Syntactic discontinuities in Latin — A treebank-based study

Dag Haug

Abstract. Syntactic discontinuities are very frequent in classical Latin and yet this data was never considered in debates on how expressive grammar formalisms need to be to capture natural languages. In this paper I show with treebank data that Latin frequently displays syntactic discontinuities that cannot be captured in standard mildly context-sensitive frameworks such as Tree-Adjoining Grammars or Combinatory Categorial Grammars. I then argue that there is no principled bound on Latin discontinuities but that they display a broadly Zipfian distribution where frequency drops quickly for the more complex patterns. Lexical-Functional Grammar can capture these discontinuities in a way that closely reflects their complexity and frequency distributions.

1 Introduction

Classical Latin, like classical Greek, is famous for its tolerance of syntactic discontinuities. One example is shown in (1).

(1) *quis*$_1$ *multa*$_2$ *gracilis*$_1$ *te* *puer*$_1$ *in* *rosa*$_2$
who.NOM much.ABL slender.NOM you.ACC boy.NOM in rose.ABL
perfusus *liquidis*$_3$ *urget* *odoribus*$_3$ *grato*$_4$, *Pyrrha,*
drenched.NOM liquid.ABL press.3SG.PRES scents.ABL delightful.ABL Pyrrha
sub *antro*$_4$
in cave.ABL
'What slender boy, drenched with perfumes, presses you on a bed of roses, Pyrrha, under the delightful cave?' (Horace, Carmina 1.5)

This example features no less than four discontinuous noun phrases, as indicated with subscript indices on the words. The syntactic dependencies inside these NPs are marked with agreement in case (and number and gender, not shown in the glossing), but not with word order.

Discontinuous NPs are in fact attested in Latin up to the twentieth century, as in (2) from Dyvik (1968).

The very model of a modern linguist. Edited by Victoria Rosén and Koenraad De Smedt. BeLLS Vol. 8 (2017), DOI 10.15845/bells.v8i1.1337.

(2) *Mihi, Paulo, nullus est terror*
me.DAT Paul.DAT none.NOM is fear.NOM
'I, Paul, have no fear.'

Examples such as (2) are reminiscent of quantifier float, a type of discontinuity which is found even in highly configurational languages such as English. While (2) is in fact the only discontinuous NP in Dyvik (1968), the focus of this article is on the classical stage of Latin, where many other types of discontinuities are attested, as shown already in (1).

At the same time as Dyvik (1968) was composed, linguists discussed whether natural languages are context-free. The debate was sparked by the definition of the Chomsky hierarchy in Chomsky (1956), which raised the question whether natural languages could be described by context-free grammars. This was an open question throughout the 1960s and 70s and it was not until the 1980s that the question was settled (in the negative).[1]

The classical languages played little role in this discussion. Occasionally, some Latin examples were cited — for example, Ross (1967/1986, p. 74) used (1) to illustrate *scrambling*. There is no obvious characterization of the elements that can intervene between the different parts of the NPs in this example, and if we assume that there is no theoretical upper bound on the intervening material, it looks like it could be possible to construct an argument for the non-contextfreeness of natural language based on such examples. Of course, assuming that there is no upper bound is a leap of faith that could never be truly justified — but in that respect, Latin is not really different from English. The common claim that finite state automata cannot model center-embedding also depends on there being no theoretical upper bound on the level of embedding, and neither in English nor in Latin can we observe infinite embeddings. In fact, corpus studies suggest that the practical upper bound on levels of center embeddings is as low as three. So the main reason for using a context-free grammar to deal with center-embedding is theoretical simplicity and elegance, as pointed out by Harris (1957): "If we were to insist on a finite language, we would have to include in our grammar several highly arbitrary and numerical conditions — saying, for example, that in a given position there are not more than three occurrences of *and* between N".

Similar considerations would apply to Latin discontinuities. However, to the extent that Latin examples featured in the scholarly discussion, scholars did not object to them because their unboundedness could not be demonstrated. Instead, Pullum (1982) argued that (1) comes from the poet Horace, who "is noted for stretching tendencies in the living Latin language beyond all grammatical limits". And so no one attempted to build an argument based on classical data. Another reason for suspicion, no doubt, was the lack of hard facts concerning the extent of syntactic discontinuity in a dead

1 See Pullum (1986) for a fascinating account of the debate.

language like Latin. The Latin grammatical tradition has been content to establish that word order is 'generally free' (not just in poetry, but also in prose) and to investigate the stylistic usage of discontinuity. Moreover, word order data from Greek and Latin used not to be very accessible. As shown in Haug (2015), it used to be the case that scholars could not even agree on the frequencies of basic word orders in Ancient Greek – never mind providing an account of them.

2 So how complex is natural language really?

Research in the Generalized Phrase Structure Grammar (GPSG) framework in the early 1980s was to a large extent motivated by the desire to keep the complexity of the formalism low and develop context-free analyses of seemingly non-context free phenomena such as long distance dependencies (Gazdar 1981). That program imploded when Shieber (1985) and Culy (1985) showed that there are phenomena in natural language that cannot be captured in a context-free grammar. There were two main responses to this discovery: either one tried to extend context-free formalisms as little as possible while achieving coverage of demonstrably non-context free phenomena such as the cross-serial dependencies from Dutch and Swiss German discussed in Shieber (1985), leading to so-called mildly context-sensitive formalisms such as (Lexicalized) Tree Adjoining Grammar (LTAG) and Combinatory Categorial Grammar (CCG); or one gave up (almost) completely on the concern about weak generative capacity, as in Lexical Functional Grammar (LFG) and Head Driven Phrase Structure Grammar (HPSG). A natural question to ask, then, is "Who was right?". Is it possible to keep the algorithmic complexity of the parsing problem low while maintaining good coverage of the data, as measured in modern treebanks?

Answering that question requires a detour into dependency grammar, since most treebanks these days – and in particular the Latin ones that we will look at here – are based on dependencies rather than phrase structures or CCG derivations. Fortunately, there are formal results that relate the complexity of formalisms like CFGs, LTAGs and CCGs to that of dependency grammars with various restrictions on non-projective (i.e. discontinuous) dependencies.

2.1 Measuring discontinuity in a dependency treebank

In order to study discontinuities in dependency trees, we need to introduce some terminology. The *projection* of a node in a dependency tree is its yield, i.e. the set of nodes in the transitive, reflexive closure of dominance, arranged in linear order. A *gap* is a discontinuity in a projection, and the *gap degree* of a node is the number of gaps in its projection. An equivalent measure is the block degree, i.e. the number of continuous *blocks* in the projection of a node, which will always be the gap degree + 1. Consider the dependency tree in Figure 1. The gap degree of *mihi* is 0, for its projection [*mihi, Paulo*] is uninterrupted. By contrast, the gap degree of *terror* is 1, for its projection [*nullus, terror*] is interrupted by *est*. Alternatively, we may say that *terror* has block

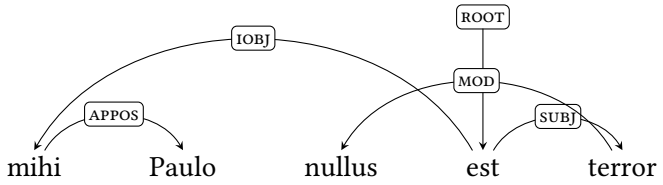

Figure 1: Dependency tree for (2)

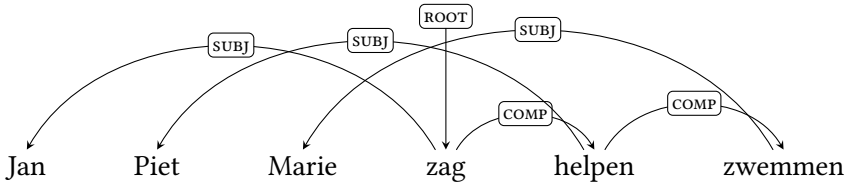

Figure 2: Dependency tree for (4)

degree 2, for it consists of the two blocks [*nullus*] and [*terror*]. Finally, we note that we may also talk about the gap degree of a dependency tree, which is defined as the highest gap degree among its nodes.

For our purposes, it will also be useful to study gap *depth*, which we define as in (3).

(3) A node d in the projection of r *introduces a discontinuity* in r iff d is in a different block b from r and there is no node in b that dominates d. The depth of the gap introduced by d is the number of edges between d and r. The *gap depth* of r is the maximum depth of a node that introduces a discontinuity in r.

In Figure 1, the gap depth of *terror* is 1, as the discontinuity is introduced by its direct dependent *nullus*. Let us now look at a deeper gap in a classic example of cross-serial dependencies in Dutch.

(4) *(…dat) Jan Piet Marie zag helpen zwemmen*
that Jan Piet Marie see-PST help.INF swim.INF
'(…that) Jan saw Piet help Marie swim.'

The nodes *helpen* and *zwemmen* both have gap degree 1. The projection of *helpen* has the two blocks {*Piet, Marie*} and {*helpen, zwemmen*}. Both *Piet* and *Marie* introduce discontinuities in the projection of *helpen*, since neither dominates the other. The depth

of those discontinuities are 1 and 2 respectively and hence the gap depth of *helpen* is 2. Thus the gap depth captures the fact that not only is *helpen* discontinuous, but it also dominates a discontinuous dependent without resolving the discontinuity. Intuitively, then, gap depth captures embedding of discontinuities e.g. in long-distance extraction, which is generally thought to be associated with human processing difficulty (see e.g. Gibson 2000). Gap depth has to my knowledge never been considered in measures of non-projectivity in dependency treebanks such as Kuhlmann and Nivre (2006), Havelka (2007) or Maier and Lichte (2011), but we will see that corpus evidence suggests this measure is useful.

2.2 Dependency structures and other grammatical formalisms

While most modern treebanks are based on dependencies, most grammatical theories are not. One early and fairly well-known result connecting dependency grammar to other grammatical formalisms is due to Gaifman (1965) and shows that projective dependency grammars, i.e. dependency grammars that allow no discontinuities, are weakly equivalent to context-free grammars. However, since the focus here is precisely on discontinuities, that result is of little value for us.

Multiple context-free grammars (MCFGs), also known as linear context-free rewriting systems, have emerged as a powerful tool to study complexity questions in the range of the Chomsky hierarchy between context-free grammars and full-blown context-sensitive grammars. Kuhlmann (2013) has established connections between dependency grammars and MCFGs which yield a close correspondence between the non-projectivity of the dependency trees admitted by a grammar on the one hand, and the parsing complexity of the grammar on the other. In the following, we briefly review these results as a background for what follows.

The MCFG formalism is a generalization of CFG which retains ordinary CFG productions for the expression of categorial structure, but uses explicit *yield functions* to compute the yield of the mother node from the yields of the daughters. In an ordinary CFG, yield computation is conflated with category formation: a rule such as DP $\rightarrow$ D NP says both that the category DP is formed of a D and an NP, and that the yield of the resulting DP is formed by concatenating the yields of D and NP. In effect, then, a CFG can be seen as an MCFG with concatenation as the only yield function.[2]

To allow for greater expressivity, MCFG allows yields to be *tuples* of strings. For example, we may want to say that the yield of DP is a pair (2-tuple) consisting of the yields of D and NP. This pair will then be the input to further yield functions that apply to productions with DP on the right-hand side. More generally, we may allow yields to be n-tuples of strings.

For our purposes, it is important to note that there is a close correspondence between yield components in an MCFG and blocks in a corresponding dependency structure. We can extract MCFG rules from dependency trees, as shown in Kuhlmann (2013),

2 See Clark (2014) for an accessible introduction for linguists.

APPOS $\rightarrow f()$	$f = \langle \textit{Paulo} \rangle$
MOD $\rightarrow g()$	$g = \langle \textit{nullus} \rangle$
IOBJ $\rightarrow h($APPOS$)$	$h = \langle \textit{mihi}\ x_1 \rangle$
SUBJ $\rightarrow i($MOD$)$	$i = \langle x_1, \textit{terror} \rangle$
ROOT $\rightarrow j($IOBJ SUBJ$)$	$j = \langle x_1 y_1\ \textit{est}\ y_2 \rangle$

Table 1: Rules extracted from the tree in Figure 1

where a formal exposition is given. Here I just provide an intuitive understanding of how the tree in Figure 1 gives rise to the rules in Table 1.

Looking at *Paulo* in Figure 1 we see that it has no dependents, hence the right-hand side of the first rule is a constant function which fixes the yield to the string *Paulo*, and similarly for *nullus*. For *mihi*, things are a bit more interesting: it takes an APPOS argument, and hence its yield depends on the yield of that argument. Concretely, the yield of the node *mihi* is computed by concatenating the string *mihi* with the yield of the APPOS argument, which is represented with x_1 according to the convention that we use x for the yield of the first argument and y for the yield of the second argument, and subscript those variables with an index referring to components of the yield. In this case, the yield of APPOS has only one component, so we use x_1. Also *terror* takes an argument, a MOD, but in this case, the resulting yield has two components, one consisting of the yield of the MOD and one consisting of the string *terror*. Finally, the verb takes two arguments, SUBJ and IOBJ. The yield is constructed by concatenating the yield of the IOBJ(i.e. x_1), the first component of the SUBJ(i.e. y_1), the string *est*, and the second component of SUBJ(y_2).

For our purposes, the primary interest of this construction lies in the fact that it provides a link between dependency treebanks and the required expressivity of corresponding grammars, as investigated in Kuhlmann (2013). On the one hand, the yield components correspond directly to blocks found in the treebanks. And on the other hand, the complexity of an MCFG grammar is easily read off the yield functions: The parsing complexity of a yield function equals the sum of the number of components in its input and output yields. For example, the parsing complexity of j in Table 1 is 4, as its two inputs have 1 and 2 component yields and it produces a 1 component yield. This yields a two-dimensional complexity hierarchy, as the complexity depends both on the number of arguments and the number of yield components of these arguments. In the presence of only *wellnested* discontinuities, we actually get a simple complexity hierarchy because any wellnested MCFG can be binarized without increasing the gap degree. A wellnested discontinuity is one whose projection does not interleave with

another non-overlapping projection.[3] (5) gives an example of an illnested discontinuity from Latin poetry.

(5) *aurea* *purpuream* *subnectit* *fibula* *vestem*
golden.NOM purple.ACC bound clasp.NOM cloak.ACC
'a golden clasp bound her purple cloak' (Vergil, Aeneid 4.139)

The projections of the subject and the object do not overlap (neither dominates the other), but they interleave, producing an illnested discontinuity. MCFGs that generate only wellnested dependencies are called wellnested MCFGs. Since they can be binarized without increasing the gap degree, their parsing complexity is uniquely determined by their gap degree. We refer to an MCFG where no argument has more than k components in its yield as a k-MCFG.

There are several results linking linguistically motivated grammatical formalisms to MCFGs. For example, TAG is weakly equivalent to a wellnested 2-MCFG. The same result applies to 'classical' CCG and linear indexed grammars (Aho 1968), since those formalisms are weakly equivalent to TAG. However, modern lexicalized CCG (i.e. the current version where (restrictions on) the combinators are not grammar-specific but all linguistic variation is captured in the lexicon) is known to be strictly less powerful than TAG (Kuhlmann, Koller, et al. 2015).

The equivalence between wellnested 2-MCFGs and established grammatical formalisms takes on significance in the light of empirical investigations on dependency treebanks. For example, Kuhlmann (2013) shows that by restricting ourselves to wellnested trees of gap degree at most 1, i.e. trees describable by a wellnested 2-MCFG, we lose only between 0.1% (Arabic) and 0.9% (Turkish) of the trees in the CoNLL 2006 treebanks. This suggests that formalisms with the power of TAG are adequate for natural languages. Similar results have been reported by others and will also be shown below for the Universal Dependencies treebanks. But we will also see that Latin behaves in a crucially different way.

2.3 Complexity in LFG

Any LFG grammar that determines an upper bound n on the number of c-structure nodes corresponding to a given f-structure (a so-called 'finite copy LFG') can be translated into a weakly equivalent MCFG. This gives us polynomial time parsing, because parsing with a (wellnested) k-MCFG can be done in time $\mathcal{O}(n^{3k})$. But in the general case, parsing with an LFG grammar is NP-complete, as can be shown with a straightforward reduction from the 3SAT problem, i.e. the problem of determining whether a formula of propositional logic in conjunctive normal form where each clause is limited to at most three literals is satisfiable: we use c-structure rules to make sure each clause

3 For a formal definition, see e.g. Kuhlmann (2013, p. 377).

contains at least one true literal and use the f-structure to keep track of the assignment of truth values across clauses.[4]

It is worth pointing out that the universal recognition problem for MCFGs is also NP-complete because, although any given MCFG is a k-MCFG, MCFG as a formalism does not bound that k. Put in other words, the difference between MCFGs and LFGs is that for any given (finite) instance of the 3SAT problem with n clauses, we can construct an MCFG that solves it, whereas we can write a general LFG that can solve any instance of the 3SAT problem.

If we think of the relations between different instances of the same literals in a 3SAT problem as analogues to discontinuous dependencies in linguistics, this means an LFG grammar can deal with an unbounded number of discontinuous dependencies across unbounded distances. We can ask ourselves whether there is any need for the expressivity that LFG gives us. As we will see in section 3.1, the answer is from one point of view negative: we can get extremely good coverage on existing dependency treebanks with a relatively low bound on the discontinuities. Nevertheless, it is worth making the point that the extra expressivity provides for extra linguistic insight. We will now show that this point holds even as we move up the complexity ladder from NP-complete to undecidable.

Undecidability was not a property of LFG originally. While unification grammars in general are Turing-equivalent and hence have an undecidable parsing problem, Kaplan and Bresnan (1982) avoided undecidability by restricting valid derivations as in (6).

(6) A c-structure derivation is valid if and only if no category appears twice in a nonbranching dominance chain, no nonterminal exhaustively dominates an optionality ϵ, and at least one lexical item or controlled e appears between two optionality ϵ's derived by the same rule element.

By disallowing nonbranching dominance chains, this constraint ensures that for any string the size and number of c-structure derivations is bounded as a function of the length of the string. The constraint seems well-motivated: after all, what could be the linguistic motivation for derivations in which e.g. some NP dominates another NP in a nonbranching structure?

As it turns out, such structures *can* be motivated. In Bresnan, Kaplan, et al. (1982), it was argued that cross-serial dependencies in Dutch cannot be given a linguistically motivated analysis in a context-free grammar. Instead, the authors proposed to give the sentence in (4) the c-structure in Figure 3.

This c-structure does not directly capture the object relation between *Piet* and *zag* or between *Marie* and *helpen*. Instead, the relationship is captured with functional annotations on the VP and $\bar{\text{V}}$ nodes which 'match up' the two branches in the f-structure and give the correct grammatical relations. So, *Marie* is the object of *helpen* by virtue of

4 See for example Francez and Wintner (2012, pp. 241–243) for details of the construction.

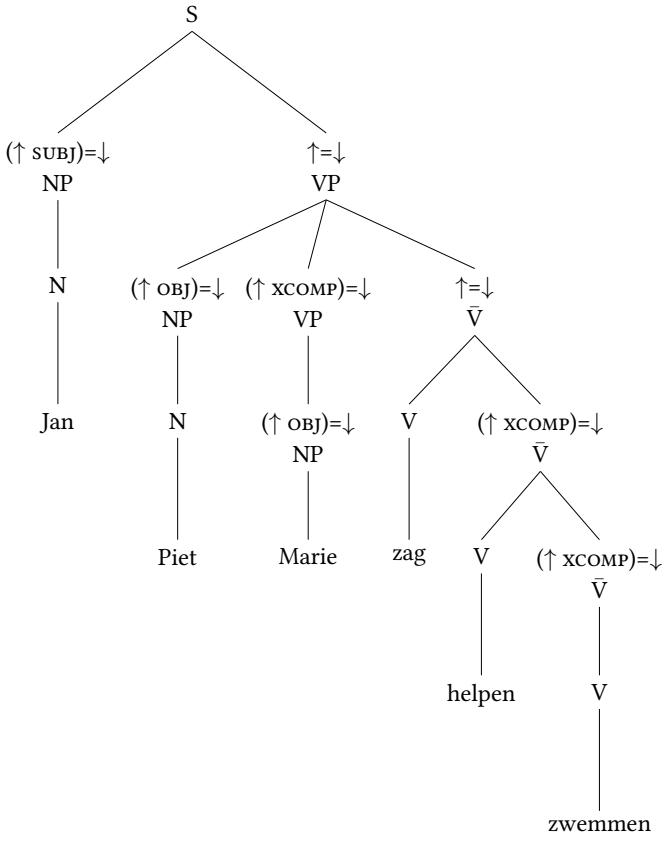

Figure 3: C-structure of (4)

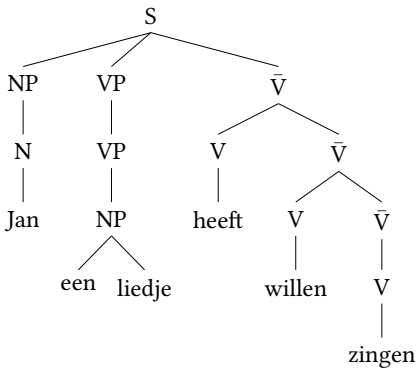

Figure 4: C-structure of (7)

being embedded under the same number of VP nodes as *helpen* is under V̄ nodes. This kind of analysis is based on what Maxwell and Kaplan (1996) call 'zipper unification'.

However, as pointed out by Johnson (1986), this analysis actually leads to non-branching dominance chains in cases where intermediate verbs in the structure are intransitive, as in (7) with the c-structure in Figure 4.

(7) *(...dat) Jan een liedje heeft willen zingen*
that Jan a song has want.INF sing.INF
'(...that) Jan has wanted to sing a song.'

So, if we want to keep the analysis from Bresnan, Kaplan, et al. (1982) we must give up the offline parsability constraint and hence the decidability of the LFG formalism. On the other hand, an alternative analysis was also proposed (Zaenen and Kaplan 1995), where NPs inside VP get the functional uncertainty annotation $(\uparrow \text{XCOMP}^* \text{OBJ}) = \downarrow$, rather than just $(\uparrow \text{OBJ}) = \downarrow$. From a linguistic point of view, there are several problems with this analysis: First, it is unclear how we can ever provide a principled structure-function mapping if we allow non-local GF assignments like this. And second, in order to capture the word order facts, we need complex f-precedence constraints.

And in fact, what happened in this case is that the analysis of Bresnan, Kaplan, et al. (1982) is still well-known and cited, whereas the alternative analysis based on non-local GF assignment and functional precedence is more or less forgotten. Both the first and the second edition of Bresnan's LFG textbook (Bresnan 2001; Bresnan, Asudeh, et al. 2015) include exercises that ask the student to reproduce Bresnan, Kaplan, et al. (1982) — even if a generalization of this analysis to intransitive verbs (not used in the exercise) would not even be LFG as defined in Kaplan and Bresnan (1982). In other

words, while 'intuitive' is a subjective notion, history lends some justification to the claim that the original analysis is more intuitive than the later one.

There are some lessons we can draw from this. First, the ban on nonbranching dominance chains looks stipulative: it can be removed from the definition of LFG without changing anything else, albeit at the cost of undecidability. An analysis like that in Figures 3 and 4 does not 'feel' substantially un-LFG-like. Second, the original analysis seems linguistically more informative than the alternative in that it captures the word order generalizations in an intuitive way while preserving locality of GF assignment. Again, this is subjective, but the fact that the analysis gets cited and is used in textbooks shows that the intuition is widespread.

Taken together, these two observations suggest that a more expressive grammatical formalism can lead to more linguistically adequate analyses — even if those analyses do not actually exploit that expressivity in a crucial way. In our case, the problem with unary branching dominance chains is that there will be no upper bound on the length of the unary VP chain in Figure 4. But chains of unbounded length are of course not crucial to the analysis. We only need VP–VP chains of a length corresponding to the number of consecutive intransitive verbs in the $\bar{\text{V}}$-chain. For practical purposes, 5 will be more than sufficient. And even from a theoretical perspective, it is not clear that banning any category α from dominating five instances of α in a nonbranching dominance chain is any more objectionable than banning it from dominating a single instance, as Bresnan and Kaplan did with (6).

3 Empirical investigation

3.1 Quantitative data

Let us now have a look at how discontinuities actually distribute in Latin treebanks. To be able to compare across languages we use the Universal Dependencies (UD) corpora,[5] in particular the version 2 release. This dataset contains three Latin treebanks, the Perseus treebank (Bamman and Crane 2011), the PROIEL treebank (Haug and Jøhndal 2008) and the Index Thomisticus Treebank (Martens and Passarotti 2014).

Table 2 shows the distribution gap degree and depth across all languages in the UD corpora.[6] As we can see, the vast majority of edges, 97.3%, are projective. Still, this means that the number of non-projective edges is high enough that we need to be able to deal with them in parsing. But at least from a practical standpoint, we can ignore everything but the simplest type of gap: restricting ourselves to edges of gap degree and depth ≤ 1 yields a coverage of 99.7%.

When we get to Latin, the picture is different. First of all, the number of simple (degree 1) non-projectivities is much higher: 9.1% of edges. More interesting is the fact

5 http://universaldependencies.org/

6 There are a few outliers of degree or depth > 3 that are not shown in the tables.

Gap degree	Gap depth			
	0	1	2	3
0	2572961 (97.3%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
1	0 (0.0%)	63729 (2.4%)	4856 (0.2%)	616 (0.0%)
2	0 (0.0%)	1584 (0.1%)	1095 (0.0%)	165 (0.0%)
3	0 (0.0%)	44 (0.0%)	56 (0.0%)	25 (0.0%)

Table 2: Gap degree and depth in the UD 2.0 corpora

Gap degree	Gap depth			
	0	1	2	3
0	60430 (90.4%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
1	0 (0.0%)	5556 (8.3%)	496 (0.7%)	61 (0.1%)
2	0 (0.0%)	134 (0.2%)	123 (0.2%)	17 (0.0%)
3	0 (0.0%)	5 (0.0%)	5 (0.0%)	1 (0.0%)
4	0 (0.0%)	0 (0.0%)	1 (0.0%)	0 (0.0%)
9	0 (0.0%)	0 (0.0%)	1 (0.0%)	1 (0.0%)

Table 3: Gap degree and depth in the UD 2.0 Latin-PROIEL treebank

that 0.4% of edges have gap degree 2 and thus reflect dependencies that cannot be captured in a TAG (or a forteriori, in a CCG).

This becomes clearer if we think about tree coverage, as shown in Table 4 for a select number of treebanks.[7] Here we see that by restricting ourselves to trees where the highest gap degree is 1, we lose 1.8% of the trees in the PROIEL treebank, compared to zero loss in the Norwegian Bokmål treebank and 0.3% loss in the Czech treebank. Overall in the UD treebanks, 0.6% of trees contain an edge of gap degree 2, but it is worth pointing out that almost three quarters of these trees are found in one of the Ancient Greek and Latin treebanks, which only make up roughly a tenth of the trees. So there clearly is something special about these languages.

Finally, we look at the illnestedness numbers in Table 5. As has been observed several times in the literature, illnestedness is a strong constraint on discontinuities in most languages. We see that this constraint is strong also in the PROIEL corpus of Latin (and Greek), but not in the Perseus corpora. As with non-projective dependencies in general, this is likely due to due to the large portions of poetry in this treebank. In (5) we saw an example of an illnested dependency from Vergil. And this was in fact no accident,

7 What I have listed as Anc.Gr.-Perseus and Latin-Perseus appear in UD as simply Ancient Greek and Latin, since they were the first treebanks for these languages. These treebanks generally have a higher degree of non-projectivity because they consist mostly of poetry.

	0	1	2	3
Anc.Gr.-Perseus	4738 (37.6%)	6983 (55.4%)	833 (6.6%)	46 (0.4%)
Anc.Gr.-PROIEL	9823 (61.9%)	5515 (34.8%)	493 (3.1%)	34 (0.2%)
Czech	3480 (87.9%)	468 (11.8%)	11 (0.3%)	0 (0.0%)
Latin-Perseus	793 (59.5%)	511 (38.3%)	27 (2.0%)	2 (0.2%)
Latin-ITTB	10367 (62.9%)	5805 (35.2%)	310 (1.9%)	9 (0.1%)
Latin-PROIEL	11213 (73.2%)	3844 (25.1%)	254 (1.7%)	11 (0.1%)
Norw.-Bokmaal	1173 (92.5%)	95 (7.5%)	0 (0.0%)	0 (0.0%)
All treebanks	353194 (87.2%)	49151 (12.1%)	2572 (0.6%)	121 (0.00%)

Table 4: Trees by gap degree in selected UD treebanks

	Illnested	Wellnested
Ancient_Greek-Perseus	1.5%	98.5%
Ancient_Greek-PROIEL	0.3%	99.7%
Czech	0.1%	99.9%
Latin-Perseus	3.8%	96.2%
Latin-ITTB	0.2%	99.8%
Latin-PROIEL	0.2%	99.8%
Norwegian-Bokmaal	0.1%	99.9%
Sum	0.1%	99.9%

Table 5: Wellnestedness

but a so-called 'golden line', a rhetorical pattern first discovered by Edward Burles in 1652: "If the Verse does consist of two Adjectives, two Substantives and a Verb only, the first Adjective agreeing with the first Substantive, the second with the second, and the Verb placed in the midst, it is called a Golden Verse." It is not clear whether Latin poets in fact preferred illnested dependencies for their own sake, or whether their frequency results from other, conspiring factors. Whatever the motivation, it is interesting that the poets regularly produced these illnested structures which are so rare in prose.

The numbers reported in this section are based on data converted to Universal Dependencies. To my knowledge there is no in-depth study of discontinuity based on the original Perseus or PROIEL data for Latin, but there is a study on Greek (Mambrini and Passarotti 2013), which finds only 25.2% projective trees, compared to 37.6% in the UD version of the same treebank. It should be noted that the UD conversion only includes a subset of the original treebank due to conversion problems. One possibility is that the conversion script was particularly likely to fail on non-projective structures,

which would explain why the projectivity rate is higher in the converted UD data. The illnestedness degree is also lower, at 1.5% versus 2.6% in the original version.

3.2 Examples

Let us now have a closer look at some examples of the discontinuities we find in Latin. An important first observation is that a large number of them arise from second-position clitics, which normally appear after the first prosodic word, even if that breaks up a syntactic constituent. The frequency of this phenomenon contributes to the number of gap degree 2 trees, since it is then enough to have one other gap resulting from some other process. An example of this is (8).

(8) *eo* *autem* *die* *credo* *aliquid* *actum* *in* *senatu*
this.ABL but day.ABL I.believe something.ACC done.ACC in senate.ABL
'But I believe that something will be done in the senate today.' (Cic. Att. 5.5.1)

In this case, we have a normal long distance dependency, where *eo die* has been displaced out the embedded clause *aliquid actum in senatu*, resulting in one gap. When the clitic then lands inside the fronted constituent, we get a second gap. Such examples are controversial as illustrations of the syntactic complexity of a language, since it is not clear to what extent clitic positioning in Latin is syntactically conditioned: prosodic factors are clearly also important. From a parsing perspective, however, we need to have some way of dealing with clitics, so a more reasonable objection may be that the set of clitic strings is finite, i.e. there is only a finite number clitics and licit combinations of clitics that can occur in the position of *autem* in (8). Therefore, we can deal with them without using the full power of a formalism that can derive syntactic discontinuities.[8]

However, trees of gap degree 2 are by no means restricted to those where clitics account for one of the gaps. Example (9) shows a discontinuous NP *multa …genera ferarum*, with an extraposed relative clause.

(9) *Multa* *que* *in* *ea* *genera* *ferarum* *nasci* *constat*
many.ACC and in it.ABL kinds.ACC beasts.GEN be born it.is.certain
quae *reliquis* *in* *locis* *visa* *non* *sint*
which.NOM other.ABL in places.ABL seen.NOM not are.SBJV
'It is certain that many kinds of beasts are born in it which have not been seen in other places' (Caes. Gal. 6.25.5)

(10) shows another example, where we get gap degree 2 because the genitive is displaced from its head noun at the same time as the wh-word *quantam* is fronted alone.

8 An approach based on MCFGs can still be more perspicuous and insightful from a linguistic point of view, see Goldstein and Haug (2016). Even so, it is likely that such a grammar could be 'compiled' to a computationally more tractable grammar by exploiting the finiteness of the set of clitic strings.

(10) *quantam* *porro* *mihi* *exspectationem* *dedisti* *convivi*
how large.ACC besides me.DAT expectation.ACC give.2S.PRF this.GEN
istius *ἀσελγοῦς*
guest.GEN wanton.GEN
'Besides, how large expectations you gave me about this wanton guest!' (Cic. Att. 2.12.2)

Example (11) shows another common pattern, where one of the gaps results from a 'postponed' coordination.

(11) *Munitis* *castris* *duas* *ibi* *legiones* *reliquit* *et*
fortified.ABL camp.ABL two.ACC there legions.ACC left.3S.PRF and
partem *auxiliorum*
part.ACC auxiliaries.GEN
'With the camp fortified, he left two legions and a part of the auxiliaries there.' (Caes. Gal. 1.49.4)

Finally, since gap degree 2 examples arise naturally, even without clitics, there are examples where a clitic intrudes in an otherwise degree 2 discontinuity, yielding gap degree 3. And there are a few gap 3 examples without clitics. We refrain from showing examples here, as they inevitably get quite complex.

When it comes to illnestedness, we saw in the previous section that examples are extremely rare in the PROIEL treebank. Nevertheless, it is worth pointing out that the ones that do occur look perfectly 'natural', in the sense that it is hard to come up with alternative analyses that make linguistic sense and capture the sentence structure without an illnested dependency. (12) shows an example where the subject appears inside the object NP, at the same time as there is an extraposed relative clause belonging to it.

(12) *Magnam* *Caesarem* *iniuriam* *facere* *qui* *suo adventu* *…*
great.ACC Caesar.ACC injustice.ACC do who.NOM by his arrival
'(He said that) Caesar was doing a great injustice, who by his arrival …' (Caes. Gal. 1.36.4)

Taken together with the metrical data discussed in section 3.1, this suggests that illnestedness is not ungrammatical in Latin, although it clearly is strongly dispreferred (in prose).

4 So how complex is Latin really?

We can clearly conclude that Latin is not a tree-adjoining language. As Table 4 shows, there are simply too many trees of gap degree > 1, and examples such as (8)–(11) show that these arise through combinations of well-established processes of Latin grammar.

However, Table 4 also shows that we do not really need the ability of LFG to transport unbounded amounts of features across unbounded distances in the tree to capture the data found in Latin treebanks: Trees of gap degree 3 are already very rare. Nevertheless they arise through well-defined grammatical processes (and are not, for example, artefacts of the annotation scheme). It is therefore impossible to define a theoretical upper bound on gap degree in Latin.

In a way, the situation is analogous to what we see in center-embedded recursion. We can deal with finite levels of center-embedded recursion in a regular (finite state) grammar by adding states to the automaton. And center-embedding of more than three levels turns out to be nonexistent in corpora (Karlsson 2007), so for practical purposes, the finite state approach could work. But linguists prefer context-free grammars both because it is hard or impossible to define a theoretical upper bound on the levels of center-embedding and, crucially, because analyses cast in terms of a CFG are linguistically more perspicuous. A similar argument applies, I contend, to syntactic discontinuities: although we could deal with them in practical terms – at least when we confine the attention to the texts in the existing Latin treebanks – by adopting a k-MCFG as our formalism for some (quite small) k, it is hard to argue theoretically for any particular k and – as we have already seen – analyses that are cast in more expressive formalisms can turn out to be more intuitive. In other words, we can adapt Harris' argument for assuming infinite levels of center embeddings to unbounded discontinuous dependencies: fixing a k is a "highly arbitrary and numerical condition" that has no place in linguistic theory. In that respect, 2 – the number that (restricted to wellnested dependencies) would give us the expressive power of TAG or classical CCG – is no different from any other number.

This gives us an argument for adopting LFG as a formalism even if that is expressive overkill in practical terms.[9] And although LFG does not provide an obvious way of restricting discontinuities, we will see that it does provide a way of analyzing them that gives us a natural metric for discontinuity complexity in the form of the number of reentrancies they require. Consider first the mock Latin sentence in (13).

(13) *Maximilianus trusit bonum Fredericum*
Max.NOM pushed good.ACC Fredrick
Maximilian pushed good Fredrik

In LFG terms, this can be analyzed with the c- and f-structure in Figure 5. A characteristic feature of this is that the ϕ mapping from maximal projections (S and NP) is injective: there are no reentrancies, i.e. distinct maximal projections mapping to the same f-structure.

9 There may be an intermediate formalism available: As shown by Kallmeyer and Satta (2009), Tree-Tuple Multiple Component TAGs (TT-MCTAGs) can describe German scrambling and have a polynomial parsing algorithm.

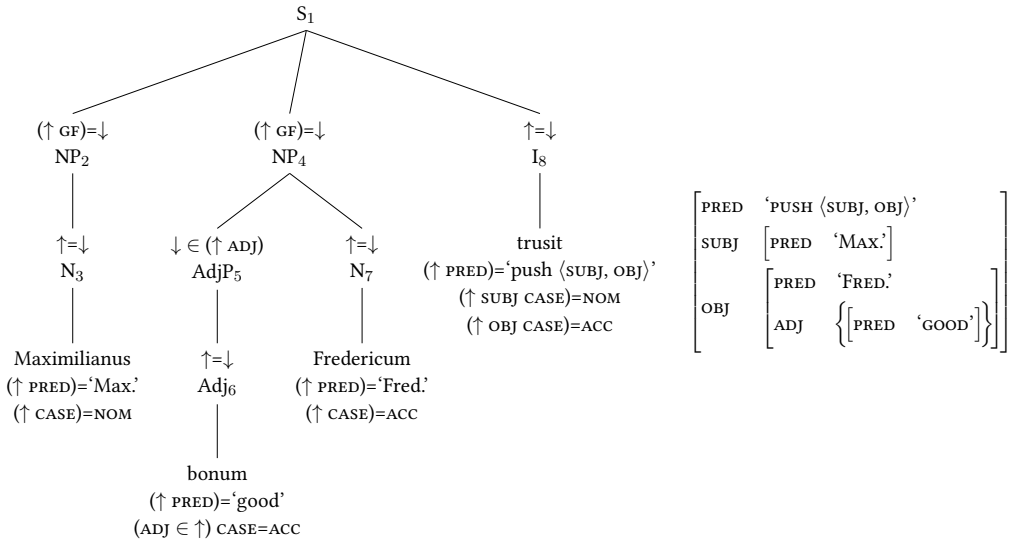

Figure 5: C- and f-structure for (13)

Now consider what happens if we permute *trusit* and *bonum* to yield a discontinuous c-structure. If we want to avoid non-local assignment of grammatical functions, the obvious way to achieve this is by using a c-structure embedding as in Figure 6. This introduces a reentrancy: for this c-structure to yield the correct f-structure (namely the same as in Figure 5), we must make sure that NP_4 and NP_8 map to the same f-structure, i.e. GF on both these nodes must be resolved to the same grammatical function.[10] In other words, the syntactic discontinuity is mirrored by structural complexity in the form of a reentrancy. Obviously, if we had yet another discontinuous dependent of *Fredericum* that was discontinuous from *bonum* so that we had a gap degree 2 discontinuity, we would need another reentrancy to capture that.

Now observe what happens if we have a deeper gap as in (14). This yields the c-structure in Figure 7. We observe that the extra depth of the discontinuity yields an extra reentrancy as compared with the otherwise similar discontinuity in Figure 6, for to get the correct f-structure from Figure 7, we must map both NP_4 and NP_5 to the same f-structures as NP_9 and NP_{10} respectively.

(14) *Maximilianus* *boni* *trusit* *Frederici* *filium*
Max.NOM good.GEN pushed Fredrick.GEN son.ACC
Maximilian pushed good Fredrick's son

10 In this particular example, case agreement will ensure that.

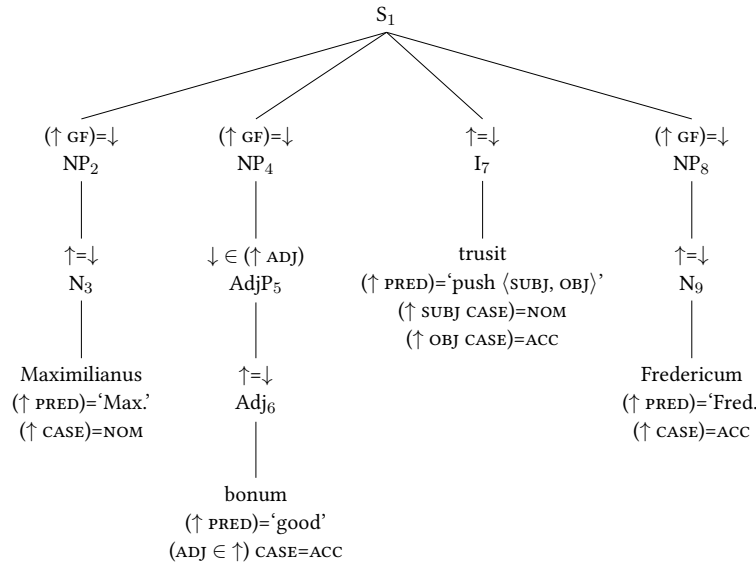

Figure 6: C-structure for permuted version of (13)

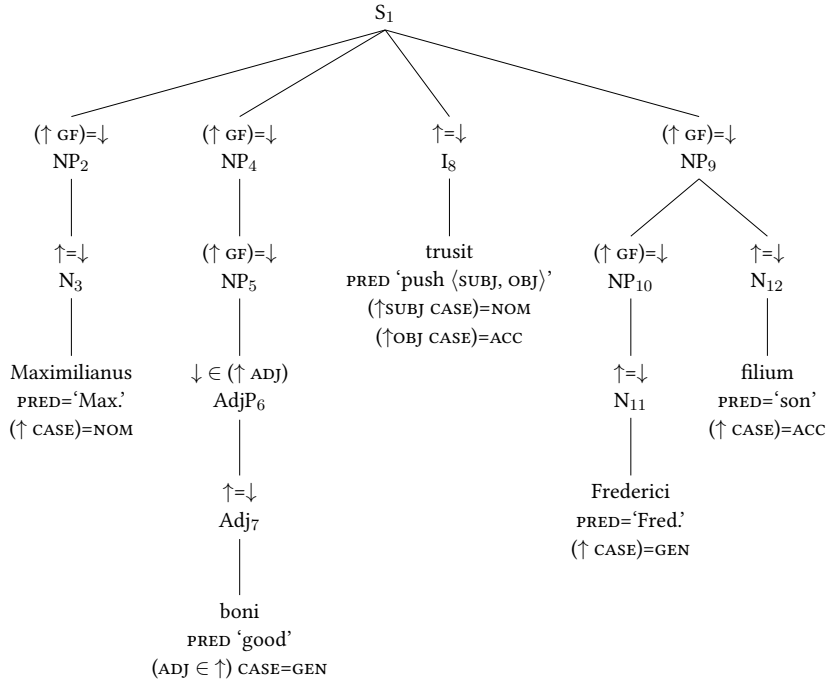

Figure 7: C-structure for (14)

The c-structure in Figure 7 clearly violates the principle in (6). Nevertheless, it is an attractive analysis compared with, say, an analysis where AdjP_6 would be directly embedded under S and annotated with $(\uparrow \text{GF}^+)$ because that would introduce non-local constraints and because Figure 7 wears its complexity on its sleeves, in the form of the length of the unit branch under NP_4. Like the analysis of Dutch cross-serial dependencies, this relies on zipper unification. As pointed out by Maxwell and Kaplan (1996, p. 24), zippers introduce computational complexity because they mean that depth of f-structures that must be unified can grow as a function of the length of the sentence. (In fact, because we allow a cyclic unit branch, it can grow even without the sentence increasing in length.) But this is really a practical problem and as such it allows a practical solution, namely a brute force bound on the length of zippers. And that is where the treebank data become interesting, for they suggest that this bound can be set quite low.

5 Conclusion and challenge

In sum, we have seen that extant Latin treebanks display syntactic discontinuities that require us to go beyond the capacity of well-known mildly context-sensitive grammar formalisms such as CCG and TAG. It has already been argued on theoretical grounds that these formalisms cannot capture data such as German scrambling (Becker et al. 1992). But as pointed out by Kuhlmann (2013), formalisms (weakly) equivalent to TAG still have very good coverage on treebanks. That, however, is not the case in Latin (and still less so in Ancient Greek), thereby verifying the inadequacy of TAG on actual treebank data.

From a theoretical point of view, this means that trees of gap degree 1 have no particular theoretical importance. Rather, the corpus data suggests that gap degrees (and depths) have a Zipfian distribution that quickly decreases beyond 1. So there is no theoretical reason to stay with k-MCFGs. And in fact we have seen that although LFG parsing is intractable, the formalism reflects the complexity of syntactic discontinuities in a rather nice and intuitive way, paving the way for empirical studies on how much of the theoretically desired expressivity is actually needed for practical purposes.

One challenge remains: we have seen that illnested discontinuities are strongly dispreferred in most treebanks, with an exception for Latin poetry. But unlike gap degree and depth, illnestedness does not correspond to any complexity in the LFG formalism. In other words, LFG as it currently stands lacks the theoretical resources to express the strong dispreference that we observe in corpora.

References

Bamman, David and Gregory Crane (2011). "The ancient Greek and Latin dependency treebanks". In: *Language Technology for Cultural Heritage: Selected Papers from the LaTeCH Workshop Series*. Berlin: Springer Verlag, pp. 79–98.

Becker, Tilman, Owen Rambow, and Michael Niv (1992). *The derivational generative power of formal systems or scrambling is beyond LCFRS*. IRCS Report 92-38. Institute for Research in Cognitive Science, University of Pennsylvania.

Bresnan, Joan (2001). *Lexical-Functional Syntax*. Blackwell Textbooks in Linguistics 16. Oxford: Blackwell.

Bresnan, Joan, Ash Asudeh, Ida Toivonen, and Stephen Wechsler (2015). *Lexical-Functional Syntax*. Malden: John Wiley & Sons.

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

Chomsky, Noam (1956). "Three models for the description of language". In: *IRE Transactions on Information Theory* 2.3, pp. 113–124.

Clark, Alexander (2014). "An introduction to multiple context free grammars for linguists". http://www.cs.rhul.ac.uk/home/alexc/lot2012/mcfgsforlinguists.pdf.

Culy, Christopher (1985). "The complexity of the vocabulary of Bambara". In: *Linguistics and Philosophy* 8.3, pp. 345–351. DOI: 10.1007/BF00630918.

Dyvik, Helge (1968). "Pauli gallinae". Available from http://folk.uib.no/hfohd/SLF/Dyvik/gallinae/gallinae.html.

Francez, Nissim and Shuly Wintner (2012). *Unification Grammars*. Cambridge: Cambridge University Press.

Gaifman, Haim (1965). "Dependency systems and phrase-structure systems". In: *Information and Control* 8.3, pp. 304–337.

Gazdar, Gerald (1981). "Unbounded dependencies and coordinate structure". In: *The Formal Complexity of Natural Language*. Ed. by W.J. Savitch, E. Bach, W.E. Marsh, and G. Safran-Naveh. Springer, pp. 183–226.

Gibson, Edward (2000). "The dependency locality theory: A distance-based theory of linguistic complexity". In: *Image, Language, Brain*. Ed. by Alec P. Marantz, Yasushi Miyashita, and Wayne O'Neil. Cambridge, MA: MIT Press, pp. 95–126.

Goldstein, David M. and Dag T. T. Haug (2016). "Second-position clitics and the syntax-phonology interface: The case of ancient Greek". In: *Proceedings of the Joint 2016 Conference on Head-driven Phrase Structure Grammar and Lexical Functional Grammar, Polish Academy of Sciences, Warsaw, Poland*. Ed. by Doug Arnold, Miriam Butt, Berthold Crysmann, Tracy Holloway King, and Stefan Müller. Stanford, CA: CSLI Publications, pp. 297–317.

Harris, Zellig S. (1957). "Co-occurrence and transformation in linguistic structure". In: *Language* 33.3, pp. 283–340.

Haug, Dag T. T. (2015). "Treebanks in historical linguistic research". In: *Perspectives on Historical Syntax*. Ed. by Carlotta Viti. Amsterdam: Benjamins, pp. 188–202.

Haug, Dag T. T. and Marius L. Jøhndal (2008). "Creating a parallel treebank of the old Indo-European bible translations". In: *Language Resources and Evaluation*. Marrakech, Morocco, pp. 27–34.

Havelka, Jiří (2007). "Beyond projectivity: multilingual evaluation of constraints and measures on non-projective structures". In: *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*. Prague, Czech Republic: Association for Computational Linguistics, pp. 608–615.

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*. Ed. by Mary Dalrymple, Jeffrey Goldberg, Kristin Hanson, Michael Inman, Chris Piñon, and Stephen Wechsler. Stanford Linguistics Association. Stanford, pp. 102–118.

Kallmeyer, Laura and Giorgio Satta (2009). "A polynomial-time parsing algorithm for TT-MCTAG". In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*. ACL '09. Suntec, Singapore: Association for Computational Linguistics, pp. 994–1002.

Kaplan, Ronald M. and Joan Bresnan (1982). "Lexical-Functional Grammar. A formal system for grammatical representations". In: *The Mental Representation of Grammatical Relations*. Ed. by Joan Bresnan. Cambridge, MA: MIT Press, pp. 173–281.

Karlsson, Fred (2007). "Constraints on multiple center-embedding of clauses". In: *Journal of Linguistics* 43.02, pp. 365–392.

Kuhlmann, Marco (2013). "Mildly non-projective dependency grammar". In: *Computational Linguistics* 39.2, pp. 355–387.

Kuhlmann, Marco, Alexander Koller, and Giorgio Satta (2015). "Lexicalization and generative power in CCG". In: *Computational Linguistics* 41.2, pp. 215–247. DOI: 10.1162/COLI_a_00219.

Kuhlmann, Marco and Joakim Nivre (2006). "Mildly non-projective dependency structures". In: *Proceedings of the COLING/ACL Main Conference Poster Sessions*. COLING-ACL '06. Sydney, Australia: Association for Computational Linguistics, pp. 507–514.

Maier, Wolfgang and Timm Lichte (2011). "Characterizing discontinuity in constituent treebanks". In: *Formal Grammar: 14th International Conference, FG 2009, Bordeaux, France, July 25-26, 2009, Revised Selected Papers*. Ed. by Philippe de Groote, Markus Egg, and Laura Kallmeyer. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 167–182. DOI: 10.1007/978-3-642-20169-1_11.

Mambrini, Francesco and Marco Passarotti (2013). "Non-projectivity in the Ancient Greek dependency treebank". In: *Proceedings of the Second International Conference on Dependency Linguistics (DepLing 2013)*. Prague, Czech Republic: Charles University in Prague, MatfyzPress, Prague, Czech Republic, pp. 177–186.

Martens, Scott and Marco Passarotti (2014). "Thomas Aquinas in the TüNDRA: Integrating the Index Thomisticus treebank into CLARIN-D". In: *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Ed. by Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Hrafn

Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis. Reykjavik, Iceland: European Language Resources Association (ELRA).

Maxwell, John T. and Ronald M. Kaplan (1996). "Unification-based parsers that automatically take advantage of context freeness". In: *Proceedings of the LFG'96 Conference*. Ed. by Miriam Butt and Tracy Holloway King. Stanford: CSLI Publications, pp. 1–31.

Pullum, Geoffrey (1982). "Free word order and phrase structure rules". In: *Proceedings of the Twelfth Annual Meeting of the North Eastern Linguistic Society*. Ed. by James Pustejovsky and Peter Sells. Graduate Linguistics Students Association. Amherst, MA, pp. 209–220.

– (1986). "Footloose and context-free". In: *Natural Language and Linguistic Theory* 4, pp. 409–414.

Ross, John R. (1967/1986). *Infinite Syntax*. Reprint of 1967 MIT dissertation. Norwood, NJ: Ablex.

Shieber, Stuart M. (1985). "Evidence against the context-freeness of natural language". In: *Linguistics and Philosophy* 8, pp. 333–343.

Zaenen, Annie and Ronald M. Kaplan (1995). "Formal devices for linguistic generalizations: West Germanic word order in LFG". In: *Formal Issues in Lexical-Functional Grammar*. Ed. by Mary Dalrymple, Ronald M. Kaplan, and John T. Maxwell. Stanford: CSLI Publications, pp. 215–239.