Tractability and Discontinuity

Ronald M. Kaplan
Stanford University

Jürgen Wedekind
University of Copenhagen

Proceedings of the LFG'19 Conference

Australian National University

Miriam Butt, Tracy Holloway King, Ida Toivonen (Editors)

2019

CSLI Publications

pages 130–148

http://csli-publications.stanford.edu/LFG/2019

Keywords: tractability, undecidability, degree of discontinuity, functional domain, finite boundedness, nonconstructivity, LCFRS, linear context-free rewriting system, finite copying grammar, X-bar theory

Kaplan, Ronald M., & Wedekind, Jürgen. 2019. Tractability and Discontinuity. In Butt, Miriam, King, Tracy Holloway, & Toivonen, Ida (Eds.), *Proceedings of the LFG'19 Conference, Australian National University*, 130–148. Stanford, CA: CSLI Publications.

Abstract

LFG rule systems that embody the explanatory principles of X-bar theory appear to allow for arbitrary repetitions of nodes, particularly complements and coheads, that map to the same f-structure. Such X-bar compliant grammars thus fail to meet the requirements for tractable computation that have been identified in recent work (Wedekind and Kaplan, to appear). This raises the question whether those grammars also fail of descriptive accuracy in that they provide for derivations with syntactic dependencies that are not attested in natural languages. We address what may be regarded as a discrepancy between explanatory and descriptive adequacy by imposing finite bounds on the degree of discontinuity of grammatical functions and the number of nodes in functional domains. We introduce additional filtering conditions on derivations, along the lines of Completeness and Coherence and the original nonbranching dominance prohibition, that make it possible to decide that derivations respect those bounds. Grammars that embody the principles of X-bar theory and satisfy these bounds and additional requirements are amenable to tractable processing.

1 Introduction

It is well known that the recognition/parsing problem is decidable for the LFG formalism restricted only by the nonbranching dominance condition (Kaplan and Bresnan, 1982), and the generation problem is decidable for arbitrary LFG grammars as long as the input f-structure is acyclic (Wedekind and Kaplan, 2012; Wedekind, 2014). But even with these restrictions these problems are also known to be computationally intractable in the worst case, as they still belong to the class of NP-complete problems (e.g. Berwick, 1982).

However, grammars for actual languages seem not to exploit all the mathematical power that the formalism makes available, as witnessed by the fact that parsing and generation systems, for example, the XLE system, have been constructed that are practical for broad-coverage grammars and naturally occurring sentences (Maxwell and Kaplan, 1996; Crouch et al., 2008). The implementations of these systems must be implicitly taking advantage of certain patterns of dependencies that are characteristic of linguistic grammars even if those properties have not yet been clearly articulated, and their computational consequences are not yet clearly understood and have not been explicitly coded.

Wedekind and Kaplan (to appear) (henceforth WK) define a subclass of LFG grammars with formal properties that, they suggest, are compatible with linguistic description but make possible the translation into equivalent grammars in the framework of linear context-free rewriting systems (LCFRS), a mildly context-sensitive grammatical formalism (Seki et al., 1991;

Kallmeyer, 2013). The recognition and emptiness problems of LCFRS grammars are decidable, and recognition is tractable in the sense that it requires time that is proportional to a polynomial (vs. an exponential) function of the length of the input. WK show that it is decidable whether an arbitrary LFG grammar in the original Kaplan and Bresnan (1982) formalism meets the requirements of the restricted subclass and thus admits of an LCFRS equivalent.

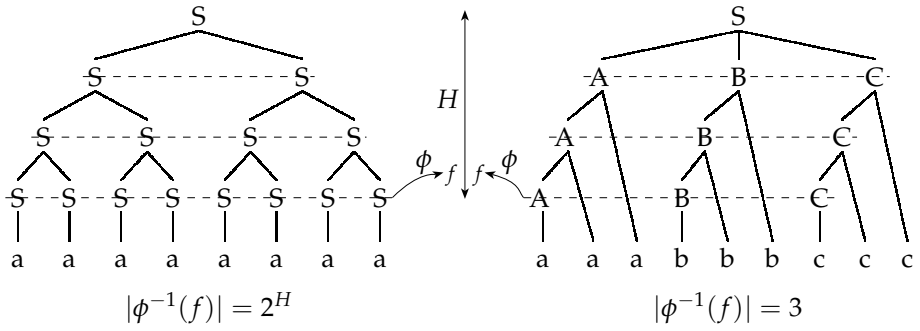
In this paper we summarize the formal properties that WK have identified, and we indicate informally why computational performance is sensitive to these particular restrictions. These restrictions include imposing finite upper bounds on the size of functional domains and the c-structure discontinuity of functional units. We observe, however, that these bounds are not realized by grammars whose rules conform to the schematic prescriptions of X-bar theory but are not otherwise constrained. We provide an LFG formalization that includes two extragrammatical parameters, the *degree of discontinuity* (Chomsky, 1953) and the *height of the functional domains*, and show that conditions for tractability are achieved if we allow only derivations that respect finite bounds on these parameters even though a language may not limit the number of modifers. Thus, if bounded derivations can account for all and only the sentences of natural languages, grammars can instantiate the explanatory principles of X-bar theory and still admit of tractable computation.

2 Tractable LFG grammars

WK take as a point of departure the severely restricted subclass of LFG grammars that were shown to be tractable by Seki et al. (1993). The functional annotations of these *finite-copying grammars* are severely restricted: they allow categories to be annotated only with at most one function assignment of the form $(\uparrow \text{G}) = \downarrow$ and feature assignments of the form $(\uparrow \text{A}) = v$ that assert feature values with a single attribute just on the mother nodes of c-structure expansions.

This notation is clearly unsuitable for linguistic description. It disallows the trivial $\uparrow = \downarrow$ annotations that mark the heads and coheads in the functional domain of a predicate, the $(\uparrow \text{XCOMP SUBJ}) = (\uparrow \text{OBJ})$ equations of functional control, and all other ways of relating the f-structures of different nodes. It also disallows longer attribute paths in the specification of mother-node feature values (e.g. $(\uparrow \text{SUBJ NUM}) = \text{SG}$) and any direct specification of feature values on daughter nodes, as in $(\downarrow \text{CASE}) = \text{NOM}$.

In terms of formal expressiveness, on the other hand, the function-assignment annotations in these grammars do allow separate nodes of the c-structure to map to the same unit of f-structure and thus to share (and require consistency of) atom-value information in a very constrained way.



a. C-structure for grammar (1) b. C-structure for grammar (2)

Figure 1: Zipper nodes in depth-balanced c-structures

This specific type of structure sharing is occasionally referred to as "zipper" unification. That is, two distinct nodes n and n' can map to the same f-structure in a valid derivation (formally, $\phi(n) = \phi(n')$) if there is a node $\hat{n}$ dominating both of these nodes and the sequences of function-assignment annotations on the c-structure paths from $\hat{n}$ to n and n', respectively, are identical, that is, form a "zipper". As a significant consequence, structure-sharing nodes for finite-copying grammars must have the same c-structure depth. This is illustrated by the simple grammars in (1) and (2) and by the depth-balanced c-structures in Figure 1, where the dashed lines link collections of nodes that map through ϕ to the same f-structure.

(1) S → S S S → a
(↑ L)=↓ (↑ L)=↓ (↑ L) = #

(2) S → A B C
(↑ L)=↓ (↑ L)=↓ (↑ L)=↓

A → A a A → a
(↑ L)=↓ (↑ L) = #

B → B b B → b
(↑ L)=↓ (↑ L) = #

C → C c C → c
(↑ L)=↓ (↑ L) = #

While these grammars both meet the very stringent finite-copying notational restrictions, the difference in these structure-sharing configurations corresponds to a difference in computational complexity. For all derivations of grammar (1) the number of nodes that map to a given f-structure f ($= |\phi^{-1}(f)|$) is an exponential in the height H of those nodes, as illustrated

in Figure 1a. In contrast, for all derivations of grammar (2) the number of nodes in a structure-sharing set is bounded by a constant (3 in this case) that is independent of their height (Figure 1b). Grammar (2) but not (1) meets the finite-boundedness property (3) that Seki et al. (1993) also included in the definition of finite-copying grammars.

(3) A grammar is **finitely bounded** if and only if there is a grammar-dependent constant d such that no more than d nodes map to the same f-structure element f in any derivation. That is, $|\phi^{-1}(f)| \leq d$.

They established that this is a decidable property of grammars with only function-assignment and mother-feature annotations and thus whether any particular grammar belongs to the finite-copying subclass. They further demonstrated that for any such finitely-bounded and notationally restricted grammar G there is an equivalent mildly context-sensitive grammar G' (an LCFRS). It follows that for this grammar class that the emptiness problem is decidable, the recognition problem is tractable, and parsing with G can be accomplished with the equivalent LCFRS G'. The key insight is that atom-based satisfiability can be tested in zipper domains of no more than d nodes, even when these are not adjacent in the c-structure. We regard the parameter d as formalizing the linguistic notion of the degree of discontinuity (Chomsky, 1953).

Finite boundedness is a necessary property for LCFRS equivalence and tractability. It is a sufficient property for finite-copying grammars, but further restrictions are required for grammars which include annotations in more elaborate formats. WK carefully extend the notation to allow for annotations that are commonly used in LFG accounts of syntactic phenomena and are thus more suitable for linguistic description. These extensions are introduced with formal restrictions that conserve the LCFRS equivalence and thus preserve the key advantages of that minimally expressive formalism. They first extend the notation to admit more elaborate atomic-value annotations on both mother and daughter nodes. This includes annotations such as $(\uparrow \text{SUBJ NUM}) = \text{SG}$ and $(\downarrow \text{NUM}) = \text{PL}$ and in general any annotations of the forms $(\uparrow \text{A B C} \ldots) = v$ and $(\downarrow \text{A B C} \ldots) = v$.

They also introduce a broader but still limited range of annotations that establish relationships between the f-structures associated with different nodes in a derivation. They allow trivial annotations $\uparrow = \downarrow$, and reentrancies of the following forms:[1]

[1]WK also allow set-membership annotations ($\downarrow \in (\uparrow \text{ADJ})$) that are typically used for modifiers. Membership annotations only map single nodes to f-structures and do not propagate atomic feature values, and thus are inert with respect to zipper formation, satisfiability, and computational complexity. We return to this point below.

(4)

$(\uparrow \text{F G}) = (\uparrow \text{H})$	functional control
$(\uparrow \text{F}) = (\uparrow \text{H})$	local-topic link
$(\downarrow \text{G}) = (\uparrow \text{H})$	SUBJ in XADJ control
$(\downarrow \text{G}) = (\downarrow \text{H})$	daughter sharing
$(\downarrow \text{G}) = \uparrow$	promotion
$(\uparrow \text{F}) = \uparrow$	mother cycle
$(\downarrow \text{G}) = \downarrow$	daughter cycle

These annotations allow designators with at most two grammatical functions, in keeping with the Principle of Functional Locality (Kaplan and Bresnan, 1982). WK hypothesize that this inventory of annotation-types is sufficient for natural language description, since the notation can account for function assignments, head and cohead identification, functional control, and agreement. This claim receives empirical support from an examination of the large-scale, broad-coverage Pargram grammars of English and German. There are only a few outliers in the 281 rules and 23,809 lexical entries of the English grammar, for example, and these appear only in still controversial accounts of oblique prepositional phrases and copular complements.

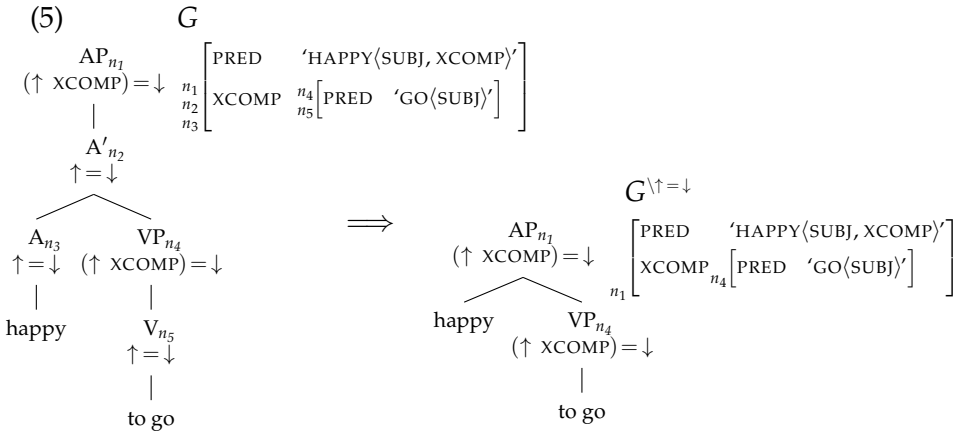
WK also demonstrate that this family of linguistically suitable annotations is also expressive enough to make the core computational problems—recognition (without the nonbranching dominance restriction), emptiness, and generation from cyclic inputs—undecidable.[2]

WK then consider a set of restrictions that together ensure LCFRS equivalence. They conjecture that these restrictions will not undermine the descriptive utility of the reentrancies and trivial annotations. WK first require, similar to Seki et al., that nonterminal categories are annotated with at most one function assignment of the form $(\uparrow \text{G}) = \downarrow$, and, furthermore, that trivial annotations and function assignments always appear in complementary distribution to keep separate the properties of a head and its complements.

Since finite boundedness is a necessary condition for LCFRS equivalence, clearly there must be a finite bound on the number of nodes in a functional domain. These nodes are annotated with $\uparrow = \downarrow$ annotations and typically carry information about heads, coheads, and the local morphosyntactic features of a functional unit. These $\uparrow = \downarrow$ annotations map all the nodes in a functional domain to the same f-structure, and thus allow

[2]Their proof strategy is quite straightforward. Following Wedekind (2014), they show that for an arbitrary context-free grammar an LFG grammar can be constructed whose c-structures simulate the context-free derivations and whose corresponding f-structures encode the terminal strings of those derivations. Grammars that encode the strings of two context-free grammars G_1 and G_2 can easily be composed so that the core computational problems for the combined LFG grammar are solvable if and only if the intersection $L(G_1) \cap L(G_2)$ is non-empty. This problem is known to be undecidable.

information to propagate up, down, and across the chain of nodes that relate to a single head. The grammar may implicitly impose a bound h on the height of a functional domain if vertical chains bottom out at lexical items or are terminated by nodes with other annotations. If the grammar does not establish an implicit bound on the height of the functional domains, one must be specified as an extragrammatical parameter. Given that such a bound has been determined, there is a simple transformation of a grammar G into a strongly equivalent LFG grammar $G^{\setminus\uparrow=\downarrow}$ that no longer contains $\uparrow=\downarrow$ annotations. The transformation is accomplished by recursively replacing a category annotated with $\uparrow=\downarrow$ in the right side of one rule by the right sides of all the rules expanding that category, and making the appropriate replacements of $\uparrow$ for $\downarrow$ to preserve the f-structure mappings. The effect of a simple case of this transformation is shown in (5).



The f-structure units are labeled here to illustrate how the size of the ϕ^{-1} node sets is reduced when $\uparrow=\downarrow$ annotations are eliminated. This simple grammar transformation makes it unnecessary to give further consideration to $\uparrow=\downarrow$ annotations.

Without further restriction, the recognition, emptiness, and generation problems are still undecidable for grammars with height-bounded functional domains. Thus, finite boundedness, a necessary condition for LCFRS equivalence, is also undecidable for these grammars.[3] WK observe that every LFG grammar G can be decomposed into two subgrammars, a *reentrancy-free kernel* $G_{\setminus\mathcal{R}}$ and an *atom-free kernel* $G_{\setminus\mathcal{A}}$, both with decidable recognition and emptiness problems, and they define the necessary and sufficient restrictions on the LFG grammars G by carefully regulating the interplay of the functional descriptions of these two grammars. The reentrancy-free kernel of G is formed by removing all reentrancies from

[3]This is because it is undecidable whether there is a valid derivation at all (undecidability of the emptiness problem).

its rules and lexical entries, leaving just function assignments and atomic-value annotations. The atom-free kernel is formed by removing all atomic-value annotations from its rules and lexical entries, so that only function assignments and reentrancies remain. We let $FD_{\setminus\mathcal{R}}$ and $FD_{\setminus\mathcal{A}}$ be the instantiated f-descriptions for derivations in $G_{\setminus\mathcal{R}}$ and $G_{\setminus\mathcal{A}}$ that correspond to a derivation in G with f-description FD.

Because d-boundedness is a necessary condition for LCFRS equivalence, WK first require the reentrancy-free kernel of $G^{\setminus\uparrow=\downarrow}$ to be d-bounded and they show that this is a decidable property. They then proceed to identify a minimally intrusive restriction that ensures that G conserves the ϕ mapping of its d-bounded reentrancy-free kernel. WK relate this restriction to another notion in the LFG literature, the concept of *nonconstructivity*. This has been discussed in the context of functional uncertainty and off-path constraints (e.g. Dalrymple et al., 1995b; Crouch et al., 2008), the idea that functional uncertainties pass information along separately motivated f-structure paths, and it is implicit in the fact that the Completeness Condition tests for the existence of independently specified grammatical functions. WK provide a technical formulation of this notion and propose it as a general condition on the operation of reentrancy annotations, a condition that is necessary to ensure LCFRS equivalence. They define nonconstructive reentrancies in terms of the interplay between the reentrancy-free and the atom-free kernel in the following way:

(6) G has **nonconstructive reentrancies** if and only if for every derivation in $G^{\setminus\uparrow=\downarrow}$ and any nodes n and n', if $\phi(n) = \phi(n')$ follows from $FD_{\setminus\mathcal{R}}$, then $\phi(n) = \phi(n')$ also follows from $FD_{\setminus\mathcal{A}}$.

This is a succinct formalization of the idea that reentrancies by themselves, without the support of function assignments, do not cause different nodes to project to the same f-structure. Thus a grammar is d-bounded if its reentrancy-free kernel is d-bounded and its reentrancies are nonconstructive. WK prove, for any LFG grammar G with only short reentrancies and a d-bounded reentrancy-free kernel, that it is decidable whether G has nonconstructive reentrancies.

The long reentrancies of the form $(\uparrow \text{F G}) = (\uparrow \text{H})$ that are found in descriptions of functional control, for example

(7) $(\uparrow \text{XCOMP SUBJ}) = (\uparrow \text{OBJ})$

appear to be just a minor enhancement to the short reentrancies in (4). However, WK demonstrate that core computational problems are undecidable for (1-)bounded grammars with long reentrancies. Thus, finite boundedness by itself is not a sufficient condition for LCFRS equivalence for grammars with long reentrancies that cannot be reduced to short ones.

WK observe that long control reentrancies such as (7) can always be shortened in derivations that meet the requirements of the Coherence Condition. They argue specifically that SUBJ is a governable function in an open (XCOMP) complement and therefore must be licensed by the complement's semantic form. These licensing semantic forms are always introduced by simple PRED equations associated with individual lexical entries, for example $(\uparrow \text{PRED}) = \text{'WALK}\langle(\uparrow \text{SUBJ})\rangle\text{'}$. Thus, $(\uparrow \text{PRED}) = \text{'WALK}\langle(\uparrow \text{SUBJ})\rangle\text{'}$ must instantiate to the equation $(\phi(n') \text{ PRED}) = \text{'WALK}\langle(\phi(n') \text{ SUBJ})\rangle\text{'}$ at some node n', and the functional description must also entail an equation $(\phi(n) \text{ XCOMP}) = \phi(n')$ that links the complement to a higher clause and is also available to shorten the control equation.

Even though it is easy to determine whether all long reentrancies can be shortened in any given derivation, it is not possible in general to construct an LCFRS that exactly simulates the set of all valid derivations of an LFG grammar. This is because it is also undecidable whether there are any derivations in which all long reentrancies can be shortened. Therefore, WK require the derivations to meet the stronger stipulation that the shortening equations ($(\phi(n) \text{ XCOMP}) = \phi(n')$) are entailed by the reentrancy-free kernel. This is formalized in (8).

(8) If FD contains $(\phi(n) \text{ F G}) = (\phi(n) \text{ H})$, then $FD_{\setminus\mathcal{R}}$ entails $(\phi(n) \text{ F}) = \phi(n')$ for some node n'.

Thus, just as LFG theory classifies as invalid derivations that do not satisfy the Completeness and Coherence Conditions (or that violate, in earlier specifications, the prohibition against nonbranching dominance chains), WK propose to remove from grammatical consideration derivations with recalcitrant control equations. This means that, as in the case of these previous conditions, some analyses will be excluded that otherwise appear to lie within scope of the normal derivational machinery of the LFG formalism. In all likelihood the derivations that this restriction eliminates would also fail to meet the Coherence Condition and thus no linguistically significant derivations will be lost.

The formal framework and result laid out by WK is summarized in (9).

(9) If G is an LFG grammar such that
    G includes only the reentrancies in (4)
    no more than one function assignment or $\uparrow = \downarrow$ annotation is attached to any category,
    G's functional domains are h-bounded,
    G's reentrancy-free kernel is d-bounded, and
    G's reentrancies are decidably nonconstructive
  then G is equivalent to an LCFRS.[4]

[4]The LCFRS is constructed in two stages. In the first stage a $\uparrow = \downarrow$-free LFG grammar

WK also demonstrate that the emptiness and cyclic generation problems for these grammars are decidable and that sentences can be recognized in polynomial time.

3 The challenge of X-bar theory

WK suggest that grammars that meet the conditions in (9) and have the associated computational advantages are also suitable for linguistic description. This suggestion appears to be undermined by the principles of X-bar theory as they appear in the literature in various forms. These principles are generally assumed to constrain the organization of c-structure and the distribution of annotations that map from c-structure to f-structure (e.g. Bresnan, 2001; Dalrymple, 2001). X-bar prescriptions are typically given as meta phrase-structure rules that govern the expansion of a generic major category XP into an X′-labeled head which expands in turn to an X-labeled lexical head. These expansions branch, either recursively or iteratively, to other major categories annotated with function assignments $(\uparrow \text{G}) = \downarrow$ (for argument functions) or set-member annotations $(\downarrow \in (\uparrow \text{ADJ})$ for modifiers). They also branch to other categories with $\uparrow = \downarrow$ annotations (for coheads).

The problem for finite boundedness is that X-bar compliant rule systems appear to allow for arbitrary repetitions of nodes, particularly complements and coheads, that map to the same f-structure. Schematically, the possibilities for specifiers, complements, and coheads are illustrated in (10). (The infix comma notation is a conventional way of abbreviating the fact that this scheme is agnostic as to the linear order of constituents, and any specific categories may or may not appear in the particular rules that instantiate this general scheme.)

(10) a. $\text{XP} \rightarrow \underset{\substack{\uparrow = \downarrow \\ \textit{head}}}{\text{X}'} \ , \ \underset{\substack{(\uparrow \text{D}) = \downarrow \\ \textit{spec}}}{\text{LP} \mid \text{FP}}$

b. $\text{X}' \rightarrow \underset{\substack{\uparrow = \downarrow \\ \textit{head}}}{\text{X}'} \ , \ \left\{ \underset{\substack{(\uparrow \text{G}) = \downarrow \\ \textit{comp}}}{\text{LP}} \ \middle| \ \underset{\substack{\uparrow = \downarrow \\ \textit{cohead}}}{\text{FP}} \right\}$

$G^{\setminus\uparrow=\downarrow}$ is created by eliminating the h-bounded $\uparrow = \downarrow$-annotated categories in favor of equivalent collections of flattened LFG rules. The second stage of the construction produces LCFRS rules for $G^{\setminus\uparrow=\downarrow}$. It hypothesizes finite sequences of $G^{\setminus\uparrow=\downarrow}$ rules that might expand the categories realizing a d-bounded zipper, and it builds an LCFRS rule that models the well-formedness conditions and the minimal f-structure for each such sequence.

c. $X' \rightarrow \underset{\uparrow=\downarrow}{X}, \underset{(\uparrow \text{G})=\downarrow}{\text{LP}}$

head *comp*

In this particular version of the meta-grammatical scheme, the lexical or functional categories LP or FP in (10a) assign a discourse function (D) to the specifier. The recursive expansion in (10b) derives complements (labeled as governable grammatical functions G) and coheads, and the recursion terminates at the lexical head in (10c).[5]

We focus our attention on the X′ recursion in (10b). This permits the object of a VP, for example, to be realized discontinuously by any number of NP's annotated with $(\uparrow \text{OBJ}) = \downarrow$, and all of those nodes would contribute features to the same f-structure. Thus, this general rule schema immediately licenses derivations that are not finitely bounded. Or, to put it in more traditional terms, the X-bar framework admits of grammatical rules that allow for an unbounded degree of discontinuity (Chomsky, 1953). This scheme also allows for functional domains of unbounded height, to derive arbitrary numbers of coheads in addition to a possible head. Grammars formulated according to this specification clearly fail to meet the tractability requirements as summarized in (9).

As theoretically appealing as it may be, this configuration may not be descriptively accurate for real languages. Real languages may have substantially less potential for repetition, and rules of this type may substantially overgenerate beyond what should be implemented in descriptively adequate grammars. Indeed, we suspect that the unrestricted X-bar schema does overgenerate, and that functional units in real languages are not arbitrarily decomposable. Individual lexical predicates subcategorize for a limited number of governable grammatical functions, there are a limited number of morphosyntactic and cohead features to be expressed with limited redundancy, and there may also be categorial cooccurrence restrictions that limit, for example, which subconstituents internal to an NP can surface independently (e.g. perhaps agreement markers and determiners must always appear together). And of course, because semantic forms are instantiated in LFG derivations, there can be at most one lexical head among the phrases that realize any grammatical function. We thus conjecture that for each language the X-bar scheme is constrained by two extragrammatical parameters: the maximum degree of discontinuity d and the maximum number c of coheads in a functional domain.

The constants d and c provide an upper bound on the height of the X′ recursion in (10b), since there is also a bound on a language-dependent parameter g, the number of grammatical functions (discourse functions and

[5]Some presentations make use of an iterative Kleene-star format instead of the recursive, binary-branching formulation here. Both formulations admit of unbounded repetition and are thus essentially the same with respect to the formal issues we are concerned with here.

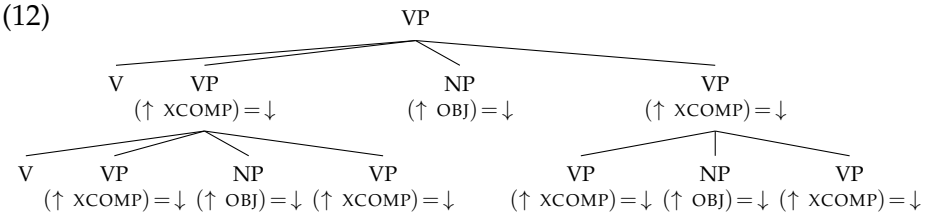
governable functions) that can appear together in any subtree.[6] The maximum height h is given by the formula $h = c + dg + 1$, with the additional 1 accounting for the lexical X expansion in (10b). If a subderivation for XP is higher than h, then the d bound must be exceeded for at least one grammatical function or the c bound must be violated. For a grammar that respects this height limit the algorithm for eliminating $\uparrow = \downarrow$ annotations from G will terminate in a grammar $G^{\setminus \uparrow = \downarrow}$ with a finite number of rules.

In the simplest case the succinct recursion in (10) would be converted to flattened rules that explicitly limit the number of repetitions of any one function assignment. For $d = 2$, for example, such a grammar might contain the V′ expansion rule (11) that maps only two nodes to XCOMP and one node to OBJ.

(11) $\mathrm{V}' \rightarrow (\mathrm{V})$ VP $(\uparrow \text{XCOMP}) = \downarrow$ NP $(\uparrow \text{OBJ}) = \downarrow$ VP $(\uparrow \text{XCOMP}) = \downarrow$

It is not difficult to see how this rule relates to the recursive X-bar specification, and the connection to an iterative Kleene-star X-bar schema would be even more obvious: a grammar's *-marked generic rules would be reduced to particular sequences whose length is restricted by h. The fragmentary V′ expansion shown in (12) demonstrates, however, that restricting the length of individual rules is by itself not sufficient to ensure that discontinuity is globally bounded.

(12)



Although only two VP nodes map to the same f-structure at each level of recursion, the overall effect of the parallel subderivations (with zipper unification) is that the four discontinuous VP nodes at the bottom map to the

[6] The grammatical functions attested in any language are drawn from a finite universal set GF, and g cannot be greater than $|\text{GF}|$. The number of governable functions included in GF can be determined for any particular language by inspecting the subcategorization frames in its lexicon. For the broad-coverage, commercial-grade Pargram grammar of English (approximately 25,000 lexical entries) there are 10 different governable functions, and for the German Pargram grammar there are 13. But a tighter bound on g comes from the fact that no word in either lexicon governs more than four functions, and very few words allow even that many (in English only the word *bet*). The inventory of lexical subcategorization frames further restricts the cooccuring combinations of governable functions. For the English and German grammars the number of combinations (32 and 41 respectively) is much less than the theoretical maximum of all up-to-four combinations of the attested set of governable functions.

same XCOMP XCOMP f-structure. While it is possible in principle to convert an X-bar compliant grammar (either recursive or iterative) to one that is descriptively accurate for a language with a global bound on the degree of discontinuity, the resulting grammar will have an elaboration of rules and a refinement of features and categories that will likely be convoluted and opaque, and its relation to the explanatory generalizations of X-bar theory will be obscure.

We propose to address this discrepancy between explanatory and descriptive adequacy, for a language with a bounded degree of discontinuity and a bounded number of coheads, in a direct and brute-force way. Such a language is described by an X-bar compliant grammar together with separate d and c values that bound its degree of discontinuity and the number of coheads under a single XP. If a grammatical function can be realized by no more than d separate XP's one of which contains a lexical head, and each XP can have no more than c coheads, then the number of nodes that can map to a single f-structure is bounded by $dc + 1$. We simply declare as invalid and disallow any derivation if $|\phi^{-1}(f)|$ for any f exceeds that number.[7] This is a further filtering condition on derivations, along the lines of Completeness and Coherence and the original nonbranching dominance prohibition of Kaplan and Bresnan (1982). Once the c-structure and the minimal solution of the f-description have been constructed, derivations are discarded if their structure-function mapping exceeds this bound.

Derivations that survive the d-c restriction also satisfy the bounding requirements as stated in (9). This at least removes the specific challenge to tractability that comes with the unbounded repetition of arguments and coheads implied by the principles of X-bar theory. If grammars for natural language also meet the constraints on notation and reentrancies in (9), as WK conjecture, then it is possible to construct linear context-free rewriting systems for such grammars that simulate all and only their valid derivations.[8]

4 Modifiers: Height and discontinuity

Modifiers are represented as sets in f-structure precisely because they are not selected by particular predicates and because there are no natural limits on how many may appear. They therefore pose a different kind of challenge to the bounding conditions necessary for LCFRS equivalence. One

[7] Note that $dc + 1$ is also an upper bound for a single grammatical function. This architecture is flexible enough to account for much more fine-grained distinctions where, for example, the major categories or grammatical functions differ in their degree of discontinuity and/or their number of coheads.

[8] In constructing an LCFRS for derivations with extragrammatical bounds, only reentrancies in the bounded derivations must be checked for nonconstructivity. This can be decided along the lines of the shrinking argument of WK.

aspect of the problem is illustrated by the Kleene-starred adjunct phrases of Bresnan's (2001) alternative X-bar schema in (13).

(13) $\text{XP} \rightarrow \underset{\uparrow=\downarrow}{\text{X}'} \quad , \quad \underset{\downarrow\in(\uparrow \text{ADJ})}{\text{YP}^*}$

These rules can be normalized to rules of the recursive form by translating the Kleene-starred adjunct phrases into right-linear expansions. The translation is done by replacing the iteration of the phrasal adjunct in (13) by a new optional trivially-annotated category YP_{ADJ}, and by introducing the rule (14) to properly expand that category.

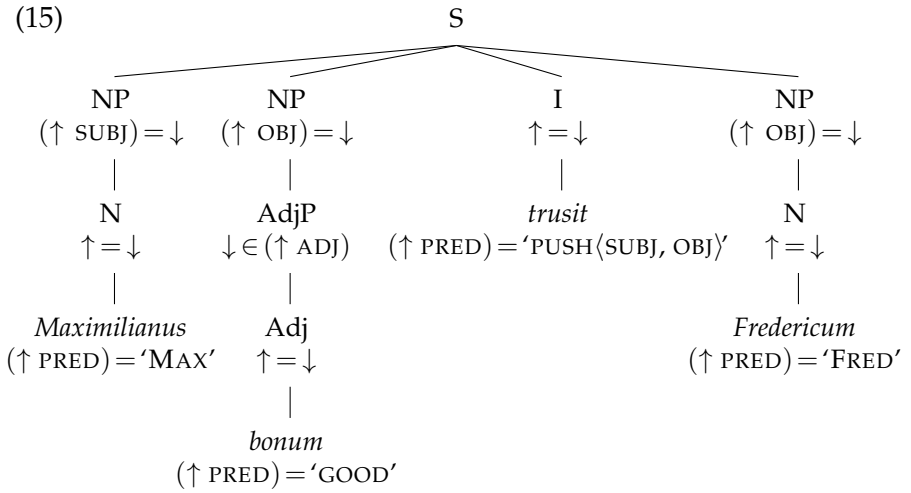
(14) $\text{YP}_{\text{ADJ}} \rightarrow \underset{\downarrow\in(\uparrow \text{ADJ})}{\text{YP}} \quad (\underset{\uparrow=\downarrow}{\text{YP}_{\text{ADJ}}})$

Immediately we see that there is no apparent bound on the height of the functional domain and thus no way of eliminating these trivial annotations. But, as it turns out, the LCFRS construction does not require the removal of trivials in this particular configuration. This is because these YP_{ADJ} nodes and their subtrees are inert with respect to zipper interactions. Therefore these nodes can be ignored when determining the height of the functional domain, the degree of discontinuity, and the finite bound as given in (3).

The expansion of a category is syntactically inert if the zippers within its subtrees in all possible derivations cannot interact with the zippers outside. This is certainly true of the YP expansion in (14), since membership annotations, unlike function assignments, form barriers that prevent daughter attributes and values from escaping to higher levels. The expansions of the trivially-annotated YP_{ADJ} categories are also inert because they dominate only inert category expansions. Thus the YP_{ADJ} nodes need not enter into the calculation of boundedness and their $\uparrow=\downarrow$ annotations need not be eliminated. We mark that fact simply by replacing those equalities with a variant $\uparrow\doteq\downarrow$ that is opaque to the trivial-elimination procedure. It can be carried along by the LCFRS translation algorithm and interpreted as a node identity only during f-structure construction.

Internal adjuncts of discontinuous NPs are another instance of inertness. A Latin example taken from Haug (2017) is shown in (15).

(15)



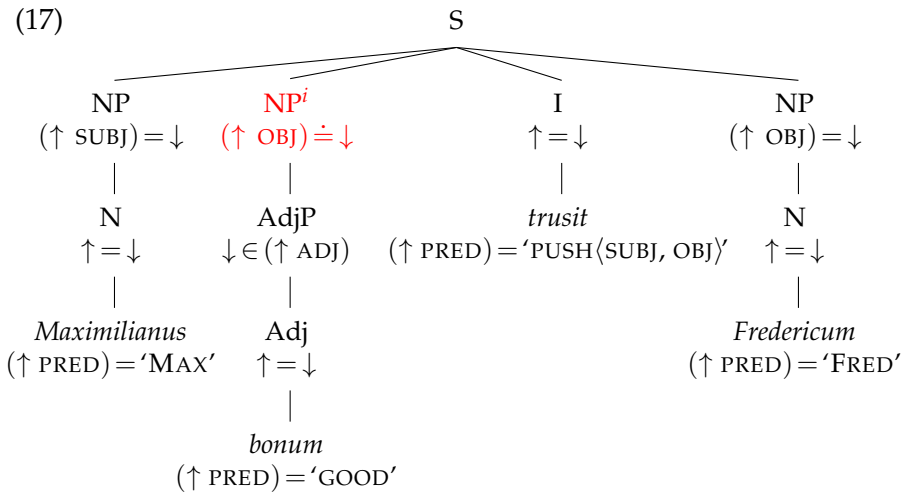
Haug (2017) investigates the degree of discontinuity of Latin based on the nonprojective dependencies occurring in several dependency treebanks and argues that there is no principled bound on Latin discontinuities. In his examples, however, the adjunct NP expansions that seem to lead to this conclusion are—as in (15)—inert with respect to zippers, because no information can escape from their AdjP daughters. This might result in an unbounded number of constituents that map to the same f-structure, but crucially only the f-structures of a bounded number of them can interact and thus give rise to zipper dependencies.

In this situation the adjunct expansion of the NP can be marked as inert, as indicated by the superscript i in (16),

(16) $\text{NP}^i \rightarrow \underset{\downarrow \in (\uparrow \text{ADJ})}{\text{AdjP}}$

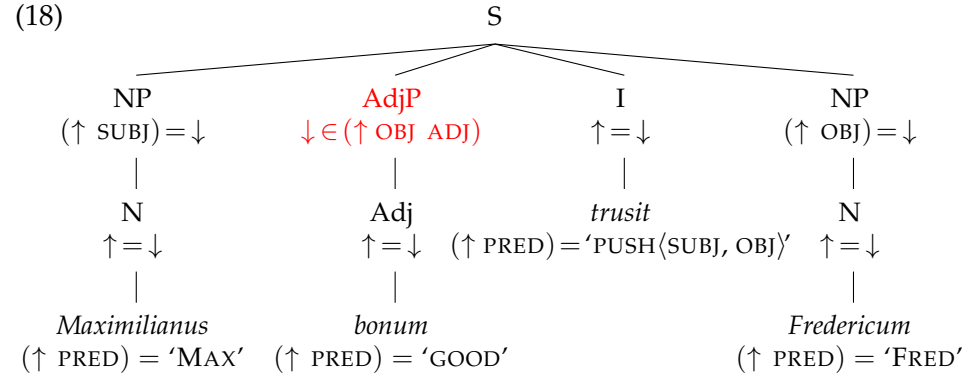
and the NP occurrence in the S rule can be replaced by NP^i with a variant annotation $(\uparrow \text{OBJ}) \doteq \downarrow$ that explicitly indicates that this function assignment does not give rise to discontinuities that prevent the construction of an equivalent LCFRS. Instead of (15) we thus obtain the annotated c-structure in (17).

(17)



Alternatively, such examples can be reanalyzed as in (18) so that inert nodes are no longer present in the c-structure, as suggested by Snijders (2016). This also ensures that the number of nodes that map to the same f-structure is finitely bounded.

(18)



There may be no limit on the number of discontinuous nominal adjuncts in Latin, but their inertness means that a *d*-bounded LFG can still account for them, contrary to Haug's supposition.

Haug also recalls Johnson's (1986) observation that the Bresnan et al. (1982) analysis of Dutch cross-serial dependencies does not extend to intransitive verb complexes without violating the nonbranching dominance constraint. In response, Zaenen and Kaplan (1995) covered both transitive and intransitive examples with a functional uncertainty solution that shortens the nonbranching chains. Haug notes that the newer formulation is less transparent, linguistically less attractive, and still less often cited than the original 1982 account. Because of this and because of apparently similar examples in Latin, Haug is willing to give up the prohibition against nonbranching dominance chains and its guarantee that recognition is decidable

for unrestricted LFG grammars, in favor of simpler nonbranching specifications. However, the simpler specifications can be implemented in the bounded-grammar framework, and WK have demonstrated that the nonbranching dominance constraint is not needed for recognition decidability and can be omitted from the restricted formalism.

5 Conclusion

Wedekind and Kaplan (to appear) have shown that there is a mildly context-sensitive grammar, an LCFRS, for every LFG grammar with the properties set forth in (9). The grammars in this class are mathematically and computationally well behaved: for languages that can be described by such grammars, the recognition problem is decidable and tractable (even without the prohibition against nonbranching dominance chains), and the emptiness and cyclic generation problems are decidable. The explanatory principles of X-bar theory admit grammars that appear not to meet the bounding conditions that WK have identified and thus seem to lie outside of the tractable class.

This may be an unintended consequence of the simple way in which the X-bar principles are typically laid out, and may in fact result in the overgeneration of strings and structures. We have provided a formal definition of a traditional linguistic notion, the degree of discontinuity for grammatical functions of a language, in terms of the c-structure to f-structure mapping of LFG, the ϕ projection. If the degree of discontinuity and the number of coheads for a language are bounded, then the derivations of an otherwise X-bar compliant grammar can be restricted to meet the conditions necessary for tractability. Notably, these bounds generally do not apply to modifier repetition because modifiers are typically isolated from their environments by set membership annotations. Thus, we suggest adding to the meta-theory of LFG these bounding requirements as a stronger replacement for the previous prohibition of nonbranching dominance chains. We conjecture that this will enable explanatory grammars that are not only descriptively accurate but also computationally tractable.

References

Berwick, Robert C. 1982. Computational complexity and Lexical-Functional Grammar. *American Journal of Computational Linguistics*, 8(3–4):97–109.

Bresnan, Joan. 2001. *Lexical-Functional Syntax*. Blackwell Publishers, Oxford.

Bresnan, Joan, Ronald M. Kaplan, Stanley Peters, and Annie Zaenen. 1982. Cross-serial dependencies in Dutch. *Linguistic Inquiry*, 13(4):613–635.

Chomsky, Noam. 1953. Systems of syntactic analysis. *Journal of Symbolic Logic*, 18(3):242–256.

Crouch, Dick, Mary Dalrymple, Ronald M. Kaplan, Tracy Holloway King, John T. Maxwell, III, and Paula S. Newman. 2008. *XLE Documentation*. Palo Alto Research Center, Palo Alto, CA.

Dalrymple, Mary. 2001. *Lexical-Functional Grammar*. Academic Press, New York.

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell, III, and Annie Zaenen, editors. 1995a. *Formal Issues in Lexical-Functional Grammar*. CSLI Publications, Stanford University.

Dalrymple, Mary, Ronald M. Kaplan, John T. Maxwell, III, and Annie Zaenen. 1995b. Non-local dependencies. In Mary Dalrymple, Ronald M. Kaplan, John T. Maxwell, III, and Annie Zaenen, editors, *Formal Issues in Lexical-Functional Grammar*. CSLI Publications, Stanford University, pages 131–135.

Haug, Dag. 2017. Syntactic discontinuities in Latin. In Victoria Rosén and Koenraad De Smedt, editors, *The Very Model of a Modern Linguist: In Honor of Helge Dyvik*. Bergen Language and Linguistic Studies, Bergen, pages 75–96.

Johnson, Mark. 1986. The LFG treatment of discontinuity and the double infinitive construction in Dutch. In *Proceedings of the Fifth West Coast Conference on Formal Linguistics*, pages 102–118, Stanford Linguistics Association, Stanford.

Kallmeyer, Laura. 2013. Linear context-free rewriting systems. *Language and Linguistics Compass*, 7(1):22–38.

Kaplan, Ronald M. and Joan Bresnan. 1982. Lexical-Functional Grammar: A formal system for grammatical representation. In Joan Bresnan, editor, *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge, Mass., pages 173–281. Reprinted in Dalrymple et al. (1995a, 98–111).

Maxwell, John T., III and Ronald M. Kaplan. 1996. Unification parsers that automatically take advantage of context freeness. In *Proceedings of the LFG'96 Conference*, CSLI publications, Stanford University.

Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science*, 88(2):191–229.

Seki, Hiroyuki, Ryuichi Nakanishi, Yuichi Kaji, Sachiko Ando, and Tadao Kasami. 1993. Parallel multiple context-free grammars, finite-state translation systems, and polynomial-time recognizable subclasses of Lexical-Functional Grammars. In *Proceedings of the 31st Annual Meeting of the Association for Computational Linguistics*, pages 130–139, Association for Computational Linguistics, Columbus, OH.

Snijders, Liselotte. 2016. An LFG account of discontinuous nominal expressions. In *Proceedings of the Workshop on Discontinuous Structures in Natural Language Processing*, pages 1–11, Association for Computational Linguistics, San Diego, California.

Wedekind, Jürgen. 2014. On the universal generation problem for unification grammars. *Computational Linguistics*, 40(3):533–538.

Wedekind, Jürgen and Ronald M. Kaplan. 2012. LFG generation by grammar specialization. *Computational Linguistics*, 38(4):867–915.

Wedekind, Jürgen and Ronald M. Kaplan. To appear. Tractable Lexical-Functional Grammar. *Computational Linguistics*.

Zaenen, Annie and Ronald M. Kaplan. 1995. Formal devices for linguistic generalizations: West Germanic word order in LFG. In Jennifer Cole, Georgia M. Green, and Jerry L. Morgan, editors, *Linguistics and Computation*. CSLI Publications, Stanford, pages 3–27. Reprinted in Dalrymple et al. (1995a, 215–239).