# Chapter 32

# HPSG and Lexical Functional Grammar

## Stephen Wechsler

The University of Texas

## Ash Asudeh

University of Rochester

Here is the abstract: concrete gets you through abstract better than abstract gets you through concrete

## 1 Introduction

Head-Driven Phrase Structure Grammar is similar in many respects to its cousin framework, Lexical Functional Grammar or LFG (Bresnan et al. 2016; Dalrymple et al. 2019). Both HPSG and LFG are lexicalist frameworks in the sense that they distinguish between the morphological system that creates words, and the syntax proper that combines those fully inflected words into phrases and sentences. Both frameworks assume a lexical theory of argument structure (Müller & Wechsler 2014) in which verbs and other predicators come equipped with valence structures indicating the kinds of complements and other dependents that the word is to be combined with. Both theories treat control (equi) and raising as lexical properties of certain control or raising predicates. Both systems are historically related to unification grammar (Kay 1984), but replace the procedural operations of unification with mutually consistent declarative descriptions of formal objects. Both frameworks use directed graphs that are often represented in the form of recursively embedded feature structures. Phonologically empty nodes of the constituent structure are avoided in both theories, with the gaps appearing in long-distance dependencies as the sole exception in some versions of the theories, and complete elimination of empty categories even in those cases, in others.

At the same time, there are interesting differences. Each theory makes available certain representational resources that the other theory lacks. LFG has output filters in the form of *constraining equations*, HPSG does not. HPSG's feature structures are *typed*, those of LFG are not. The feature descriptions (directed graphs) are fully integrated with the phrase structure grammar in the case of HPSG, while in LFG they are intentionally separated in an autonomous level of representation in the form of a *functional structure* or f-structure. These differences lead some linguists to feel that certain types of generalization are more perspicuously stated in one framework than the other. Because LFG's functional structure is autonomous from the constituent structure whose terminal yield gives the order of words in a sentence, that functional structure can instead serve as a representation of the grammatical functions played by various components of a sentence. This makes LFG more amenable to a functionalist motivation, and also provides a standard representation language for capturing the more cross-linguistically invariant properties of syntax. Meanwhile, HPSG is more deeply rooted in phrase structure grammar, and thus provides a clearer representation of the locality conditions that are important for the proper functioning of grammars.

This chapter presents a comparison of the two theories with an emphasis on contrasts between the two. It is organized by grammatical topic.

## 2 Design principles of HPSG and LFG

The HPSG and LFG frameworks are motivated by rather different design considerations. In order to make a meaningful comparison between the frameworks, it is essential to understand those differences.

HPSG grew out of the tradition established by Chomsky (1957), of studying the computational properties of natural language syntax as a window onto the capabilities of human cognitive processes. The advent of Generalized Phrase Structure Grammar (Gazdar et al. 1985) was an important milestone in that tradition, bringing as it did the surprising prospect that the entire range of syntactic phenomena known to exist at the time could be described with a context-free grammar (CFG). If this could be maintained it would mean that natural language is context-free in the sense of the Chomsky Hierarchy (Chomsky 1957)— an answer to the question raised by Chomsky that was very different from, and more interesting than, Chomsky's own answer. Then Shieber (1985) and Culy (1985) showed that certain phenomena such as recursive cross-serial dependencies in Swiss German and Bambara exceeded the generative capacity of a context-free

grammar.

Despite that development, it was nonetheless clear that languages for the most part hewed rather closely to the context free design. Thus began the search for more powerful grammatical formalisms that would preserve the insight of a context-free grammar while allowing for certain phenomena that exceed that generative capacity.[1] HPSG grew out of this search. As Sag et al. (2003: 83) observe, HPSG "is still closely related to standard CFG." In fact, an HPSG grammar basically consists of constraints on local sub-trees (i.e., trees consisting of a node and her immediate daughters), which would make it a context-free grammar were it not that the nodes themselves are complex, recursively-defined feature structures. This CFG-like character of HPSG means that the framework itself embodies an interesting theory of locality.

The architecture of LFG is motivated by rather different concerns: an interest in typological variation, the search for linguistic universals and a desire to explain the prevailing grammatical patterns found across languages. For this reason two levels of representation are discerned, a functional structure or *f-structure* representing the internal logical relations in a sentence that are largely invariant across languages; and a categorial constituent structure or *c-structure* representing the external morphological and syntactic expression of those relations, which vary, often rather dramatically, across various typological varieties of language. For example, probably all (or nearly all) languages have subjects and objects, hence those relations are represented in f-structure. But languages vary as to the mechanisms for signaling subjecthood and objecthood, the three main mechanisms being word order, head-marking, and dependent-marking (Nichols 1986), hence those mechanisms are distinguished at c-structure. The word *functional* in the name of the LFG framework is a three-way pun, referring to the grammatical *functions* that play such an important role in the framework, the mathematical *functions* that are the basis for the representational formalism, and the generally *functionalist*-friendly nature of the LFG approach.

Evidence for these different underlying motivations can be found in the later proposals for further theory development by the principle architects of the respective theories. In some of his last work before his death, Ivan Sag worked on the development of Sign-Based Construction Grammar, a variant of HPSG in which those locality conditions are even more deeply built into the representational system (Boas & Sag 2012). Meanwhile, Joan Bresnan had developed Opti-

---

[1]There were also some problems with the GPSG theory of the lexicon, in which complement selection was assimilated to the phrase structure grammar. For discussion see Müller & Wechsler (2014).

mality Theoretic LFG (Bresnan 1997), a framework in which the OT *input* was defined as an underspecified f-structure representing the universal aspects of a syntactic structure, the *candidates set* is the set of external structures mapping to that input, and cross-linguistic variation is modeled with constraint re-rankings. So while Sag sought to strengthen the locality conditions of the framework, Bresnan sought a more explicit account of universality and variation.

In the remainder of this chapter we will survey various syntactic and semantic phenomena, showing how the the approaches of the different frameworks reflect those differing design motivations.

## 3  Phrases and Endocentricity

A phrasal node shares certain grammatical features with specific daughters. In HPSG this is accomplished by means of structure-sharing (reentrancies) in the immediate dominance schemata and other constraints on local sub-trees such as the Head Feature Principle. LFG employs essentially the same mechanism for feature sharing in a local sub-tree but implements it slightly differently, so as to better address the design motivations of LFG. Each node in a phrase structure is paired with a so-called functional structure or *f-structure*, which is formally a set of attribute-value pairs. It is through the f-structure that the nodes of the phrase structure share features. The phrase structure is referred to as *c-structure*, for categorial or constituent structure, in order to distinguish it from f-structure. The grammar, in the form of a standard rewriting system, directly generates only c-structures, not f-structures. Those c-structure rules introduce equations that form a projection function from c-structure to f-structure. For example, the phrase structure grammar in (1) and lexicon in (2) generate the tree in Figure 1.

(1)   a.  S   →   NP          VP
                  (↑ SUBJ)=↓    ↑ =↓

     b.  NP  →   $\begin{pmatrix} \text{Det} \\ \uparrow =\downarrow \end{pmatrix}$  N
                                      ↑ =↓

     c.  VP  →   V        $\begin{pmatrix} \text{NP} \\ (\uparrow \text{OBJ})=\downarrow \end{pmatrix}$
                  ↑ =↓

(2)   a.  *lion*: N        (↑ PRED) = 'lion'
                           (↑ NUM) = SG

b.  *rule*: V          ($\uparrow$ PRED) = 'rule$\langle$($\uparrow$ SUBJ)'

    *-s*: *infl*          ($\uparrow$ TENSE) = PRES

                  ($\uparrow$ SUBJ) = $\downarrow$

                         ($\downarrow$ PERS) = 3
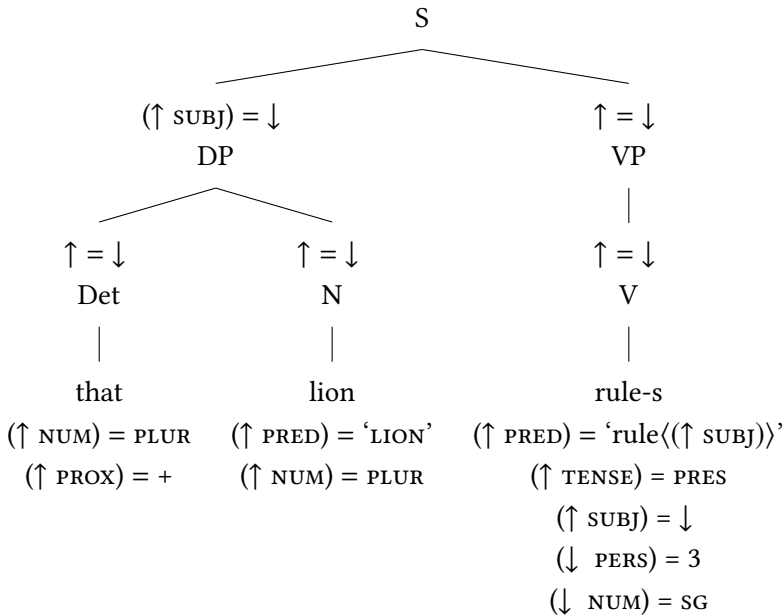
                         ($\downarrow$ NUM) = SG



Figure 1: Add caption

Each node in the c-structure maps to a function, that is, to a set of attribute-value pairs. Within the equations, the up and down arrows are metavariables over function names, interpreted as follows: the up arrow refers to the function to which the mother node maps, and the down arrow refers to the function that its own node maps to. To derive the f-structure from (1), we instantiate the metavariables to specific function names and solve for the function associated with the root node (here, S). In Figure 2, the function names *f1, f2,* etc. are subscripted to the node labels. The arrows have been replaced with those function names. Collecting all the equations from this tree and solving for *f1,* we arrive at the f-structure in (3):
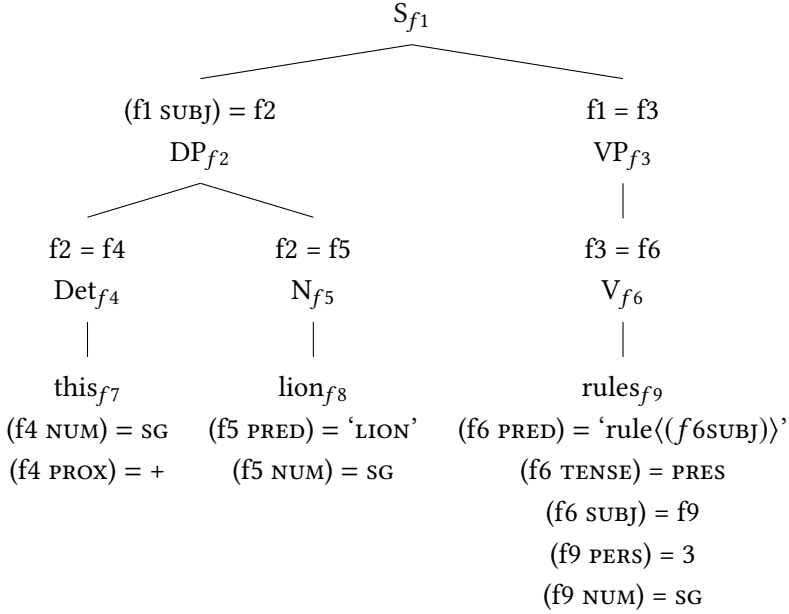
$S_{f1}$

$(f1 \text{ SUBJ}) = f2$      $f1 = f3$

$DP_{f2}$      $VP_{f3}$

$f2 = f4$    $f2 = f5$    $f3 = f6$

$Det_{f4}$      $N_{f5}$      $V_{f6}$

$this_{f7}$      $lion_{f8}$      $rules_{f9}$

$(f4 \text{ NUM}) = \text{SG}$    $(f5 \text{ PRED}) = \text{‘LION’}$    $(f6 \text{ PRED}) = \text{‘rule}\langle(f6\text{SUBJ})\rangle\text{’}$

$(f4 \text{ PROX}) = +$    $(f5 \text{ NUM}) = \text{SG}$    $(f6 \text{ TENSE}) = \text{PRES}$

$(f6 \text{ SUBJ}) = f9$

$(f9 \text{ PERS}) = 3$

$(f9 \text{ NUM}) = \text{SG}$

Figure 2: Add caption

$$
(3) \quad
\begin{bmatrix}
\text{SUBJ} &
\begin{bmatrix}
\text{PRED} & \text{‘LION’} \\
\text{NUM} & \text{SG} \\
\text{PERS} & 3 \\
\text{PROX} & +
\end{bmatrix} \\
\text{PRED} & \text{‘rule}\langle(\uparrow \text{ SUBJ})\rangle\text{’} \\
\text{TENSE} & \text{PRES}
\end{bmatrix}
$$

Since the up and down arrows refer to nodes of the local subtree, LFG annotated phrase structure rules like those in (1) can often be directly translated into HPSG immediate dominance schemata and principles constraining local subtrees. By way of illustration, let FS (for *f-structure*) be an HPSG attribute corresponding to the f-structure projection function. Then the LFG rule in (4a) (repeated from (1a above)) is equivalent to the HPSG rule in (4b):

(4)    a.   S    →      DP      VP

                    $(\uparrow \text{ SUBJ})=\downarrow$    $\uparrow =\downarrow$

   b.   S[FS ①]→    DP[FS ②]    VP[FS ①[SUBJ ②]]

Let us compare the two representations with respect to heads and dependents.

Taking heads first, the VP node annotated with ↑ = ↓ is an *f-structure head*, meaning that the features of the VP are identified with those of the mother S. This effect is equivalent to the tag ① in (4b). Hence ↑ = ↓ has an effect similar to HPSG's Head Feature Principle. However, in LFG the part of speech categories and their projections such as N, V, Det, NP, VP, DP, etc. belong to the c-structure and not the f-structure. As a consequence those features are not subject to sharing, and any principled correlations between such categories, such as the fact that N is the head of NP, V the head of VP, C the head of CP, and so on, are instead captured in an explicit version of (extended) X-bar theory applying to the c-structure (Grimshaw 1998). The LFG based theory of endocentricity is considerably weaker (more permissive) than what is typically found in most transformation based grammars. The version of extended X-bar theory in Bresnan et al. (2016: Chapter 6) assumes that all nodes on the right side of the arrow of the phrase structure rule are optional, with many unacceptable partial structures ruled out in the f-structure instead. Also not all structures need to be endocentric (i.e. not all structures have a head daughter in c-structure). The LFG category S shown in (4a) is inherently exocentric, lacking a c-structure head, and is used for the analysis of copulaless clauses. (English is also assumed to have endocentric clauses of category IP, where an auxiliary verb of category I (for Inflection) serves as the c-structure head.) S is also used for flat structures in non-configurational clauses found in languages such as Warlpiri.

Functional projections like DP, IP, and CP are typically assumed to form a 'shell' over the lexical projections NP, VP, AP, and PP (plus CP can appear over S). In fact this idea of extending X-bar to functional categories has its origin in the LFG work of the late Yehuda Falk (Falk 1984), from which it then spread to transformational theories. This is formally implemented by having the functional head (such as Det) and its lexical complement (such as NP) be f-structure co-heads. See for example the DP *that lion* in (1), where Det and N are both annotated with ↑ = ↓ . The DP, Det, and N nodes all map to the same f-structure, namely the subsidiary structure serving as the value of SUBJ in (3). What makes this unification possible is that function words lack a PRED (for 'predicate') feature that would otherwise indicate a semantic form. Content words such as *lion* have such a feature ([PRED 'LION']), and so if the Det had one as well then they would clash in the f-structure. Note more generally that the f-structure flattens out much of the hierarchical structure of the corresponding c-structure.

Complementation works a little differently in LFG from HPSG. Note that the LFG rule (4a) indicates the SUBJ grammatical function on the subject DP node, while the pseudo-HPSG rule (4b) indicates the SUBJ function on the VP functor

selecting the subject. A consequence of the use of functional equations in LFG is that a grammatical relation such as SUBJ can be locally associated with its formal exponents, whether a configurational position in phrase structure (as in (1)), head-marking (agreement, see (2b)), or dependent marking (case). A subject-marking case affix such as a nominative inflection can introduce a so-called 'inside out' functional designator, (SUBJ ↑ ), which requires that the f-structure of the DP bearing that case ending be the value of a SUBJ attribute (Nordlinger 1998).[2] This aspect of LFG representations makes it convenient for functionalist and typological work on grammatical relations.

## 4 Valence

In LFG a lexical predicator such as a verb selects its complements via f-structure rather than c-structure. A transitive verb selects a SUBJ and OBJ, which are features of f-structure, but it cannot select for the category 'DP' because such part of speech categories belong only to c-structure. For example the verb stem rule in (2b) has a PRED feature whose value contains (↑ SUBJ), which has the effect of requiring a SUBJ function in the f-structure. The f-structure (shown in (3)) is built using the defining equations, as described above. Then that f-structure is checked against any *existential constraints* such as the expression (↑ SUBJ), which requires that the f-structure contain a SUBJ feature. That constraint is satisfied, as shown in (3). Moreover, the fact that (↑ SUBJ) appears in the angled brackets means that it expresses a semantic role of the 'rule' relation, hence the SUBJ value is required to contain a PRED feature, which is satisfied by the feature [PRED 'LION'] in (3).

Selection for grammatical relations instead of formal categories enables LFG to capture the flexibility in the expression of a given grammatical relation described at the end of the previous section. As noted there, in many languages the subject can be expressed either as an independent DP phrase as in English, or as a pronominal affix on the verb. As long as the affix introduces a PRED feature and is designated by the grammar as filling the SUBJ relation, then it satisfies the subcategorization requirements imposed by a verb. A more subtle example of flexible expression of grammatical functions can be seen in English constructions where an argument can in principle take the form of either a DP (as in (5a)) or a clause (as in (5b)) (example (5) is from Bresnan et al. (2016) PAGE).

(5)    a.   That problem, we talked about for days.

---

[2]For function *f*, attribute *a* and value *v*, $(f\,a) = v$ iff $(a\,v) = f$.

b. That he was sick, we talked about for days.

c. We talked about that problem for days.

d. *We talked about that he was sick for days.

The variant of *talk* taking an *about*-PP selects neither a DP nor a clausal complement, but rather an object (OBJ) of an oblique (OBL$_{about}$) function:

(6)  *talk*: V     (↑ PRED) = 'talk-about⟨(↑ SUBJ)(↑ OBL$_{about}$OBJ)⟩'

It is not the verb but the local c-structure environment that conditions the category of that argument: the canonical object position right-adjacent to *about* can only house a DP, while the topic position allows either DP or clause (as seen by comparing (5c) and (5d)). In LFG the grammatical functions such as SUBJ and OBJ represent equivalence classes across various modes of c-structure expression.

HPSG captures this variability in the expression of arguments in essentially the same way as LFG, despite some terminological differences. The HPSG correspondent of the LFG SUBJ specification is not an HPSG SUBJ valence list item, but rather the item of the ARG-ST list that the (optional) SUBJ item maps to, when it appears. The same applies to complements. The disjunction between DP and clausal expression of an argument is encoded in the ARG-ST; the restriction to DP (observed in examples (5c,d)) is encoded in the relevant VALENCE list item.

## 5 Head mobility

The lexical head of a phrase can sometimes appear in an alternative position apparently outside of what would normally be its phrasal projection. Assuming that an English finite auxiliary verb is the (category I) head of its (IP) clause, then that auxiliary appears outside its clause in a yes/no question:

(7)  a. [$_{IP}$ she is mad].

    b. Is [$_{IP}$ she ___ mad]?

Transformational grammars capture the systematic relation between these two structures with a head-movement transformation that leaves the source IP structure intact, with a trace replacing the moved lexical head of the clause. The landing site of the moved clausal head is often assumed to be C, the complementizer position, as motivated by complementarity between the fronted verb and a lexical complementizer. This complementarity is observed most strikingly in German verb-second versus verb-final alternations, but is also found in other languages, including some English constructions such as the following:

(8)  a. I wonder whether [$_{\text{IP}}$ she is mad].

  b. I wonder, is [$_{\text{IP}}$ she ___ mad]?

  c. *I wonder whether is she mad.

In HPSG the sentences in (8) are treated as displaying two distinct structures generated by the grammar. For example, assuming ternary branching in (7b) then the subject DP *she* and predicate AP *mad* would normally be assumed to be sisters of the fronted auxiliary *is*. On that analysis the structure is flattened out so that *she mad* is not a constituent. In fact for English the fronting of *is* can even be seen as a consequence of that flattening: English is a head-initial language so the two dependents *she* and *mad* are expected to follow their selecting head *is*.

Although LFG is non-transformational, it can express the intuition behind the I-to-C movement analysis due to the separation of autonomous c- and f-structures. Recall from the above discussion of the DP in Figure 1 that functional heads such as determiners, auxiliaries and complementizers do not introduce new f-structures, but rather map to the same f-structure as their complement phrases. The finite auxiliary can therefore appear in either I or C without affecting the f-structure, as we will see presently. Recall also that c-structure nodes are optional and can be omitted as long as a well-formed f-structure is generated. Comparing the non-terminal structures of Figure 3 and Figure 4, the I node is omitted from the latter structure but otherwise they are identical.

(Most of the lexical equations are omitted from Figure 4 for clarity.) Given the many ↑ = ↓ annotations, the C, I, and AP nodes (as well as IP and CP) all map to the same f-structure, namely the one shown in (9).

$$(9) \quad \begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{`PRO'} \\ \text{NUM} & \text{SG} \\ \text{PERS} & 3 \\ \text{GEN} & \text{FEM} \end{bmatrix} \\ \text{PRED} & \text{`mad}\langle(\uparrow \text{SUBJ})\rangle\text{'} \\ \text{FIN} & + \end{bmatrix}$$

The C and I positions are appropriate for markers of clausal grammatical features such as finiteness ([FIN ±]), encoded either by auxiliary verbs like finite *is* or complementizers like finite *that* and infinitival *for*: *I said that/*for she is present* vs. *I asked for/*that her to be present*. English has a specialized class of auxiliary verbs for marking finiteness from the C position, while in languages like German all finite verbs, including main verbs, can appear in a C position that is unoccupied by a lexical complementizer. Summarizing, the LFG framework enables a theory
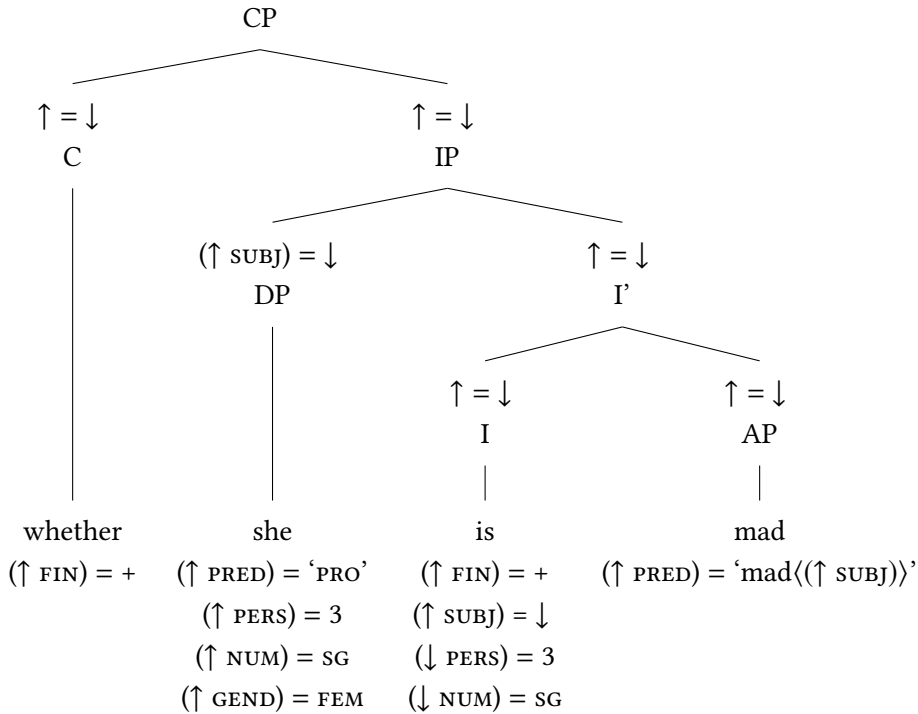
Figure 3: Add caption

of head mobility based on the intuition that a clause has multiple head positions where inflectional features of the clause are encoded.

## 6 Case, agreement, and constraining equations

The basic theory of agreement is the same in LFG and HPSG (see Wechsler 2020, Chapter 6 of this volume): agreement occurs when multiple feature sets arising from distinct elements of a sentence specify information about a single abstract object, so that the information must be mutually consistent (Kay 1984). The two forms are said to agree when the values imposed by the two constraints are compatible, while ungrammaticality results when they are incompatible. An LFG example is seen in (1) above, where the noun, determiner and verbal suffix each specify person and/or number features of the same SUBJ value.

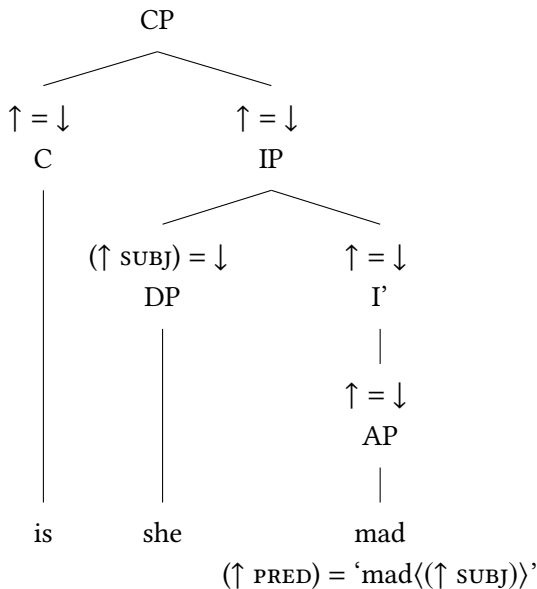The basic mechanism for case marking works in essentially the same way as

Figure 4: Add caption

agreement, in both frameworks: in case marking, distinct elements of a sentence specify case information about a single abstract object, hence that information must be compatible. To account for the contrast in (10a), nominative CASE equations are associated with the pronoun *she* and added to the entry for the verbal agreement suffix *-s*:

(10)  a.  She/*Her/*You rules.

    b.  *she*: Pron    ($\uparrow$ CASE) = NOM
                             ($\uparrow$ PERS) = 3
                             ($\uparrow$ NUM) = SG
                             ($\uparrow$ GEND) = FEM

    c.  *her*: Pron    ($\uparrow$ CASE) = ACC
                             ($\uparrow$ PERS) = 3
                             ($\uparrow$ NUM) = SG
                             ($\uparrow$ GEND) = FEM

d.  *-s: infl*　　　($\uparrow$ TENSE) = PRES

($\uparrow$ SUBJ) = $\downarrow$

($\downarrow$ PERS) = 3

($\downarrow$ NUM) = SG

($\downarrow$ CASE) = NOM

The variant of (10a) with *her* as subject is ruled out due to a clash of CASE features within the value of SUBJ in the f-structure. The variant with *you* as subject is ruled out due to a clash of PERS features. This mechanism is essentially the same as in HPSG, where it operates via the VALENCE feature.

This account allows for underspecification of both the case assigner and the case bearing element, and of both the controller and target of agreement. In English, for example, gender is marked on some pronouns but not on a verbal affix; and (nominative) case is marked on the verbal affix but not on nominals, with the exception of the pronouns. But certain case and agreement phenomena do not tolerate underspecification, and for those phenomena LFG offers an account using a *constraining equation*, a mechanism absent from HPSG and indeed ruled out by current principles of HPSG theory. (Some early precursors to HPSG included a special feature value called ANY that functioned much like an LFG constraining equation (Shieber 1986: 36-7), but that device has been eliminated from HPSG.) The functional equations described so far in this chapter function by building the f-structure, as illustrated in (1) and (3) above; such equations are called *defining equations*. A constraining equation has the same syntax as a defining equation, but it functions by checking the completed f-structure for the presence of a feature. An f-structure lacking the feature designated by the constraining equation is ill-formed.

The following lexical entry for *she* is identical to the one in (10b) above, except that the CASE equation has been replaced with a constraining equation, notated with $=_c$.

(11)　*she: Pron*　　　($\uparrow$ CASE) $=_c$ NOM

($\uparrow$ PERS) = 3

($\uparrow$ NUM) = SG

($\uparrow$ GEND) = FEM

The f-structure is built from the defining equations, after which the SUBJ field is checked for the presence of the [CASE NOM] feature, as indicated by the constraining equation. If this feature has been contributed by the finite verb, as in

(10), then the sentence is predicted to be grammatical; if there is no finite verb (and there is no other source of nominative case) then it is ruled out. This predicts the following grammaticality pattern:

(12)   Who won the popular vote in the 2016 election?

    a.   She did! / *Her did!

    b.   * She! / Her!

English nominative pronouns require the presence of a finite verb, here the finite auxiliary *did*. Constraining equations operate as output filters on f-structures and are the primary way to grammatically specify the obligatoriness of a form, especially under the assumption that all daughter nodes are optional in the phrase structure. As described in Section 4 above, obligatory dependents are specified in the lexical form of a predicator using existential constraints like (↑ SUBJ) or (↑ OBJ). These are equivalent to constraining equations in which the particular value is unspecified, but some value must appear in order for the f-structure to be well-formed.

    A constraining equation for case introduced by the case-*assigner* rather than the case-bearing element, predicts that the appropriate case-bearing element must appear. A striking example from Serbo-Croatian is described by Wechsler & Zlatić (2003: 134), who give this descriptive generalization:

(13)   Serbo-Croatian Dative/Instrumental Case Realization Condition.

    If a verb or noun assigns dative or instrumental case to an NP, then that case must be morphologically realized by some element within the NP.

In Serbo-Croatian most common nouns, proper nouns, adjectives, and determiners are inflected for case. An NP in a dative position must contain at least one such item morphologically inflected for dative case, and similarly for instrumental case. The verb *pokloniti* 'give' governs a dative object, such as *ovom studentu* in (14a). But a quantified NP like *ovih pet studenata* 'these five students' has invariant case, namely genitive on the determiner and noun, and an undeclinable numeral *pet* 'five'. Such a quantified NP can appear in any case position— except when it fails to satisfy the condition in (13), such as this dative position (Wechsler & Zlatić 2003: 125):

(14)   a.   pokloniti knjige    ovom    studentu
          give.INF  books.ACC this.DAT.SG student.DAT.SG
          'to give books to this student'

b. * pokloniti knjige    [ovih      pet studenata]
      give.INF  books.ACC this.GEN.PL five student.GEN.PL
      ('to give books to these five students')

Similarly, certain foreign names such as *Miki* and loanwords such as *braon* 'brown, brunette' are undeclinable, and can appear in any case position, except those ruled out by (13). Thus the dative example in (15a) is unacceptable unless the inflected possessive adjective *mojoj/mojom* 'my' appears. When the possessive adjective realizes the case feature, it is acceptable. In (15b) we contrast the undeclined loan word *braon* 'brown' with the inflected form *lepoj* 'beautiful'. The example is acceptable only with the inflected adjective (Wechsler & Zlatić 2003: 134).

(15)  a. Divim    se   *(mojoj)   Miki.
         admire.1SG REFL my.DAT.SG Miki
         'I admire (my) Miki.'

      b. Divim    se   *braon/ lepoj          Miki.
         admire.1sg REFL brown/ beautiful.DAT.SG Miki
         ('I admire brunette/ beautiful Miki')

This complex distribution is captured simply by positing that the dative (and instrumental) case assigning equations on verbs and nouns, such as the verbs *pokloniti* and *divim* in the above examples, are constraining equations:

(16)  $(\uparrow \text{OBL}_{dat} \text{CASE}) =_c \text{DAT}$

Any item in dative form within the NP, such as *ovom* or *studentu* in (14)a or *mojoj* or *lepoj* in (15), could introduce the [CASE DAT] feature that satisfies this equation, but if none appears then the sentence fails. In contrast, other case-assigning equations (e.g. for nominative, accusative, or genitive case, or for cases assigned by prepositions) are defining equations, which therefore allow the undeclined NPs to appear. This sort of phenomenon is easy to capture using an output filter such as a constraining equation, but rather difficult otherwise. See Wechsler & Zlatić (2001) for further examples and discussion.

# 7 Agreement and affixal pronouns

Agreement inflections that include the person feature derive historically from incorporated pronominal affixes. Distinguishing between agreement markers and

affixal pronouns can be a subtle and controversial matter. LFG provides a particular formal device for representing this distinction within the f-structure: a true pronoun, whether affixal or free, introduces a semantic form (formally a PRED feature) with the value 'PRO', while an agreement inflection does not. For example, [Bresnan & Mchombo (1987)](#) argue that the Chicheŵa (Bantu) object marker (OM) is an incorporated pronoun, while the subject marker (SM) alternates between agreement and incorporated pronoun, as in this example:

(17)  Njûchi zi-ná-wá-lum-a     a-lenje.
      10.bee  10.SM-PST-2.OM-bite-FV 2-hunter
      'The bees bit them, the hunters.'

According to [Bresnan & Mchombo (1987)](#) the class 2 object marker *wá-* is a pronoun, so the phrase *alenje* 'the hunters' is not the true object but rather a postposed topic cataphorically linked to the object marker, with which it agrees in noun class (a case of *anaphoric agreement*). Meanwhile, the class 10 subject marker *zi-* alternates: when an associated subject NP (*njûchi* 'bees' in (2)) appears then it is a grammatical agreement marker, but when no subject NP appears then it functions as a pronoun. This is captured in LFG with the simplified lexical entries in ([18](#)):

(18)   a.  *lum*: V       ($\uparrow$ PRED) = 'bite$\langle(\uparrow$ SUBJ$)(\uparrow$ OBJ$)\rangle$'

       b.  *wá-*: Aff     ($\uparrow$ OBJ GEND) = 2
                          ($\uparrow$ OBJ PRED) = 'PRO'

       c.  *zi-*: Aff     ($\uparrow$ SUBJ GEND) = 10
                          (($\uparrow$ SUBJ PRED) = 'PRO')

The PRED feature in ([18](#)b) is obligatory while that of ([18](#)c) is optional, as indicated by the parentheses around the latter. These entries interact with the grammar in the following manner. The two grammatical functions governed by the verb in ([18](#)a) are SUBJ and OBJ (the *governed* functions are the ones designated in the predicate argument structure of a predicator). According to the *Principle of Completeness*, a PRED feature must appear in the f-structure for each governed grammatical function that appears within the angle brackets of a predicator (indicating assignment of a semantic role). By the uniqueness condition it follows that there must be *exactly one* PRED feature, since a second such feature would cause a clash of values.[3]

---

[3]Each PRED value is assumed to be unique, so that two 'PRO' values cannot unify.

The OM *wá-* introduces the [PRED 'PRO'] into the object field of the f-structure of this sentence; the word *alenje* 'hunters' introduces its own PRED feature with value 'HUNTERS', so it cannot be the true object, and instead is assumed to be in a topic position. Bresnan & Mchombo (1987) note that the OM can be omitted from the sentence, in which case the phrasal object (here, *alenje*) is fixed in the immediately post-verbal position, while that phrase can alternatively be preposed when the OM appears. This is explained by assuming that the post-verbal position is an OBJ position, while the adjoined TOPIC position is more flexible.

The subject *njûchi* can be omitted from (17), yielding a grammatical sentence meaning 'They (some plural entity named with a class 10 noun) bit them, the hunters.' The optional PRED feature equation in (18b) captures this pro-drop property: when the equation appears then a phrase such as *njûchi* cannot appear in the subject position, since this would lead to a clash of PRED values ('PRO' versus 'BEES'); but when the equation is not selected then *njûchi* must appear in the subject position in order for the f-structure to be complete.

The diachronic process in which a pronominal affix is reanalyzed as agreement has been modeled in LFG as the historic loss of the PRED feature, along with the retention of the pronoun's person, number, and gender features (Coppock & Wechsler 2010). The anaphoric agreement of the older pronoun with its antecedent then becomes reanalyzed as grammatical agreement of the inflected verb with an external nominal. Finer transition states can also be modeled in terms of selective feature loss. Clitic doubling can be modeled as optional loss of the PRED feature with retention of some semantic vestiges of the pronominal, such as specificity of reference.

The LFG analysis of affixal pronouns and agreement inflections can be approximated in HPSG by associating the former but not the latter with pronominal semantics in the CONTENT field. However, because HPSG lacks output filters like the LFG Principle of Completeness, it is more difficult for the HPSG framework to directly capture the complementarity between the appearance of a pronominal affix and an analytic phrase filling the same grammatical relation.

# 8 Lexical mapping

LFG and HPSG both adopt *lexical approaches to argument structure* in the sense of Müller & Wechsler (2014): a verb or other predicator is equipped with a valence structure indicating the grammatical expression of its semantic arguments as syntactic dependents. Both frameworks have complex systems for mapping semantic arguments to syntactic dependents that are designed to capture pre-

vailing semantic regularities within a language and across languages. The respective systems differ greatly in their notation and formal properties but it is unclear whether there are any theoretically interesting differences, such as types of analysis that are available in one but not the other. This section identifies some of the most important analogues across the two systems, namely LFG's Lexical Mapping Theory (Bresnan et al. 2016: Chapter 14) and the theory of macro-roles proposed by Davis and Koenig for HPSG (see Wechsler, Koenig & Davis 2020, Chapter 9 of this volume).[4]

In Lexical Mapping Theory, the argument structure is a list of a verb's argument slots, each labeled with a thematic role type such as Agent, Instrument, Recipient, Patient, Location, and so on, in the tradition of Charles Fillmore's *Deep Cases* (Fillmore 1968; 1977) and Pāṇini's *kārakas* (Kiparsky & Staal 1969). The ordering is determined by a thematic hierarchy that reflects priority for subject selection.[5] The thematic role type influences a further classification by the features $[\pm o]$ and $[\pm r]$ that conditions the mapping to syntactic functions (this version is from Bresnan et al. (2016: p. 331)):

(19)   **Semantic classification of argument structure roles for function**:

patientlike roles:               $\theta$
                                 $[-r]$

secondary patientlike roles:     $\theta$
                                 $[+o]$

other semantic roles:            $\theta$
                                 $[-o]$

The features $[\pm r]$ (thematically *restricted*) and $[\pm o]$ (*objective*) cross-classify grammatical functions: subject is $[-r, -o]$, object is $[-r, +o]$, obliques are $[+r, -o]$ and restricted objects are $[+r, +o]$. A monotonic derivation (where feature values cannot be changed) starts from the argument list with the Intrinsic Classification (*I.C.* in example (20) below), then morpholexical operations such as passivization can suppress a role (not shown), then the thematically highest role (such

---

[4]A recent alternative to LMT based on Glue Semantics has been developed by Asudeh & Giorgolo (2012) and Asudeh et al. (2014). It preserves the monotonicity of LMT, discussed below, but uses Glue Semantics to model argument realization and valence alternations.

[5]The particular ordering proposed in Bresnan & Kanerva (1989) and Bresnan et al. (2016) is the following: *agent > beneficiary > experiencer/goal > instrument > patient/theme > locative*

*Draft of 31st March 2020, 22:50*

as the Agent), if [−o], is selected as Subject (*Sbj.* in example (20)) and then any remaining features receive positive values by default (*Def.* in example (20)).

(20)   Derivation of *eat* as in *Pam ate a yam.*

|  | | | | |
|---|---|---|---|---|
| a-structure: | *eat* | < | *ag* | *th* | > |
| | I.C. | | [−o] | [−r] | |
| | Sbj. | | [−r] | | |
| | Def. | | | [+o] | |
| | | | │ | │ | |
| f-structure: | | | SUBJ | OBJ | |

In the macro-role theory formulated for HPSG, the analogues of [−o] and [−r] are the macro-roles 'Actor' (ACT) and 'Undergoer' (UND), respectively. The names of these features reflect large general groupings of semantic role types, but there is not a unique semantic entailment such as 'agency' or 'affectedness' associated with each of them. 'Actor' (ACT) and 'Undergoer' (UND), name whatever semantic roles map to the subject and object, respectively, of a transitive verb.[6] On the semantic side they are disjunctively defined: *x* is the Actor and *y* is the Undergoer iff '*x* causes a change in *y*, or *x* has a notion of *y*, or …' (quoted from Wechsler, Koenig & Davis 2020, Chapter 9 of this volume, example (15)). Such disjunctive definitions are the HPSG analogues of the Lexical Mapping Theory 'semantic classifications' shown in (19) above. In the HPSG macro-role system, linking constraints dictate that the ACT argument maps to the first element of ARG-ST, and that the UND argument maps to some nominal element of ARG-ST (21) and (22) are from Wechsler, Koenig & Davis 2020, Chapter 9 of this volume, examples (16) and (17)):

(21)
$$\begin{bmatrix} \text{CONTENT|KEY} & \begin{bmatrix} \text{ACT} & \boxed{1} \end{bmatrix} \\ \text{ARG-ST} & \left\langle \text{NP}_{\boxed{1}}, \dots \right\rangle \end{bmatrix}$$

(22)
$$\begin{bmatrix} \text{CONTENT|KEY} & \begin{bmatrix} \text{UND} & \boxed{2} \end{bmatrix} \\ \text{ARG-ST} & \left\langle \dots, \text{NP}_{\boxed{2}}, \dots \right\rangle \end{bmatrix}$$

The first element of ARG-ST maps to the subject of an active voice verb, so (21)–(22) imply that the subject is the ACT if there is one, and otherwise it is the UND.

---

[6]Note, for example, that within this system the 'Undergoer' argument of the English verb *undergo*, as in *John underwent an operation*, is the object— and not the subject, as one might expect if being an 'Undergoer' involved actually undergoing something.

Similarly, in Lexical Mapping Theory as described above, the subject is the [−*o*] highest argument, if there is one, and otherwise it is the [−*r*] argument. In this simple example we can see how the two systems accomplish exactly the same thing. A careful examination of more complex examples might point up theoretical differences, but it seems more likely that the two systems can express virtually the same set of mappings.

In LFG the *argument structure* (or *a-structure*) contains the predicator and its argument roles classified and ordered by thematic role type and further classified by Intrinsic Classification. It is considered a distinct level of representation, along with c-structure and f-structure. As a consequence the grammar can make reference to the initial item in a-structure, such as the agent (*ag*) in (20), which is considered the 'most prominent' role and often called the *a-subject* ('argument structure subject') in LFG parlance. To derive the passive voice mapping, the a-subject is suppressed in a morpholexical operation that crucially takes place before the subject is selected:

(23)   Derivation of *eaten* as in *A yam was eaten (by Pam)*.

| a-structure: | *eat* | < | *ag* | *th* | > |
|---|---|---|---|---|---|
| | I.C. | | [−*o*] | [−*r*] | |
| | passive | | Ø | | |
| | Sbj. | | | [−*o*] | |
| | | | | | | |

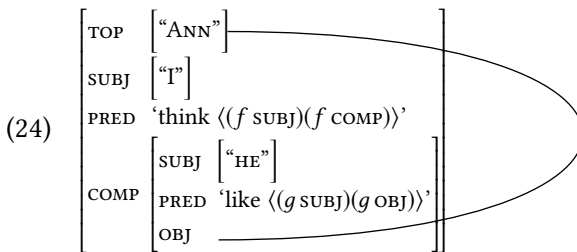f-structure:                                    SUBJ

(The optional *by*-phrase is considered to be an adjunct referring to the passivized a-subject.) Note that the passive alternation is *not* captured by a procedural rule that replaces one grammatical relation (such as OBJ) with another (such as SUBJ). The mapping from word strings to f-structures in LFG is monotonic, in the sense that information cannot be destroyed or changed. As a result the mapping between internal and external structures is said to be transparent in the sense that the grammatical relations of parts of the sentence are preserved in the whole (for discussion of this point, see Bresnan et al. (2016: Chapter 5)). In early versions of LFG, monotonicity was assumed for the syntax proper, while destructive procedures were permitted in the lexicon. This was canonized in the *Principle of Direct Syntactic Encoding*, according to which all grammatical relation changes are lexical (Bresnan 1982). At that time an LFG passive rule operated on fully specified predicate argument structures, replacing OBJ with SUBJ, and SUBJ with an $OBL_{by}$ or an existentially bound variable. The advent of LMT brought monotonicity to the lexicon as well.

# 9 Long distance dependencies

In LFG a long distance dependency is modeled as a reentrancy in the f-structure, a structure lacking from HPSG. The HPSG theory of long distance dependencies is based on that of GPSG and uses the percolation of a 'slash' feature through the constituent structure. But the two frameworks are essentially very similar, both working by decomposing a long distance dependency into a series of local dependencies. As we will see, there are nevertheless some minor differences with respect to what hypothetical extraction patterns can be expressed.

Both frameworks allow either a 'gap' or 'gapless' account: regarding LFG see Bresnan et al. (2016) for gap and Dalrymple et al. (2019) for gapless; regarding HPSG see Pollard & Sag (1994) for gap and Sag et al. (2003) for gapless. Gaps have been motivated by the (controversial) claim that the linear position of an 'empty category' matters for the purpose of weak crossover and other binding phenomena (Bresnan et al. 2016: Chapter 9). In this section I compare gapless accounts.

LFG has two grammaticalized discourse functions, TOP ('topic') and FOC ('focus'). A sentence with a left-adjoined topic position is depicted in Figure 5. The topic phrase *Ann* serves as the object of the verb *like* within the clausal complement of *think*. This dependency is encoded in the second equation annotating the topic node, where the variable $x$ ranges over strings of attributes representing grammatical functions such as SUBJ, OBJ, OBL, or COMP. These strings describe paths through the f-structure. In this example $x$ is resolved to the string COMP OBJ, so this equation has the effect of adding to the f-structure in (24) the curved line representing token identity.

(24)
$$
\begin{bmatrix}
\text{TOP} & \begin{bmatrix} \text{"ANN"} \end{bmatrix} \\
\text{SUBJ} & \begin{bmatrix} \text{"I"} \end{bmatrix} \\
\text{PRED} & \text{'think } \langle (f \text{ SUBJ})(f \text{ COMP}) \rangle \text{'} \\
\text{COMP} & \begin{bmatrix}
\text{SUBJ} & \begin{bmatrix} \text{"HE"} \end{bmatrix} \\
\text{PRED} & \text{'like } \langle (g \text{ SUBJ})(g \text{ OBJ}) \rangle \text{'} \\
\text{OBJ} & 
\end{bmatrix}
\end{bmatrix}
$$

HPSG accounts are broadly similar. One HPSG version relaxes the requirement that the arguments specified in the lexical entry of a verb or other predicator must all appear in its VALENCE lists. Verbal arguments are represented by elements of the ARG-ST list, so the list for the verb *like* contains two DPs, one each for the subject and object. In a sentence with no extraction, those ARG-ST list items map
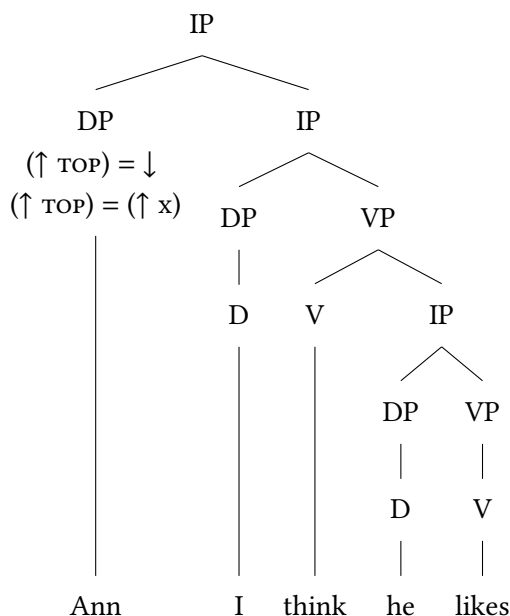
Figure 5: Add cpation

to the SMALLCAPS VALENCE lists, the first item appearing in SUBJ and any remaining ones in COMPS. To allow for extraction, one of those ARG-ST list items is permitted to appear on the SLASH list instead. The SLASH list item is then passed up the tree by means of strictly local constraints, until it is bound by the topicalized phrase.

The LFG dependency is expressed in the f-structure, not the c-structure. Bresnan et al. (2016: Chapter 2) note that this allows for category mismatches between the phrases serving as filler and those in the canonical, unextracted position. This is illustrated in example (5) above. The lexical entry for *talk (about)* in (6) selects an OBL$_{about}$ OBJ but does not specify the part of speech category of that argument; meanwhile, the phrase structure rules dictate that the top position allows (at least) DP and CP, while the position right adjacent to *about* can only house a DP. Category mismatches pose a problem for transformational theories that assume the Projection Principle, since the 'moved' constituent should satisfy the conditions imposed on the phrase in its source position. But HPSG seems to permit essentially the same analysis as the LFG analysis just sketched. In HPSG, the preposition *about* would have a disjunctive DP/CP on its ARG-ST list item, but only DP is selected for its COMPS list item; and the topic position would allow

either DP or CP.

Constraints on extraction such as accessibility conditions and island constraints ▮ can be captured in LFG by placing constraints on the attribute string $x$ (Dalrymple et al. 2019). If subjects are not accessible to extraction then we stipulate that SUBJ cannot be the final attribute in $x$; if subjects are islands then we stipulate that SUBJ cannot be a non-final attribute in $x$. If the f-structure is the only place such constraints are stated then this makes the interesting (but unfortunately false; see presently) prediction that the theory of extraction cannot distinguish between constituents that map to the same f-structure. For example, as noted in Section 5 function words like determiners and their contentful phrases like NP are usually assumed to be f-structure co-heads, so the DP *the lion* maps to the same f-structure as its NP daughter *lion* (see diagram Figure 1. This predicts that if the DP can be extracted then so can the NP, but of course that is not true:

(25) a. The lion, I think she saw.
   b. * Lion, I think she saw the.

These two sentences have exactly the same f-structures, so any explanation for the contrast in acceptability must involve some other level. For example, one could posit that the phrase structure rules can introduce obligatory daughters (see Snijders (2015: 239)), contra the assumptions of Bresnan et al. (2016).
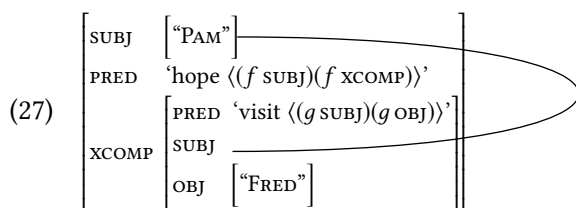
## 10 Control and raising

Raising and control (equi) words are treated in virtually the same way in LFG and HPSG: a subject control word (such as *hope* in (26)) specifies that its subject is (also) the subject of its predicate complement.

(26) Pam hopes to visit Fred.
  a. *hope*:  ($\uparrow$ PRED) = 'hope$\langle(\uparrow$ SUBJ)($\uparrow$ XCOMP)$\rangle$'
       ($\uparrow$ SUBJ) = ($\uparrow$ XCOMP SUBJ)
  b. *hope*:  [ ARG-ST $\langle$ ①NP, VP[*inf*; SUBJ $\langle$ ① $\rangle$]$\rangle$]

The LFG entry for *hope* in (26a) contains the grammatical function XCOMP ('open complement'), the function reserved for predicate complements such as the infinitival phrase *to visit Fred*. The control equation specifies that its subject is equivalent to the subject of the verb *hope*; the tag ① plays the same role in the simplified HPSG entry in (26b).

The f-structure for (26) is shown here:

(27)

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{"PAM"} \end{bmatrix} \\ \text{PRED} & \text{'hope } \langle (f\text{ SUBJ})(f\text{ XCOMP}) \rangle \text{'} \\ \text{XCOMP} & \begin{bmatrix} \text{PRED} & \text{'visit } \langle (g\text{ SUBJ})(g\text{ OBJ}) \rangle \text{'} \\ \text{SUBJ} & \underline{\quad\quad} \\ \text{OBJ} & \begin{bmatrix} \text{"FRED"} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

One interesting difference between the two frameworks is that the HPSG representation allows only for control (or raising) of subjects and not complements. More precisely, it allows for control of the outermost or final dependent to be combined with the verbal projection. This is because of the list cancellation regime that operates with valence lists. The expression 'VP' in (26b) represents an item with an empty COMPS list. In a simple English clause the verb combines with its complement phrases to form a VP constituent, which which the subject is then combined to form a clause. Assuming the same order of combination in the control structure, it is not possible to control the complement of a structure that contains a structural subject, as in (28a):

(28)   a.   * Fred hopes Pam to visit.

    b.   Fred hopes to be visited by Pam.

    c.   ?! ($\uparrow$ SUBJ) = ($\uparrow$ XCOMP OBJ)

The intended meaning would be that of (28b). The passive voice is needed in order to make the intended controllee (*Fred*) the subject of *visit* and thus available to be controlled. This restriction of controllees to subjects follows from the HPSG theory, while in LFG it follows only if control equations like the one in (28c) are systematically excluded.

This raises the question of whether the restriction to subject controllees is universal. In fact it appears that some languages allow the control of non-subjects, but it is still unclear whether such languages justify equations such as (28c). For example, Kroeger (1993) shows that Tagalog has two types of control relation. In the more specialized type, which occurs only with a small set of verbs or in a special construction in which the downstairs verb appears in non-volitive mood, both the controller and controllee must be subjects. Kroeger analyzes this type as functional control using a control equation like the one in (26a). In the more common type of Tagalog control, the controllee must be the Actor argument, while the grammatical relations of controllee and controller are not restricted. (Tagalog has a rich voice system, often called a focus marking system, regulating which argument of a verb is selected as its subject.) This latter type of Tagalog

control is defined on argument structure (Actors, etc.), so a-structure rather than f-structure is appropriate for representing the control relations.

Instead of functional control, Kroeger analyzes this latter type of control within LFG as *anaphoric control*, in which the controllee has the status of a null pronoun whose antecedent is the controller. In anaphoric control the f-structures of the controller and controllee are not identified, and instead each has its own PRED feature. An anaphoric control analysis of (26) would involve a structure resembling (27) except that the curved line would be eliminated and the SUBJ of *visit* would contain the feature [PRED 'PRO']. In the HPSG analogue of anaphoric control, an element of a VALENCE list represents a null pronoun in the following sense: the rules mapping the VALENCE to the phrase structure do not project such an element; and it is accorded the semantic and syntactic properties of a pronoun, such as definiteness and/or specificity, and binding features.

## 11  Semantics

HPSG was conceived from the start as a theory of the *sign* (de Saussure 1916), wherein each constituent is a pairing of form and meaning. So semantic representation and composition was built into HPSG (and the subsequent sister framework of Sign-Based Construction Grammar Boas & Sag 2012), as reflected in the title of the first HPSG book (Pollard & Sag 1987), *Information-Based Syntax and Semantics*. LFG was not founded as a theory that included semantics, but a semantic component was developed for LFG shortly thereafter (Halvorsen 1983). The direction of semantics for LFG changed some ten years later and the dominant tradition is now Glue Semantics (Dalrymple et al. 1993; Dalrymple 1999; 2001; Asudeh 2012; Dalrymple et al. 2019).

This section presents a basic introduction to Glue Semantics. Our goal is to present enough of the approach for readers to grasp the main intuitions behind it, without presupposing much knowledge of formal semantic theory. The references listed at the end of the previous paragraph (especially Dalrymple et al. (2019)) are good places to find additional discussion and references. The rest of this section is organized as follows. In Section 11.1 we present some more historical background on semantics for LFG and HPSG. In Section 11.2, we present Glue Semantics (Glue), as a general compositional system in its own right. Then, in Section 11.3 we look at the syntax–semantics interface with specific reference to an LFG syntax. For further details on semantic composition and the syntax–semantics interface in constraint-based theories of syntax, see Koenig & Richter (2020), Chapter 23 of this volume for semantics for HPSG and Asudeh (2020) for

semantics for LFG.

## 11.1 Brief history of semantics for LFG and HPSG

Various theories of semantic representation have been adopted by the different non-transformational syntactic frameworks over the years. The precursor to HPSG, GPSG, was paired by its designers with a then fairly standard static Montogovian semantics (Gazdar et al. 1985), but GPSG itself was subsequently adopted as the syntactic framework for Discourse Representation Theory (Kamp & Reyle 1993), a dynamic theory of semantics. Initial work on semantics for LFG also assumed a Montogovian semantics (Halvorsen 1983; Halvorsen & Kaplan 1988). But with the increasing interest in Situation Semantics (Barwise & Perry 1983) in the 1980s at Stanford University and environs (particularly SRI International and Xerox PARC), the sites of the foundational work on both HPSG and LFG, both frameworks also became associated with Situation Semantics. In the case of LFG, this association was not foundational, manifesting primarily in the form of Fenstad et al. (1987), and soon waned. In contrast, as noted above, HPSG is a theory of signs and therefore had semantics incorporated into the theory from the outset. Moreover, the chosen semantic theory was Situation Semantics, and this also carried over into the second main HPSG book (Pollard & Sag 1994) and further (Ginzburg & Sag 2000).

Beginning in the 90s, the focus subsequently shifted in new directions due to a new interest in computationally tractable theories of the syntax–semantics interface, to support efforts at large-scale grammar development, such as the ParGram project for LFG (Butt et al. 1999; 2002) and the LinGO/Grammar Matrix projects for HPSG (Flickinger 2000; Bender et al. 2002; 2010).[7] This naturally led to an interest in underspecified semantic representations, so that semantic ambiguities such as scope ambiguity could be compactly encoded without the need for full enumeration of all scope possibilities. Two examples for HPSG are *Lexical Resource Semantics* (Richter 2004; Penn & Richter 2011) and *Minimal Recursion Semantics* (Copestake et al. 2005). Similarly, focus in semantics for LFG shifted to ways of encoding semantic ambiguity compactly and efficiently. This

---

[7]Readers can explore the current incarnations of these projects at the following links (checked January 13, 2020):

**ParGram**  https://pargram.w.uib.no

**LinGO**  http://lingo.stanford.edu

**Grammar Matrix**  http://matrix.ling.washington.edu/index.html

*Draft of 31st March 2020, 22:50*

led to the development of Glue Semantics.

## 11.2 General Glue Semantics

In this section, we briefly review Glue Semantics itself, without reference to a particular syntactic framework. Glue Semantics is a general framework for semantic composition that requires *some* independent syntactic framework but does not presuppose anything about syntax except headedness, which is an uncontroversial assumption across frameworks. This makes the system flexible and adaptable, and it has been paired not just with LFG, but also with Lexicalized Tree-Adjoining Grammar (Frank & van Genabith 2001), HPSG (Asudeh & Crouch 2002b), Minimalism (Gotham 2018), and Universal Dependencies (Gotham & Haug 2018).

In Glue Semantics, meaningful linguistic expressions — including lexical items but also possibly particular syntactic configurations — are associated with *meaning constructors* of the following form:[8]

(29)   $\mathcal{M} : G$

$\mathcal{M}$ is an expression from a *meaning language* which can be anything that supports the lambda calculus; G is an expression of *linear logic* (Girard 1987), which specifies the semantic composition (it "glues meanings together"), based on a syntactic parse. By convention a colon separates them. Glue Semantics is related to (Type-Logical) Categorial Grammar (Carpenter 1997; Morrill 1994; 2011; Moortgat 1997), but it assumes a separate syntactic representation for handling word order, so the terms of the linear logic specify just semantic composition without regard to word order (see Asudeh 2012 for further discussion). Glue Semantics is therefore useful in helping us focus on semantic composition in its own right.

The principal compositional rules for Glue Semantics are those for the linear implication connective, $\multimap$, which are here presented in a natural deduction format:

(30)   Functional application : Implication elimination (modus ponens)

$$\frac{f : A \multimap B \quad a : A}{f(a) : B} \; \multimap\!\varepsilon$$

---

[8]It is in principle possible for a linguistic expression to have a phonology and syntax but not contribute to interpretation, such as expletive pronouns like *there* and *it* or the *do*-support auxiliary in English; see Asudeh (2012: 113) for some discussion of expletive pronouns in the context of Glue.

(31)   Functional abstraction : Implication introduction (hypothetical reasoning)

$$\frac{\begin{array}{c}[a : A]^1\\ \vdots \\ f : B\end{array}}{\lambda a.f : A \multimap B} \, \multimap_{I,1}$$

In each of these rules, the inference over the linear logic term, $G$, corresponds to an operation on the meaning term, via the Curry-Howard Isomorphism between formulas and types (Curry & Feys 1958; 1995; Howard 1980). The rule for eliminating the linear implication, which is just modus ponens, corresponds to functional application. The rule for introducing the linear implication, i.e. hypothetical reasoning, corresponds to functional abstraction. These rules will be seen in action shortly.

In general, given some head $h$ and some arguments of the head $a_1, \ldots, a_n$, an implicational term like the following models consumption of the arguments to yield the saturated meaning of the head: $a_1 \multimap \ldots \multimap a_n \multimap h$. For example, let us assume the following meaning constructor for the verb *likes* in the sentence *Max likes Sam*:

(32)   $\lambda y \lambda x.\textbf{like}(y)(x) : s \multimap m \multimap l$

Let's also assume that $s$ is mnemonic for the semantic correspondent of the (single word) phrase *Sam*, $m$ similarly mnemonic for *Max*, and $l$ for *likes*. In other words, the meaning constructor for *likes* would be associated with the lexical entry for the verb and specified in some general form such that it can be instantiated by the syntax (we will see an LFG example shortly); here we are assuming that the instantiation has given us the meaning constructor in (32).

Given this separate level of syntax, the glue logic does not have to worry about word order and is permitted to be commutative (unlike the logic of Categorial Grammar). We could therefore freely reorder the arguments for *likes* above such that we instead first compose with the subject and then the object, but still yield the meaning appropriate for the intended sentence *Max likes Sam* (rather than for *Sam likes Max*):

(33)   $\lambda x \lambda y.\textbf{like}(y)(x) : m \multimap s \multimap l$

As we will see below, the commutativity of the glue logic yields a simple and elegant treatment of quantifiers in non-subject positions, which are challenging for other frameworks (see, for example, the careful pedagogical presentation of the issue in Jacobson 2014: 244–263).

First, though, let us see how this argument reordering, otherwise known as Currying or Schönfinkelization, works in a proof, which also demonstrates the rules of implication elimination and introduction:

(34)
$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{\lambda y.\lambda x.f(y)(x) : a \multimap b \multimap c \quad [v : a]^1}{\lambda x.f(v)(x) : b \multimap c} \multimap_{\mathcal{E}} \quad [u : b]^2}{f(v)(u) : c} \multimap_{\mathcal{E}}}{\lambda v.f(v)(u) : a \multimap c} \multimap_{\mathcal{I},1}}{\lambda y.f(y)(u) : a \multimap c} \Rightarrow_\alpha}{\lambda u.\lambda y.f(y)(u) : b \multimap a \multimap c} \multimap_{\mathcal{I},2}}{\lambda x.\lambda y.f(y)(x) : b \multimap a \multimap c} \Rightarrow_\alpha$$

The general structure of the proof is as follows. First, an assumption (hypothesis) is formed for each argument, in the order in which they originally occur, corresponding to a variable in the meaning language. Each assumed argument is then allowed to combine with the implicational term by implication elimination. Once the implicational term has been entirely reduced, the assumptions are then discharged in the same order that they were made, through iterations of implication introduction. The result is the original term in curried form, such that the order of arguments has been reversed but without any change in meaning. The two steps of $\alpha$-equivalence, notated $\Rightarrow_\alpha$, are of course not strictly necessary, but have been added for exposition.

This presentation has been purposefully abstract to highlight what is intrinsic to the glue logic, but we of course need to see how this works with a syntactic framework to see how Glue Semantics actually handles semantic composition and the syntax–semantics interface. So next, in Section 11.3, we will review LFG+Glue, as this is the predominant pairing of syntactic framework and Glue.

## 11.3 Glue Semantics for LFG

Glue for LFG will be demonstrated by analyses of the following three examples:

(35)   Blake called Alex.

(36)   Blake called everybody.

(37)   Everybody called somebody.

Example (35) is a simple case of a transitive verb with two proper name arguments, but is sufficient to demonstrate the basics of the syntax–semantics interface in LFG+Glue. Example (36) is a case of a quantifier in object position, which

is challenging to compositionality because there is a type clash between the simplest type we can assign to the verb, $\langle e, \langle e, t \rangle \rangle$, and the simplest type that would be assigned to the quantifier, $\langle \langle e, t \rangle, t \rangle$. In other theories, this necessitates either a syntactic operation which is syntactically undermotivated, e.g. Quantifier Raising in interpretive theories of composition, such as Logical Form semantics (Heim & Kratzer 1998), or a type shifting operation of some kind in directly compositional approaches, as in categorial or type-logical frameworks; see Jacobson (2014) for further discussion and references. Example (37) also demonstrates this point, but it more importantly demonstrates that quantifier scope ambiguity can be handled in Glue without a) positing an undermotivated syntactic ambiguity and b) while maintaining the simplest types for both quantifiers.

The relevant aspects of the lexical entries involved are shown in Table 1. Other syntactic aspects of the lexical items, such as the fact that *called* has a subject and an object, are specified in its meaning constructor. Minimal f-structures are provided below for each example. The subscript $\sigma$ indicates the semantic structure that corresponds to the annotated f-description. The types for the lexical items are the minimal types that would be expected. Note that in Glue these are normally associated directly with the semantic structures, for example $\uparrow_{\sigma_e}$ and $(\uparrow \text{OBJ})_{\sigma_e} \multimap (\uparrow \text{SUBJ})_{\sigma_e} \multimap \uparrow_{\sigma_t}$, but they have been presented separately for better exposition; see Dalrymple et al. (2019: 299–305) for further discussion. We do not show semantic structures here, as they are not necessary for this simple demonstration.

The generalized quantifier functions associated with *everybody* and *somebody* are, respectively, **every** and **some** in the meaning language. The universal symbol $\forall$ in the glue logic/linear logic terms for the quantifiers ranges over semantic structures of type $t$. It is unrelated to the generalized quantifiers. Hence even the existential word *somebody* has the universal $\forall$ in its linear logic glue term. The $\forall$-terms thus effectively say that *any* type $t$ semantic structure $S$ that can be found by application of proof rules such that the quantifier's semantic structure implies $S$ can serve as the scope of the quantifier; see Asudeh (2005: 393–394) for basic discussion of the interpretation of $\forall$ in linear logic. This will become clearer when quantifier scope is demonstrated shortly.

Let us assume the following f-structure for (35):

$$(38) \quad c \begin{bmatrix} \text{PRED} & \text{'call'} \\ \text{SUBJ} & b \begin{bmatrix} \text{PRED} & \text{'Blake'} \end{bmatrix} \\ \text{OBJ} & a \begin{bmatrix} \text{PRED} & \text{'Alex'} \end{bmatrix} \end{bmatrix}$$

Note that here, unlike in previous sections, the PRED value for the verb does

Table 1: Relevant lexical details for the three examples in (35–37)

| Expression | Type | Meaning Constructor |
|---|---|---|
| *Alex* | $e$ | $\mathbf{alex} : \uparrow_\sigma$ |
| *Blake* | $e$ | $\mathbf{blake} : \uparrow_\sigma$ |
| *called* | $\langle e, \langle e, t \rangle \rangle$ | $\lambda y.\lambda x.\mathbf{call}(y)(x) : (\uparrow \text{OBJ})_\sigma \multimap (\uparrow \text{SUBJ})_\sigma \multimap \uparrow_\sigma$ |
| *everybody* | $\langle \langle e, t \rangle, t \rangle$ | $\lambda Q.\mathbf{every}(\mathbf{person}, Q) : \forall S.(\uparrow_\sigma \multimap S) \multimap S$ |
| *somebody* | $\langle \langle e, t \rangle, t \rangle$ | $\lambda Q.\mathbf{some}(\mathbf{person}, Q) : \forall S.(\uparrow_\sigma \multimap S) \multimap S$ |

not list its subcategorization information. This is because we've made the standard move in much Glue work to suppress this information.[9] The f-structures are named mnemonically by the first character of their PRED value. All other f-structural information has been suppressed for simplicity. Based on these f-structure labels, the meaning constructors in the lexicon in Table 1 are instantiated as follows ($\sigma$ subscripts suppressed):

(39) **Instantiated meaning constructors**
   $\mathbf{blake} : b$
   $\mathbf{alex} : a$
   $\lambda y.\lambda x.\mathbf{call}(y)(x) : a \multimap b \multimap c$

These meaning constructors yield the following proof, which is the only available normal form proof for the sentence:[10]

(40) **Proof**

$$
\cfrac{
  \cfrac{
    \cfrac{
      \lambda y.\lambda x.\mathbf{call}(y)(x) : a \multimap b \multimap c \quad \mathbf{alex} : a
    }{
      (\lambda y.\lambda x.\mathbf{call}(y)(x))(\mathbf{alex}) : b \multimap c
    } \; {\multimap}\mathcal{E}
  }{
    \lambda x.\mathbf{call}(\mathbf{alex})(x) : b \multimap c
  } \; {\Rightarrow}_\beta \qquad \mathbf{blake} : b
}{
  \cfrac{
    (\lambda x.\mathbf{call}(\mathbf{alex})(x))(\mathbf{blake}) : c
  }{
    \mathbf{call}(\mathbf{alex})(\mathbf{blake}) : c
  } \; {\Rightarrow}_\beta
} \; {\multimap}\mathcal{E}
$$

---

[9] Indeed, one could go further and argue that PRED values do not list subcategorization at all, in which case the move is not just notational, and that the Principles of Completeness and Coherence instead follow from the resource-sensitivity of Glue Semantics; for some discussion, see Asudeh (2012); Asudeh & Giorgolo (2012); Asudeh et al. (2014).

[10] The reader can think of the normal form proof as the minimal proof that yields the conclusion, without unnecessary steps of introducing and discharging assumptions; see Asudeh & Crouch (2002a) for some basic discussion.

The final meaning language expression, **call**(**alex**)(**blake**), gives the correct truth conditions for *Blake called Alex*, based on a standard model theory.

Let us next assume the following f-structure for (36):

(41) $c\begin{bmatrix} \text{PRED 'call'} \\ \text{SUBJ} \quad b\begin{bmatrix} \text{PRED 'Blake'} \end{bmatrix} \\ \text{OBJ} \quad e\begin{bmatrix} \text{PRED 'everybody'} \end{bmatrix} \end{bmatrix}$

Based on these f-structure labels, the meaning constructors in the lexicon are instantiated as follows ($\sigma$ subscripts again suppressed):

(42) **Instantiated meaning constructors**

$\lambda y.\lambda x.\textbf{call}(y)(x) : e \multimap b \multimap c$

$\lambda Q.\textbf{every}(\textbf{person}, Q) : \forall S.(e \multimap S) \multimap S$

**blake** : $b$

These meaning constructors yield the following proof, which is again the only available normal form proof:[11]

(43) **Proof**

$$\cfrac{\lambda Q.\textbf{every}(\textbf{person}, Q) : \forall S.(e \multimap S) \multimap S}{\cfrac{\lambda Q.\textbf{every}(\textbf{person}, Q) :}{(e \multimap c) \multimap c}} \forall_{\mathcal{E}}[c/S] \qquad \cfrac{\cfrac{\cfrac{\cfrac{\lambda y.\lambda x.\textbf{call}(y)(x) :}{e \multimap b \multimap c} \quad [z : e]^1}{\lambda x.\textbf{call}(z)(x) : b \multimap c} \multimap_{\mathcal{E}}, \Rightarrow_\beta \quad \textbf{blake} : b}{\textbf{call}(z)(\textbf{blake}) : c} \multimap_{\mathcal{E}}, \Rightarrow_\beta}{\lambda z.\textbf{call}(z)(\textbf{blake}) : e \multimap c} \multimap_{\mathcal{I},1}}{\textbf{every}(\textbf{person}, \lambda z.\textbf{call}(z)(\textbf{blake})) : c} \multimap_{\mathcal{E}}, \Rightarrow_\beta$$

The final meaning language expression, **every**(**person**, $\lambda z.\textbf{call}(z)(\textbf{blake})$), again gives the correct truth conditions for *Blake called everybody*, based on a standard model theory with generalized quantifiers.

Notice that the quantifier need not be moved in the syntax — it's just an OBJECT in f-structure — and no special type shifting was necessary. This is because the proof logic allows us to temporarily fill the position of the object quantifier with a hypothetical meaning constructor that consists of a type $e$ variable paired with the linear logic term for the object; this assumption is then discharged to return the scope of the quantifier, $e \multimap c$, and the corresponding variable bound, to yield the function that maps individuals called by Blake to a truth value. In other words,

---

[11]We have not presented the proof rule for Universal Instantiation, but it is trivial; see Asudeh (2012: 396).

we have demonstrated that this approach scopes the quantifier without positing an *ad hoc* syntactic operation and without complicating the type of the object quantifier or the transitive verb. This is ultimately due to the commutativity of the glue logic, linear logic, since the proof does not have to deal with the elements of composition (words) in their syntactic order, because the syntax is separately represented by c-structure (not shown here) and f-structure.

Lastly, let us assume the following f-structure for (37):

(44) $c\begin{bmatrix} \text{PRED} & \text{`call'} \\ \text{SUBJ} & e\begin{bmatrix} \text{PRED} & \text{`everybody'} \end{bmatrix} \\ \text{OBJ} & s\begin{bmatrix} \text{PRED} & \text{`somebody'} \end{bmatrix} \end{bmatrix}$

Based on these f-structure labels, the meaning constructors in the lexicon are instantiated as follows:

(45) **Instantiated meaning constructors**

$\lambda y.\lambda x.\textbf{call}(y)(x) : s \multimap e \multimap c$

$\lambda Q.\textbf{some}(\textbf{person}, Q) : \forall S.(s \multimap S) \multimap S$

$\lambda Q.\textbf{every}(\textbf{person}, Q) : \forall S.(e \multimap S) \multimap S$

These meaning constructors yield the following proofs, which are the only available normal form proofs, but there are two distinct proofs, because of the scope ambiguity:[12]

(46) **Proof 1 (subject wide scope)**

$$
\cfrac{
\lambda Q.\textbf{every}(\textbf{person}, Q) : \forall S.(e \multimap S) \multimap S \quad
\cfrac{
\lambda Q.\textbf{some}(\textbf{person}, Q) : \forall S.(s \multimap S) \multimap S \quad
\cfrac{
\cfrac{
\cfrac{
\cfrac{\lambda y.\lambda x.\textbf{call}(y)(x) : s \multimap e \multimap c \quad [v:s]^1}{\lambda x.\textbf{call}(v)(x) : e \multimap c} {\scriptstyle \multimap\mathcal{E}, \Rightarrow_\beta} \quad [u:e]^2
}{\textbf{call}(v)(u) : c} {\scriptstyle \multimap\mathcal{E}, \Rightarrow_\beta}
}{\lambda v.\textbf{call}(v)(u) : s \multimap c} {\scriptstyle \multimap\mathcal{I},1}
}{\textbf{some}(\textbf{person}, \lambda v.\textbf{call}(v)(u)) : c} {\scriptstyle \multimap\mathcal{E}, \forall\mathcal{E}[c/S], \Rightarrow_\beta}
}{\lambda u.\textbf{some}(\textbf{person}, \lambda v.\textbf{call}(v)(u)) : e \multimap c} {\scriptstyle \multimap\mathcal{I},2}
}{
\cfrac{
\cfrac{
\textbf{every}(\textbf{person}, \lambda u.\textbf{some}(\textbf{person}, \lambda v.\textbf{call}(v)(u)))
}{\textbf{every}(\textbf{person}, \lambda u.\textbf{some}(\textbf{person}, \lambda y.\textbf{call}(y)(u)))} {\scriptstyle \Rightarrow_\alpha}
}{\textbf{every}(\textbf{person}, \lambda x.\textbf{some}(\textbf{person}, \lambda y.\textbf{call}(y)(x)))} {\scriptstyle \Rightarrow_\alpha}
} {\scriptstyle \multimap\mathcal{E}, \forall\mathcal{E}[c/S], \Rightarrow_\beta}
$$

---

[12]We have made the typical move in Glue work of not showing the trivial universal instantiation step this time.

(47)  **Proof 2 (object wide scope)**

$$
\cfrac{
\lambda Q.\mathbf{some}(\mathbf{person}, Q) : \atop \forall S.(s \multimap S) \multimap S
\qquad
\cfrac{
\cfrac{
\lambda Q.\mathbf{every}(\mathbf{person}, Q) : \atop \forall S.(e \multimap S) \multimap S
\qquad
\cfrac{
\cfrac{
\lambda y.\lambda x.\mathbf{call}(y)(x) : \atop s \multimap e \multimap c
\qquad
[v : s]^1
}{
\lambda x.\mathbf{call}(v)(x) : e \multimap c
} \multimap_{\mathcal{E}}, \Rightarrow_{\beta}
}{
\mathbf{every}(\mathbf{person}, \lambda x.\mathbf{call}(v)(x)) : c
} \multimap_{\mathcal{E}}, \forall_{\mathcal{E}}[c/S], \Rightarrow_{\beta}
}{
\lambda v.\mathbf{every}(\mathbf{person}, \lambda x.\mathbf{call}(v)(x)) : s \multimap c
} \multimap_{\mathcal{I},1}
}{
\cfrac{
\mathbf{some}(\mathbf{person}, \lambda v.\mathbf{every}(\mathbf{person}, \lambda x.\mathbf{call}(v)(x)))
}{
\mathbf{some}(\mathbf{person}, \lambda y.\mathbf{every}(\mathbf{person}, \lambda x.\mathbf{call}(y)(x)))
} \Rightarrow_{\alpha}
} \multimap_{\mathcal{E}}, \forall_{\mathcal{E}}[c/S], \Rightarrow_{\beta}
$$

The final meaning language expressions, **every(person**, $\lambda x.$**some(person**, $\lambda y.$**call**$(y)(x)))$▮ and **some(person**, $\lambda y.$**every(person**, $\lambda x.$**call**$(y)(x)))$, give the two possible readings for the scope ambiguity, again based on a standard model theory with generalized quantifiers. Once more, notice that neither quantifier need be moved in the syntax — they are respectively just a SUBJECT and an OBJECT in f-structure. And once more, no special type shifting is necessary. It is a key strength of this approach that even quantifier scope ambiguity can be captured without positing *ad hoc* syntactic operations (and, again, without complicating the type of the object quantifier or the transitive verb). This is once more ultimately due to the commutativity of the linear logic.

## 12  Conclusion

HPSG and LFG are rather similar syntactic frameworks, both of them important lexicalist alternatives to transformational grammar. They allow for the expression of roughly the same set of substantive analyses, where analyses are individuated in terms of deeper theoretical content rather than superficial properties. The same sort of analytical options can be compared under both systems, answers to questions such as whether a phenomenon is to be captured on a lexical level or in the syntax, whether a given word string is a constituent or not, the proper treatment of complex predicates, and so on. Analyses in one framework can often be translated to the other, preserving the underlying intuition of the account. This stands in sharp contrast to frameworks such as Distributed Morphology and Minimalism, whose literature includes many theory-internal issues whose substantive content can be difficult to discern. For example, phrasal constituency is directly represented in both LFG and HPSG, but not in Distributed Morphology or Minimalism, because the tree diagrams in those theories include many putative constituents for which evidence is non-existent.

Against the backdrop of a general expressive similarity, we have pointed out a

few specific places where one framework makes certain modes of analysis available that are not found in the other. The main thesis of this chapter is that the differences between the frameworks stem from different design motivations, reflecting subtly different methodological outlooks. LFG is based on the notion of functional similarity or equivalence between what are externally rather different structures. For example, fixed phrasal positions, case markers, and agreement inflections can all function similarly to signal grammatical relations. LFG makes that functional similarity highly explicit, reflecting its functionalist-friendly outlook on grammar.

## Abbreviations

## Acknowledgements

## References

Asudeh, Ash. 2005. Relational nouns, pronouns, and resumption. *Linguistics and Philosophy* 28(4). 375–446.

Asudeh, Ash. 2012. *The logic of pronominal resumption.* Oxford: Oxford University Press.

Asudeh, Ash. 2020. Glue semantics. In Mary Dalrymple (ed.), *Lexical Functional Grammar: the handbook.* Berlin: Language Science Press. To appear.

Asudeh, Ash & Richard Crouch. 2002a. Derivational parallelism and ellipsis parallelism. In Line Mikkelsen & Christopher Potts (eds.), *Proceedings of the 21st West Coast Conference on Formal Linguistics*, 1–14. Somerville, MA: Cascadilla Press.

Asudeh, Ash & Richard Crouch. 2002b. Glue Semantics for HPSG. In Frank van Eynde, Lars Hellan & Dorothee Beermann (eds.), *Proceedings of the 8th International HPSG Conference*, 1–19. Stanford, CA: CSLI Publications.

Asudeh, Ash & Gianluca Giorgolo. 2012. Flexible composition for optional and derived arguments. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG12 conference*, 64–84. Stanford, CA: CSLI Publications.

Asudeh, Ash, Gianluca Giorgolo & Ida Toivonen. 2014. Meaning and valency. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG14 conference.* Stanford, CA: CSLI Publications.

Barwise, Jon & John Perry. 1983. *Situations and attitudes.* Cambridge, MA: MIT Press.

Bender, Emily M., Scott Drellishak, Antske Fokkens, Laurie Poulson & Safiyyah Saleem. 2010. Grammar customization. *Research on Language & Computation* 8(1). 23–72. http://dx.doi.org/10.1007/s11168-010-9070-1. 10.1007/s11168-010-9070-1.

Bender, Emily M., Dan Flickinger & Stephan Oepen. 2002. The grammar matrix: an open-source starter-kit for the rapid development of cross-linguistically consistent broad-coverage precision grammars. In John Carroll, Nelleke Oostdijk & Richard Sutcliffe (eds.), *Proceedings of the workshop on grammar engineering and evaluation at the 19th international conference on computational linguistics*, 8–14. Taipei, Taiwan.

Boas, Hans C. & Ivan A. Sag (eds.). 2012. *Sign-based Construction Grammar* (CSLI Lecture Notes 193). Stanford, CA: CSLI Publications.

Bresnan, Joan (ed.). 1982. *The mental representation of grammatical relations* (MIT Press Series on Cognitive Theory and Mental Representation). Cambridge, MA: MIT Press.

Bresnan, Joan. 1997. Optimal syntax. In Joost Dekkers, Frank van der Leeuw & Jeroen van de Weijer (eds.), *The pointing finger: conceptual studies in Optimality Theory*. The MIT Press. http://csli-www.stanford.edu/users/bresnan/pt2.ps. To appear.

Bresnan, Joan, Ash Asudeh, Ida Toivonen & Stephen Mark Wechsler. 2016. *Lexical-functional syntax*. 2nd edn. Oxford: Blackwell Publishers Ltd. DOI:10.1002/978111910566⁴

Bresnan, Joan & Jonni M. Kanerva. 1989. Locative inversion in Chichewa: A case study of factorization in grammar. *Linguistic Inquiry* 20(1). 1–50.

Bresnan, Joan & Sam Mchombo. 1987. Topic, pronoun, and agreement in Chicheŵa▮ *Language* 63. 741–82. DOI:10.2307/415717

Butt, Miriam, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi & Christian Rohrer. 2002. The Parallel Grammar Project. In *Proceedings of coling-2002 workshop on grammar engineering and evaluation*, 1–7.

Butt, Miriam, Tracy Holloway King, María-Eugenia Niño & Frédérique Segond. 1999. *A grammar writer's cookbook*. Stanford, CA: CSLI Publications.

Carpenter, Bob. 1997. *Type-logical semantics*. Cambridge, MA: MIT Press.

Chomsky, Noam. 1957. *Syntactic structures* (Janua Linguarum / Series Minor 4). The Hague/Paris: Mouton.

Copestake, Ann, Dan Flickinger, Carl Pollard & Ivan A. Sag. 2005. Minimal recursion semantics: an introduction. *Research on Language and Computation* 3(4). 281–332.

Coppock, Elizabeth & Stephen Wechsler. 2010. Less-travelled paths from pronoun to agreement: the case of the Uralic objective conjugations. In Tracy Holloway

King (ed.), *The proceedings of the LFG '10 conference*, 165–85. Stanford: CSLI Publications.

Curry, Haskell B. & Robert Feys. 1995. The basic theory of functionality. analogies with propositional algebra. In Philippe de Groote (ed.), *The Curry-Howard isomorphism*, vol. 8 (Cahiers du Centre de Logique), 9–13. Louvain-la-neuve, Belgium: Academia. Reprint of Curry & Feys (1958: Chapter 9, Section E).

de Groote, Philippe (ed.). 1995. *The Curry-Howard isomorphism*. Vol. 8 (Cahiers du Centre de Logique). Louvain-la-neuve, Belgium: Academia.

Curry, Haskell B. & Robert Feys. 1958. *Combinatory logic*. Vol. 1. Amsterdam: North-Holland.

Dalrymple, Mary (ed.). 1999. *Semantics and syntax in Lexical Functional Grammar: the resource logic approach*. Cambridge, MA: MIT Press.

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

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

Dalrymple, Mary, John Lamping & Vijay Saraswat. 1993. LFG semantics via constraints. In *Proceedings of the Sixth Meeting of the European ACL*, 97–105. University of Utrecht.

Dalrymple, Mary, John J. Lowe & Louise Mycock. 2019. *The Oxford Reference Guide to Lexical Functional Grammar*. Oxford: Oxford University Press.

de Saussure, Ferdinand. 1916. *Cours de linguistique générale* (Bibliothèque Scientifique Payot). Paris: Payot. Edited by Charles Bally and Albert Sechehaye.

Falk, Yehuda N. 1984. The English auxiliary system: A Lexical-Functional analysis. *Language* 60(3). 483–509.

Fenstad, Jens Erik, Per-Kristian Halvorsen, Tore Langhold & Johan van Benthem. 1987. *Situations, language and logic*. Dordrecht: D. Reidel.

Fillmore, Charles J. 1968. The case for case. In Emmon Bach & Robert T. Harms (eds.), *Universals of linguistic theory*, 1–88. New York: Holt, Rinehart, & Winston.

Fillmore, Charles J. 1977. The case for case reopened. In Peter Cole & Jerrold M. Sadock (eds.), *Grammatical relations* (Syntax and Semantics 8), 59–81. New York, San Francisco, London: Academic Press.

Flickinger, Dan. 2000. On building a more efficient grammar by exploiting types. *Natural Language Engineering* 6(1). 15–28.

Frank, Anette & Josef van Genabith. 2001. LL-based semantics construction for LTAG — and what it teaches us about the relation between LFG and LTAG.

In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG01 conference*, 104–126. Stanford, CA: CSLI Publications. %7B%5Ctexttt%7Bhttp://csli-publications.stanford.edu/LFG/6/lfg01.html%7D%7D.

Gazdar, Gerald, Ewan Klein, Geoffrey K. Pullum & Ivan A. Sag. 1985. *Generalized Phrase Structure Grammar*. Cambridge, MA: Harvard University Press.

Ginzburg, Jonathan & Ivan A. Sag. 2000. *Interrogative investigations: the form, meaning and use of English interrogatives*. Stanford, CA: CSLI Publications.

Girard, Jean-Yves. 1987. Linear logic. *Theoretical Computer Science* 50(1). 1–102.

Gotham, Matthew. 2018. Making logical form type-logical: glue semantics for minimalist syntax. *Linguistics and Philosophy*. https://doi.org/10.1007/s10988-018-9229-z. DOI:10.1007/s10988-018-9229-z

Gotham, Matthew & Dag Trygve Truslew Haug. 2018. Glue semantics for Universal Dependencies. In Miriam Butt & Tracy Holloway King (eds.), *Proceedings of the LFG11 conference*, 208–226. Stanford, CA: CSLI Publications.

Grimshaw, Jane. 1998. Locality and extended projection. In Peter Coopmans, Martin Everaert & Jane Grimshaw (eds.), *Lexical specification and insertion*, 115–133. Mahwah, New Jersey: Lawrence Erlbaum Associates.

Halvorsen, Per-Kristian. 1983. Semantics for Lexical-Functional Grammar. *Linguistic Inquiry* 14(4). 567–615.

Heim, Irene & Angelika Kratzer. 1998. *Semantics in generative grammar*. Oxford: Blackwell.

Howard, William A. 1980. The formulae-as-types notion of construction. In Jonathan P. Seldin & J. Roger Hindley (eds.), *To H.B. Curry: essays on combinatory logic, lambda calculus and formalism*, 479–490. London: Academic press. Circulated in unpublished form from 1969. Reprinted in de Groote (1995: 15–26).

Jacobson, Pauline. 2014. *Compositional semantics: An introduction to the syntax/semantics interface*. Oxford: Oxford University Press.

Kamp, Hans & Uwe Reyle. 1993. *From discourse to logic: introduction to modeltheoretic semantics of natural language, formal logic and Discourse Representation Theory*. Dordrecht: Kluwer.

Kay, Martin. 1984. Functional Unification Grammar: A formalism for machine translation. In Yorick Wilks (ed.), *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, 75–78. Stanford University, California: Association for Computational Linguistics.

Kiparsky, Paul & Johan F Staal. 1969. Syntactic and semantic relations in Pāṇini. *Foundations of Language* 5(1). 83–117.

*Draft of 31st March 2020, 22:50*

Koenig, Jean-Pierre & Frank Richter. 2020. Semantics. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 897–936. Berlin: Language Science Press. DOI:??

Kroeger, Paul. 1993. *Phrase structure and grammatical relations in tagalog*. Stanford CA: Center for the Study of Language & Information (CSLI).

Halvorsen, Per-Kristian & Ronald M. Kaplan. 1988. Projections and semantic description in Lexical-Functional Grammar. In *Proceedings of the international conference on fifth generation computer systems*, 1116–1122. Tokyo. Reprinted in Dalrymple, Kaplan, Maxwell III & Zaenen (1995: 279–292).

Moortgat, Michael. 1997. Categorial type logics. In Johan van Benthem & Alice ter Meulen (eds.), *Handbook of logic and language*, 1st edn., 93–177. Amsterdam: North-Holland.

Morrill, Glyn V. 1994. *Type Logical Grammar*. Dordrecht: Kluwer.

Morrill, Glyn V. 2011. *Categorial grammar: logical syntax, semantics, and processing*. Oxford: Oxford University Press.

Müller, Stefan & Stephen Mark Wechsler. 2014. Lexical approaches to argument structure. *Theoretical Linguistics* 40(1–2). 1–76. DOI:10.1515/tl-2014-0001

Nichols, Johanna. 1986. Head-marking and dependent-marking grammar. *Language* 62(1). 56–119.

Nordlinger, Rachel. 1998. *Constructive case: Evidence from Australia* (Dissertations in Linguistics). Stanford, CA: CSLI Publications.

Penn, Gerald & Frank Richter. 2011. Lexical resource semantics: from theory to implementation. In Stefan Müller (ed.), *Proceedings of the 18th international conference on Head-Driven Phrase Structure Grammar*, 423–443. Stanford, CA: CSLI Publications.

Pollard, Carl J. & Ivan A. Sag. 1987. *Information-based syntax and semantics*. CSLI Publications.

Pollard, Carl J. & Ivan A. Sag. 1994. *Head-Driven Phrase Structure Grammar* (Studies in Contemporary Linguistics). Chicago: The University of Chicago Press.

Richter, Frank. 2004. *Foundations of Lexical Resource Semantics*. Habilitation Thesis. Seminar für Sprachwissenschaft, Universität Tübingen.

Sag, Ivan A., Thomas Wasow & Emily M. Bender. 2003. *Syntactic theory: A formal introduction*. 2nd edn. (CSLI Lecture Notes 152). Stanford, CA: CSLI Publications.

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

Snijders, Liselotte. 2015. *The nature of configurationality in LFG*. University of Oxford. (Doctoral dissertation).

Wechsler, Stephen. 2020. Agreement. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 171–195. Berlin: Language Science Press. DOI:??

Wechsler, Stephen Mark & Larisa Zlatić. 2003. *The many faces of agreement* (Stanford Monographs in Linguistics). Stanford, CA: CSLI Publications.

Wechsler, Stephen, Jean-Pierre Koenig & Anthony Davis. 2020. Argument structure and linking. In Stefan Müller, Anne Abeillé, Robert D. Borsley & Jean-Pierre Koenig (eds.), *Head-Driven Phrase Structure Grammar: The handbook*, 269–319. Berlin: Language Science Press. DOI:??

Wechsler, Stephen & Larisa Zlatić. 2001. Case realization and identity. *Lingua* 111. 539–560.