a function, since a function maps arguments to single values (thus defining a one-to-one or many-to-one, but not a one-to-many or many-to-many correspondence).

3.4.1.2 Multiple specification of a value

Uniqueness does not in any way imply that multiple specification of an attribute value is ruled out. When the *same* value is assigned to an attribute two or more times, the resulting f-structure is valid, as seen in (35).

(35)

$$
\begin{array}{l}
(f\ \textsc{a1}\ \textsc{a2}) = \textsc{l}\\
(f\ \textsc{a1}\ \textsc{a2}) = \textsc{l}\\
(g\ \textsc{a2}) = \textsc{l}\\
(g\ \textsc{a3}) = \textsc{m}\\
(f\ \textsc{a1}) = g
\end{array}
\qquad
{}_{f}\!\left[\begin{array}{ll}\textsc{a1} & {}_{g}\!\left[\begin{array}{ll}\textsc{a2} & \textsc{l}\\ \textsc{a3} & \textsc{m}\end{array}\right]\end{array}\right]
$$

[19] A kind of hybrid approach is proposed in Andrews (2008), which introduces a variant of Glue Semantics where meaning is at least in part derived from f-structure feature values; in this approach, PRED features do play a prominent role in semantic composition.

In (35), the attribute (*g* A2) is assigned its value three times and referred to in two different ways, but this "history" of its origin is not displayed in the resulting f-structure and is not recoverable from it in any way. This is an illustration of the LFG distinction between a *description* and the *object* that it describes, a crucial feature of LFG that separates it from most other frameworks, where syntactic constraints are usually encoded in the structure itself in one way or another.

Turning to a linguistically meaningful example, this distinction between description and object is manifest in the standard LFG approach to agreement (see Haug 2023 [this volume] for more detail). Agreement targets do not normally have a "copy" of their controller's features; they only lexically specify the same features that are separately specified by the controller. If there is a conflict, the resulting f-structure is invalid. If there is no conflict, the agreement features are displayed in the f-structure once and there is nothing in the f-structure indicating that agreement feature checking has taken place. Compare the Italian examples (49) and (50) below, which map to the same f-structure even though the person-number features are described in two positions in (49) but defined once in (50).

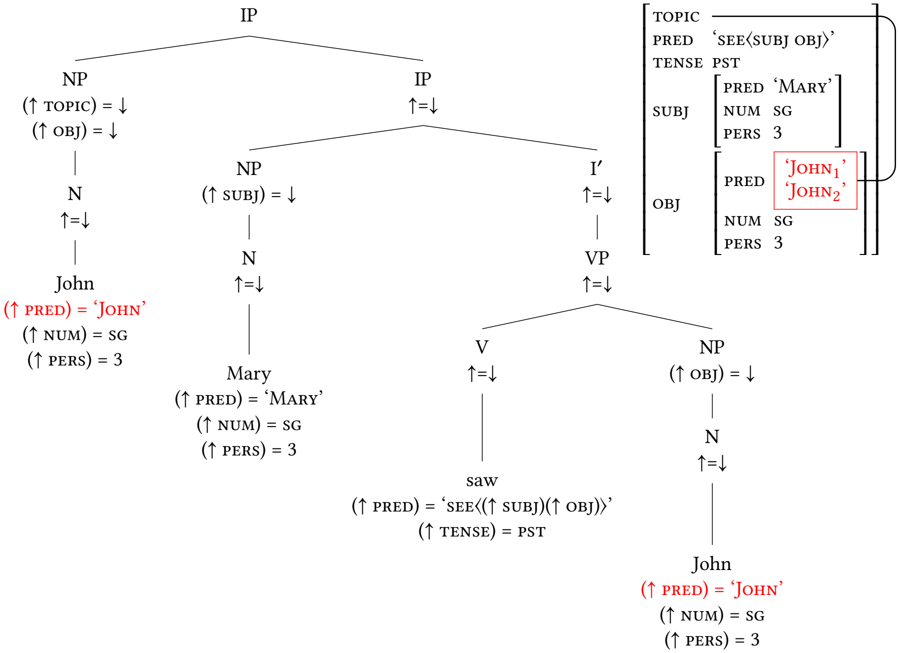
3.4.1.3 Uniqueness and PRED values

One place where multiple specification is virtually prohibited is PRED features, whose values are special objects called semantic forms. As described above in Section 3.3.4, each assignment of a PRED value is treated as a unique object; it is thus impossible to assign a PRED value more than once, even if the value to be assigned has the same functor name.

The reason why PRED values are treated in this way is to ensure that each argument position, and each predicative element in general, is instantiated by exactly one lexical head. Since there is no one-to-one correspondence between c-structure positions and f-structure functions, this cannot, in the general case, be ensured by phrase structure rules alone. Even in a configurational language like English, a displaced constitutent is not directly linked to its "original" (normal, unmarked) position at c-structure; consequently, the c- to f-structure correspondence allows introducing it twice, as in (36).[20]

[20] For the sake of exposition, I assume that the topicalized direct object appears as an IP adjunct – this carries no theoretical significance.

(36) Ill-formed f-structure for **John, Mary saw John*:

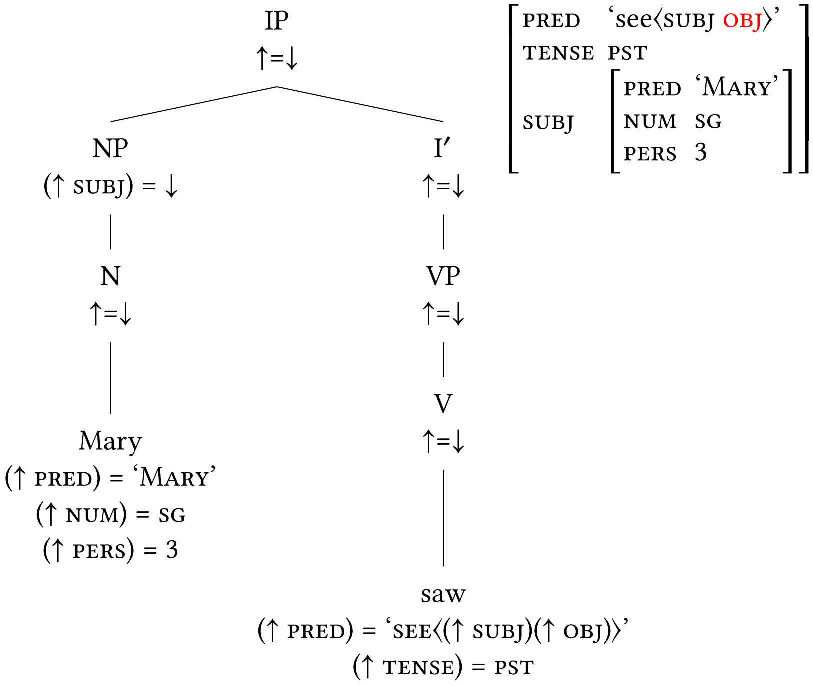


What ensures the ungrammaticality of (36) is precisely the uniqueness of PRED values. This effect is even more pronounced in non-configurational languages, where no c-structure position is tied to any grammatical function, and any number of NPs may be freely mapped to any grammatical function; see Andrews 2023 [this volume] for detail.

3.4.2 Completeness

The Completeness condition requires every grammatical function governed by the PRED value of a given f-structure to exist in this f-structure. In other words, all arguments of a predicate must be "filled" by f-structures. This disallows examples such as (37).

(37) Ill-formed f-structure for **Mary saw*:



C-structure rules cannot be conditioned by argument structure; hence, Completeness violation is the only reason why this sentence is ungrammatical. The c- to f-structure correspondence is otherwise entirely valid.

It is important to understand that completeness only refers to f-structure and has nothing to do with whether arguments are expressed overtly or covertly. Since LFG avoids empty nodes, covert subjects in pro-drop languages do not correspond to any c-structure NP or DP, but Completeness still has to be satisfied at f-structure. This is normally done via equations introducing the pronominal PRED of the subject in the verb's lexical entry: see (48) below.

An additional Completeness constraint has to do with the parameter of semantic argumenthood. It states that semantic arguments (i.e. those whose names stand within angled brackets in the PRED) have to themselves contain a PRED. Conversely, non-arguments (those whose names stand outside angled brackets) are required not to contain a PRED, unless these f-structures are arguments or adjuncts elsewhere (such as, for example, in raising constructions). This is meant to exclude, respectively, expletive arguments in positions where semantic roles are assigned (38), and meaningful NPs in expletive positions (39).

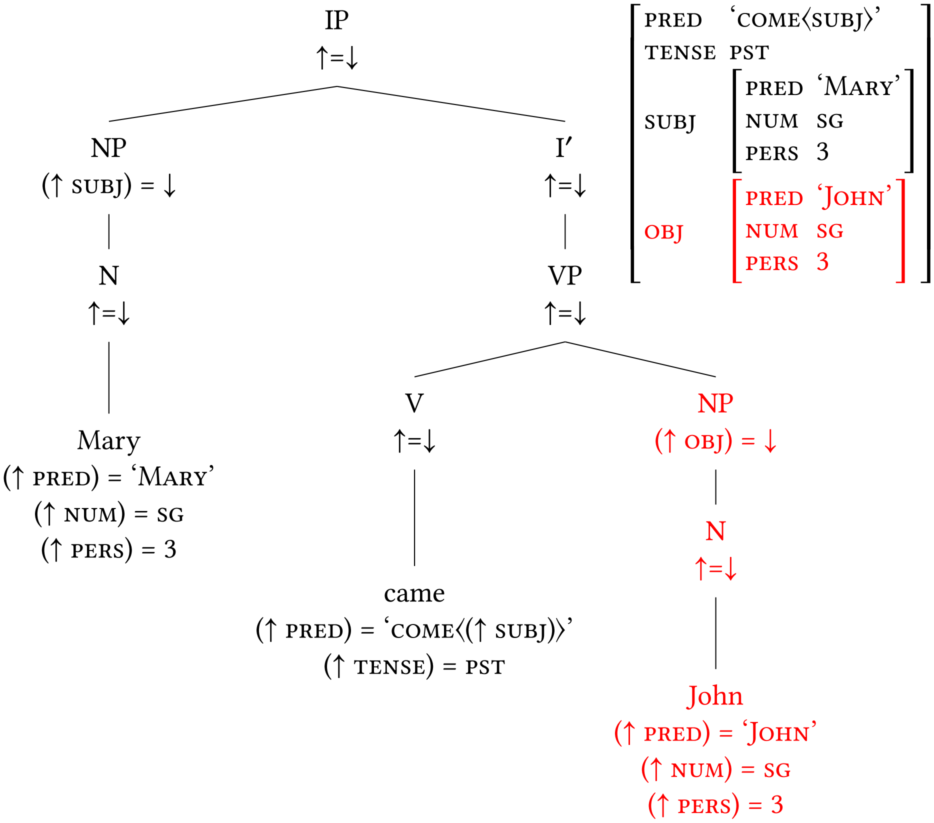
(38) *I saw **there**.

(39) ***The sky** rained.

3.4.3 Coherence

The Coherence condition is the converse of Completeness: no governable functions (i.e. f-structure functions representing grammatical functions such as SUBJ, OBJ, etc., see Belyaev 2023a [this volume]) may appear in an f-structure without being listed in a PRED value. This ensures that no "orphaned" arguments appear in an f-structure, disallowing examples such as (40).

(40) Ill-formed f-structure for **Mary came John*:



Here, again, the c- to f-structure correspondence itself is valid, but the resulting f-structure is incoherent.

The coherence condition only applies to argumental grammatical functions and does not say anything about adjuncts or discourse functions. Where these elements may appear is constrained by a separate condition called Extended Coherence (Bresnan et al. 2016: 63). Extended Coherence requires that the f-structure where adjuncts appear have a PRED value. This ensures that no adjuncts appear in PRED-less f-structures. Discourse / overlay functions (DIS in more recent approaches, TOPIC and FOCUS in earlier work) are required to be linked to a grammatical function in some way: either functionally (via structure sharing) or anaphorically. For more information on the differences between various types of grammatical functions, see Belyaev 2023a [this volume].

3.4.4 Redundancy of PRED?

The description of Completeness and Coherence in this chapter follows the traditional LFG model, which had little to say about semantics; therefore, all valency restrictions had to be modeled at f-structure. Since at least the papers in Dalrymple et al. (1993), Glue Semantics (see Asudeh 2023 [this volume]) has been gaining acceptance in LFG as the model of the syntax-semantics interface. Glue Semantics is resource-sensitive, which automatically ensures both Completeness and Coherence: Completeness, because all premises of the meaning constructor introducing the main predicate have to be saturated; Coherence, because no unused resources have to be left. The role of uniqueness of PRED for ensuring lack of multiple argument introduction / duplicate heads (Section 3.4.1) also follows from Glue semantics due to the fact that any resource can only be consumed once. Therefore, many authors, among others Kuhn (2001), Asudeh & Giorgolo (2012), Asudeh et al. (2014), have argued that PRED features in their original form are no longer necessary in LFG. At least argument lists can, for the most part, be safely dispensed with.[21] Many authors, therefore, continue to use PRED values but only include the name of the functor, not arguments in angled brackets; the remaining role of PRED values is only to provide an index for the f-structure, guaranteeing its uniqueness (that may be relevant for purely syntactic purposes that are not handled in semantics), and to provide information on the lexical content of its head.

[21] Non-thematic arguments like *it* in *it rained* might still be relevant insofar as they are not selected by any semantic predicate. However, these arguments may be forced to appear using existential constraining statements.

4 The c- to f-structure mapping

4.1 Annotated c-structure rules

The metalanguage discussed in the preceding section can describe individual f-structures, but cannot, by itself, generate or evaluate natural language expressions. F-descriptions must come from somewhere. The only generative component in LFG is c-structure; therefore, phrase structure rules must be coupled with some mechanism that specifies how the nodes in the c-structure tree are mapped to f-structures – the projection function ϕ. In LFG, this is normally done using ANNOTATED PHRASE STRUCTURE RULES where nodes at the right-hand side are supplemented by f-descriptions that reference the c- to f-structure mapping. This referencing is done by introducing two additional notational symbols:

(41) the current c-structure node: $*$
the immediately dominating c-structure node: $\hat{*}$

These are normally not used directly in LFG grammars; instead, two metavariables $\downarrow$ and $\uparrow$ are used, which signify the following:

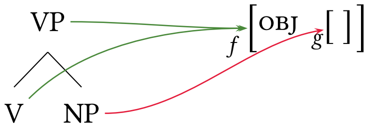
(42)
$\downarrow = \phi(*)$ (the f-structure corresponding to the current c-structure node)
$\uparrow = \phi(\hat{*})$ (the f-structure corresponding to the immediately dominating c-structure node)

This notation allows formulating rules of the type:

(43)
$$\begin{array}{lccc} \text{VP} & \longrightarrow & \text{V} & \text{NP} \\ & & \uparrow=\downarrow & (\uparrow \text{OBJ})=\downarrow \end{array}$$

In (43), the annotation for V stands for “this node (V) maps to the same f-structure as the dominating node (VP)”, while the annotation for NP stands for “this node (NP) maps to the OBJ attribute of the f-structure of the dominating node (VP)”. The mapping that this rule defines is illustrated in (44). The nodes VP and V map to the same f-structure labeled as f, while NP maps to the f-structure labeled as g – the direct object of the clause.

(44)



The LFG metalanguage also allows for a notation for the inverse projection ϕ^{-1}, that maps f-structures to the c-structure node(s) that map to them. This mapping is one-to-many and thus, unlike the direct projection, not a function. For example, in (44), $\phi^{-1}(f)$ refers to two nodes: VP and V. The inverse projection is, by design, seldom used and, in fact, rarely required; but it is indispensable for certain construction which place selective requirements on the categorial status of their elements, such as the verb *wax* in examples like *wax poetical*, which is only compatible with an AP complement (the construction is discussed in Pollard & Sag 1994; for an LFG implementation, see Dalrymple et al. 2019: 6.10.3).

4.2 Some consequences of the mapping

4.2.1 Locality

The annotated rule format described in the preceding section is not merely a question of notation; it defines a rather rigid constraint on the way c-structure nodes can be mapped to f-structure. Namely, the mapping is strictly local: it can refer only to the nodes that are involved in a given phrase structure rule. It is not possible to freely traverse the tree and refer to, say, the node dominating the mother node, the child node of the current node, or the root node. LFG assumes that no linguistically meaningful generalizations can be captured using such "long-distance" references. For the majority of cases, this is clearly true, and consequently, there have been no serious attempts to extend the LFG metalanguage in this direction.[22]

However, the strict locality of c-structure to f-structure mapping does create problems for the analysis of certain idiomatic combinations – multi-word expressions (MWEs), as they are called in the literature. Such MWEs often span whole syntactic phrases, and the lexical constraints involved cannot be captured locally. One solution that has been proposed in LFG is to replaced the context-free c-structure by a variant of Tree-Adjoining Grammar (TAG), see Findlay (2017, 2019); this proposal is described in some detail in Findlay 2023 [this volume].

Within the local domain of c-structure rules, the mapping to f-structure is further constrained in that it is only possible to refer to the immediately dominating and current nodes, but not to any of the sister nodes. Unlike the locality constraint, this has been challenged in some LFG literature. For example, Dalrymple

[22] As observed by an anonymous reviewer, if ↑ and ↓ are only abbreviations of $\phi(\hat{*})$ and $\phi(*)$, it is possible to also use $\phi(\hat{\hat{*}})$ and so on, making annotated rules potentially non-local. As mentioned above, low-level "designators" like $*$ are not normally used in LFG analyses: grammars are expected to operate only with ↑ and ↓.

(2001: 120), Dalrymple et al. (2019: 222–223), developing the ideas of Nordlinger (1998), extend this notation by defining the metavariables <* and *> for "left sister of the current node" and "right sister of the current node", respectively; the corresponding f-structures are $\phi(<*)$ and $\phi(*>)$. Similarly, XLE defines the metavariables LS* and RS* for the same concepts.

The status of such innovations in the general LFG framework is uncertain. On the one hand, the analyses that introduce such notational conventions make convincing cases that they are necessary for analyzing certain phenomena, or at least vastly simplify such analyses. On the other hand, it is telling – and usually implied – that their use is somewhat exceptional and limited to a handful of specific phenomena. The fact that phrase structure nodes and lexical items do not refer to the information contributed by their left or right sisters in the vast majority of cases seems to be an important cross-linguistic generalization – one that is lost if this possibility is introduced in the formalism. If such formal devices are necessary, additional theoretical stipulations should supposedly constrain their use, but in practice, this possibility is almost never explored.

4.2.2 Monotonicity

As Bresnan et al. (2016: 73ff.) observe, the limitations of the metalanguage described above (even if additional designations like <* and *> are included) lead to several important consequences for grammatical architecture. Specifically, the locality of the c- to f-structure mapping leads to the monotonicity of information flow in the syntax: the f-structure of a larger fragment is always more specific than the f-structure of a smaller fragment.

Let us first consider what "being more specific" means for an f-structure. By definition, f-structures are sets of feature-value pairs. It is clear, then, that g in (45) is more specific than f, as it has exactly the same features and values as f and one additional feature.

(45) $$_f\begin{bmatrix}\textsc{a} & \textsc{x}\\ \textsc{b} & \textsc{y}\end{bmatrix} \qquad _g\begin{bmatrix}\textsc{a} & \textsc{x}\\ \textsc{b} & \textsc{y}\\ \textsc{c} & \textsc{z}\end{bmatrix}$$

Now consider a more complex case. In (46), f and g have the same features, but intuitively, g is more specific than f because the f-structure value of A in g is more specific than the value of A in f.

(46) $$_f\begin{bmatrix}\textsc{a} & \begin{bmatrix}\textsc{c} & \textsc{z}\end{bmatrix}\\ \textsc{b} & \textsc{y}\end{bmatrix} \qquad _g\begin{bmatrix}\textsc{a} & \begin{bmatrix}\textsc{c} & \textsc{z}\\ \textsc{d} & \textsc{m}\end{bmatrix}\\ \textsc{b} & \textsc{y}\end{bmatrix}$$

Thus, specificity can be defined recursively: g is at least as specific as f if for every attribute a in f, $(g\,a) = (f\,a)$ or $(g\,a)$ is at least as specific as $(f\,a)$ (Bresnan et al. 2016: 74). This relation is essentially equivalent to subsumption (Dalrymple et al. 2019: 240), and can be notated accordingly: $g \sqsupseteq f$ or $f \sqsubseteq g$ means that g is at least as specific as f, or f subsumes g.

Now recall that every f-description can in principle correspond to infinitely many f-structures that satisfy it. Let us, then, define $\phi(d)$ to be the smallest f-structure ϕ that satisfies d; this gives the mapping ϕ from the set of functional descriptions D to the set of f-structures F. This mapping is MONOTONIC: the larger the f-description d, the more specific the corresponding f-structure f. In other words, $\phi : D \rightarrow F$ has the property that if $d \subseteq d'$ and both d and d' have f-structure solutions, then $\phi(d) \sqsubseteq \phi(d')$.

This property of the mapping between f-descriptions and the corresponding f-structures follows from the nature of the f-structure equations: New equations can only specify additional information about the f-structure or check existing information; they cannot, as it were, "delete" existing feature values or otherwise make the structure less specific.

4.2.3 Fragmentability

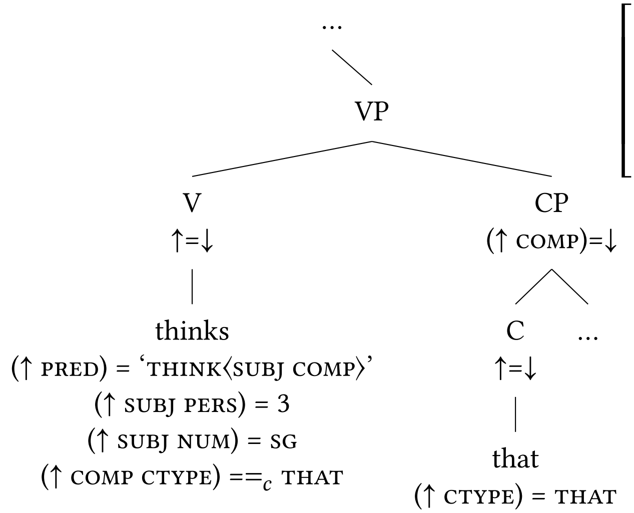
Another feature of syntax in the LFG architecture that follows, in part, from monotonicity is FRAGMENTABILITY of language (Bresnan et al. 2016: 79–82). Recall that f-descriptions in annotated c-structure rules can only refer to the f-structures of the nodes involved in the rule (the node at the left side of the rule – the dominating node – and its daughters). This means that, the larger the tree, the longer its f-description; due to monotonicity, the f-structure of a larger tree fragment is, then, always more specific than the f-structure of any of its subtrees. Therefore, a valid f-structure can be constructed for any tree fragment dominating an arbitrary sequence of terminal nodes (a substring of a complete sentence), and this f-structure will not be overridden by any additional information that is contained in the complete sentence (unless it renders the f-structure ill-formed, in which case the sentence is ungrammatical).

Note that this property of the c- to f-structure correspondence does not depend on whether the tree fragment corresponds to a sequence of terminal nodes in a complete sentence; it may even not be a constituent. Any sentence fragment is "self-contained" in the sense that its content is not modified by additional nodes in the tree.[23] Consider the c- and f-structures in (47). Here, the combina-

[23] Note, however, that a tree fragment may be ambiguous between two or more interpretations; this ambiguity may be resolved by further material in the tree.

tion “thinks that”, which is not even a constituent in a fully formed sentence, contributes the argument structure of the matrix clause, the person and number features of the subject, and the complement type. It can be extended both upwards (with the addition of a subject) and downwards (with the addition of a complement clause), with f-structure information increasing monotonically in both cases.

(47)



$$\begin{bmatrix} \text{PRED} & \text{‘THINK}\langle\text{SUBJ COMP}\rangle\text{’} \\ \text{TENSE} & \text{PRS} \\ \text{SUBJ} & \begin{bmatrix} \text{PERS} & 3 \\ \text{NUM} & \text{SG} \end{bmatrix} \\ \text{COMP} & \begin{bmatrix} \text{CTYPE} & \text{THAT} \end{bmatrix} \end{bmatrix}$$

Within the non-transformational architecture of LFG, the properties of monotonicity and fragmentability may seem trivial. But this is not so for transformational frameworks, where elements may be extracted from within constituents, thus violating the principle of fragmentability: sentence fragments may become modified during derivation, losing some of the information they initially contain. Fragmentability captures the fact that sentence fragments frequently occur in natural discourse and are parsed without effort by native speakers.

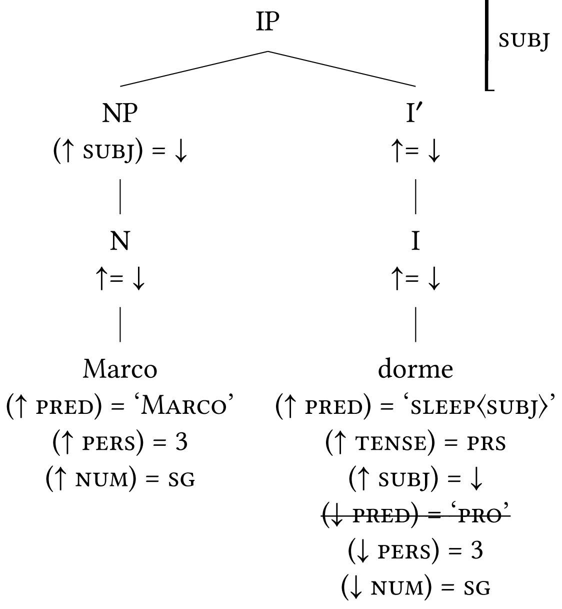
4.2.4 Non-configurationality

Another consequence of the mapping between c- and f-structure is NON-CONFIGURATIONALITY of language. This property means that information in the f-structure does not necessarily correspond to specific positions in the tree. Thus, features of a single constituent may be “collected” from several nodes or assigned several times in different positions. This is usually related to the interaction between syntactic and morphological encoding.

For example, in Italian, a pro-drop language, a third person singular verb form might be defined as in (48) – with the optional assignment of a pred feature to the subject (the standard analysis of pro-drop in LFG). If there is a subject NP in Spec,IP, this annotation is not selected, as it would lead to a pred conflict. If, however, there is no overt subject, this annotation *must* be used, because otherwise the resulting f-structure would violate Coherence: subj would have no pred value. Both options can be seen in (49) and (50).

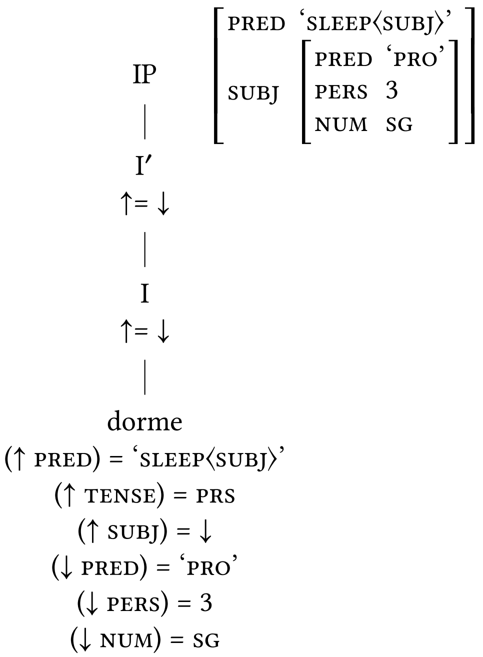
(48) *dorme* V (↑ pred) = ‘sleep⟨subj⟩’
(↑ tense) = prs
(↑ subj) = ↓
((↓ pred) = ‘pro’)
(↓ pers) = 3
(↓ num) = sg

(49)



$$\begin{bmatrix} \text{pred} & \text{‘sleep⟨subj⟩’} \\ \text{subj} & \begin{bmatrix} \text{pred} & \text{‘Marco’} \\ \text{pers} & 3 \\ \text{num} & \text{sg} \end{bmatrix} \end{bmatrix}$$

(50)

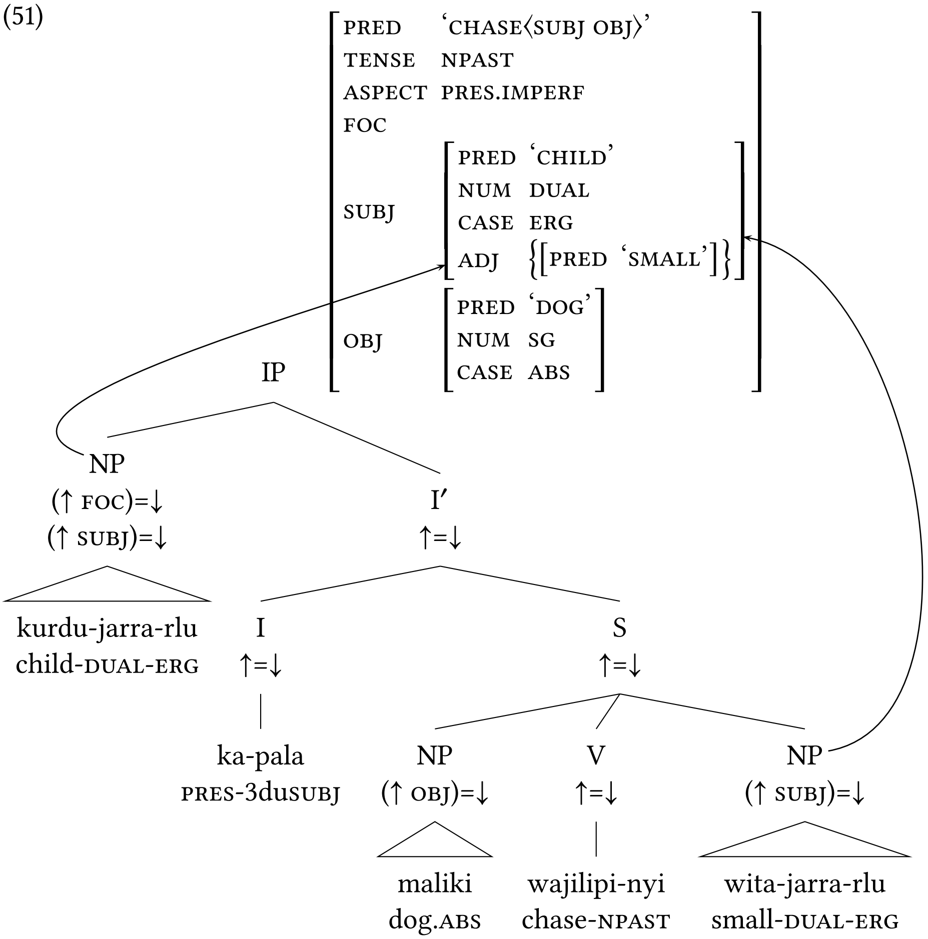


$$\begin{bmatrix} \textsc{pred} & \text{‘}\textsc{sleep}\langle\textsc{subj}\rangle\text{’} \\ \textsc{subj} & \begin{bmatrix} \textsc{pred} & \text{‘}\textsc{pro}\text{’} \\ \textsc{pers} & 3 \\ \textsc{num} & \textsc{sg} \end{bmatrix} \end{bmatrix}$$

Thus, in Italian, the PERS and NUM features of the subject are always assigned at the I node, and they may also be assigned at the N head, if it is present. The PRED feature, in contrast, can be supplied either at the I node (if no overt head is present) or at the N node. This means that even in languages with relatively rigid word order and clausal phrase structure such as Italian (and English, although examples are less illustrative; see Bresnan et al. 2016), there is no universal mapping between c-structure positions and f-structure features.

“Non-configurationality” is usually understood in a more narrow sense, describing languages with no evidence for a hierarchical clause structure, such as Warlpiri (Hale 1983, Austin & Bresnan 1996). In (51), from Austin & Bresnan (1996: 229), two NPs, one having a head and the other only specifying an adjunct, map to the same f-structure function SUBJ. Thus information that is split at f-structure is collected together at f-structure.

(51)



These, of course, are only more radical manifestations of the phenomenon illustrated above. In Italian, the features of certain grammatical functions can be defined in different positions, but these positions, at least, are generally fixed, such that the overt subject, if present, occupies the Spec,IP postion, the full NP direct object occupies Comp,VP, and the verb provides the agreement and PRED features of the subject. In radically non-configurational languages, in contrast, there is no association between c-structure positions and grammatical functions at all: any NP daughter of the S node can be mapped to any grammatical function, and any category, not only the verb, can function as the predicate of the clause. Non-configurational syntax and its challenges are described in more detail in Andrews 2023 [this volume].

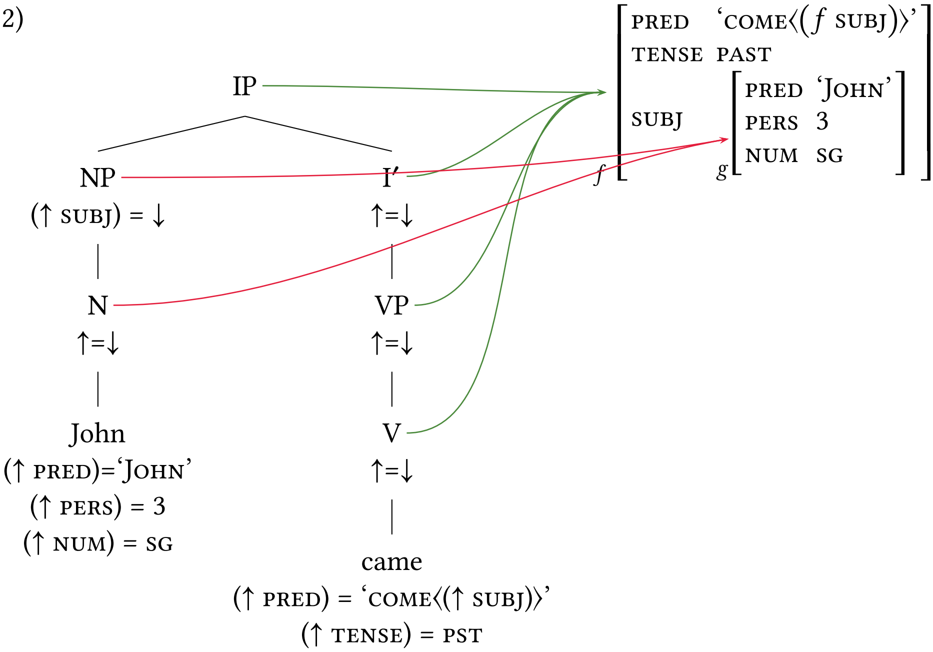
4.2.5 Equality, unification, and non-compositionality

As seen in Belyaev 2023b [this volume] and elsewhere above, statements specifying the equality of one f-structure to another – most prominently, ↑=↓ – play a key role in the LFG c- to f-structure mapping and syntactic analyses. These kinds of statements allow mapping more than one c-structure node to the same f-structure and permit structure sharing and the checking of compatibility of f-structure features. Equality in LFG is very similar in its effects to UNIFICATION found in many other non-transformational formalisms – such that LFG itself is included in the class of UNIFICATION-BASED GRAMMARS in Shieber (1986).

However, as Kaplan (1989: 8ff.) points out, there is a crucial difference between LFG grammars and most unification-based frameworks (GPSG, HPSG, etc.): namely, the distinction between linguistic representations and the *descriptions* of said representations. The clearest case of this distinction are constraining equations, which impose additional constraints on admissible f-structures which, if not violated, do not show up anywhere in the f-structure. Defining equations behave similarly: the same feature may be defined several times in the tree, but the f-structure will contain no trace of its "pedigree": only the resulting feature value will be included.

Another way in which LFG grammars are different from unification grammars is their NON-COMPOSITIONALITY. Even if a c-structure node is annotated with the "unificational" statement ↑=↓, the f-structure it maps to in the complete sentence may contain additional values that are introduced higher in the tree. Thus, in (52) the VP node maps to an f-structure that includes a SUBJ feature that is not introduced anywhere in the VP subtree.

(52)



In a single-tier unificational model like GPSG or HPSG, where the counterpart to f-structure information directly occupies phrase structure nodes together with categorial information, the flow of information would be different: The content of a dominating node would be a function of the content of its children, hence, information contained in VP would be a subset of the information contained in IP. In LFG, as discussed above, *f-descriptions* do indeed increase monotonically, and a *fragment* associated with a node like VP does indeed contain a subset of the information contained in a larger constituent. However, in the full structure, this is not the case: every node mapped to a given f-structure maps to *all* the information contained in this f-structure, even to the information that is introduced only higher above.

4.3 Regularities in the c- to f-structure correspondence

In Section 2.1, I briefly described X′ theory in the way that it is used in most LFG work. However, given that c-structure plays a limited role in LFG compared to the frameworks for which X′ theory was originally devised, in this form it amounts to little more than a system for labelling nodes. In order to give significance to the notion of being a head, a specifier, a complement, or an adjunct, X′ theory must be augmented by f-structure mapping principles.[24] A set of such principles is broadly accepted in LFG, although some details vary. For a more detailed exposition of X′ theory, see Dalrymple et al. (2019), Bresnan et al. (2016).

4.3.1 Heads

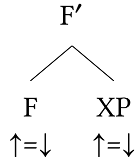
Headedness is a key concept of X′ theory; all projecting nodes, i.e. preterminal nodes (X^0) and intermediate projections (X′), are heads. We saw in all examples above that all projections of a single X′ category are mapped to the same f-structure, and this is for good reason: X-bar theory aims to model endocentricity, and so heads map to a “matrix” f-structure while specifiers, adjuncts, and complements (with the exception of functional categories) map to its dependents. Thus, heads are always annotated as ↑=↓. This principle was first proposed in Bresnan (1982a) and further developed in Zaenen (1983), where it is called the Head Convention. Additionally, XP is also annotated as ↑=↓ when other categories are adjoined to it.

This principle of head annotation allows us to formalize endocentricity as the requirement that every lexical category have a head (Bresnan et al. 2016), or, more correctly, an extended head (see below), because some phrases can have a lexically instantiated functional head but no lexically instantiated lexical head.

4.3.2 Complements

Complements are annotated differently depending on whether they are attached to functional or lexical heads. In essence, functional projections are little more than extensions of lexical projections, and generally map to the same f-structure: for example, CP, IP and VP map to the same clausal structure, while DP and NP map to the same nominal structure. Thus, complements of functional projections are f-structure co-heads, annotated as ↑=↓ (53). The heads of functional categories are known as extended heads of lexical categories; a formal definition of extended head can be found in Bresnan et al. (2016: 136).

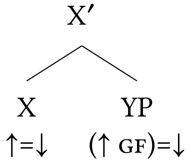
(53)



[24] It is by no means implied that these principles dictate the *only* annotations that can be associated with a given node: additional annotations are not only possible, but sometimes even required to produce a valid f-structure (for example, dis must usually be associated with a grammatical function).

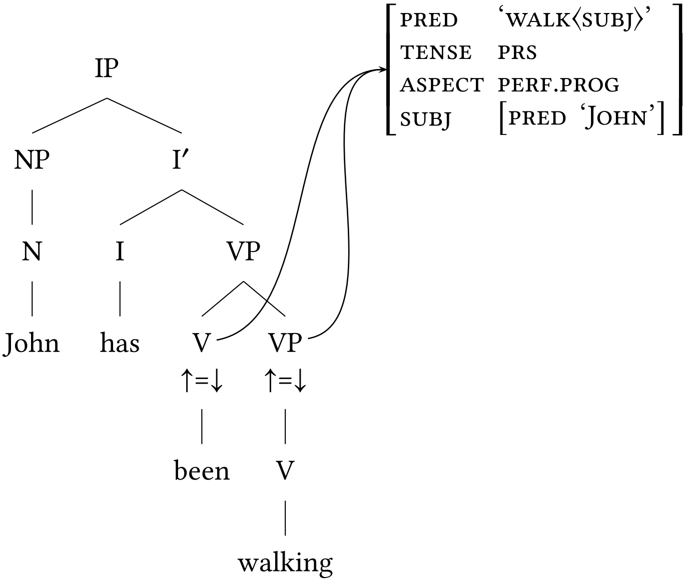
Complements of lexical projections are assigned to various functions of their heads' f-structures. Most typically these are, more specifically, grammatical functions, i.e. those functions that are governed by predicates and have no additional discourse significance (54); the label gf stands for "grammatical function" and includes such notions as subject (subj), direct object (obj), secondary object (obj$_\theta$) and oblique (obl$_\theta$). In Bresnan et al. (2016), this is formulated as a strict requirement that the complement may be any grammatical function except subj (which, in their model, is both a grammatical and a discourse function, see Belyaev 2023a [this volume]). However, this restricted understanding of lexical complements is not universally accepted. For example, Laczkó (2014) analyzes postverbal subjects in Hungarian as occupying the same position as postverbal direct objects, i.e. VP complements.

(54)



Complements of lexical heads may also behave in the same way as complements of functional projections, i.e. be annotated as ↑=↓. This possibility should be allowed for to handle cases where the same f-structure extends over more than two projections, e.g. in certain English auxiliary constructions (55), see Bresnan et al. (2016: 111).

(55)



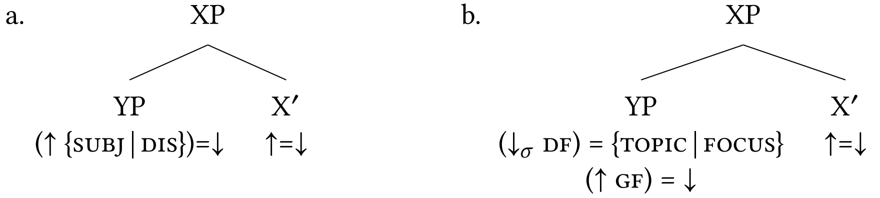
The higher VP in this case thus operates as a kind of intermediate functional projection. An alternative solution would be to introduce an additional functional projection for English, but this does not seem justified as the forms used in these positions are identical to V complements of simpler auxiliary constructions. At the same time, the X′ model itself is obviously too simplistic to describe the full system of constraints on the English system of verbal periphrasis. This requires reference to morphological features of c-structure nodes; see Forst & King 2023: Section 2.2 [this volume] for a discussion of complex c-structure categories which can encode such information.

4.3.3 Specifiers

Specifiers are similar to complements in that they are mapped to f-structure positions in the f-structure of their heads. In the literature on LFG, there are two views on exactly what functions specifiers can be mapped to. The traditional approach as described in Dalrymple (2001), Bresnan et al. (2016) is that specifiers map to discourse functions (DF), which consist of topic, focus and subj (which is unique in being simultaneously a grammatical function and a discourse function). However, a trend in much LFG work (King 1997, Butt & King 1997, Dalrymple & Nikolaeva 2011) is to eliminate information structure functions from syntax, instead relegating them to a separate projection, i-structure. Thus Dalrymple et al. (2019) instead propose that specifiers must be *either* syntactically prominent or prominent in information-structure terms. Syntactic prominence means that the f-structure of the specifier is either the subject, or it bears the overlay function dis (which replaces the earlier topic and focus and handles long-distance dependencies). Discourse prominence means that the specifier occupies the discourse functions topic or focus at i-structure.[25] This question is discussed in more detail in Kaplan 2023 [this volume].

According to this approach, then, specifiers can be given annotations as in either (56a) or (56b):

(56)

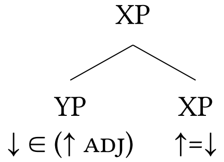


[25] In contemporary LFG, discourse functions are usually modeled not in f-structure but in a separate projection: see Kaplan 2023 [this volume] for more information. The notation in (56) follows the model of information structure in Dalrymple & Nikolaeva (2011).

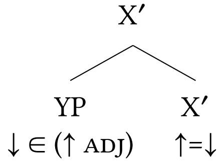
4.3.4 Adjunction

Unlike specifiers and complements, adjuncts may be freely iterated.[26] Naturally, then, they tend to be associated with the only grammatical function that is always set-valued,[27] ADJ (or XADJ), see (57). As new adjuncts are added to the tree, they get added to the adjunct set, thereby not violating uniqueness.

(57) a.



b.



C-structure adjuncts do not always map to f-structure adjuncts, however. Extraposed focused or topicalized material is often adjoined at c-structure, especially at XP level; it is then associated with an information structure function like TOPIC or FOCUS, a grammatical function, and the overlay function DIS.

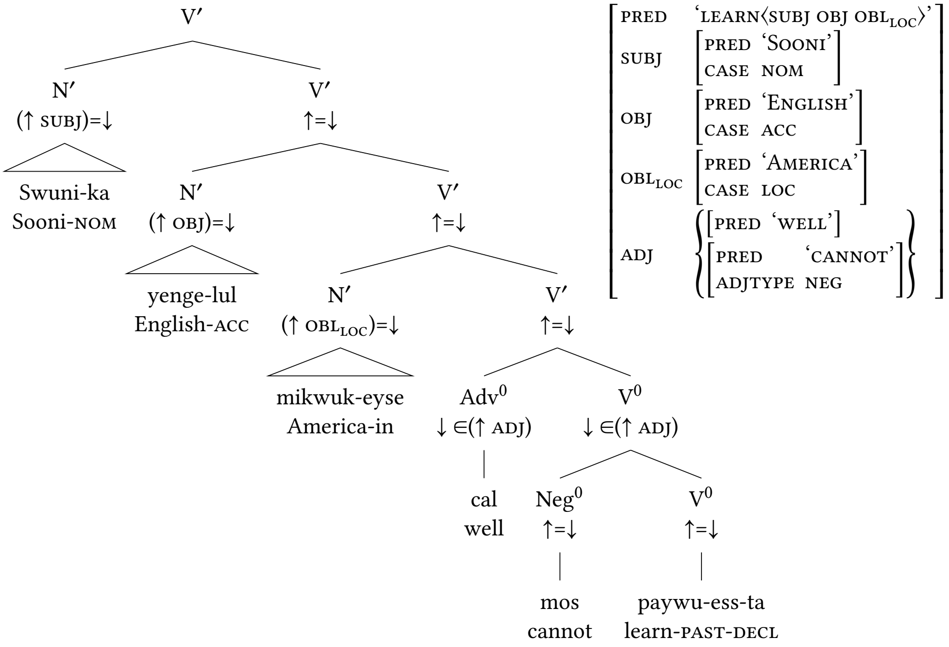
Some analyses also use adjunction as the main mechanism of introducing grammatical functions, not only adjuncts, into the f-structure, without them having any special information structure role. A prominent example is the analysis of Japanese and Korean in Sells (1994, 1995). Building on the ideas of Fukui (1986), Sells proposes that the maximal projection in Japanese and Korean is X′, and that the main sentence-building operation is the adjunction of verbal arguments and adjuncts to V′, and nominal dependents to N′. Adjunction of this sort can be described in LFG notation by rules such as (58), where GF is any grammatical function. Unlike flat structures of non-configurational languages, the resulting structures like (59), from Korean, are binary-branching, but the use of unrestricted adjunction of this kind ensures that the order of constituents is free.

[26] As noted in Section 2.1, some versions of LFG X′ theory allow multiple complements or specifiers. However, this is not the same as adjunct iterations, because, if multiple complements or specifiers are used in a grammar, these receive different annotations, thereby not causing a conflict. In contrast, multiple application of the same adjunct rule will lead to a uniqueness violation if it selects the same grammatical function.

[27] Due to the possibility of coordination, all grammatical functions can be set-valued. However, this requires the use of a special syntactic configuration at c-structure, whereas adjuncts are set-valued “by definition”.

(58) $X' \longrightarrow \begin{array}{c} Y' \\ (\uparrow \textsc{gf})=\downarrow \end{array} \quad \begin{array}{c} X' \\ \uparrow=\downarrow \end{array}$ (Sells 1994: 354)

(59)



‘Sooni did not learn English well in America.’ (Sells 1994: 355)

4.3.5 The category S

As discussed above, the category S, being by definition exocentric, does not have a head in the X′-theoretic sense. This does not mean, however, that it has no head in the sense of c- to f-structure mapping, i.e. no node that is annotated as ↑=↓. In fact, S usually includes at least one such node that represents the predicate; for example, in (7), representing the clause structure of Tagalog, the predicative XP is annotated as ↑=↓, which causes the f-structure of the clause to be unified with the f-structure of the predicate, regardless of what its c-structure category may be.[28] Moreover, unlike X′-theoretic structures, a nonconfigurational S node can have more than one head: for example, a V node representing the lexical verb

[28] The actual developed analysis can be somewhat more complex, as there are several views on nonverbal predication in LFG, and the (non-)identity of its structure to that of verbal predications.

and an Aux node representing an auxiliary that contributes tense, agreement and other grammatical information.

It is remarkable that S is the only systematic exception from the X′ schema[29] that is admitted in mainstream LFG, at least in theory. While the use of S for both nonconfigurational and "partially non-endocentric" languages like Tagalog or Irish is universally accepted as a valid and theoretically solid decision, there has been no discussion of exocentric NPs or other categories in the literature. Whether this represents a lack of empirical evidence for such structures in languages of the world, or is simply the result of a lack of focus and a kind of predetermined conviction, is not clear.

4.3.6 Optionality of c-structure positions

Now that X′ theory is supplemented by f-structure well-formedness constraints and annotation principles, we can introduce an additional feature of LFG c-structures: ECONOMY OF EXPRESSION, which amounts to optionality of most nodes, because the relevant grammatical constraints are for the most part captured at f-structure. This broad principle is formulated in the most radical way in Bresnan et al. (2016: 90), who state that *all* nodes (including nonbranching intermediate X′ projection nodes, heads, complements and specifiers) are optional:

(60) ECONOMY OF EXPRESSION:
All syntactic phrase structure nodes are optional and are not used unless required by independent principles (completeness, coherence, semantic expressivity). (Bresnan et al. 2016: 90)

Note that this is a *theoretical* principle whose *formal* implementation is a separate issue, partly discussed in Section 5.2.1. For example, in the standard phrase stucture rule formalism, the notions of complement and specifier crucially depend on the presence of intermediate X′ nodes, even if these are redundant in the sense of unary branching. Thus, as Dalrymple et al. (2015) observe in their detailed discussion of economy of expression, this principle leads to a proliferation of rules, such as in (61).

(61) X′ ELISION (Dalrymple et al. 2015: 384)
If an LFG grammar G_G contains an annotated rule of the form

[29] It is also the only consistent exception from endocentricity, although, as an anonymous reviewer observes, the X′ theory elaborated in Bresnan et al. (2016) only requires endocentricity for lexical, not functional, projections (p. 137), thereby allowing, among other things, the standard treatment of mixed categories (Bresnan et al. 2016: 311ff.).

$$\begin{array}{cccccc} \text{XP} & \longrightarrow & \alpha & \text{X}' & \beta \\ & & & \uparrow=\downarrow & \end{array}$$

it also contains a rule of the form

$$\begin{array}{cccccc} \text{XP} & \longrightarrow & \alpha & \text{X} & \beta \\ & & & \uparrow=\downarrow & \end{array}$$

In general, Dalrymple et al. (2015) conclude that economy of expression is plausible as an informal principle that emerges through the interaction of other, more basic principles, and that grammars, in general, tend to obey; but it is not plausible as a formal principle to be incorporated into the theory of grammar, because it not only introduces additional complexity into the framework, but also fails to account for cases of genuine non-optionality (such as, for example, in configurational languages where certain nodes are obligatory regardless of independent principles).

Still, the degree of optionality commonly allowed in LFG grammars is rather large and certainly greater than what is assumed by most other phrase-structure-based frameworks. I will now go through each of the X′ theoretic categories and show why they can be optional (except adjuncts, because these are optional by definition, by virtue of the rules that introduce them).

4.3.6.1 Complements and specifiers

Complements and specifiers not only can but must, as a rule, be optional because the c-structure does not contain any valency information and there is no way to verify at c-structure if, for example, the verb has a direct object. Thus, the rule in (43), repeated in (62), will hold for all English sentences, but the NP complement will only be licensed in transitive clauses.

(62) $$\text{VP} \longrightarrow \begin{array}{c} \text{V} \\ \uparrow=\downarrow \end{array} \left(\begin{array}{c} \text{NP} \\ (\uparrow \textsc{obj})=\downarrow \end{array} \right)$$

If the verb is transitive (i.e. its PRED feature has OBJ in the list of arguments), omitting the complement will result in a violation of Completeness (unless the object is introduced in another position). By contrast, if the verb is intransitive, introducing the object here will lead to a Coherence violation, because the grammatical function OBJ will not be selected by any argument.

Optionality of complement and specifier positions, and c-structure positions where arguments are introduced in general, is also required because the material that they “canonically” contain may be displaced elsewhere, for example, to a position designated for wh-movement or information structure function. In this case, only one position must be filled, otherwise conflict of PRED values will lead to a Uniqueness violation. Thus (43) may produce a single V node even in a transitive clause, provided that the direct object is introduced in another position (such as wh-movement *Whom did you see?* or topicalization *John, I saw.*).

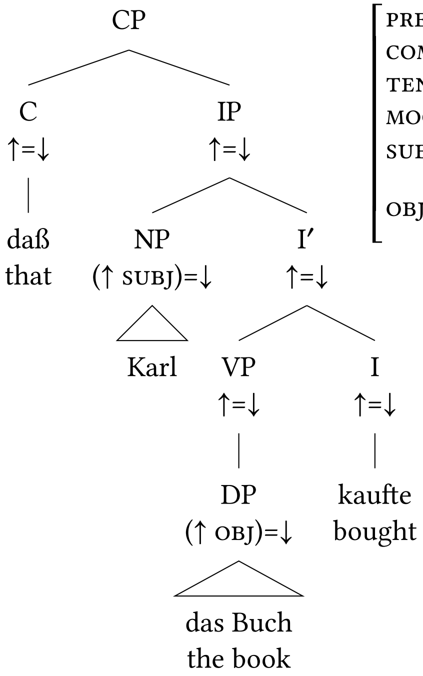
4.3.6.2 Heads

Similarly, c-structure heads can be optional in LFG because of Completeness and Coherence. PRED features are almost always[30] introduced by head nodes, i.e. nodes carrying the unificational annotation ↑=↓. Therefore, a structure lacking a head (without its PRED features introduced elsewhere) will be PRED-less and will not be able to include any grammatical functions, because that would violate Coherence.

Headless XPs are quite widespread at clause level; their role is to account for variation in head positions in configurational languages. For example, in English lexical verbs always appear in V, but the I head can be filled or not depending on whether the verb form is periphrastic or synthetic. In German and other V2 languages, the distribution is more complex: the V head is only occupied if the verb form is periphrastic, and the auxiliary, or the finite verb in synthetic forms, stands in the I node in subordinate clauses (63) and in the C node in main clauses (64). Examples are from Bresnan et al. (2016: 448–450).

[30] It is technically possible to introduce a PRED feature in a different position. For example, the annotation of a complement or specifier might include an additional annotation like (↑ OBJ PRED)='PRO'. I am not aware of any analyses utilizing this possibility; "external" PRED assignment normally only happens in verbal heads assigning PRED features to pro-dropped subjects and in similar such structures. However, Mary Dalrymple (p.c.) points out that such annotations seem to be required in asyndetic relative clauses like *The man John saw*, where the pronominal OBJ in the relative clause has to be introduced by a phrase structure rule since there is no lexical material that could plausibly contribute its content.

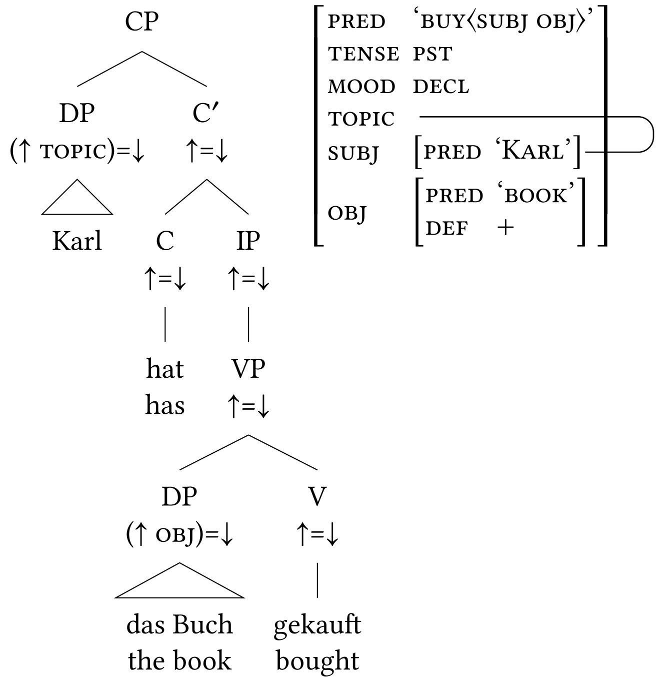
(63)



$$\left[\begin{array}{ll} \textsc{pred} & \text{‘buy}\langle\textsc{subj obj}\rangle\text{’} \\ \textsc{comptype} & \textsc{that} \\ \textsc{tense} & \textsc{past} \\ \textsc{mood} & \textsc{decl} \\ \textsc{subj} & \left[\begin{array}{ll} \textsc{pred} & \text{‘}\textsc{Karl}\text{’} \end{array}\right] \\ \textsc{obj} & \left[\begin{array}{ll} \textsc{pred} & \text{‘}\textsc{book}\text{’} \\ \textsc{def} & + \end{array}\right] \end{array}\right]$$

‘that Karl bought the book’

(64)



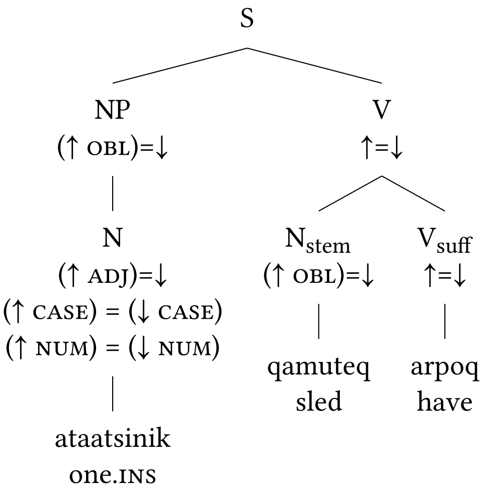
‘Karl has bought the book.’

This analysis corresponds quite closely to the standard view of German word order in GB / Minimalism, such as Vikner (1995). The key difference is that there is no verb movement in LFG; verbs and auxiliaries are always “base-generated” in C, I, or V depending on clause types and the verb form. The correct word order is ensured by feature licensing; multiple occurences of a verb form or verbless sentences are excluded at f-structure through Uniqueness and Coherence.

Another type of headless XP occurs in languages which allow freely discontinuous constituents, like the example from Warlpiri in (51) above. Non-configurational languages like Warlpiri allow freely assigning any grammatical function to Spec,IP (which is additionally interpreted as a focus) and to any NP children of S. Hence, two or more NPs might be mapped to the same grammatical function; if there is no PRED clash or case mismatch, the resulting sentences will be grammatical and these multiple NPs will be mapped to the same f-structure. For more information on non-configurational languages, see Andrews 2023 [this volume].

Finally, headless constituents appear in certain instances of incorporation, such as in West Greenlandic (65), where an incorporated noun head can have non-incorporated dependents (here, agreeing in instrumental case with the incorporated argument).

(65) Greenlandic (Bresnan et al. 2016: 446)



$$\left[\begin{array}{ll} \text{PRED} & \text{‘HAVE}\langle\text{SUBJ OBJ}\rangle\text{’} \\ \text{MOOD} & \text{INDIC} \\ \text{SUBJ} & \left[\begin{array}{ll} \text{PRED} & \text{‘PRO’} \\ \text{NUM} & \text{SG} \\ \text{PERS} & 3 \\ \text{CASE} & \text{ABS} \end{array}\right] \\ \text{OBL} & \left[\begin{array}{ll} \text{PRED} & \text{‘SLED’} \\ \text{CASE} & \text{INS} \\ \text{NUM} & \text{PL} \\ \text{ADJ} & \left[\begin{array}{ll} \text{PRED} & \text{‘ONE’} \\ \text{CASE} & \text{INS} \\ \text{NUM} & \text{PL} \end{array}\right] \end{array}\right] \end{array}\right]$$

‘I have one sled.’

5 Extensions of the core architecture

The core architecture of LFG has remained remarkably stable since the framework was first introduced in Bresnan (1982a); the only major innovations are the introduction of various additional projections, briefly described in Belyaev 2023b: Section 5 [this volume], and functional uncertainty (earlier LFG used traces to model long-distance dependencies). Nevertheless, there have been proposals to alter and extend the core architecture, mainly from three directions: to adopt a view of c-structure different from context-free grammar; to introduce construction-based approaches to LFG using templates; to eliminate PRED values, fully relegating their work to semantics. None of these approaches have been adopted by mainstream LFG practitioners, with the exception of templates, which have gained some acceptance. Nevertheless, these proposals may represent venues in which LFG could develop in the future.

5.1 Constructions and LFG: Templates

In many ways, LFG is close in spirit to other non-transformational frameworks such as HPSG (Pollard & Sag 1994) or various versions of construction grammar (see Hoffmann & Trousdale 2013). All these frameworks, unlike mainstream generative grammar, are not committed to cross-linguistically universal structures and instead define syntactic rules on a language-by-language basis. However, LFG is crucially different from these other approaches in lacking any concept comparable to the notion of construction. The basic building blocks of syntax are phrase structure rules and lexical entries (which formally are a subtype of phrase structure rules); there is a general set of principles governing the mapping from phrase structure positions to f-structure. It is, of course, possible to define separate phrase structure rules and lexical entries to handle specific phenomena and constructions, but these will not be formally related to other rules – there is no hierarchy of phrase structure rules that would allow defining, for example, an exceptional subtype of a specifier rule. In general, most theoretical principles in LFG (such as the principles of c- to f-structure mapping described above) are formulated in such a way as to define a structure that obtains by default, but which can be overriden in individual languages. This is at odds with the main tenets of construction-based approaches, where no general or universal principles or structures are usually assumed, and each construction hierarchy is language-specific.

Furthermore, while it is possible to define rules that are specific to individual constructions or lexical items, it is impossible to directly define a construction that spans more than the scope of one phrase structure rule (e.g., a specific combination of a specifier, head and complement). Of course, the same effect may be achieved by using combinations of defining and constraining equations, as in analyses of idioms; for an example, see Falk (2001: 77). But such analyses do not treat idioms or constructions as theoretical objects in their own right; the collocation is only enforced by the combination of equations acting at different levels.

These "limitations" related to the c-structure to f-structure correspondence are not necessarily disadvantages of the LFG system: they are the result of a conscious design decision that influences the way LFG analyses are structured; in most cases, it is possible to account for "construction-based" phenomena in LFG, but the description will be different than in Construction Grammar and related frameworks. However, there are certainly genuine cases of construction-specific phenomena, such as so-called multi-word expressions (MWEs); these are difficult to describe in standard LFG. A possible, but radical, solution is the replacement of context-free grammar by Tree-Adjoining Grammar (TAG) at c-structure, as described in Findlay 2023 [this volume].

Another reason why some counterpart to the notion of construction might be useful in LFG is that f-structure equations associated with rules and lexical items are not generalized in any way. Thus, nouns may have annotations such as (↑ NUM)=SG and (↑ NUM)=PL, and verbs, (↑ TENSE)=PST, but nothing in the grammar *requires* nouns and verbs to introduce these equations, and there is no place where such generalizations are stated explicitly – in effect, they are only the result of consistency on the part of the grammar writer.[31] This limitation, again, cannot be overcome using a kind of type inheritance system common to construction-based approaches, because that would require a "hierarchy" of f-descriptions. But f-descriptions are only sets of expressions, not objects that can be manipulated or inherit information from each other.

A possible compromise between the description-based approach of LFG and constructions, explored in Asudeh et al. (2013), is based on the use of TEMPLATES – bundles of grammatical descriptions extensively used in computational LFG, such as in XLE, but also in some theoretical work (Dalrymple et al. 2004, Asudeh 2012). Templates are basically symbols that serve as shorthands for f-descriptions that are substituted for the template call wherever it is invoked in an f-description. For example, the combination of third person and singular number agreement, highly relevant for English grammar, can be abbreviated as the template 3SG (66). This template can then be called as in (67). Furthermore, just like an f-description, a template can be negated; thus, as Asudeh et al. (2013: 19) propose, English unmarked present-tense forms can be naturally captured as in (68a), which resolves to (68b).[32]

(66) 3SG ≡[33] $(\uparrow \text{SUBJ PERS})=3$
$(\uparrow \text{SUBJ NUM})=\text{SG}$

(67) *laughs* V $(\uparrow \text{PRED}) = \text{`LAUGH}\langle\text{SUBJ}\rangle\text{'}$
@3SG

(68) a. *laugh* V $(\uparrow \text{PRED}) = \text{`LAUGH}\langle\text{SUBJ}\rangle\text{'}$
¬@3SG

b. *laugh* V $(\uparrow \text{PRED}) = \text{`LAUGH}\langle\text{SUBJ}\rangle\text{'}$
$\{ (\uparrow \text{SUBJ PERS}) \neq 3$
$\mid (\uparrow \text{SUBJ NUM}) \neq \text{SG} \}$

[31] Note that in LFG, this issue is distinct from the issue of permissible f-structure attributes and values discussed in Section 3.3. The two are, of course, related, and would have been the same issue in other frameworks, but not in LFG, where, as discussed above, structures are distinct from the descriptions that license them. An LFG grammar may not generalize over *structures* directly (unless feature declarations are used), but it may well generalize over *descriptions*.

[32] Note that such negation tacitly changes the equation type from defining to constraining, because negative statements can only be constraining. This change is not formally problematic, but care should be taken to ensure that other parts of grammar, which may depend on these defining equations, are not compromised.

[33] Asudeh et al. (2013) use := for template assignment, which is a standard assignment operator in some programming languages (e.g. Pascal), also used in computer science.

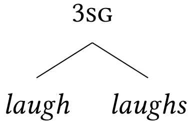
Templates can also be parametric, with parameters supplied in parentheses, as in programming languages. When a template is called, all mentions of each parameter are replaced by the string given in the parentheses. Note that this is done via simple string substitution,[34] and the parameters can be any kind of symbol; often, a reference to an f-structure, but not necessarily. For example, Asudeh et al. (2013) define the following template for intransitive verbs:

(69) INTRANS(P) ≡ (↑ PRED) = ‘P⟨SUBJ⟩’

(70) *laughs* V @INTRANS(LAUGH)
@3SG

Templates by themselves are not theoretical objects: they are a simple mechanism for reusing common parts of f-descriptions. Nevertheless, if used consistently, they can serve as a powerful mechanism for capturing generalizations in grammatical structure. In particular, a kind of hierarchy of templates can be defined if the use of a template in a lexical item, phrase structure rule, or in another template is viewed as inheritance from that template. For example, both *laugh* and *laughs* inherit from the 3SG template:[35]

(71)



Asudeh et al. (2013) use this template system to develop a detailed analysis of the traversal / result construction (*Smithy drank his way through university*, Jackendoff 1992, Goldberg 1995) in English, Swedish, and Dutch. Since this seminal work, templates have been widely used in LFG literature, although their adoption is not universal. Importantly, an advantage of the template-based approach to constructions is that they only introduce a purely notational convention; they

[34] For this reason, if the parameter is an f-structure reference, it may be ambiguous within a template if it includes Functional Uncertainty. To ensure that the same f-structure is referred to in all expressions, the template should first assign the parameter to a local name.

[35] This may seem counterintuitive, given that *laugh* is not a third person singular form. However, inheritance in this approach is purely a matter of calling a template in the f-description: it does not matter in what context it is called (under negation, in disjunction, etc.). This graph captures the intuition that the English unmarked Present Simple form is defined with reference to the third person singular features (as opposed to, e.g., being a disjunction of all alternative person-number combinations). Note that “inheritance” here is purely a matter of visualization and metagrammatical analysis; it has no special status in the formalism itself.

do not change the architecture of LFG in any way. Thus template-based analyses are fully compatible with non-template-based ones.

This simplicity can also be perceived as a disadvantage, in that constructions are not "first class citizens" of the theory: the template mechanism is unconstrained, and its use is fully optional. However, this follows the overall spirit of LFG: As seen above, the core architecture and metalanguage are relatively unconstrained and certainly more expressive than is needed for the purposes of describing natural languages. Constraints on possible languages are meant to be captured by theoretical generalizations (such as the regularities of c- to f-structure mapping described in Section 4.3) that are not part of the formal framework itself. Likewise, templates only serve as a useful mechanism of generalizing over f-descriptions; what these templates should look like and how consistently they should be used are theoretical decisions that should be viewed as additional constraints on LFG grammars, not part of the formal architecture itself.

5.2 Modifications of c-structure

Compared to developments in other frameworks, such as Minimalism (cf. Adger 2013), there have been few advances in the development of constituent structure in LFG. Apart from the introduction of non-projecting words in Toivonen (2003), the version of X′ theory used in most LFG work is the same as the original version developed in transformational grammar. However, there have been several alternative approaches to c-structure proposed in the literature, some relatively minor while others quite radical. In this section, I will describe two approaches – minimal c-structure (Lovestrand & Lowe 2017) and lexical sharing (Wescoat 2002). Another modification (Findlay 2017, 2019), which replaces context-free grammar with tree-adjoining grammar (TAG) while preserving core features of the LFG formalism, is described in Findlay 2023 [this volume]. Several categorial grammar-based approaches have been proposed (Oehrle 1999, Muskens 2001, Kokkonidis 2007), but have not gained much traction, possibly because they are no longer compatible with standard LFG and have to be regarded as separate, though related, frameworks.

5.2.1 Minimal c-structure

Lovestrand & Lowe (2017) propose a modification of X′ theory to account for two shortcomings that they perceive in its standard LFG version. First, X′ categories and projection levels are stipulated by the theory but not actually represented as discrete features; in formal terms, c-structure node labels are just monolithic symbols, even though they are given a theoretical interpretation. Second, consistent application of X′ theoretic principles leads to many redundant nodes, e.g. unary branching X′ nodes have to be used if an XP has a complement but no specifier or adjuncts. This redundancy is sometimes eliminated by appealing to economy of expression, either by "pruning" the superfluous nodes (Bresnan et al. 2016) or by introducing additional rules into the grammar (such as XP ⟶ X ZP in addition to XP ⟶ X′ YP and X′ ⟶ X ZP). However, both solutions introduce additional complexity into LFG and could be avoided. Third, some analyses work with fewer than two levels of X′ structure: for example, Bresnan et al. (2016: 130) take Welsh IP to lack a specifier, dominating only I and S. Sells (1994, 1995) similarly assumes that all phases in Japanese and Korean have X′ as their maximal projection. This kind of "deficiency" is not formalized in traditional X′ theory.

An earlier attempt to refine X′ theory in LFG is Marcotte (2014), which, however, has been criticized in Lovestrand & Lowe (2017) for failing to account for some common syntactic structures, such as adjunction and non-projecting words. Lovestrand and Lowe propose, following Kaplan (1989), that additional categorial features are projected in a separate feature structure (l-structure) via the function λ. L-structure contains the features L (for LEVEL) and P (for PROJECTION) that represent the "current" bar level of the node and the maximal level that this particular phrase has in the sentence. C-structure itself only contains syntactic category information; thus X, X′, and XP are all represented as X. Lovestrand and Lowe then define a set of templates and rule schemas that describe all the positions allowed by X′ theory. For example, the template EXT in (72a) is a conjunction of the tem1lates LPM (72b) and LP (72c), which mean that the annotated node is a maximal projection (LP) that is a daughter of a maximal projection (LPM). This applies to specifiers and adjuncts. The template HEADX (73a) is used on all X′ theoretic heads and consists of the templates LDOWN (73b) and PUD (73c), which mean that, first, the bar-level of the annotated node is lower than the level of the mother by 1; (b) the maximal projection level is inherited from the head to the overall structure. These templates allow us to define the specifier rule template in (74).[36]

(72) templates for specifier

a. EXT ≡ $@\text{LPM} \wedge @\text{LP}$

b. LPM ≡ $(\hat{*}_{\lambda}\ \text{L}) = (\hat{*}_{\lambda}\ \text{P})$

c. LP ≡ $({*}_{\lambda}\ \text{L}) = ({*}_{\lambda}\ \text{P})$

[36] For clarity, conjunction is explicitly represented as ∧ in (72a) and (73a).

(73) templates for head

a. $\textsc{headx} \equiv @\textsc{ldown} \wedge @\textsc{pud}$

b. $\textsc{ldown} \equiv \{(*_{\lambda}\ \textsc{l}) = 0 \wedge (\hat{*}_{\lambda}\ \textsc{l}) = 1 \mid (*_{\lambda}\ \textsc{l}) = 1 \wedge (\hat{*}_{\lambda}\ \textsc{l}) = 2\,\}$

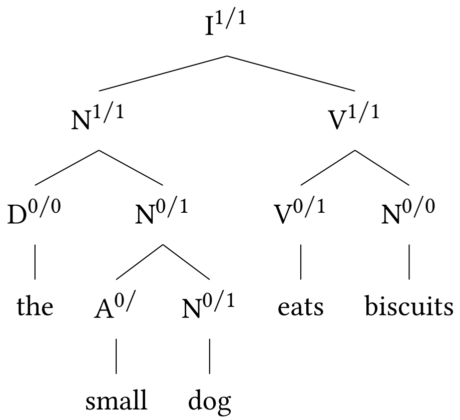
c. $\textsc{pud} \equiv (\hat{*}_{\lambda}\ \textsc{p}) = (*_{\lambda}\ \textsc{p})$

(74) specifier rule

$$\begin{array}{cccc} \mathrm{X} & \longrightarrow & \mathrm{Y} & \mathrm{X} \\ & & @\textsc{ext} & @\textsc{headx} \end{array}$$

The application of this approach leads to c-structures notated as in (75), where the superscript numbers are shorthand for L/P feature values of the node.

(75)



In this example, prenominal A in English is treated as a non-projecting category, hence it lacks the P feature altogether.[37] It is seen from this example that the "maximal" projection level (P) is inherited bottom-up and represents the highest projection that the phrase has in this specific sentence. For example, the specifier noun phrase *the small dog* has a specifier, hence its dominating node has the category $\mathrm{N}^{1/1}$, while the complement *biscuits* has no modifiers, and its head is only $\mathrm{N}^{0/0}$. Thus the system results in minimal c-structures solely by using standard LFG mechanisms of templates and projections, without employing additional formal devices such as Economy of Expression.

[37] While Lovestrand and Lowe assume no DP in English, D is not treated as a non-projecting word: in their theory, *'s* possessors can attach to D as internal arguments (complements).

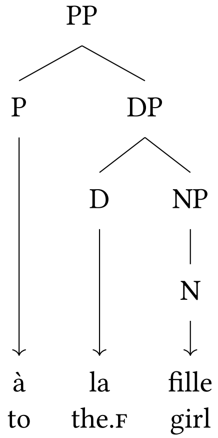
5.2.2 Lexical sharing

The principle of lexical integrity, and the general idea that there is a definite boundary between morphology and syntax, has long been criticized in the generative literature (perhaps the most recent such attempt is Bruening 2018) and, recently, in typological approaches (see Haspelmath 2011). Not all of the objections to lexicalism are necessarily applicable to LFG, but one persistent problem is the putative existence of syntactic structure where one lexical item (either completely idiosyncratic or derived in the morphology) occupies two or more syntactic heads. One example are preposition-determiner contractions in languages like French and German (Wescoat 2007): Items like French *au* [o] 'to the (masculine)' (⟵ *à* + *le*) are clearly idiosyncratic, historically motivated mergers of the preposition and the article (compare *à le faire* 'to do it', where *le*, identical in form to the masculine singular definite article, is the object proclitic of *faire* 'do', and thus does not trigger merger), but syntactically, they obey all the constraints that are independently imposed on prepositions and determiners in the language.

To account for such phenomena, Wescoat (2002) proposed LEXICAL SHARING: a modification of the LFG architecture to allow a single word (supplied by the lexicon) to occupy more than one c-structure node. In Wescoat's system, lexical items are no longer part of c-structure; category nodes like N, V, I (preterminals in the standard system) are now terminal nodes that are mapped, via the projection function λ, to morphological words that comprise an ordered list at a separate level of representation, l-structure.[38] In the simplest and most common case, each terminal c-structure node corresponds to exactly one word:

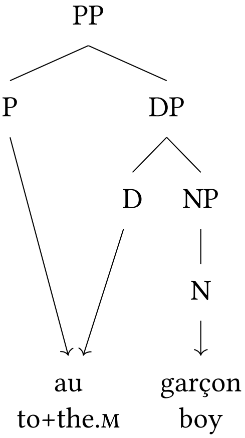
[38] It is unfortunate that the same name of the level and the projection function were independently used in Lovestrand & Lowe's (2017) proposal of minimal c-structure, which creates confusion. However, as will be shown below, Wescoat's approach can be integrated into the contemporary LFG architecture without stipulating an additional level.

(76)



Lexical sharing occurs when two or more terminal c-structure nodes are mapped to one morphological word:

(77)



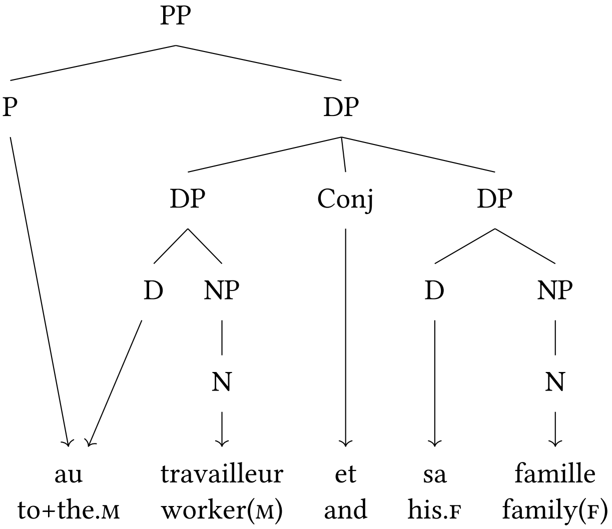
To avoid excessive reorderings, Wescoat puts a constraint on the correspondence between c-structure and l-structure which he calls the *order preservation axiom*: For all n_1 and n_2 in the set of terminal nodes, if $\lambda(n_1)$ precedes $\lambda(n_2)$, then n_1 precedes n_2. This means that words cannot be reordered. It also follows from this axiom that only adjacent nodes may be shared. Thus lexical sharing is, in fact, rather constrained and does not seem to introduce much additional complexity into the system.

Lexical entries in lexical sharing analyses are defined as in (78), with each node having a separate f-description. The syntactic analysis then proceeds according to the standard f-structure rules defined by the grammar; lexical sharing configurations are licensed if a word is defined as coinstantiating adjacent nodes.

(78) $$\textit{au} \longleftarrow \begin{array}{cc} \mathrm{P} & \mathrm{D} \\ (\downarrow \textsc{pcase})=\textsc{to} & (\downarrow \textsc{spec})=\textsc{def} \\ \Downarrow=\downarrow & (\downarrow \textsc{gend}) =_c \textsc{m} \\ & (\downarrow \textsc{num}) =_c \textsc{sg} \end{array}$$

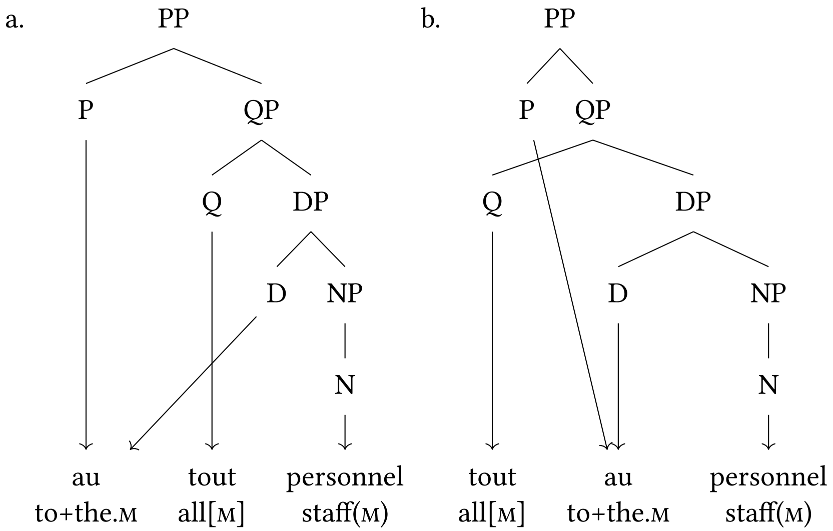
This correctly predicts the scope difference between the preposition and definite article in examples like (79). The order preservation axiom also predicts that structures like (80a) and (80b) are ungrammatical, because the shared nodes are not adjacent; the only possible word order is (81).

(79)



‘to the worker and his family’

(80) Ill-formed c-structures:



(81)

á	tout	le	personnel
to	all[M]	the.M	staff

‘to all the staff’

Note that Wescoat assumes that the correspondence function ϕ should have l-structure in its domain, hence the use of ↓ in annotations, instead of ↑ in standard LFG analyses. This assumption also motivates the symbol ⇓; this stands for the abbreviation $\phi(\lambda(\downarrow))$, i.e. “the f-structure of the lexical exponent of the current node” – this is needed to determine which of the co-instantiated f-structures the word itself maps to. However, this is not actually required, and Lowe (2016), in his analysis of the English “Saxon genitive” *’s*, proposed a modification of lexical sharing that dispenses with both these additional notations and integrates the proposal into modern mainstream LFG. Lowe observes that Wescoat’s “l-structure” in fact serves the exact same function as the s-string – the set of morphosyntactic words that map to terminal c-structure nodes – in the LFG projection architecture, including the recent proposal of Dalrymple & Mycock (2011). Ordinarily, the s-string in LFG maps to terminal tree nodes that are occupied by morphosyntactic words; lexical sharing can be implemented by assuming that the c-structure tree terminates in category labels (preterminals), to which elements of the s-string are mapped. The replacement of l-structure by the s-string means that the symbol ⇓ and all the related machinery is no longer needed, because s-structure does not map to f-structure.[39] For the same reason, lexical entries use ↑, as in normal LFG, instead of ↓.[40] In Lowe's version of lexical sharing, the entry in (78) will look as follows:

(82) *au*: P D
P (↑ PCASE) = TO
D (↑ SPEC) = DEF
(↑ GEND) $=_c$ M
(↑ NUM) $=_c$ SG

While lexical sharing has been used to analyze several phenomena, including auxiliary reduction (Wescoat 2005), preposition-determiner contractions (Wescoat 2007), suspended affixation (Broadwell 2008, Belyaev 2014, 2021), endoclitics (Wescoat 2009), and morphologically bound complementation (Panova 2020), it has not been adopted as part of mainstream LFG, mainly, it seems, due to its apparent violation of lexical integrity and the potential to vastly increase the number of possible analyses. Indeed, if unconstrained, lexical sharing can be used to produce structures where every morphological category has its separate functional projection that is shared with the lexical head, reminiscent of Distributed Morphology (DM, Halle & Marantz 1993). However, as both Broadwell (2008) and Lowe (2016) observe, lexical sharing can be constrained to be used only when there is independent syntactic evidence in favour of a separate lexical head. Under this interpretation, lexical sharing analyses present an advantage over non-lexicalist analyses in that functional heads like CaseP or NumP are only stipulated as needed; for example, in Broadwell's (2008) analysis of suspended affixation, there is an empirical difference between languages where case is realized by a cointantiated head (these allow suspended affixation) and languages where it is purely morphological (these do not); this opposition is lost in non-lexicalist approaches, where other, arguably less intuitively plausible mechanisms have to be used, such as a constraint on coordinating sub-CaseP constituents, feature deletion (Kharytonava 2012), or morphological ellipsis (Erschler 2012).

Furthermore, as mentioned in Section 2.2, lexical sharing does not really violate lexical integrity as formulated in Bresnan & Mchombo (1995), see (8), i.e. as

[39] This seems rather harmless, because lexical sharing entries overwhelmingly just use ⇓=↓ on one of the nodes, which doesn't seem to influence anything. However, Wescoat (2007) does use constraints on the l-structure to f-structure mapping to model certain limitations on preposition-determiner contractions in German.

[40] In fact, while standard LFG allows using ↓ in lexical entries, this model does not. This means that analyses that make use of ↓ in lexical entries, such as the Italian example in (48), have to be reformulated to use ↑. In most cases, this should not influence anything, although the definition and application of f-precedence might require some modification.

the general principle that words are built from different blocks and according to different rules than syntactic units. Indeed, syntax does not have any access to internal word structure in lexical sharing analyses, and coinstantiated nodes map to words as complete units, not to morphemes, disembodied features, or anything similar.[41] This gives lexical sharing analyses a distinct flavour that separates them from both mainstream LFG analyses and from truly non-lexicalist analyses that situate morphemes or features in functional projections (which have also been proposed in LFG: see Melchin et al. 2020 for a DM-like approach to LFG morphology). Notably, it still allows an independent morphological module (usually described in LFG in terms of a lexicalist realizational framework like PFM, see Belyaev (2021) and Asudeh & Siddiqi 2023 [this volume]) to do its work.

6 Conclusion

In this article, I have tried to summarize the state of the art of the core syntactic representations of LFG – the c- and f-structures. While the understanding of various phenomena has considerably changed in almost all areas of grammar (for example, in semantics and information structure: see Asudeh 2023 [this volume] and Zaenen 2023 [this volume]), the formal underpinnings of LFG have remained remarkably stable over the years. The only fundamental innovation to the original c- and f-structure architecture of Kaplan & Bresnan (1982) is the introduction of functional uncertainty in Kaplan & Zaenen (1989b). Since then, new levels of projection were introduced, and the architecture extended in various ways, but the core mechanisms of c- and f-structure – notation, featurehood, even the basic set of GFs – have remained constant. This serves as an impressive testimony of the versatility of the architecture proposed in Kaplan & Bresnan (1982), and its remarkable suitability to describing natural languages.

The architecture of LFG is both similar to that of other constraint-based frameworks and very different from them in various ways. The main difference is the parallel architecture of LFG, and the related emphasis on the distinction between descriptions (a set of syntactic constraints) and the structures that are licensed by

[41] In fact, from a certain perspective this might be viewed as a disadvantage of lexical sharing analyses in that they fail to capture the fact that coinstantiated material usually corresponds to a well-defined, segmental, agglutinatively attached element of the wordform. For example, Ossetic affixal case features are realized on the Case head, while stem-based ones are realized on N (Belyaev 2014, 2021). I am not aware of any analyses where coinstantiated heads encode features that are realized by stem change, suppletion, or apophony. This fact might be explained diachronically, however, since lexical sharing usually reflects an ongoing process of (de)grammaticalization.

these descriptions. While constructions and lexical entries are *structures* in most other frameworks, in LFG they are sets of *statements* that describe a range of possible structures. This architectural feature enables LFG to make use of mechanisms such as functional uncertainty and inside-out application, which are unavailable in other frameworks.

While the empirical coverage of LFG work is impressive, and a number of important developments are now taking place in several theoretical directions, not all areas of syntax have been researched to the same extent. The focus on f-structure and the view of GFs as theoretical primitives has prompted a lot of fruitful and insightful work on subjects and other core grammatical relations. Functional uncertainty and structure sharing have also proved to be efficient mechanisms for describing long-distance dependencies. The notion of sets and feature distributivity allow for elegant analyses of coordination – an area traditionally underrepresented in mainstream syntactic frameworks. In contrast, c-structure has seen much less attention,[42] although here important developments are also taking place. The notion of lexical integrity, assumed as a stipulation early in the history of LFG, has not been extensively discussed and refined, in spite of numerous challenges. These challenges will have to be dealt with if LFG is to compete with other frameworks for the originally envisaged role of "a theoretically justified representation of the native speaker's linguistic knowledge" (Kaplan & Bresnan 1982).

Acknowledgements

I am grateful to two anonymous reviewers, whose comments made a significant contribution to the final form of this chapter, and to Mary Dalrymple for her attentive reading and insightful comments. This research has been supported by the Interdisciplinary Scientific and Educational School of Moscow University "Preservation of the World Cultural and Historical Heritage".

[42] The reason for this might be that the range of phenomena handled by c-structure is much less than those handled by f-structure, as c-structure only directly models word order and embedding. However, as an anonymous reviewer observes, c-structure in LFG is analogous to Merge in Minimalism, being the main generative component that connects different projections together while also providing codescription for the semantics. This role can hardly be described as minor, but the existing model handles this purpose rather adequately.

References

Adger, David. 2013. *A syntax of substance* (Linguistic Inquiry Monographs 64). Cambridge, MA: The MIT Press. DOI: 10.7551/mitpress/9780262018616.001.0001.

Andrews, Avery D. 2008. The role of PRED in LFG+Glue. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '08 conference*, 46–67. Stanford: CSLI Publications.

Andrews, Avery D. 2018. Sets, heads and spreading in LFG. *Journal of Language Modelling* 6(1). 131–174. DOI: 10.15398/jlm.v6i1.175.

Andrews, Avery D. 2023. Clause structure and configurationality. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 375–424. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185952.

Asudeh, Ash. 2012. *The logic of pronominal resumption* (Oxford Studies in Theoretical Linguistics). Oxford: Oxford University Press. DOI: 10.1093/acprof:oso/9780199206421.001.0001.

Asudeh, Ash. 2023. Glue semantics. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 651–697. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185964.

Asudeh, Ash, Mary Dalrymple & Ida Toivonen. 2013. Constructions with Lexical Integrity. *Journal of Language Modelling* 1(1). 1–54. DOI: 10.15398/jlm.v1i1.56.

Asudeh, Ash & Gianluca Giorgolo. 2012. Flexible composition for optional and derived arguments. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '12 conference*, 64–84. Stanford: CSLI Publications.

Asudeh, Ash, Gianluca Giorgolo & Ida Toivonen. 2014. Meaning and valency. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '14 conference*, 68–88. Stanford: CSLI Publications.

Asudeh, Ash & Daniel Siddiqi. 2023. Morphology in LFG. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 855–901. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185976.

Asudeh, Ash & Ida Toivonen. 2006. Symptomatic imperfections. *Journal of Linguistics* 42. 395–422.

Austin, Peter K. & Joan Bresnan. 1996. Non-configurationality in Australian aboriginal languages. *Natural Language & Linguistic Theory* 14(2). 215–268. DOI: 10.1007/bf00133684.

Belyaev, Oleg. 2013. Optimal agreement at m-structure. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '13 conference*, 90–110. Stanford: CSLI Publications.

Belyaev, Oleg. 2014. Osetinskij kak jazyk s dvuxpadežnoj sistemoj: gruppovaja fleksija i drugie paradoksy padežnogo markirovanija [Ossetic as a two-case language: Suspended affixation and other case marking paradoxes]. *Voprosy jazykoznanija* 6. 31–65.

Belyaev, Oleg. 2021. Paradigm structure influences syntactic behaviour: Ossetic case inflection. In I Wayan Arka, Ash Asudeh & Tracy Holloway King (eds.), *Modular design of grammar: Linguistics on the edge*, 251–281. Oxford: Oxford University Press. DOI: 10.1093/oso/9780192844842.003.0016.

Belyaev, Oleg. 2023a. Grammatical functions in LFG. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 97–161. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185938.

Belyaev, Oleg. 2023b. Introduction to LFG. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 3–22. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185934.

Belyaev, Oleg, Mary Dalrymple & John J. Lowe. 2015. Number mismatches in coordination. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '15 conference*, 26–46. Stanford: CSLI Publications.

Blackburn, Patrick & Claire Gardent. 1995. A specification language for Lexical-Functional Grammars. In *Proceedings of the 7th conference of the European chapter of the ACL (EACL 1995)*, 39–44. Association for Computational Linguistics. DOI: 10.3115/976973.976980.

Bögel, Tina, Miriam Butt, Ronald M. Kaplan, Tracy Holloway King & John T. III Maxwell. 2010. Second position and the prosody-syntax interface. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '10 conference*, 106–126. Stanford: CSLI Publications. http://csli-publications.stanford.edu/LFG/15/papers/lfg10boegeletal.pdf.

Bresnan, Joan. 1982a. Control and complementation. *Linguistic Inquiry* 13(3). Reprinted in Bresnan 1982b, 343–434.

Bresnan, Joan (ed.). 1982b. *The mental representation of grammatical relations*. Cambridge, MA: The MIT Press.

Bresnan, Joan, Ash Asudeh, Ida Toivonen & Stephen Wechsler. 2016. *Lexical-Functional Syntax*. 2nd edn. (Blackwell Textbooks in Linguistics 16). Malden, MA: Wiley-Blackwell.

Bresnan, Joan, Ronald M. Kaplan & Peter Peterson. 1985. Coordination and the flow of information through phrase structure. Unpublished manuscript, Xerox PARC.

Bresnan, Joan & Sam A. Mchombo. 1995. The lexical integrity principle: Evidence from Bantu. *Natural Language & Linguistic Theory* 13(2). 181–254. DOI: 10.1007/bf00992782.

Broadwell, George Aaron. 2008. Turkish suspended affixation is lexical sharing. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '08 conference*, 198–213. Stanford: CSLI Publications.

Bruening, Benjamin. 2014. Precede-and-command revisited. *Language* 90(2). 342–388. DOI: 10.1353/lan.2014.0037.

Bruening, Benjamin. 2018. The lexicalist hypothesis: Both wrong and superfluous. *Language* 94(1). 1–42. DOI: 10.1353/lan.2018.0000.

Butt, Miriam & Tracy Holloway King. 1997. Null elements in discourse structure. Unpublished manuscript intended for publication in the Proceedings of the NULLS Seminar. http://ling.uni-konstanz.de/pages/home/butt/main/papers/nulls97.pdf.

Carpenter, Robert L. 1992. *The logic of typed feature structures*. Cambridge, UK: Cambridge University Press. DOI: 10.1017/cbo9780511530098.

Chomsky, Noam. 1970. Remarks on nominalization. In Roderick A. Jacobs & Peter S. Rosenbaum (eds.), *Readings in English transformational grammar*, 184–221. Waltham, MA: Ginn.

Corbett, Greville G. 2012. *Features*. Cambridge, UK: Cambridge University Press. DOI: 10.1017/cbo9781139206983.

Crouch, Richard, Mary Dalrymple, Ronald M. Kaplan, Tracy Holloway King, John T. III Maxwell & Paula S. Newman. 2011. *XLE Documentation*. Xerox Palo Alto Research Center. Palo Alto, CA. https://ling.sprachwiss.uni-konstanz.de/pages/xle/doc/xle_toc.html.

Crouch, Richard & Tracy Holloway King. 2008. Type-checking in formally non-typed systems. In *Proceedings of the ACL workshop on Software Engineering, Testing, and Quality Assurance for Natural Language Processing*, 3–4. Association for Computational Linguistics. DOI: 10.3115/1622110.1622112.

Dalrymple, Mary. 1993. *The syntax of anaphoric binding*. Stanford: CSLI Publications.

Dalrymple, Mary. 2001. *Lexical Functional Grammar* (Syntax and Semantics 34). New York: Academic Press. DOI: 10.1163/9781849500104.

Dalrymple, Mary & Ronald M. Kaplan. 2000. Feature indeterminacy and feature resolution. *Language* 76. 759–798. DOI: 10.2307/417199.

Dalrymple, Mary, Ronald M. Kaplan & Tracy Holloway King. 2004. Linguistic generalizations over descriptions. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '04 conference*, 199–208. Stanford: CSLI Publications.

Dalrymple, Mary, Ronald M. Kaplan & Tracy Holloway King. 2015. Economy of Expression as a principle of syntax. *Journal of Language Modelling* 3(2). 377–412. DOI: 10.15398/jlm.v3i2.82.

Dalrymple, Mary, Ronald M. Kaplan, John T. III Maxwell & Annie Zaenen (eds.). 1995. *Formal issues in Lexical-Functional Grammar*. Stanford: CSLI Publications.

Dalrymple, Mary, John Lamping & Vijay Saraswat. 1993. LFG semantics via constraints. In *Proceedings of the 6th conference of the European chapter of the ACL (EACL 1993)*, 97–105. Association for Computational Linguistics. DOI: 10.3115/976744.976757.

Dalrymple, Mary, John J. Lowe & Louise Mycock. 2019. *The Oxford reference guide to Lexical Functional Grammar*. Oxford: Oxford University Press. DOI: 10.1093/oso/9780198733300.001.0001.

Dalrymple, Mary & Louise Mycock. 2011. The prosody-semantics interface. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '11 conference*, 173–193. Stanford: CSLI Publications.

Dalrymple, Mary & Irina Nikolaeva. 2011. *Objects and information structure* (Cambridge Studies in Linguistics). Cambridge, UK: Cambridge University Press. DOI: 10.1017/cbo9780511993473.

Erschler, David. 2012. Suspended affixation in Ossetic and the structure of the syntax-morphology interface. *Acta Linguistica Hungarica*. DOI: 10.1556/aling.59.2012.1-2.7.

Everett, Dan. 2015. Review of *A Syntax of Substance* by David Adger. *American Anthropologist* 117(2). 414–449. DOI: 10.1111/aman.12251.

Falk, Yehuda N. 2001. *Lexical-Functional Grammar: An introduction to parallel constraint-based syntax*. Stanford: CSLI Publications.

Falk, Yehuda N. 2010. An unmediated analysis of relative clauses. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '10 conference*, 207–227. Stanford: CSLI Publications.

Findlay, Jamie Y. 2017. Multiword expressions and lexicalism. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '17 conference*, 200–229. Stanford: CSLI Publications.

Findlay, Jamie Y. 2019. *Multiword expressions and the lexicon*. Oxford: University of Oxford. (D.Phil. Thesis).

Findlay, Jamie Y. 2023. LFG and Tree-Adjoining Grammar. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 2069–2125. Berlin: Language Science Press. DOI: 10.5281/zenodo.10186054.

Forst, Martin & Tracy Holloway King. 2023. Computational implementations and applications. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 1083–1123. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185986.

Fukui, Naoki. 1986. *A theory of category projection and its applications*. Cambridge, MA: Massachusetts Institute of Technology. (Doctoral dissertation).

Goldberg, Adele E. 1995. *Constructions: A Construction Grammar approach to argument structure*. Chicago: University of Chicago Press.

Hale, Ken. 1983. Warlpiri and the grammar of non-configurational languages. *Natural Language & Linguistic Theory* 1. 5–47. DOI: 10.1007/bf00210374.

Halle, Morris & Alec Marantz. 1993. Distributed morphology and the pieces of inflection. In Kenneth Hale & Samuel Jay Keyser (eds.), *The view from Building 20: Essays in linguistics in honor of Sylvain Bromberger*, 111–176. Cambridge, MA: The MIT Press.

Haspelmath, Martin. 2011. The indeterminacy of word segmentation and the nature of morphology and syntax. *Folia Linguistica* 45(1). 31–80. DOI: 10.1515/flin-2017-1005.

Haug, Dag. 2023. Agreement. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 193–218. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185942.

Haug, Dag & Tatiana Nikitina. 2015. Feature sharing in agreement. *Natural Language & Linguistic Theory* 34. 865–910. DOI: 10.1007/s11049-015-9321-9.

Hoffmann, Thomas & Graeme Trousdale (eds.). 2013. *The Oxford handbook of Construction Grammar*. Oxford: Oxford University Press. DOI: 10.1093/oxfordhb/9780195396683.001.0001.

Jackendoff, Ray. 1977. *$\bar{X}$ syntax: A study of phrase structure* (Linguistic Inquiry Monographs 2). Cambridge, MA: The MIT Press.

Jackendoff, Ray. 1992. Babe Ruth homered his way into the hearts of America. In Tim Stowell & Eric Wehrli (eds.), *Syntax and the lexicon* (Syntax and Semantics 26), 155–178. San Diego, CA: Academic Press. DOI: 10.1163/9789004373181.

Kameyama, Megumi. 1985. *Zero anaphora: The case of Japanese*. Stanford: Stanford University. (Doctoral dissertation).

Kaplan, Ronald M. 1989. The formal architecture of Lexical-Functional Grammar. *Journal of Information Science and Engineering* 5. 305–322. Revised version published as Kaplan (1995).

Kaplan, Ronald M. 1995. The formal architecture of Lexical-Functional Grammar. In Mary Dalrymple, Ronald M. Kaplan, John T. III Maxwell & Annie Zaenen (eds.), *Formal issues in Lexical-Functional Grammar*, 7–27. Stanford: CSLI Publications. Earlier version published as Kaplan (1989).

Kaplan, Ronald M. 2023. Unbounded dependencies. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 425–481. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185954.

Kaplan, Ronald M. & Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan (ed.), *The mental rep-*

resentation of grammatical relations, 173–281. Cambridge, MA: The MIT Press. Reprinted in Dalrymple, Kaplan, Maxwell & Zaenen (1995: 29–130).

Kaplan, Ronald M. & John T. III Maxwell. 1996. *LFG Grammar Writer's Workbench*. Xerox Palo Alto Research Center. Palo Alto, CA. https://www.researchgate.net/profile/John_Maxwell5/publication/2760068_Grammar_Writer's_Workbench/links/0c96052405e97928e9000000.pdf.

Kaplan, Ronald M. & Jürgen Wedekind. 1993. Restriction and correspondence-based translation. In *Proceedings of the 6th conference of the European chapter of the ACL (EACL 1993)*, 193–202. Association for Computational Linguistics. DOI: 10.3115/976744.976768.

Kaplan, Ronald M. & Annie Zaenen. 1989a. Functional precedence and constituent structure. In Chu-Ren Huang & Keh-Jiann Chen (eds.), *Proceedings of ROCLING II*, 19–40. Taipei.

Kaplan, Ronald M. & Annie Zaenen. 1989b. Long-distance dependencies, constituent structure, and functional uncertainty. In Mark Baltin & Anthony Kroch (eds.), *Alternative conceptions of phrase structure*, 17–42. Chicago: University of Chicago Press. Reprinted in Dalrymple, Kaplan, Maxwell & Zaenen (1995: 137–165).

Kharytonava, Volha (Olga). 2012. Taming affixes in Turkish: With or without residue? In Thomas Stolz, Hitomi Otsuka, Aina Urdze & Johan van der Auwera (eds.), *Irregularity in morphology (and beyond)* (Studia Typologica 11), 167–185. Berlin: De Gruyter.

King, Tracy Holloway. 1997. Focus domains and information structure. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '97 conference*, 2–13. Stanford: CSLI Publications. http://csli-publications.stanford.edu/LFG/LFG2-1997/lfg97king.pdf.

Kokkonidis, Miltiadis. 2007. Towards a more lexical and functional type-logical theory of grammar. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '07 conference*, 271–292. Stanford: CSLI Publications.

Kroeger, Paul R. 1993. *Phrase structure and grammatical relations in Tagalog*. Stanford: CSLI Publications.

Kuhn, Jonas. 2001. Resource sensitivity in the syntax-semantics interface: Evidence from the German split NP construction. In W. Detmar Meurers & Tibor Kiss (eds.), *Constraint-based approaches to Germanic syntax* (Studies in Constraint-Based Lexicalism), 177–216. Stanford: CSLI Publications.

Kuhn, Jonas. 2003. *Optimality-Theoretic Syntax – A declarative approach*. Stanford: CSLI Publications.

Laczkó, Tibor. 2014. An LFG analysis of verbal modifiers in Hungarian. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '14 conference*, 346–366. Stanford: CSLI Publications.

Langacker, Ronald W. 1969. On pronominalization and the chain of command. In David A. Reibel & Sanford A. Schane (eds.), *Modern studies in English*, 160–186. Englewood Cliffs, NJ: Prentice-Hall.

Lovestrand, Joseph & John J. Lowe. 2017. Minimal c-structure: Rethinking projection in phrase structure. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '17 conference*, 285–305. Stanford: CSLI Publications.

Lowe, John J. 2011. Ṛgvedic clitics and 'prosodic' movement. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '11 conference*, 360–380. Stanford: CSLI Publications.

Lowe, John J. 2016. English possessive *'s*: Clitic *and* affix. *Natural Language & Linguistic Theory* 34. 157–195. DOI: 10.1007/s11049-015-9300-1.

Lyons, John. 1968. *Introduction to theoretical linguistics*. Cambridge, UK: Cambridge University Press. DOI: 10.1017/cbo9781139165570.

Marcotte, Jean-Philippe. 2014. Syntactic categories in the correspondence architecture. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '14 conference*, 408–428. Stanford: CSLI Publications.

Melchin, Paul B., Ash Asudeh & Daniel Siddiqi. 2020. Ojibwe agreement in Lexical-Realizational Functional Grammar. In Miriam Butt & Ida Toivonen (eds.), *Proceedings of the LFG '20 conference*, 268–288. Stanford: CSLI Publications. https://csli-publications.stanford.edu/LFG/LFG-2020/lfg2020-mas.pdf.

Mohanan, K. P. 1982. Grammatical relations and clause structure in Malayalam. In Joan Bresnan (ed.), *The mental representation of grammatical relations*, 504–589. Cambridge, MA: The MIT Press.

Muskens, Reinhard. 2001. Categorial Grammar and Lexical-Functional Grammar. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '01 conference*, 259–279. Stanford: CSLI Publications.

Nordlinger, Rachel. 1998. *Constructive case: Evidence from Australian languages*. Stanford: CSLI Publications.

Oehrle, Richard T. 1999. LFG as labeled deduction. In Mary Dalrymple (ed.), *Semantics and syntax in Lexical Functional Grammar: The resource logic approach* (Language, Speech, and Communication), 319–357. Cambridge, MA: The MIT Press.

Panova, Anastasia. 2020. A case of morphologically bound complementation in Abaza: An LFG analysis. In Miriam Butt & Ida Toivonen (eds.), *Proceedings of the LFG '20 conference*, 289–306. Stanford: CSLI Publications.

Patejuk, Agnieszka. 2023. Coordination. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 309–374. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185948.

Patejuk, Agnieszka & Adam Przepiórkowski. 2014. Control into selected conjuncts. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '14 conference*, 448–460. Stanford: CSLI Publications.

Pollard, Carl & Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar*. Chicago: University of Chicago Press & CSLI Publications.

Przepiórkowski, Adam & Agnieszka Patejuk. 2012. On case assignment and the coordination of unlikes: The limits of distributive features. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '12 conference*, 479–489. Stanford: CSLI Publications.

Pullum, Geoffrey K. 2013. The central question in comparative syntactic metatheory. *Mind & Language* 28(4). 492–521. DOI: 10.1111/mila.12029.

Rákosi, György. 2023. Anaphora. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 165–192. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185940.

Sadler, Louisa & Doug Arnold. 1994. Prenominal adjectives and the phrasal/lexical distinction. *Journal of Linguistics* 30(1). 187–226. DOI: 10.1017/s0022226700016224.

Sells, Peter. 1994. Sub-phrasal syntax in Korean. *Language Research* 30(2). 351–386.

Sells, Peter. 1995. Korean and Japanese morphology from a lexical perspective. *Linguistic Inquiry* 26(2). 277–325.

Shieber, Stuart M. 1986. *An introduction to unification-based approaches to grammar* (CSLI Lecture Notes 4). Stanford: CSLI Publications.

Spencer, Andrew. 2005. Case in Hindi. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '05 conference*, 429–446. Stanford: CSLI Publications.

Toivonen, Ida. 2003. *The phrase structure of non-projecting words: A case study of Swedish particles* (Studies in Natural Language and Linguistic Theory 58). Dordrecht: Kluwer Academic Publishers.

Vikner, Sten. 1995. *Verb movement and expletive subjects in the Germanic languages*. Oxford: Oxford University Press.

Wescoat, Michael T. 2002. *On lexical sharing*. Stanford: Stanford University. (Doctoral dissertation).

Wescoat, Michael T. 2005. English nonsyllabic auxiliary contractions: An analysis in LFG with lexical sharing. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '05 conference*, 468–486. Stanford: CSLI Publications.

Wescoat, Michael T. 2007. Preposition-determiner contractions: An analysis in Optimality-Theoretic Lexical-Functional Grammar with lexical sharing. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '07 conference*, 439–459. Stanford: CSLI Publications.

Wescoat, Michael T. 2009. Udi person markers and lexical integrity. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG '09 conference*, 604–622. Stanford: CSLI Publications.

Zaenen, Annie. 1983. On syntactic binding. *Linguistic Inquiry* 14(3). 469–504.

Zaenen, Annie. 2023. Information structure. In Mary Dalrymple (ed.), *Handbook of Lexical Functional Grammar*, 823–853. Berlin: Language Science Press. DOI: 10.5281/zenodo.10185972.