Creating and exploring LFG treebanks

Victoria Rosén
University of Bergen

Helge Dyvik
University of Bergen

Paul Meurer
University of Bergen

Koenraad De Smedt
University of Bergen

Proceedings of the LFG'20 Conference

On-Line

Miriam Butt, Ida Toivonen (Editors)

2020

CSLI Publications

pages 328–348

http://csli-publications.stanford.edu/LFG/2020

Keywords: treebanks, parsed corpora, Lexical-Functional Grammar, Universal Dependencies

Rosén, Victoria, Dyvik, Helge, Meurer, Paul, & De Smedt, Koenraad. 2020. Creating and exploring LFG treebanks. In Butt, Miriam, & Toivonen, Ida (Eds.), *Proceedings of the LFG'20 Conference, On-Line*, 328–348. Stanford, CA: CSLI Publications.

Abstract

While corpora are increasingly used in grammar studies, LFG treebanks have been underused, despite their high level of detail and solid theoretical grounding. The INESS platform provides access to LFG treebanks for several languages, as well as tools to construct and explore LFG treebanks. We present the main features of treebank building and search in INESS and end with a comparison of search in LFG and Universal Dependencies treebanks.

1 Introduction

Research in linguistics is informed by a variety of data, increasingly in digital form. Corpora annotated at the syntactic level, also called treebanks, have been used in many grammar studies. Treebanks come, however, in different varieties, depending on the grammar formalisms that are the basis for their annotation. LFG treebanks are essentially collections of LFG analyses, which means they have at least c-structures and f-structures for a number of sentences.

In our experience, LFG treebanks are instrumental in grammar development and testing, and they are also very useful for quantitative syntactic studies and applications such as lexicography. There are, however, few studies that exploit LFG treebanks and the computational LFG grammars that these are based on. This is perhaps because most existing LFG treebanks are still limited in size, and because researchers have been familiar neither with LFG treebanks nor with tools to search them efficiently. As a case in point, the introductory textbook on LFG by Börjars et al. (2019) has less than two pages on computational work, including only one sentence on treebank-based work.

The main aim of this paper is therefore to provide some guidance to linguists—especially but not exclusively linguists working with LFG—regarding access to treebanks and their potential for grammar research. We will approach this aim by describing INESS (the Infrastructure for the Exploration of Syntax and Semantics), which is currently the largest treebanking platform with extensive support for LFG treebanks.[1] While various design aspects of this infrastructure have been described in the literature (for an overview, see Rosén et al. 2012), the current paper will focus on the available treebanks and tools, and will demonstrate some of the potential for exploring them.

2 LFG treebanks accessible in INESS

INESS hosts approximately 700 treebanks of different types for more than 90 languages. These include LFG treebanks for English, Georgian, German, Hungarian, Indonesian, Norwegian, Polish, Portuguese, Tamil, Turkish, Urdu and Wolof, among others. Some of these are substantial treebanks, while others are test suites for grammar development.

[1] https://clarino.uib.no/iness; INESS is offered as a service of the CLARINO Bergen Center.

The Treebank Selection page on the INESS website gives an overview of available treebanks in INESS, as shown in Figure 1. Treebanks are grouped according to three criteria: language, collection and type. A collection is a group of treebanks which have something in common. It could be that they were constructed from the same grammar, such as the *POLFIE* collection, a group of treebanks that were parsed with POLFIE, the Polish LFG grammar (Patejuk and Przepiórkowski 2014). Another type of collection is exemplified by *Sofie*, which consists of translations of the first part of the novel *Sophie's World* by Jostein Gaarder; this is a parallel treebank, with the component treebanks pairwise aligned for various languages.

Figure 1: Treebank selection page with LFG treebanks chosen

Treebanks are selected by clicking on languages, collections or types. The screenshot in Figure 1 shows the Treebank Selection page after the user has clicked on *lfg* under *Treebank Types*; this choice results in only the languages and collections with LFG treebanks being displayed in boldface. The numbers in parentheses show how many treebanks there are per language, collection and type. The number before the slash gives the number of chosen treebanks, and the number after the slash, the total number of treebanks. For instance, *English (6/39)* means that of the 39 English treebanks in INESS, six are LFG treebanks and are part of the current selection. There are 118 LFG treebanks for 18 different languages.

The largest LFG treebanks are the Norwegian NorGramBank (Dyvik et al. 2016), the LFG Structure Bank of Polish (Patejuk and Przepiórkowski 2014), and

the LFG TIGER treebank for German (Brants et al. 2002). Most of the others are small test suites created for parallel grammar development. As far as we know, all LFG treebanks hosted in INESS are corpora parsed with manually constructed LFG grammars. An example analysis from the TIGER treebank for the sentence in (1) is shown in Figure 2.

(1) *Kambodscha hat Beobachterstatus.*
Cambodia has observer status
'Cambodia has observer status.'

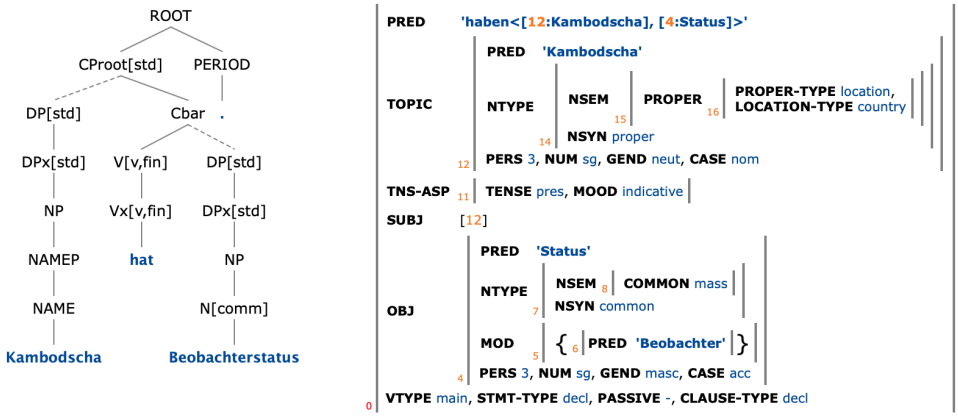

Figure 2: Analysis of the sentence in (1) from the LFG TIGER treebank for German

An example analysis from the Polish treebank for the sentence in (2) is shown in Figure 3. The Polish c-structure follows different principles than the German c-structure—note that the German tree exhibits extensive unary branching—but the buildup of both f-structures is quite similar. The visualizations help us to quickly inspect examples which illustrate similarities and differences in descriptive approaches.[2]

(2) *Ciągle popijali kawę.*
continuously sip.3PL.M1 coffee.ACC
'They were sipping (their) coffee all the time.'

The Parallel Grammar Project (ParGram) has been involved in the development of parallel LFG grammars for more than twenty years (Butt et al. 1999, 2002). The aim has been to build grammars based on common principles, so that, ideally, language-specific characteristics of grammatical structure stand out from quasi-universal ones. The parallelism is mainly on the level of f-structure. Initially the languages involved were English, French and German. Later other languages

[2]For more on visualization in INESS, see Meurer et al. (2020).

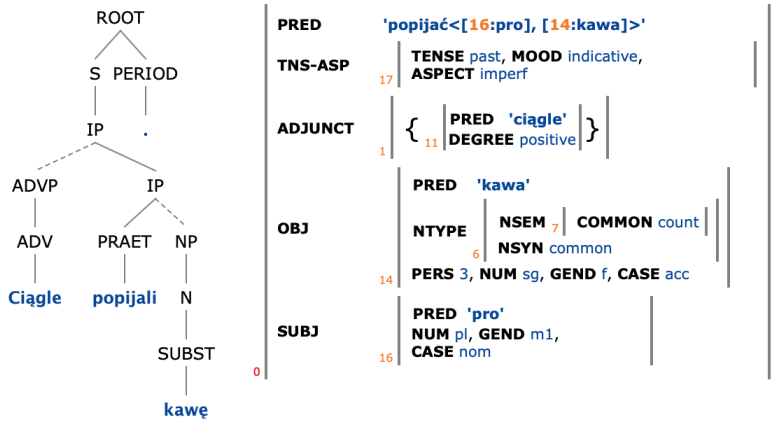

Figure 3: Analysis of the sentence in (2) from the LFG Structure Bank of Polish

joined the project: Georgian, Hungarian, Indonesian, Japanese, Norwegian, Polish, Tamil, Turkish, Urdu and Wolof, among others.

Two sets of parallel test suites in INESS have been developed by ParGram participants: the ParGram collection (Sulger et al. 2013) and the ParTMA collection. These test suites consist of sentences translated from English to the other ParGram languages. The sentences chosen illustrate important linguistic phenomena such as transitivity, voice alternations, interrogatives, copula constructions, etc.

An example of aligned sentences from the ParGram collection is shown in Figure 4, with analyses of the English sentence *Did the farmer sell his tractor?* and its translation into Norwegian *Solgte bonden traktoren sin?* The Norwegian c-structure is displayed to the right of the English c-structure, and the Norwegian f-structure is shown below the English one. The f-structures are displayed in the simplified 'PREDs only' mode, where only attributes related to PREDs are included, in order to make the structures more compact. Both the c-structures and the f-structures are aligned according to the principles developed in the XPAR project (Dyvik et al. 2009), whereby the alignment of c-structures is automatically derived from aligned f-structures. The alignment is done on the level of f-structure. When two f-structures fulfill certain requirements, they may be manually aligned by the user dragging the index of one f-structure onto the corresponding index of another f-structure. The alignment of the f-structures is shown in the indices, where the index of one f-structure points to the index of the other f-structure with an arrow. Once f-structure nodes have been aligned, the corresponding c-structure nodes are automatically aligned by the system, and the c-structure alignment is shown by the green lines between nodes. These alignments are intended to support cross-linguistic studies of grammatical structure.

INESS also offers XLE-Web, which is an online version of XLE (the Xerox Linguistic Environment) for sentence analysis with LFG grammars (Crouch et al. 2011). XLE-Web hosts grammars for the following languages: English, French,

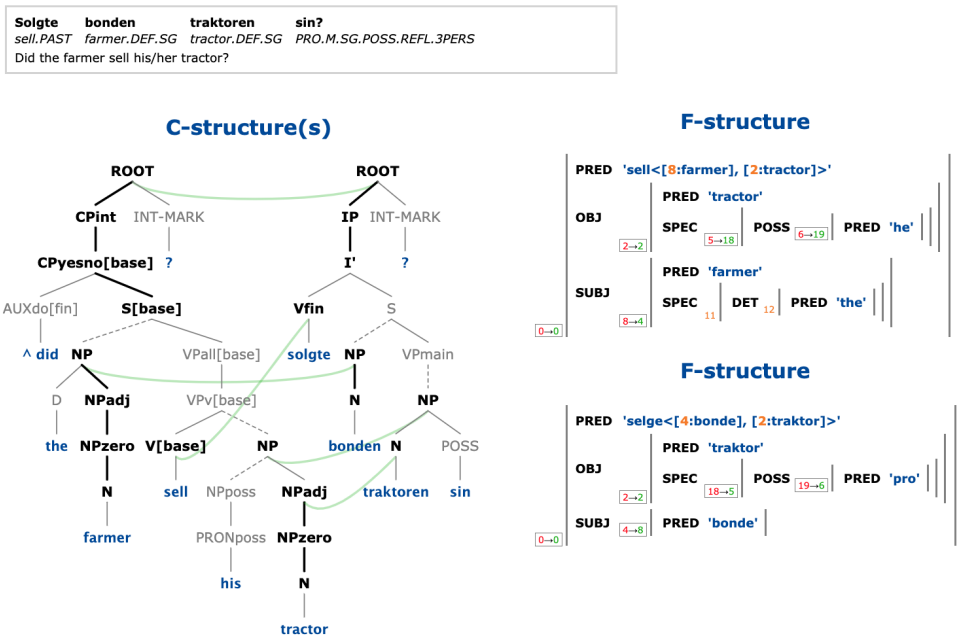

Figure 4: Aligned c- and f-structures for English (left and top) and Norwegian (right and bottom)

German, Georgian, Indonesian, Malagasy, Norwegian, Polish, Tamil, Turkish, Urdu and Wolof. These grammars have been used to create treebanks by parsing corpora and test suites, including the abovementioned ParGram treebanks. XLE-Web offers discriminant disambiguation as described below.

3 Building LFG treebanks

Building an LFG treebank as a parsebank, i.e. by parsing a corpus, is an excellent way of testing the correctness and coverage of a grammar. Assuming a full-coverage grammar and lexicon, as well as perfect disambiguation preferences, parsing a corpus should result in a treebank with correct LFG analyses. In practice, the grammar and lexicon will need to be incrementally revised, and regular reparsing of the corpus should be undertaken in tandem with these revisions. Eventually, stochastic disambiguation can be trained on a gold standard treebank in order to parse and disambiguate a larger corpus fully automatically.

The LFG Parsebanker in INESS is a platform for parsing and disambiguating LFG treebanks (Rosén et al. 2007, 2009). Since sentences tend to receive multiple analyses, sometimes in the thousands, manual disambiguation is supported by the automatic identification of discriminants, which are minimal differences between analyses. Annotators disambiguate manually with discriminants, and the intended analysis is saved in the treebank. If the intended analysis is not present, this may

be corrected by making changes to the lexicon and/or the grammar. The corpus can then be reparsed and earlier disambiguation choices can be automatically reused.

Such a method has been followed in the construction of several treebanks, including the largest one, NorGramBank (Dyvik et al. 2016), which was constructed by means of the XLE-based LFG Parsebanker (Rosén et al. 2009) with discriminant disambiguation in INESS. The LFG Structure Bank of Polish (Patejuk and Przepiórkowski 2014) has also been developed using INESS, while the LFG TIGER Treebank (Brants et al. 2002) was constructed through similar parsebanking with XLE and disambiguation, but in a different environment.

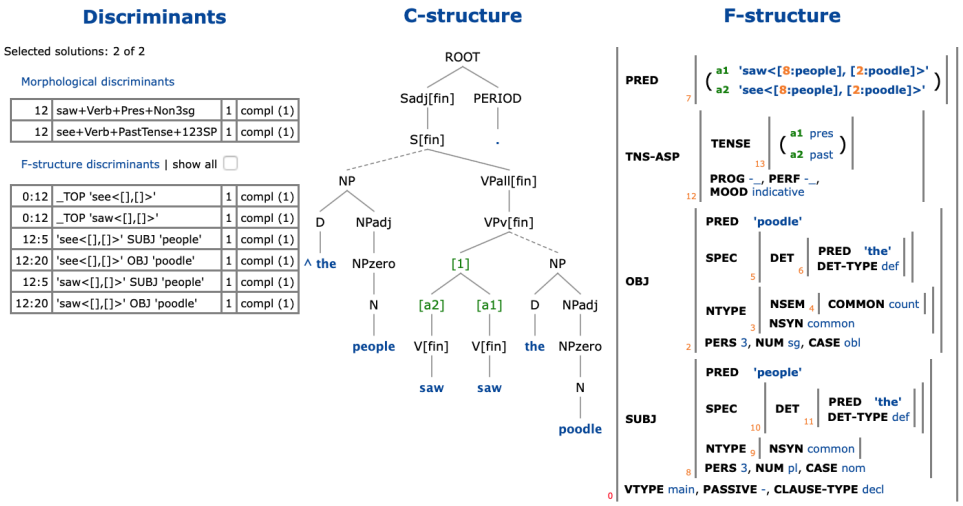

Figure 5: Discriminants for the sentence *The people saw the poodle.*

An illustration of discriminant disambiguation is provided in Figure 5 for the sentence *The people saw the poodle.* This sentence has been parsed on XLE-Web with the English grammar. It gets two analyses, or solutions, meaning either 'The people observed the poodle' or 'The people cut the poodle with a saw'. Note that both the c-structure and the f-structure are packed, which means that they include all alternative c- and f-structure analyses, with choice points indicated (King et al. 2004, Meurer et al. 2020, pp. 63–65). On the left there is a table of discriminants before disambiguation. Discriminants are properties of analyses computed from the different solutions. They make it possible for the user to choose a property that the analysis should have, or reject a property that it shouldn't have. Eight different discriminants are displayed for this sentence, all of which are related to the choice between the verbs *saw* and *see*. In this case disambiguation can be achieved by choosing either a morphological discriminant or an f-structure discriminant. Clicking on any of the discriminants which mention the predicate 'see' will result in complete disambiguation and display of the correct c- and f-structures, as shown in Figure 6.

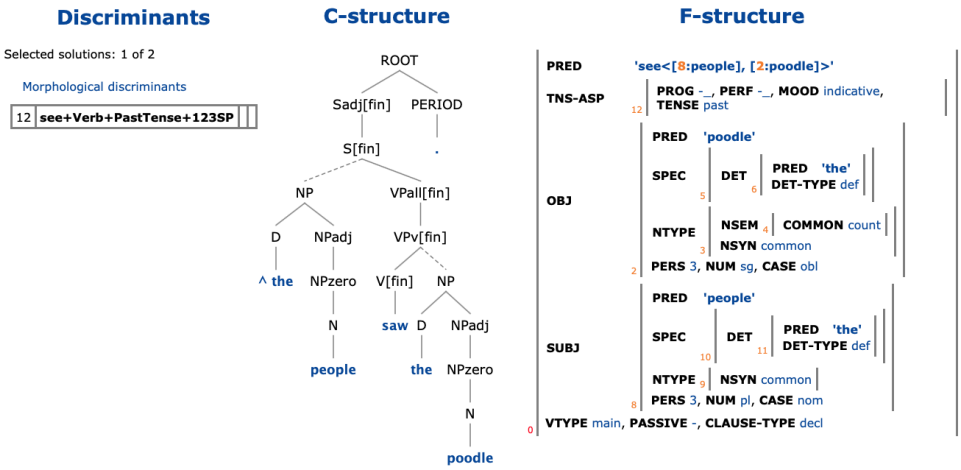

Figure 6: Completed disambiguation for the sentence *The people saw the poodle.*

4 Searching LFG treebanks

Some phenomena are difficult to search for in corpora that lack detailed syntactic annotation, for instance, inversion, passives, clefts, and dependent clauses without complementizers. Such phenomena should be more easily retrievable from treebanks, but that presupposes a powerful and insightful search facility.

INESS Search (Meurer 2012) is a reimplemented, expanded and syntactically simplified online version of TIGERSearch (Lezius 2002). It allows search in different formalisms, including LFG c- and f-structures and their interrelations. INESS Search supports existential and universal quantification as well as negation (with some restrictions concerning universal quantification over disjunctions). This means that it is possible to search for all sentences for which something is the case, and something else at the same time is not the case—for instance, noun phrases where the head noun is not elided (i.e. a noun phrase whose PRED is not 'pro') or dependent clauses without complementizers.

INESS offers two modes for the display and further exploration of search results. In sentence overview mode, a list of sentences that match the query can be displayed and the user can click on sentences to display their structures. In tabular mode, a table is displayed where values of selected query parameters are aggregated with their frequencies, so that quantitative studies are facilitated.

Formulating search expressions obviously requires some knowledge of the annotation in the treebank.[3] XLE-Web can be useful in this respect; you can type in a sentence with the phenomenon you are interested in, and study the analysis showing the structural characteristics of the phenomenon. These structural charac-

[3]For NorGramBank, the documentation (in Norwegian) can be consulted on the INESS website.

teristics can then be specified in a query expression for searching treebanks which were created with the same grammar.

Two examples from NorGramBank will serve to illustrate how easily some structural characteristics may be found in a treebank. In the ParGram grammars, clefts receive the feature VCONSTR with the value *cleft*, so searching for this feature will identify all cleft sentences in the treebank. The search expression in (3) may be read 'There is an f-structure $\#x$ that has an attribute VCONSTR with the value $\#y$ = *cleft*.' One of the matches is the cleft sentence in (4), and its analysis is given in Figure 7. When we specify values with variables such as $\#x$ and $\#y$ in the search expression, these are marked in red in the results.

(3) `#x >VCONSTR #y:'cleft'`

(4) *Det var bussen som kom.*
it was the bus that came
'It was the bus that came.'

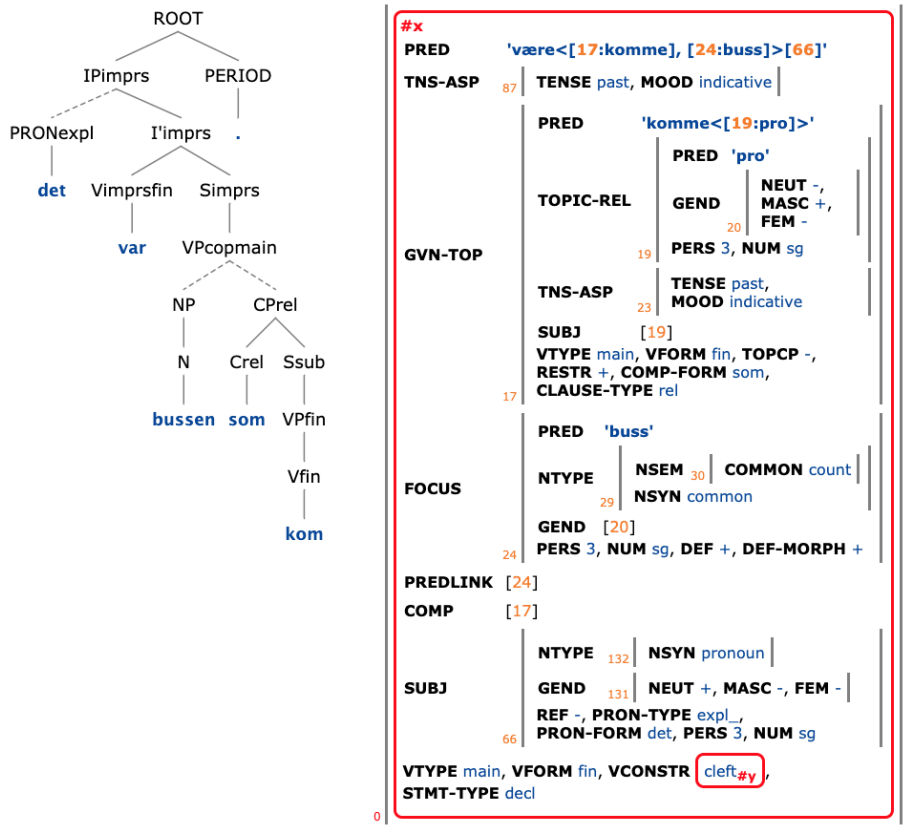

Figure 7: C- and f-structures for the sentence in (4)

A second example involves searching for *that*-clauses without complementizers, for which (5) shows a suitable query. This expression searches for an f-structure $\#x$ which has the attribute CLAUSE-TYPE with the value *nominal*, and which, crucially, does not have an attribute COMP-FORM. For this, we need the negation operator, which is the exclamation point. One of the matches is the sentence in (6); its analysis is shown in Figure 8.

(5) `#x >CLAUSE-TYPE 'nominal' & !(#x >COMP-FORM)`

(6) *Jeg trur han fleipa.*
I think he kidded
'I think he was kidding.'

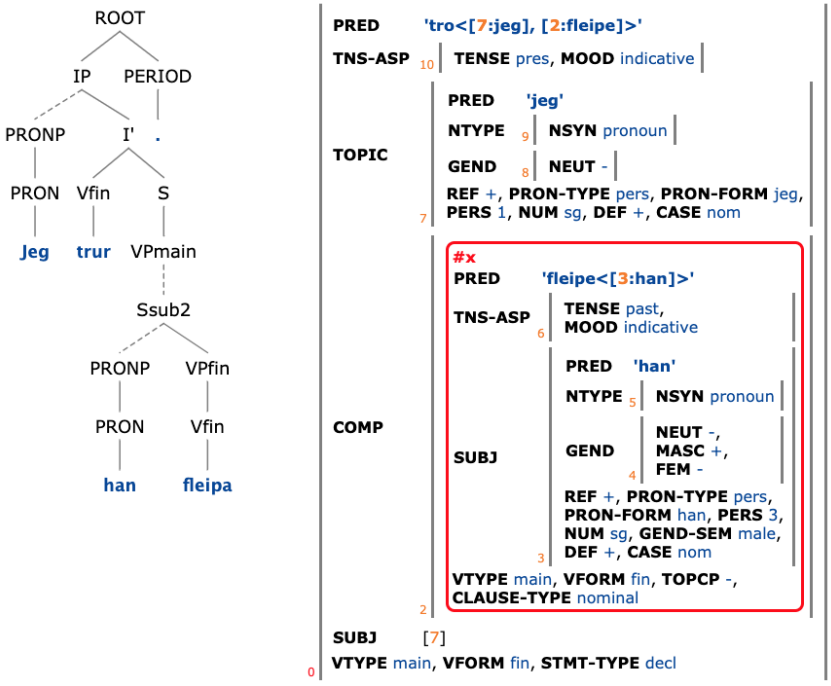

Figure 8: C- and f-structures for the sentence in (6)

Not all query expressions are so simple, however; some can be quite complex. Suppose that we want to find all the different argument frames of a given verb, e.g. Norwegian *overlate* 'leave something to someone', and list them with the lexical categories that fill each argument position. The expression in (7) will accomplish this search. Note that ARG1, ARG2, etc. are not explicit attributes in an f-structure. INESS Search introduces them as a way to refer to the elements in the argument list of the PRED value. The reason (7) is quite complex is that many disjunctions are necessary in order to find all the possible argument frames.

(7)
```
#f_ >PRED #a:'overlate' & #f_ >VFORM &
(  (#f_ >(ARG1 NTYPE NSYN) #arg1
   | #f_ >(ARG1 VFORM) #arg1) &
   !(#f_ >ARG2)
|  (#f_ >(ARG1 NTYPE NSYN) #arg1
   | #f_ >(ARG1 VFORM) #arg1
   | !(#f_ >ARG1)) &
   (#f_ >(ARG2 NTYPE NSYN) #arg2
   | #f_ >(ARG2 VFORM) #arg2) &
   !(#f_ >ARG3)
|  (#f_ >(ARG1 NTYPE NSYN) #arg1
   | #f_ >(ARG1 VFORM) #arg1
   | !(#f_ >ARG1)) &
   (#f_ >(ARG2 NTYPE NSYN) #arg2
   | #f_ >(ARG2 VFORM) #arg2) &
   (#f_ >(ARG3 NTYPE NSYN) #arg3
   | #f_ >(ARG3 VFORM) #arg3) )
```

The richness of the syntactic annotation may lead to query expressions of forbidding complexity for many users. A recent development to overcome this problem consists in templates which can be filled in online.[4] A query template is a parameterized query expression in which some values (e.g. word or lemma forms, predicates, or feature values) are represented by placeholders. It typically represents a technically complex query, and is accompanied by a description of its function and possible parameter values. Upon choosing a template from a menu, the user simply supplies values for the parameters and activates the search.

An example of a template is `V-argframes(@V)`, shown in the template search interface in Figure 9. This template does the same work as the search expression in (7), but the verb is parameterized so that the template can easily be used to explore the frames of different verbs. After filling in the desired value for the `@V` parameter, i.e. one or more verbs, the user may click on *Run query*.

This search in NorGramBank resulted in a table with 58 frames; the top of the table is displayed in Figure 10. This figure also shows what happens when a particular frame is selected for further inspection. The user has clicked on the sixth line (#arg1: *common*, #arg2: *pronoun*, #arg3: *inf*) to display the examples for that frame, the first of which is the sentence in (8). Search results like these are particularly useful for lexicographers, and, in fact, they are currently being used in the construction and further development of dictionaries for Norwegian.

(8) *Mor overlater til meg å lage kveldsmat.*
mother leaves to me to make supper
'Mother leaves it to me to make supper.'

Templates may have more than one parameter. The template for searching for filler–gap constructions, for instance, requires the user to supply the function of the

[4] https://folk.uib.no/hfohd/INESS-Sketch-veiledning-2020.pdf

Template: *** V-argframes(@V)**

Description: **Argument frames of a verb**

Lists, with frequencies, all argument frames (valency frames) of the verb @V by means of columns headed 'arg1', 'arg2' and 'arg3'. The frames are further sorted within the columns according to whether the argument is:

a common noun (**common**)
a proper noun (**proper**)
a pronoun (**pronoun**)
an infinitival (**inf**)
a supine (**sup**)
a past participle (**pastpart**)
a present participle (**prespart**)
a finite clause (**fin**)
or a web address (**uri**).

Verbal expressions with @V, i.e. @V with selected prepositions, with selected particles, or in idioms, are listed as separate predicates in the column '#a'. The search is limited to Bokmål texts.
The search may take several minutes with frequent verbs.

RECOMMENDATION: search only in non-fragmented analyses ('fragments' = 'none').

Parameters:

@V: overlate

Run query

Figure 9: The template `V-argframes(@V)` with a value for the parameter `@V` filled in by the user

Count	#a: *atom*	#arg1: *value*	#arg2: *value*	#arg3: *value*
54	overlate	common	common	common
9	overlate	common	common	inf
12	overlate	common	common	pronoun
1	overlate	common	common	proper
25	overlate	common	pronoun	common
4	overlate	common	pronoun	inf

Download

Click on a row to go to the sentence. Mouse over a row to see the structures.

Treebank	*Document*	*Trans.*	*Id*	*Sentence*	
nob-child	nb100560601	yes	**50**	Mor overlater til meg å lage kveldsmat.	Copy
nob-novel	oai:bibsys.no:biblio…	yes	**115**	Foreldrene hadde forelsket seg i Frankrike og overlatt til ham å drive familiens ranch og sølvgruvene i Nevada.	Copy
nob-novel_4	oai:bibsys.no:biblio…	no	**3348**	De høyvelbårne kuppmakere hadde overlatt til ham, en alminnelig mann, å rake kastanjene ut av ilden.	Copy
nob-novel_4	oai:bibsys.no:biblio…	no	**403**	Mens hun skjenket sprudlende kjellerkald champagne, fortalte hun at ektemannen hadde overlatt til henne å avgjøre om hun skulle selge huset som det sto eller få det pusset opp først.	Copy

Figure 10: Result of clicking on the sixth row in the list of search results for frames of the verb *overlate*

filler, the path of functions between filler and gap, and the function of the gap. Possible parameter settings are illustrated in Figure 11, where (XCOMP)+ means one or more XCOMPs. This query results in 542 matches in NorGramBank, including the sentence in (9).

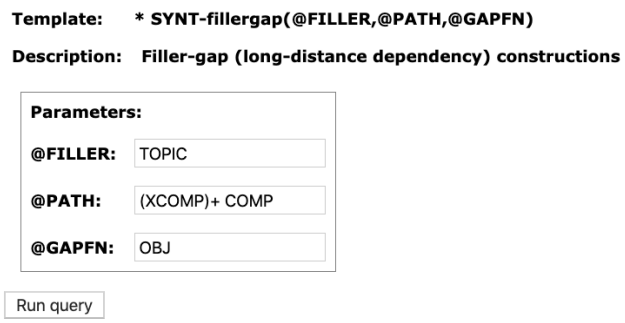

Figure 11: Template for filler–gap constructions with parameter values filled in

(9) *Det vil jeg også tro at de fleste i denne salen faktisk synes.*
that will I also believe that the most in this hall actually think
'That I also believe most people in this room actually think.'

5 Comparison with dependency treebanks

An important development in recent years has been a broad international effort to build treebanks in the Universal Dependency (UD) framework. So far this effort has resulted in more than 150 treebanks for 90 languages, and the list is growing fast (Nivre et al. 2020). Dependency treebanks do not encode phrase structure, but dependencies between words. Given the popularity of UD treebanks, one may wonder if there is a need for LFG treebanks. In this section, we will suggest an answer by comparing how certain phenomena are analyzed, something which directly reflects on their searchability. The texts in the Norwegian UD treebank (Øvrelid and Hohle 2016) are also included in NorGramBank (with about 90% of the sentences analyzed), so that we can make a direct comparison of analyses in both treebanks. We start by comparing the LFG and UD analyses of the Norwegian sentence in (10).

(10) *Familien satt rundt middagsbordet.*
the family sat around the dinner table
'The family was sitting around the dinner table.'

Figure 12 presents the LFG analysis of (10) with fairly detailed c- and f-structure information. The c-structure shows the hierarchical configuration of a

rich inventory of syntactic categories, while the corresponding f-structure displays predicate–argument structures, syntactic functions and grammatical features in an attribute–value format which is in some ways comparable to a dependency structure. In the screenshot in Figure 12 an aspect of the projection from c-structure to f-structure has been visualized by mousing over the NP dominating *Familien*. This results in the substructure with index 17 being highlighted, and shows that the NP corresponds to both the TOPIC and the SUBJ in the f-structure.

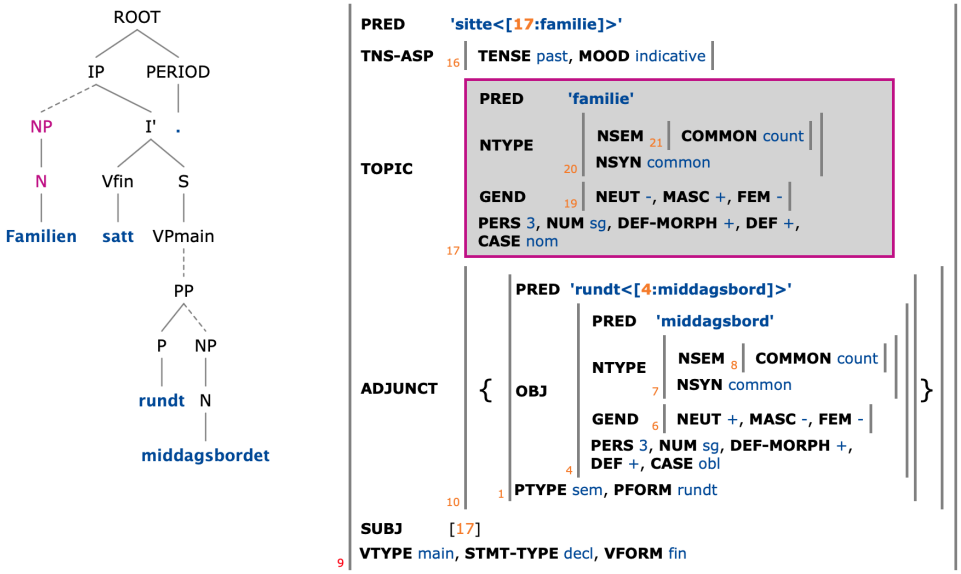

Figure 12: LFG analysis of the sentence in (10)

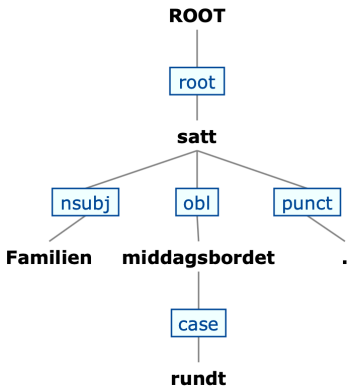

Figure 13: Tree view of the UD analysis of the sentence in (10)

Figure 13 shows the UD representation of the same sentence. The dependencies go from the bottom of one node to the top of another node, with a label on the edge. For instance, there is a dependency from the node *satt* to the node *Familien*, with

the label `nsubj` for subject. This view does not preserve word order, but there is also an alternative linear view of the same analysis, shown in Figure 14, in which the word order is retained, and the dependencies are shown by labeled arrows.

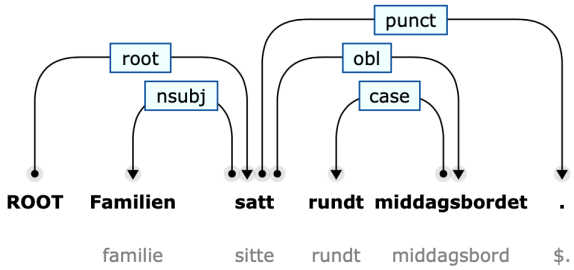

Figure 14: Linear view of the UD analysis of the sentence in (10)

In more complex examples, there can be many arrows entering and leaving nodes. Color coding makes it easier to distinguish arrows at selected nodes. Figure 15 shows the result of mousing over the word *satt*; outgoing dependencies from the selected node are then shown in blue, while incoming dependencies are shown in red. Figure 16 shows the result of mousing over and clicking on the word *Familien*. The lemma, part of speech and morphological features are then displayed.

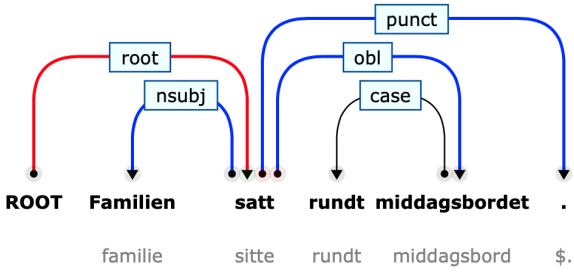

Figure 15: Linear view of the sentence in (10) with highlighting of incoming and outgoing dependencies at the word *satt*

The deeper analysis in an LFG treebank improves the search possibilities as compared to the shallower analysis in a dependency treebank. As an illustration we can look at ways to search for all examples of the first argument of a particular verb. In LFG treebanks the first argument of a verb can be searched for directly, but since there is no direct annotation of the first argument of a verb in UD treebanks, we will need to search for their possible syntactic realizations, as described below.

Assume that we want to find the first argument of the Norwegian verb *dominere* 'to dominate'. In other words, who or what tends to dominate? For NorGramBank the search expression in (11) will basically do the trick.

(11) `#f_ >PRED #x:'dominere' & #f_ >(ARG1 PRED) #p`

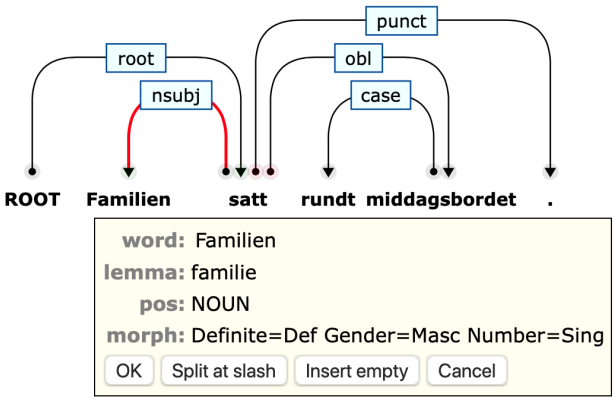

Figure 16: Display of morphological information for *Familien* in sentence (10)

This expression finds each f-structure $\#f\_$ that has a PRED 'dominere' and whose ARG1 has the PRED $\#p$. The search output will list all values of $\#p$, i.e. the ARG1 predicates. In (12) the expression $(\$)*$ has been added to allow ARG1 to be expressed by a coordinated phrase; thus the PRED of each conjunct is retrieved.[5]

(12) `#f_ >PRED #x:'dominere' & #f_ >(ARG1 ($)* PRED) #p`

An example match for the search expression in (12) is the sentence in (13); the PREDs-only version of its f-structure is given in Figure 17. The first position in the argument list of 'dominere' has the index 92, pointing to the value of OBL-AG, which is thus identified as ARG1. The highlighting matches the variables in the search expression. Since the sentence has a coordinated phrase as ARG1, each PRED value is found and highlighted, and listed in the search output.[6]

(13) *Vegetasjonen domineres av småvokste urter, lav og moser.*
the vegetation is dominated by stunted herbs lichen and mosses
'The vegetation is dominated by stunted herbs, lichen and mosses.'

In contrast, the Norwegian UD treebank does not annotate predicate–argument structures, so it is not obvious how to search for the first argument of the verb *dominere* in this treebank. When a property is not annotated in a treebank, the only way to search for it is by specifying all the ways in which the property may be realized in the texts, which will frequently be an insurmountable task for a user. The first argument of a verb can, for instance, be the subject of an active verb, the

[5] $ encodes the set membership operator $\in$, while the Kleene star allows zero or more occurrences of it in order to search within possibly nested sets of f-structures; this is relevant for searches within binary branching coordinations.

[6] In the full NorGramBank (about 160 mill. words, accessed December 15, 2020) there are 5137 matches for the LFG query, with 1818 different ARG1 predicates.

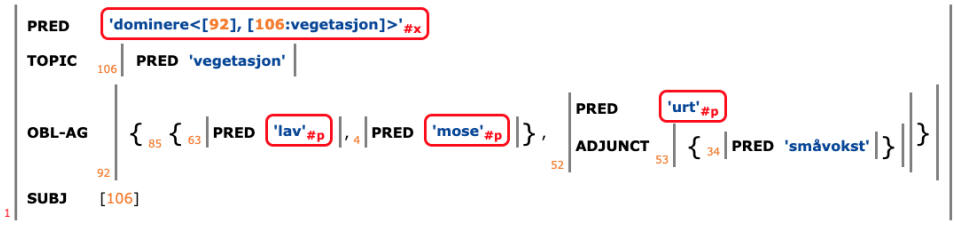

Figure 17: Simplified f-structure for (13), as displayed in results of query (12)

oblique agent of a passive verb, the head of an attributive present participle, or the subject of a predicative present participle. To search for these various possibilities, we need to know how they are annotated. Examining the UD analyses of the sentences found in NorGramBank by the search expression in (12), as well as the documentation,[7] shows that at least the following annotations are possible:

- subjects are coded as *nsubj*;
- agents are coded as *obl*, *obl:agent*, and sometimes *advmod*;
- heads and subjects of participles are coded as taking the participle as an *amod* or an *xcomp*.

An example is the UD treebank analysis of the sentence in (13), given in Figure 18. In this example the first of the coordinated agents, *urter*, is coded as *obl*, while the following agents can be found via *conj* arcs.

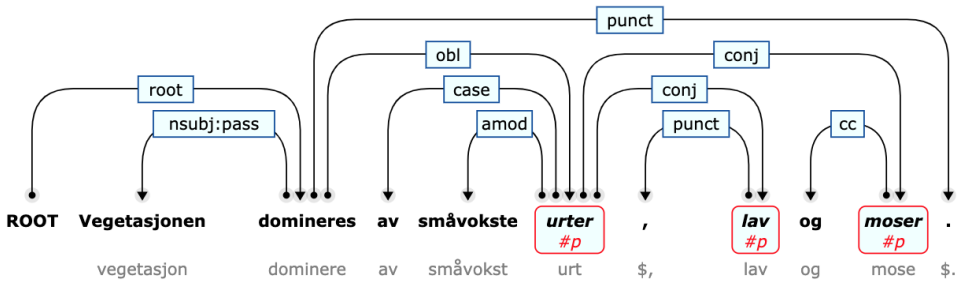

Figure 18: The UD treebank analysis for (13), as displayed in results of query (14)

It is complicated to construct a search expression that captures all possible cases for UD, but a reasonable approximation in INESS Search is provided in (14). The expression tries to restrict search to the relevant contexts for the specified features. For instance, *advmod* and *obl* are not only found with passive agents in the treebank. Lines 6–12 in (14) attempt to restrict their occurrence to analyses

[7] https://universaldependencies.org/guidelines.html

where they do represent passive agent phrases. Also, *amod* and *xcomp* do not only occur with present participles (ending in *-ende/-ande*). Lines 13 and 14 in (14) are meant to restrict the occurrence of these labels to forms that are present participles.

(14)
```
#x_:/dominere(nde)?/ &
( #x_:[morph=".*(Fin|Inf|Degree).*"] >(nsubj (conj)*) #p
| #x_:[morph=".*Part.*"] >(nsubj (conj)*) #p &
      #x_ >aux /ha/
| #x_ >((obl | advmod) (conj)*) #p:[lemma] &
      (#p >case "av"
      | #q_ >case "av" & #q_ >conj #p) &
      (#x_:[morph=".*Pass.*"]
      | #x_:[morph=".*Part.*"] &
         (#x_ >aux:pass
         | #x_ >aux #r_:/være|vere/
         | !(#x_ >aux)))
| #p >amod #x_:".*ende|.*ande"
| #y_ >nsubj #p & #y_ >xcomp #x_:".*ende|.*ande")
```

Querying the UD treebank with (14) finds 19 sentences out of the 20 found by (12) in the LFG analyses of the same texts, with the first arguments identified. The sentence which was not found has *dominere* as a noncontrolled infinitive (a COMP, not an XCOMP, in LFG terms). Such infinitives get no subject in the UD treebank, while LFG assigns 'pro' as a subject, and hence as ARG1.

There is, of course, no guarantee that all constructions expressing first arguments of the chosen verb are covered by a search expression like (14) based on a selection of examples from the UD treebank. Some first-argument cases may be irretrievable in UD, or alternatively, retrievable only with noise. Writing the expression (14) for UD presupposes detailed knowledge about the different ways in which the first-argument relation is realized in the UD treebank, whereas (12) for LFG simply mentions ARG1.

6 Conclusion

LFG treebanks constructed as parsebanks are collections of LFG analyses; they show the effects of choices of grammatical analysis when applied to a corpus of authentic sentences or constructed examples. A substantial LFG parsebank can serve as an empirical testing ground in LFG grammar and lexicon development (e.g. Losnegaard et al. 2012, Rosén et al. 2016b) and for a variety of corpus studies (e.g. Rosén and Borthen 2017). LFG treebanks can contribute to a basis for cross-linguistic and cross-theoretic studies of multiword expressions and constructions, both with respect to the distribution of types and the treatment of such constructions across languages and formalisms (Rosén et al. 2016a). LFG treebanks can also be used for grammar induction or treebank conversion (e.g. Meurer 2017). With appropriate search tools, they can provide frequencies and examples of constructions for applications such as lexicography, as mentioned in Section 4.

Although LFG treebanks have so far not been widely exploited, we believe they nevertheless have great potential due to the high level of detail and theoretical grounding that LFG analyses offer as compared to, for instance, those in Universal Dependency treebanks. It would therefore be worthwhile to develop more and larger LFG treebanks for many languages and to promote their use in the LFG and other communities.

References

Brants, Sabine, Dipper, Stefanie, Hansen, Silvia, Lezius, Wolfgang and Smith, George. 2002. The TIGER Treebank. In *Proceedings of the 1st Workshop on Treebanks and Linguistic Theories*, pages 24–41.

Butt, Miriam, Dipper, Stefanie, Frank, Anette and King, Tracy Holloway. 1999. Writing Large-Scale Parallel Grammars for English, French and German. In Miriam Butt and Tracy Holloway King (eds.), *Proceedings of the LFG99 Conference*, CSLI Publications.

Butt, Miriam, Dyvik, Helge, King, Tracy Holloway, Masuichi, Hiroshi and Rohrer, Christian. 2002. The Parallel Grammar Project. In John Carroll, Nelleke Oostdijk and Richard Sutcliffe (eds.), *COLING-GEE '02 Proceedings of the 2002 Workshop on Grammar Engineering and Evaluation*, pages 1–7, Association for Computational Linguistics.

Börjars, Kersti, Nordlinger, Rachel and Sadler, Louisa. 2019. *Lexical-Functional Grammar: An Introduction*. Cambridge, UK: Cambridge Univ. Press.

Crouch, Dick, Dalrymple, Mary, Kaplan, Ron, King, Tracy, Maxwell, John and Newman, Paula. 2011. XLE Documentation. Technical Report, Palo Alto Research Center, [Online; accessed October 23, 2015].

Dyvik, Helge, Meurer, Paul, Rosén, Victoria and De Smedt, Koenraad. 2009. Linguistically Motivated Parallel Parsebanks. In Marco Passarotti, Adam Przepiórkowski, Sabine Raynaud and Frank Van Eynde (eds.), *Proceedings of the Eighth International Workshop on Treebanks and Linguistic Theories*, pages 71–82, EDUCatt.

Dyvik, Helge, Meurer, Paul, Rosén, Victoria, De Smedt, Koenraad, Haugereid, Petter, Losnegaard, Gyri Smørdal, Lyse, Gunn Inger and Thunes, Martha. 2016. NorGramBank: A 'Deep' Treebank for Norwegian. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk and Stelios Piperidis (eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3555–3562.

King, Tracy Holloway, Dipper, Stefanie, Frank, Anette, Kuhn, Jonas and Maxwell III, John T. 2004. Ambiguity Management in Grammar Writing. *Research on Language and Computation* 2(2), 259–280.

Lezius, Wolfgang. 2002. TIGERSearch – Ein Suchwerkzeug für Baumbanken. In Stephan Busemann (ed.), *Proceedings der 6. Konferenz zur Verarbeitung Natürlicher Sprache (KONVENS 2002)*.

Losnegaard, Gyri Smørdal, Lyse, Gunn Inger, Thunes, Martha, Rosén, Victoria, De Smedt, Koenraad, Dyvik, Helge and Meurer, Paul. 2012. What We Have Learned from Sofie: Extending Lexical and Grammatical Coverage in an LFG Parsebank. In Jan Hajič, Koenraad De Smedt, Marko Tadić and António Branco (eds.), *META-RESEARCH Workshop on Advanced Treebanking at LREC 2012*, pages 69–76.

Meurer, Paul. 2012. INESS-Search: A Search System for LFG (and Other) Treebanks. In Miriam Butt and Tracy Holloway King (eds.), *Proceedings of the LFG'12 Conference*, LFG Online Proceedings, pages 404–421, CSLI Publications.

Meurer, Paul. 2017. From LFG Structures to Dependency Relations. In Victoria Rosén and Koenraad De Smedt (eds.), *The Very Model of a Modern Linguist — in Honor of Helge Dyvik*, volume 8 of *Bergen Language and Linguistics Studies (BeLLS)*, pages 183–201, University of Bergen, Norway.

Meurer, Paul, Rosén, Victoria and De Smedt, Koenraad. 2020. Interactive Visualizations in INESS. In Miriam Butt, Annette Hautli-Janisz and Verena Lyding (eds.), *LingVis: Visual Analytics for Linguistics*, pages 55–85, Stanford, California: CSLI Publications.

Nivre, Joakim, de Marneffe, Marie-Catherine, Ginter, Filip, Hajic, Jan, Manning, Christopher D., Pyysalo, Sampo, Schuster, Sebastian, Tyers, Francis and Zeman, Daniel. 2020. Universal Dependencies v2: An Evergrowing Multilingual Treebank Collection. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 4027–4036, Marseille, France: European Language Resources Association (ELRA).

Patejuk, Agnieszka and Przepiórkowski, Adam. 2014. Synergistic Development of Grammatical Resources: A Valence Dictionary, an LFG Grammar, and an LFG Structure Bank for Polish. In *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT13)*, pages 113–126, Department of Linguistics (SfS), University of Tübingen.

Rosén, Victoria and Borthen, Kaja. 2017. Norwegian Bare Singulars Revisited. In Victoria Rosén and Koenraad De Smedt (eds.), *The Very Model of a Modern Linguist — in Honor of Helge Dyvik*, volume 8 of *Bergen Language and Linguistics Studies (BeLLS)*, pages 220–240, University of Bergen, Norway.

Rosén, Victoria, De Smedt, Koenraad, Losnegaard, Gyri Smørdal, Bejček, Eduard, Savary, Agata and Osenova, Petya. 2016a. MWEs in Treebanks: From Survey to Guidelines. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk and Stelios Piperidis (eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, pages 2323–2330, ELRA.

Rosén, Victoria, De Smedt, Koenraad, Meurer, Paul and Dyvik, Helge. 2012. An Open Infrastructure for Advanced Treebanking. In Jan Hajič, Koenraad De Smedt, Marko Tadić and António Branco (eds.), *META-RESEARCH Workshop on Advanced Treebanking at LREC 2012*, pages 22–29.

Rosén, Victoria, Meurer, Paul and De Smedt, Koenraad. 2007. Designing and Implementing Discriminants for LFG Grammars. In Tracy Holloway King and Miriam Butt (eds.), *The Proceedings of the LFG'07 Conference*, pages 397–417, CSLI Publications.

Rosén, Victoria, Meurer, Paul and De Smedt, Koenraad. 2009. LFG Parsebanker: A Toolkit for Building and Searching a Treebank as a Parsed Corpus. In Frank Van Eynde, Anette Frank, Gertjan van Noord and Koenraad De Smedt (eds.), *Proceedings of the Seventh International Workshop on Treebanks and Linguistic Theories (TLT7)*, pages 127–133, LOT.

Rosén, Victoria, Thunes, Martha, Haugereid, Petter, Losnegaard, Gyri Smørdal, Dyvik, Helge, Meurer, Paul, Lyse, Gunn Inger and De Smedt, Koenraad. 2016b. The Enrichment of Lexical Resources Through Incremental Parsebanking. *Language Resources and Evaluation* 50(2), 291–319.

Sulger, Sebastian, Butt, Miriam, King, Tracy Holloway, Meurer, Paul, Laczkó, Tibor, Rákosi, György, Dione, Cheikh Bamba, Dyvik, Helge, Rosén, Victoria, De Smedt, Koenraad, Patejuk, Agnieszka, Çetinoglu, Özlem, Arka, I Wayan and Mistica, Meladel. 2013. ParGramBank: The ParGram Parallel Treebank. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 550–560, Association for Computational Linguistics.

Øvrelid, Lilja and Hohle, Petter. 2016. Universal Dependencies for Norwegian. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk and Stelios Piperidis (eds.), *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1579–1585.