CONSTRUCTING A PARSED CORPUS WITH A LARGE LFG GRAMMAR

Victoria Rosén, Paul Meurer and Koenraad de Smedt
University of Bergen and AKSIS

Proceedings of the LFG'05 Conference
University of Bergen
Miriam Butt and Tracy Holloway King (Editors)

2005
CSLI Publications
http://csli-publications.stanford.edu

Abstract

The TREPIL project (Norwegian treebank pilot project 2004-2008) is aimed at developing and testing methods for the construction of a Norwegian parsed corpus. Annotation of c-structures, f-structures and mrs-structures is based on automatic parsing with human validation and disambiguation. Parsing is done with a large LFG grammar and the XLE parser. We propose a method for efficient disambiguation based on discriminants and we have implemented a set of computational tools for this purpose.

1 Treebanks and parsed corpora

We use the term *treebank* to mean a corpus annotated with sentence structures beyond the part of speech level. Even though the term refers to syntactic tree structures, it is in current usage extended to corpora with all kinds of structural annotation at syntactic and even semantic levels, such as constituent structures, grammatical functions, or predicate-argument relations (Nivre, De Smedt, and Volk, 2005). Our work in the context of the TREPIL project (Norwegian treebank pilot project 2004-2008) is aimed at developing and testing methods for the construction of a Norwegian treebank based on deep parsing. Before going into details about this project and its results so far, we first provide some background on previous related work.

Currently, linguists and language engineers have easy access to large text and speech corpora, many of which are annotated at the word level, mostly by parts of speech. Although searching in large corpora for certain words and sequences of words with given categories may yield valuable information, two problems can be discerned (Abeillé, 2003). Firstly, part of speech tagging is of limited use to syntacticians, as it fails to distinguish boundaries of clauses, of phrases and even of compound words that are written separately. Secondly, as automatic part of speech tagging is normally based on a shallow analysis or statistical processor, the quality of the annotated corpus is likely to be unsatisfactory.

To overcome the limitations of corpora with word-level annotation only, efforts have been made towards more sophisticated linguistic annotation of corpora. Whereas the first syntactically annotated corpora were developed mostly with manual methods, the development of more sophisticated linguistic models prompted the application of such models to treebank construction (Abeillé, 2003). This has led to the term *parsed corpus* which is usually reserved for a treebank that is grounded in a computational grammar model. Treebank construction on the basis of automatic parsing with a computational grammar is desirable for both practical and theoretical reasons. Indeed, manual annotation has the disadvantage of being costly and prone to human error, and it is difficult to achieve satisfactory consistency both within and between human annotators (van der Beek et al., 2002a). Moreover, an annotation scheme which is only verbally defined and is not grounded in a computational grammar model risks isolating the corpus from the very applications for which it could be useful.

Fully automatic annotation, on the other hand, only works to the extent that the analyses chosen by the parser are correct. Since perfect coverage is not attainable in practice, many current approaches to treebank construction are semi-automatic, in the sense that parser output is validated by a human annotator. Furthermore, automatic parsing usually produces more than one possible analysis, since many sentences can be analyzed in a variety of ways that may be infelicitous and can neither be excluded on purely syntactic grounds nor completely avoided by statistical learning techniques. Therefore, manual disambiguation is a necessity. In the Alpino treebank, for instance, the corpus is automatically parsed with a dependency grammar, assisted by interactive tools for manual checking, including disambiguation and extension of the lexicon (van der Beek et al., 2002b). The TREPIL project has devoted significant efforts to disambiguation methods, as discussed below.

Several treebank projects have prominently used the LFG and HPSG formalisms. The PARC 700 Dependency Bank (King et al., 2003) was constructed by a two-step approach. First, a corpus was parsed with an LFG grammar and the best parse for each sentence was chosen manually and stored. Then, from each

stored functional structure, a corresponding dependency structure was automatically derived, modified as needed, and validated by a human annotator. The PARC 700 has subsequently been used as an external standard in the evaluation of other f-structure annotations (Burke et al., 2004b). One of the points illustrated by the PARC 700 is that a treebank constructed by parsing with a certain language model (in this case, based on the LFG formalism) nevertheless can be convertible into different linguistic models. A different example of treebanking by conversion is the derivation of Estonian phrase structures on the basis of Constraint Grammar function tags (Bick, Uibo, and Müürisep, 2004). It may perhaps be concluded that aspirations for neutrality with respect to grammatical theory are as unneccessary as they are illusory.

Treebanks are currently receiving a lot of attention because they provide highly valuable empirical data for many research questions in linguistics and language technology (Nivre, De Smedt, and Volk, 2005). Provided they are composed and annotated as reference corpora rather than special purpose collections, treebanks allow for multiple uses in the various sciences of language as well as in language technology. Linguists may want to search for examples or counterexamples of syntactic constructions under investigation, whereas psycholinguists may be interested in relative frequencies of various possible attachments of prepositional phrases or relative clauses (Abeillé, 2003). Formal and computational linguists can evaluate the correctness and coverage of grammars and lexicons against the analyses stored in a treebank, and at a more general level, the adequacy of linguistic theories and formalisms can be assessed (Bouma, 2004).

From the grammatical information stored in treebanks, other resources such as grammars and lexicons can be induced. Stochastic grammars can be trained using frequency information about the parse choices. Other research has focused on the induction of LFG grammars from existing manually annotated treebanks, with the goal of deriving robust, wide-coverage grammars from treebanks rather than having to hand code them (Burke et al., 2004a). Claims that the performance of automatically induced LFG grammars may surpass that of hand-coded LFG grammars (Cahill, 2004) have to be weighed against questions concerning the generality and explanatory power of the linguistic theories embodied by the induced grammars.

More important than the particular formalism used is the level of detail of the analyses in treebanks. While some of the earliest syntactically annotated corpora contain syntactic boundaries, others contain for instance constituent structures (Abeillé, Clément, and Toussenel, 2003), functional dependency structures (Hajič, 1998) or, in addition to syntactic structures, also predicate-argument structures (Marcus et al., 1994). TREPIL is cooperating with the LOGON project on Norwegian-English machine translation (Oepen et al., 2004), which has produced a small treebank containing semantic structures. In the context of translation and contrastive linguistics, we also want to mention the potential of parallel treebanks of translated texts, where detailed and deep analyses offer an interesting domain of study.

2 Treebanking goals in the TREPIL project

The TREPIL project is a research project on treebanking methods, aimed at building a Norwegian parsed corpus. The current project is a preparatory project; it will not produce a full-scale treebank, but a methodology, a set of computational tools, and a demonstration corpus. Our hope is that the resulting methods and tools will be put to use in a subsequent project for building a large scale Norwegian treebank which will form part of a future Norsk Språkbank (Norwegian Language Bank).

The TREPIL project uses the LFG formalism and explores a tight relation between a grammar and a corpus, but our focus is different from earlier LFG-banking projects. Our method for treebank construction is based on the testing and further improvement of an existing hand-coded grammar and parser, and its extension with additional treebanking tools, primarily for disambiguation. For this purpose, we use the NorGram LFG grammar for Norwegian, together with the Xerox Linguistic Environment (XLE). Our motivation for using NorGram as a starting point is twofold. Firstly, NorGram is currently the only deep grammar for Norwegian with large coverage. Secondly, the grammar is developed in the international ParGram project

(Butt et al., 2002) which attains a certain level of generality across languages through agreements on similar feature structures and the existence of a transfer formalism for f-structure based translation. However, we do not want to overemphasize the choice of formalism for reasons outlined above.

An innovative characteristic of TREPIL is that, in contrast to the single stratum approaches of most other treebanks, the Norwegian grammar generates three separate but interrelated structures for each sentence: a constituent structure, a functional structure, and a semantic structure. The semantic projection is based on Minimal Recursion Semantics (MRS) (Copestake et al., in preparation), which allows a deeper level of semantic description than the predicate-argument coding in the Penn treebank (Marcus et al., 1994). MRS represents the semantics of a sentence as a bag of elementary predications, underspecified for scope. The mrs-structures are derived by co-description and may contain information that cannot be derived from the c- or f-structures, such that the mrs-projection represents an autonomous level of structure.

The triple stratum annotation generated by our grammar represents a rich, layered description of the syntax and semantics of each sentence, which allows for multiple uses. However, this sophistication comes at a price, because disambiguation and validation are nontrivial and manual annotation would be quite difficult and inefficient. Thus, our treebanking method is strongly dependent on computational systems, including an efficient parser and a grammar of which the coverage and precision is being continually improved.

One sometimes comes across scepticism with respect to the possibility of deep (full) parsing, often by adherents of shallow parsing as an approximation. However, it has also been pointed out that much of the scepticism is unwarranted for the XLE parser (Zaenen, 2004). Although full parsers may be slower, the XLE parser is still fast enough for off-line parsing of a corpus. Also, it has been pointed out that some full parsers are too brittle to deal with anomalous input, but the XLE parser allows fragment parses, so any input can receive some form of analysis. It has also been claimed that full parsers may yield so many parses that applications have difficulty coping with them. While this problem is being addressed by the inclusion of probability based disambiguation components, such automatic disambiguation is not feasible in our situation, since it would need to be bootstrapped from a treebank which is not yet built. Instead, we are focusing our attention on an altogether different method aimed at highly efficient manual disambiguation. This method, which will be the primary focus of the next section of this paper, is supported by a computational tool which we have implemented in a working first version.

Also, we are working towards a system which allows the automatic reanalysis of the corpus as the grammar develops. In as far as TREPIL involves the synchronous evolution of a treebank and a grammar, our approach is similar to that of LinGO Redwoods (Oepen et al., 2003), which is based on HPSG and the LKB parser environment. LinGO has developed a set of advanced tools that allow the automated update of the treebank after reparsing with a new version of the grammar, but without having to fully disambiguate the corpus over again. This is achieved by reapplying earlier recorded choices by the annotator in the selection of the preferred parse, based on techniques proposed by Carter (1997). A crucial point is that not only the preferred analysis of the sentence is recorded, but all decisions made as part of the annotation in the database.

We believe there are important methodological advantages to our approach. Instead of building a treebank incrementally and improving the grammar independently, we develop an efficient way to successively reannotate the corpus with each version of the grammar, thus obtaining a parsed corpus that is fully consistent with the grammar. The end result is therefore not only a treebank, but also a grammar that can be deployed in other applications, for example machine translation, especially since it produces semantic analyses. In view of such applications, we believe it is advantageous to retain manual control over the grammar in order to obtain the kind of abstraction and readability required by a linguist, rather than inducing an entirely new grammar from the treebank.

3 Disambiguation with XLE

One of the main challenges in using parser output for treebanking is selecting the desired parse among a potentially large number of parses. It is worth remembering that the number of parses is exponential to the number of ambiguities, such that up to 2^n analyses may be produced for n binary choices. Six unresolved, independent parsing choices can for instance give rise to 64 analyses. Consequently, a disambiguation strategy that concentrates on local ambiguities might be more efficient than one that only looks at the whole set of resulting analyses.

XLE has a built-in facility for disambiguation in the form of packed representations. When a sentence is parsed, XLE displays the analyses one at a time in the c-structure and f-structure windows. This allows the user to browse through all the analyses and inspect each c-structure and its corresponding f-structures in turn. In addition, *f-structure chart* windows show *packed* representations of all analyses. There are two different formats in which this compact information is shown, but we will concentrate on the f-structure chart window that indexes the analyses by constraints, providing a view of choices listed as alternatives. When a sentence contains a single ambiguity, this type of representation makes it easy to spot the source of the ambiguity, as shown in figure 1 for example 1.

(1) *Hun er barn.*
she is child/children
"She is a child." / "She is children."

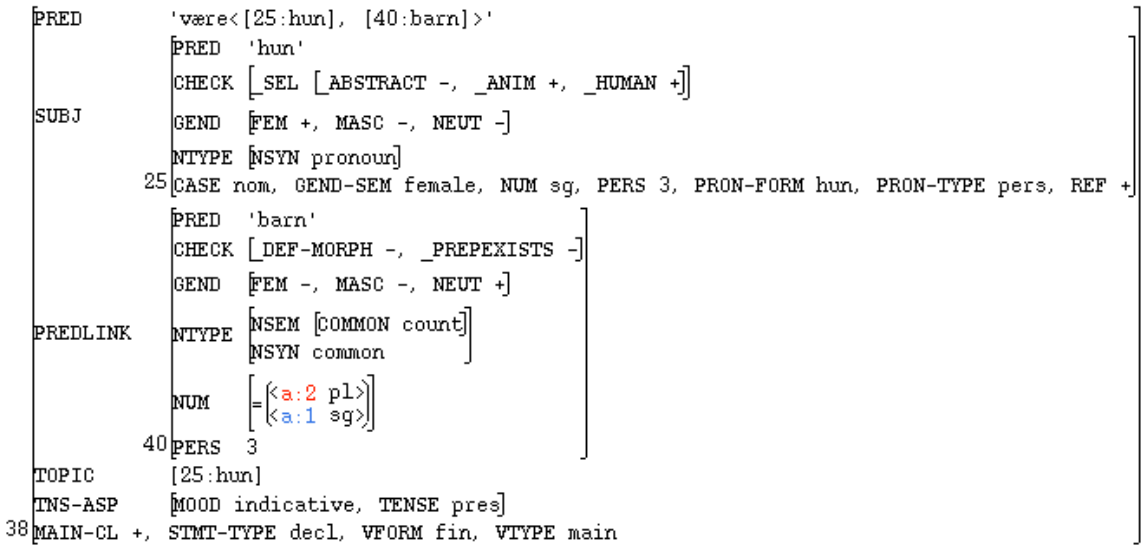

Figure 1: F-structure chart for (1) *Hun er barn.*

This sentence has two analyses, identical except for the value of the feature number, which may be singular or plural. In the f-structure chart, the two values are displayed as alternatives, labeled with indices *a:1* and *a:2*. The choices in these windows are active, so that the user can click on a choice and have a solution corresponding to it displayed in the c-structure and f-structure windows. This facility is easy to use for disambiguation when there are only a few choices.

The sentence in example 2 has two local ambiguities, resulting in four analyses. The packed f-structure chart is shown in figure 2.

(2) *Hun kjøper klær i den dyre butikken.*
she buys clothes in the expensive store
"She buys clothes in the expensive store."

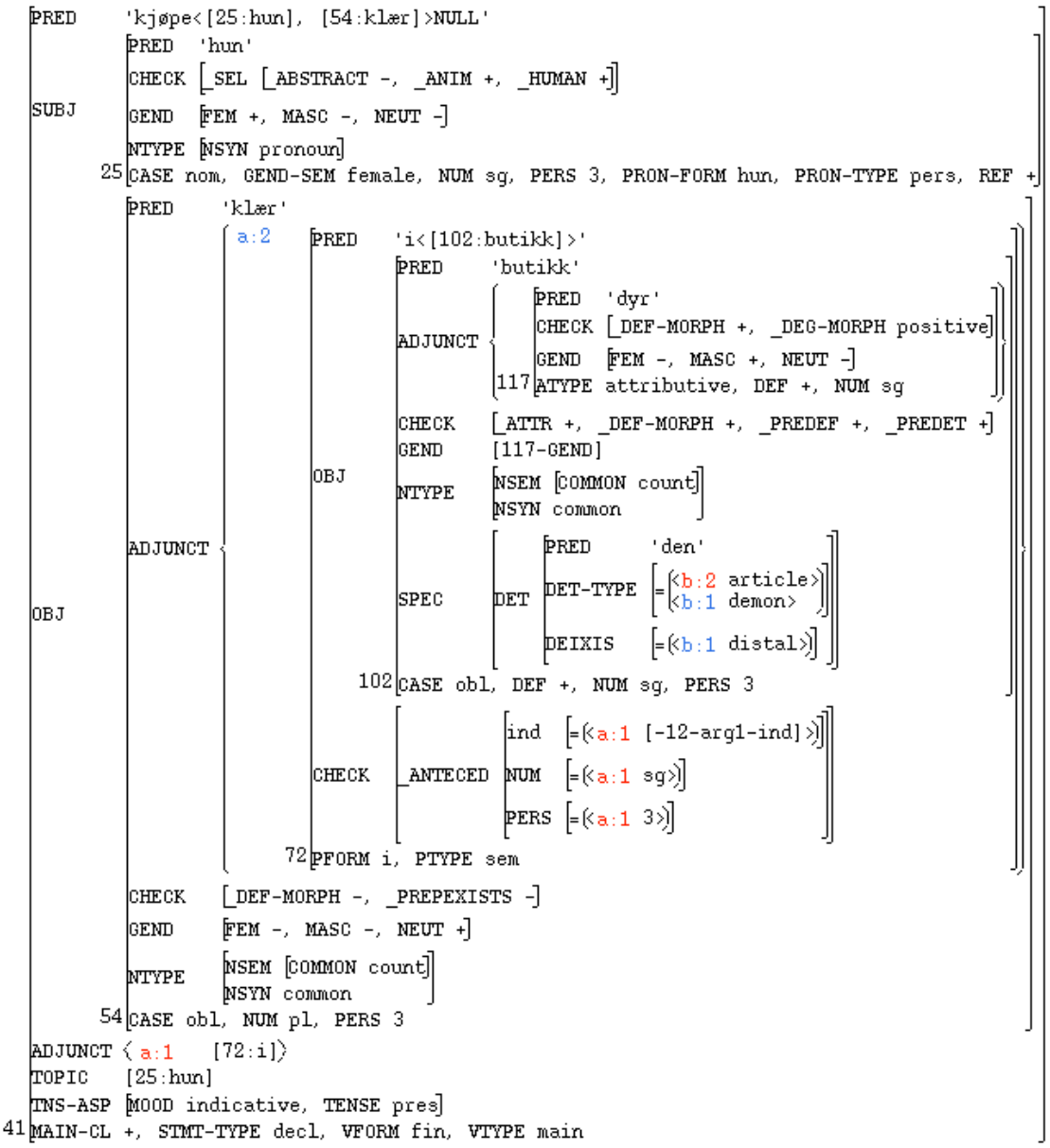

Figure 2: F-structure chart for (2) *Hun kjøper klær i den dyre butikken.*

The word *den* may either be a simple definite determiner or a demonstrative. There is also a syntactic ambiguity: the prepositional phrase may be attached to the NP or to the VP. When the PP is part of the NP, it is an ADJUNCT in the OBJ; when it is part of the VP, it is an ADJUNCT on the outer level of the f-structure. In the f-structure chart shown in figure 2, this is represented by having the attribute ADJUNCT in two places, each with an alphabetic index, *a:1* for the sentential ADJUNCT and *a:2* for the ADJUNCT in the OBJ.

An only slightly more complicated example is given in example 3.

(3) *En jente kjøper klær i den dyre butikken.*
a girl buys clothes in the expensive store
"A girl buys clothes in the expensive store."

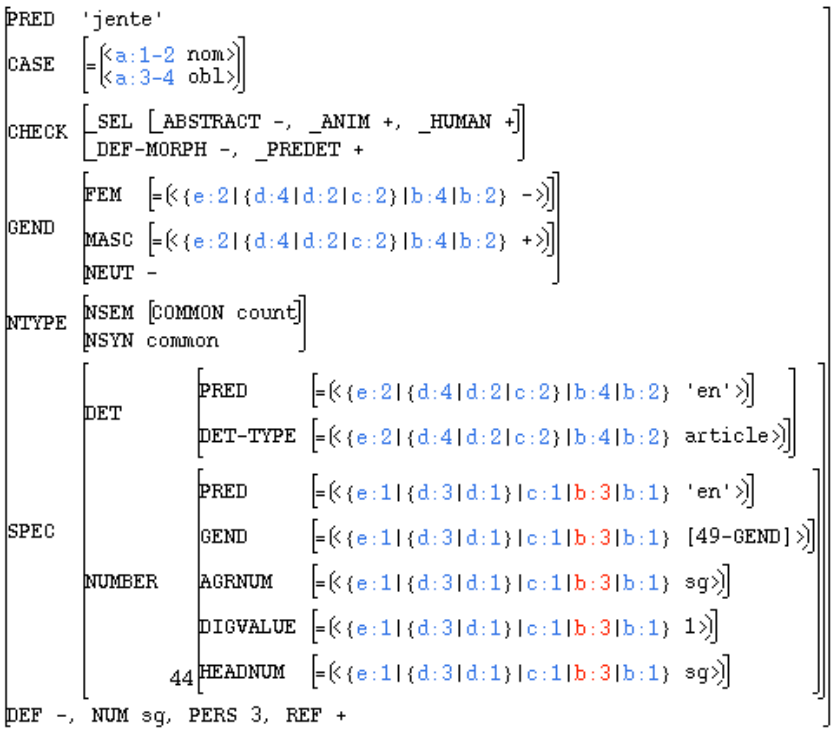

Figure 3: Partial f-structure chart for (3) *En jente kjøper klær i den dyre butikken.*

In addition to the lexical ambiguity and the PP-attachment ambiguity we observed in example 2, there are two more ambiguities here. The word *en* is either the indefinite article or the numeral "one". And since Norwegian is a V2 language, and there is no person or number inflection on verbs, the phrases before and after the finite verb may each either be SUBJ or OBJ. These four local ambiguities result in 16 possible analyses. For this sentence, the f-structure chart requires more space than will fit on even a large computer screen, and we show only a small part of it in in figure 3.

A somewhat more compact display may be obtained from an alternative f-structure chart showing a tree of choices, but the main problem for a human disambiguator is that regardless of which method of display is chosen, the packed representations always contain all the information in all analyses. The disambiguator can choose to mark certain choices as dispreferred, turning their labels gray. But the information still remains in the display, forcing the disambiguator to keep track of many kinds of information about the analyses at the same time. These few examples should make it clear that this method of disambiguation requires expert competence in using XLE and detailed knowledge of the grammar. Even for an annotator with that kind of competence, disambiguating sentences with hundreds of analyses in this way can be a formidable task.

4 The XLE Web Interface

The XLE Web Interface (XLE-Web) is a web-based tool for parsing with XLE and viewing c-structures, f-structures and mrs-structures. Initially developed in the LOGON project (Oepen et al., 2004), it has been

the starting point for the work in TREPIL discussed in the following section. XLE-Web allows the user to choose a grammar and type in a sentence to be analyzed. The sentence is then processed by the XLE parser, and the resulting c-structures, f-structures and mrs-structures are displayed, either one solution at a time, or all solutions together in the form of packed c- and f-structure representations (there is no packed mrs-representation at present).

Packed f-structures were first implemented in XLE in order to provide a compact internal representation of the set of solutions of a sentence. The XLE display system uses this packing to simultaneously display all f-structures in one graph, and the packed f-structures in XLE-Web have been tightly modeled after XLE's packed f-structure display. An innovation in XLE-Web is the display of *packed c-structures* as directed acyclic graphs, namely, a set of c-structure trees where nodes that are equal across solutions are identified and where additional nodes indicate in which contexts their subnodes are valid.

The XLE-Web server software runs on Linux and MacOS; it is implemented in Common Lisp and uses a shared library version of the XLE core parsing engine which is dynamically linked into the server program. Internally, the interface web pages are generated as XML files, which are converted on the server side to HTML by means of XSLT. The interactive features of the displayed structures are implemented in Javascript. For instance, when mousing over a QEQ relation in an mrs-structure, the corresponding variables in the elemenary predicates are highlighted. In the same fashion, structure sharing in f-structures is made visible, and mousing over a c-structure node highlights both the f-structure projection of that node and all other nodes having the same projection. The c-structure trees (and graphs in the case of packed representations) are drawn using the XML-compliant standard SVG (Scalable Vector Graphics).

Screenshots from XLE-Web for the analysis of the ambiguous sentence in example 1 are provided in figure 4. The analyses are by default shown successively on separate web pages. By clicking on *Previous* and *Next* buttons, the user browses through the various c- and f-structures, as well as the mrs-structures. The c- and f-structures are displayed side by side. The mrs-structure is displayed by clicking on the *Show MRS* button.

There are a number of options which may be chosen by ticking off the boxes on the main page. Most LFG grammars that have been implemented in XLE make use of optimality marks to prefer some analyses over others. By ticking the *Disable Optimality marks* box, the user chooses to have all analyses displayed rather than just the 'optimal' analyses. For the purpose of treebanking it may be desirable to run the grammar without the optimality marks, to make sure that all possible analyses are presented for manual disambiguation.

Another way of examining ambiguous analyses in XLE-Web is chosen by ticking off the *Packed representation* box. This displays all analyses of a sentence on one page. For the two-way ambiguous sentence in example 1, we get the packed representation in figure 5. There is no c-structure distinction between the two analyses, so that only one c-structure is displayed. The difference is, as noted above, the value of the attribute NUM, which may be either *sg* or *pl*. This is displayed in the packed f-structure in a way very similar to the standard XLE packed representation shown above in figure 1. In addition, the display in 5 shows a list of discriminants, which are the topic of the following section.

5 Disambiguation with discriminants

The technique of disambiguating with discriminants was first described by Carter (1997), who points out that there may be "dozens of analyses that are variations on a small number of largely independent themes: choices of word sense, modifier attachment, conjunction scope and so on." Carter argues that disambiguation may be achieved quickly and without expert competence if it is based on elementary linguistic properties which the disambiguator may accept or reject independently of other properties. The disambiguation is then done on the individual properties rather than on the analyses themselves.

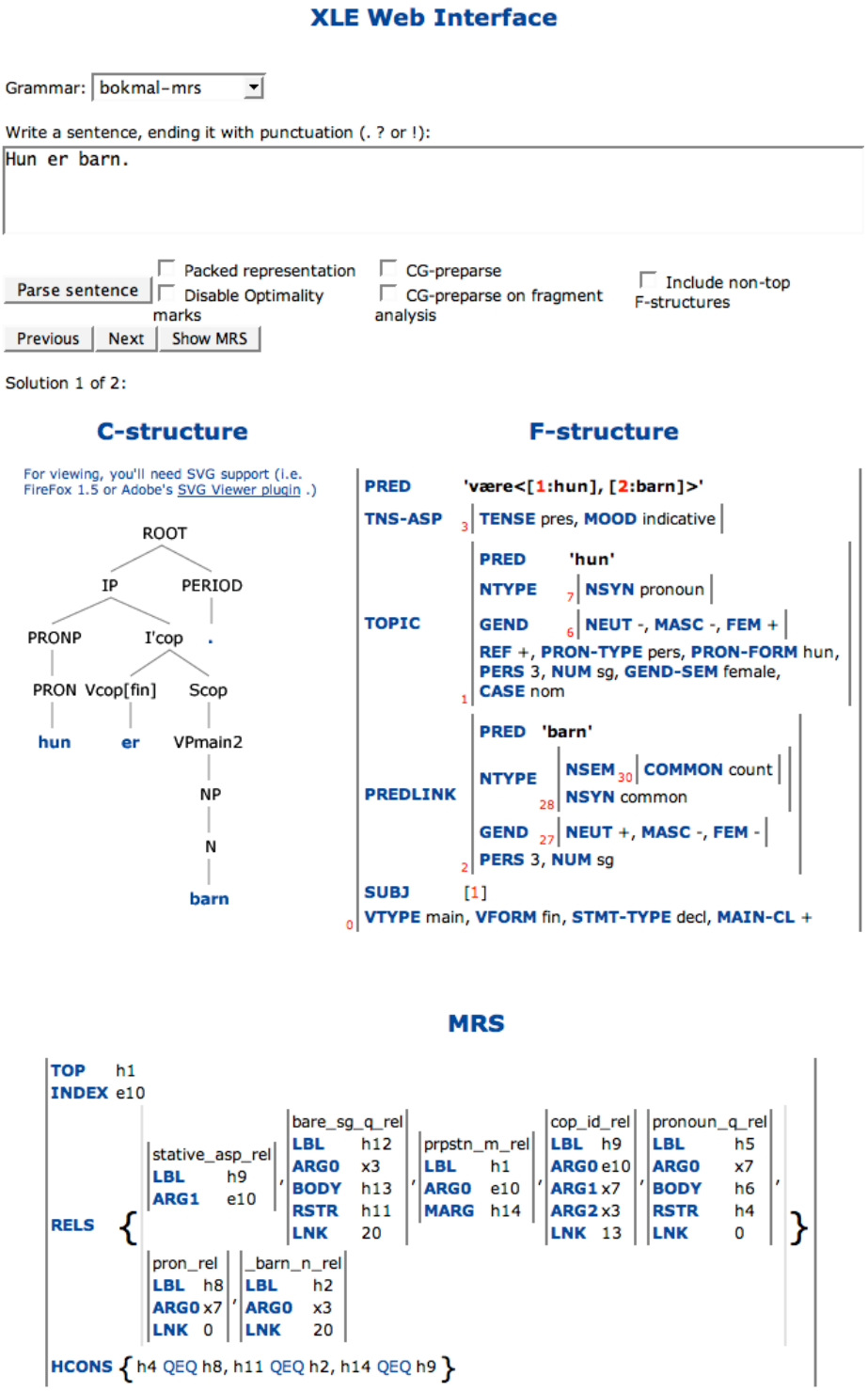

Figure 4: Screenshots from XLE-Web

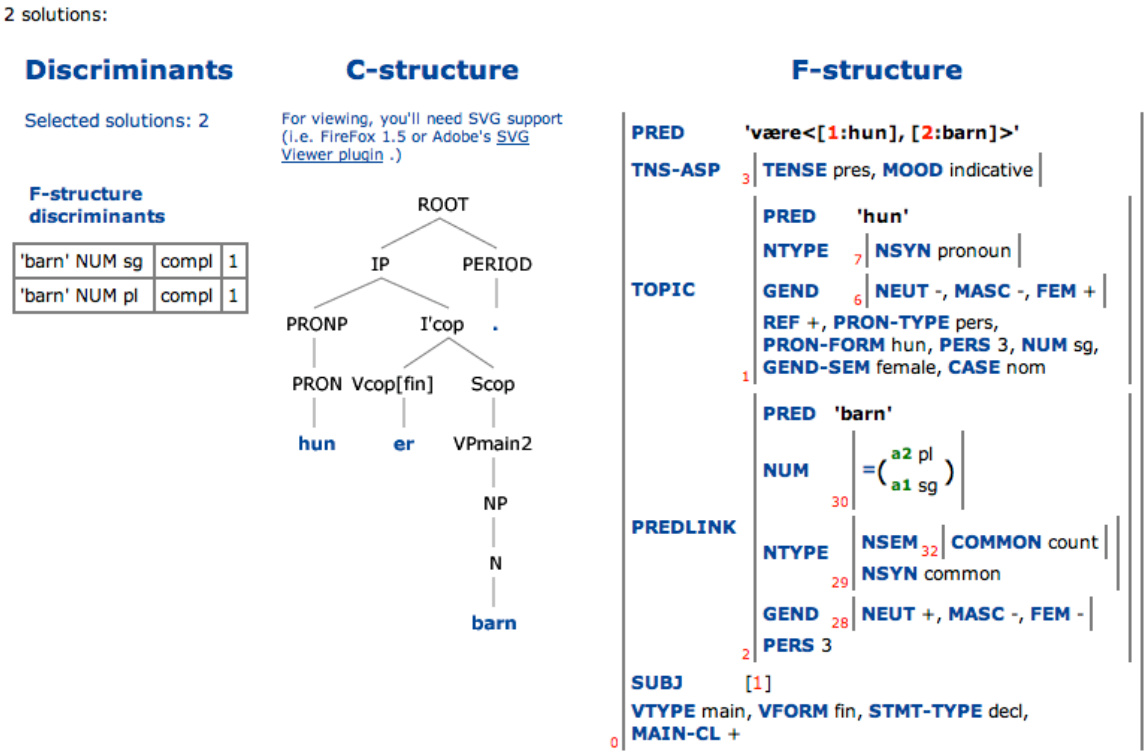

Figure 5: Packed representation of (1) *Hun er barn.*

In LFG terms a discriminant is, in general, any local property of a c-structure or f-structure that not all analyses share. We have implemented three types of discriminants in TREPIL: c-structure discriminants, f-structure discriminants and morphology discriminants. A c-structure discriminant is the segmentation of a surface constituent string induced by a minimal subtree (a node with its immediate subnodes); in addition, the rule that gives rise to this subtree is a discriminant. An f-structure discriminant is a direct path in an f-structure from a PRED value to an embedded PRED value or from a PRED value to an atomic value. A morphology discriminant is a word with the tags it receives from morphological preprocessing. Examples of all three types of discriminants will be given below.

As a first step towards developing a treebanking tool for disambiguation, we have implemented discriminants in XLE-Web. To the left in figure 5 is a list of discriminants. This sentence has only f-structure discriminants. In this example, the discriminants are paths from a PRED value to two alternative atomic values. Next to each discriminant in the display there are two columns, one where it says *compl* (for 'complement') and one with a number. The disambiguator may choose a discriminant by clicking on it, or reject a discriminant by clicking on *compl*. The number in the third column gives the number of analyses that will remain if that discriminant is chosen. When a discriminant choice has been made, the chosen discriminant is boldfaced, and only the discriminants still compatible with that choice are redisplayed. Since there is only one local ambiguity in this sentence, it will be fully disambiguated after one discriminant choice has been made.

The sentence in example 2 has, as mentioned above, two local ambiguities and four analyses. Figure 6 shows the seven discriminants which distinguish these analyses from each other and, in addition to a packed

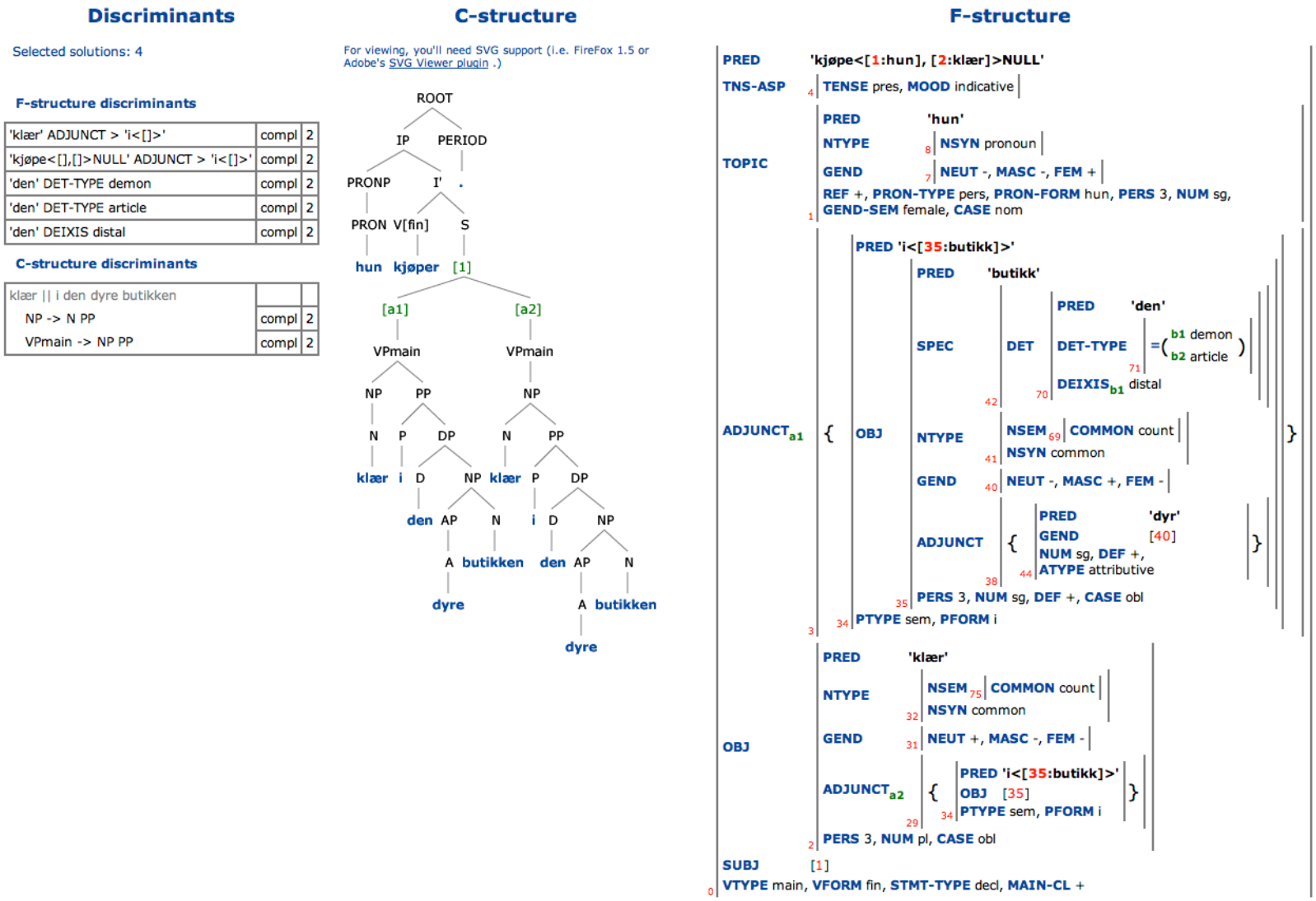

Figure 6: Packed representation for (2) *Hun kjøper klær i den dyre butikken.*

f-structure, also a packed c-structure. For this example, the PP attachment ambiguity is reflected in both c-structure and f-structure discriminants. At the f-structure level, the disambiguator can choose the first discriminant, which may be read: the PRED 'klær' ("clothes") has an ADJUNCT whose PRED is 'i' ("in"), or the second discriminant, which may be read: the PRED 'kjøpe<[ ],[ ]>NULL' ("buy" with two arguments) has an ADJUNCT whose PRED is 'i' ("in").

Alternatively, the disambiguator may choose one of the c-structure discriminants. The bracketing (represented by ||) of the string in the c-structure discriminant table in figure 6 may be attributed either to the NP rule or the VPmain rule listed underneath this string. If the annotator decides to disambiguate the PP attachment by choosing the *complement* of the first c-structure discriminant (NP → N PP), a new set of discriminants and structures will be displayed as in figure 7. There are now only f-structure discriminants left for distinguishing between the two readings of *den* as either demonstrative or article. If the article reading is chosen, the interface redisplays the fully disambiguated structures as in figure 8.

In addition to c-structure and f-structure discriminants, we have also implemented morphology discriminants. As mentioned above, a morphology discriminant is a word with the tags it receives from morphological preprocessing. Consequently, only words that have morphological features receive morphology discriminants. Consider example 4, which has many possible readings due to multiple lexical ambiguities. The *noun* readings of the ambiguous words receive the morphological discriminants shown in 5. By choosing the complement of each of these discriminants, we can eliminate all noun readings, thereby reducing the number of analyses from 45 to 6.

(4)

To	*av*	*disse*	*ga*	*henne*	*tre.*
two/stuff	of	these/swing	gave	her	three/wood

(5)

to+SP+Noun+Neut+Indef	compl	18
disse+Sg+Noun+MF+Indef	compl	18
tre+SP+Noun+Neut+Indef	compl	30

We have demonstrated that even in sentences with a small number of analyses, discriminant disambiguation is easier and more efficient for a human disambiguator than examining full analyses. The true power of discriminant analysis becomes apparent when one considers sentences with a large number of analyses. The previous example showed that 39 analyses could be eliminated through three simple discriminant decisions. Consider also sentence 6, which has many local ambiguities.

(6)

Sjefen	*har*	*drevet*	*og*	*sendt*	*invitasjoner*	*til*	*alle*	*han*	*kjenner.*
boss-the	has	driven	and	sent	invitations	to	everyone	he	knows

"The boss has been sending invitations to everyone he knows."

This sentence gets 86 solutions. The number of discriminants is also large: there are 5 c-structure discriminants, 14 morphology discriminants, and 77 f-structure discriminants, which is too large a number to be shown here. However, it may be fully disambiguated to the intended analysis by making choices concerning only two discriminants, for instance those shown in examples 7 and 8. The discriminant in 7 chooses the pseudo-coordination analysis of the progressive, while the one in 8 specifies that the noun *sjef* "boss" is the subject of the verb *sende* "send".

(7) 'sende<[ ],[ ],[ ]>' TNS-ASP ASP progressive

(8) 'sende<[ ],[ ],[ ]>NULL' SUBJ 'sjef'

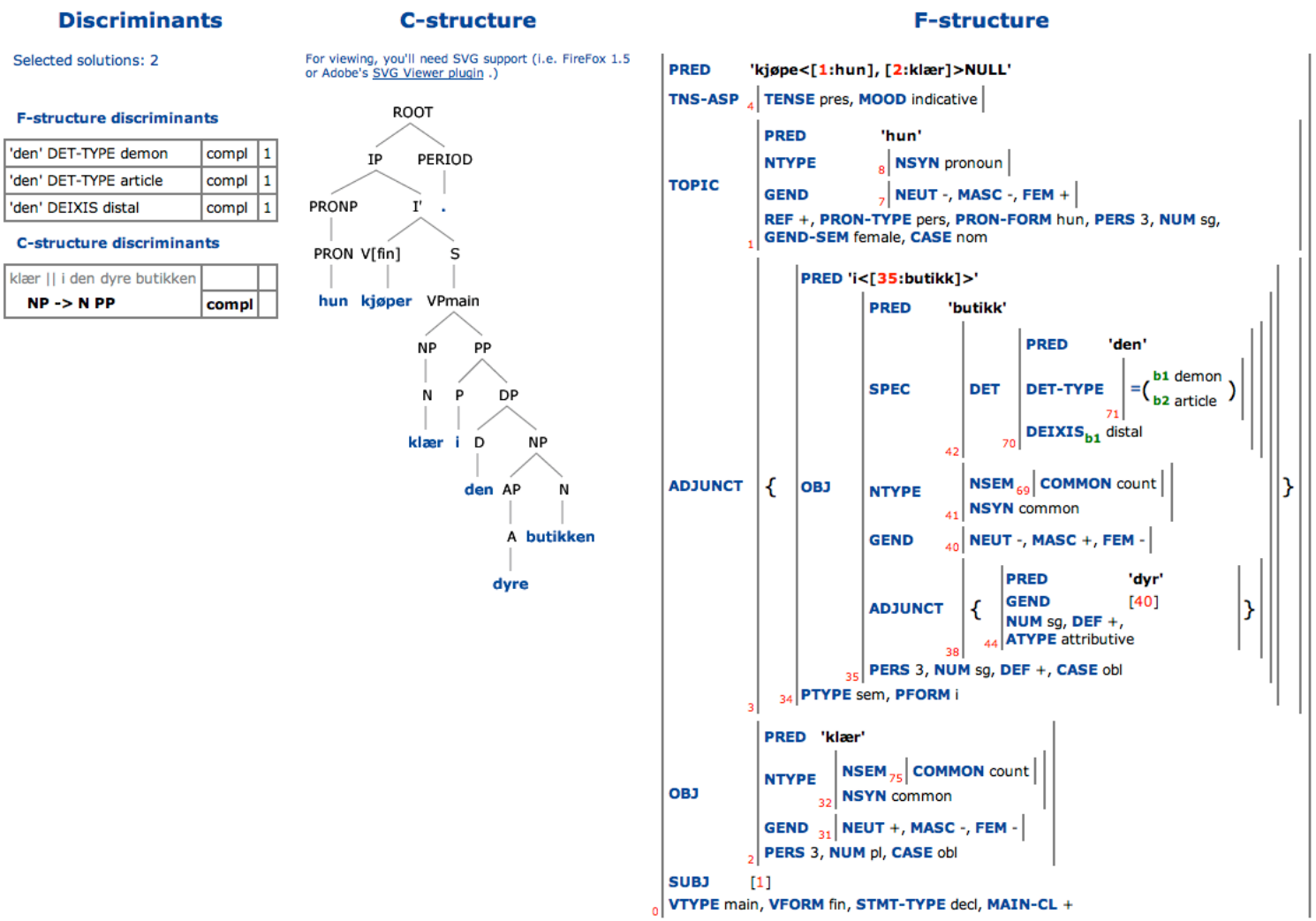

Figure 7: Partially disambiguated structures for (2) *Hun kjøper klær i den dyre butikken.*

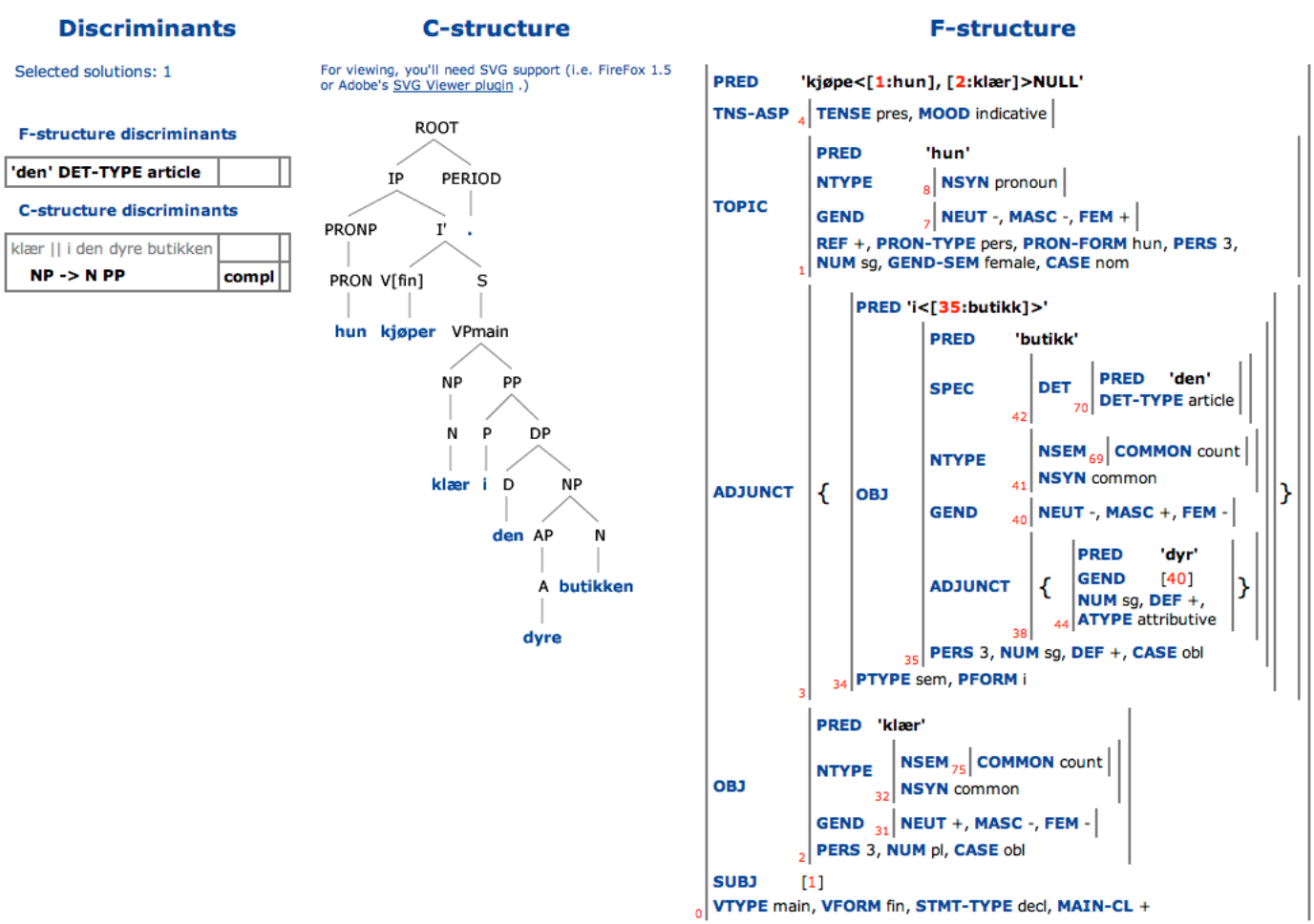

Figure 8: Fully disambiguated structures for (2) *Hun kjøper klær i den dyre butikken.*

6 Conclusion and further work

The TREPIL project is aimed at a methodology for the incremental development of a treebank in synchrony with a wide-coverage LFG grammar. This methodology is dependent on the XLE parser in conjunction with a disambiguation tool for recording all structural choices when the annotator selects one parse over other parses. To our knowledge, the TREPIL project is the first to develop a discriminant-based tool for LFG.

The XLE Web Interface provides either a browsing display or a packed representation, and has been extended with discriminants in TREPIL. When the packed representation is used for disambiguation, it is gradually unpacked as discriminants are chosen until disambiguation is complete and only one analysis is left. This tool is grammar and language independent, so that other LFG grammars developed on the XLE platform will be able to use it to create their own parsed corpora.

We have shown that a small number of discriminant choices may be sufficient to disambiguate a large number of analyses. It is therefore unnecessary to display all discriminants at one time. We will experiment with different ways of presenting selected discriminants to the annotator. Another important kind of functionality will be that the annotator should be able to record the degree of confidence with which decisions have been made.

One of the most interesting aspects of disambiguation by local discriminants is that such annotator decisions may be saved in a database and reused for automatic disambiguation after the grammar has been revised (Carter, 1997; Oepen et al., 2003). The decisions on local properties have been shown to be remarkably stable over revisions of the grammar. This means that the treebank may be produced in new versions as the grammar develops. This solves one serious problem with many treebanks, namely that they become obsolete as linguistic theories evolve. A treebank that can be updated semiautomatically as the grammar (and the theory behind the grammar) evolves is therefore dynamic. This approach has been followed in the Redwoods initiative (Oepen et al., 2003), and it will also be the aim in TREPIL.

7 Acknowledgments

This work was supported in part by a grant from the Research Council of Norway. We would like to thank John Maxwell at PARC, who has always been willing to discuss implementation issues and has provided extensions to the XLE software as we needed them.

References

Abeillé, Anne. 2003. Introduction. In Anne Abeillé, editor, *Treebanks: Building and Using Parsed Corpora*. Kluwer Academic Publishers, Dordrecht, pages xiii–xxvi.

Abeillé, Anne, Lionel Clément, and François Toussenel. 2003. Building a treebank for French. In *Treebanks: Building and Using Parsed Corpora*. Kluwer Academic Publishers.

Bick, Eckhard, Heli Uibo, and Kaili Müürisep. 2004. Arborest – a VISL-style treebank derived from an Estonian Constraint Grammar corpus. In *Proceedings of the 3rd Workshop on Treebanks and Linguistic Theories*. Seminar für Sprachwissenschaft, Universität Tübingen.

Bouma, Gosse. 2004. Treebank evidence for the analysis of PP-fronting. In *Third Workshop on Treebanks and Linguistic Theories, Seminar für Sprachwissenschaft, Tübingen, 2004*, pages 15–26.

Burke, Michael, Aoife Cahill, Ruth O' Donovan, Josef Van Genabith, and Andy Way. 2004a. Treebank-based acquisition of wide-coverage, probabilistic LFG resources: Project overview, results and eval-

uation. In *The First International Joint Conference on Natural Language Processing (IJCNLP-04), Workshop "Beyond shallow analyses – Formalisms and statistical modeling for deep analyses", March 22-24, 2004 Sanya City, Hainan Island, China*.

Burke, Michael, Aoife Cahill, Ruth O'Donovan, Josef Van Genabith, and Andy Way. 2004b. Evaluation of an automatic f-structure annotation algorithm against the parc 700 dependency bank. In *Proceedings of the LFG04 Conference, Christchurch, New Zealand*.

Butt, Miriam, Helge Dyvik, Tracy Holloway King, Hiroshi Masuichi, and Christian Rohrer. 2002. The Parallel Grammar project. In *Proceedings of COLING-2002 Workshop on Grammar Engineering and Evaluation, Taipei, Taiwan*.

Cahill, Aoife. 2004. *Parsing with Automatically Acquired, Wide-Coverage, Robust, Probabilistic LFG Approximations*. Ph.D. thesis, School of Computing, Dublin City University.

Carter, David. 1997. The TreeBanker: A tool for supervised training of parsed corpora. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 598–603, Providence, Rhode Island.

Copestake, Ann, Dan Flickinger, Ivan A. Sag, and Carl Pollard. in preparation. Minimal Recursion Semantics: An introduction. Manuscript.

Hajič, Jan. 1998. Building a syntactically annotated corpus: The Prague dependency treebank. In *Issues of Valency and Meaning*. Karolinum, Praha, pages 106–132.

King, Tracy Holloway, Richard Crouch, Stefan Riezler, Mary Dalrymple, and Ronald M. Kaplan. 2003. The PARC 700 dependency bank. In *Proceedings of the 4th International Workshop on Linguistically Interpreted Corpora, held at the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03), Budapest*.

Marcus, Mitchell, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The Penn Treebank: Annotating predicate argument structure. In *Proceedings of the ARPA Human Language Technology Workshop*.

Nivre, Joakim, Koenraad De Smedt, and Martin Volk. 2005. Treebanking in northern europe: A white paper. In Henrik Holmboe, editor, *Nordisk Sprogteknologi 2004. Årbog for Nordisk Sprogteknologisk Forskningsprogram 2000-2004*. Museum Tusculanums Forlag, Copenhagen, pages 97–112.

Oepen, Stephan, Helge Dyvik, Jan Tore Lønning, Erik Velldal, Dorothee Beermann, John Carroll, Dan Flickinger, Lars Hellan, Janne Bondi Johannessen, Paul Meurer, Torbjørn Nordgård, and Victoria Rosén. 2004. Som å kapp-ete med trollet? Towards MRS-based Norwegian–English Machine Translation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD, October.

Oepen, Stephan, Dan Flickinger, Kristina Toutanova, and Christopher D. Manning. 2003. LinGO Redwoods, a rich and dynamic treebank for HPSG. In Joakim Nivre and Erhard Hinrichs, editors, *Proceedings of the 2nd Workshop on Treebanks and Linguistic Theories*, pages 117–128. Växjö University Press.

van der Beek, Leonoor, Gosse Bouma, Jan Daciuk, Tanja Gaustad, Robert Malouf, Gertjan van Noord, Robbert Prins, and Begoña Villada. 2002a. Algorithms for linguistic processing: NWO PIONIER progress report, August 2002. Technical report, NWO.

van der Beek, Leonoor, Gosse Bouma, Robert Malouf, and Geertjan van Noord. 2002b. The Alpino dependency treebank. In *Computational Linguistics in the Netherlands (CLIN) 2001*, Twente University.

Zaenen, Annie. 2004. . . . But full parsing is impossible. *Elsnews*, 13(1):9–10.