ADAPTING STOCHASTIC LFG INPUT FOR SEMANTICS

Annette Hautli and Tracy Holloway King
Universität Konstanz Microsoft

Proceedings of the LFG09 Conference

Miriam Butt and Tracy Holloway King (Editors)

2009

CSLI Publications

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

Abstract

LFG c(onstituent)-structure and f(unctional)-structure analyses provide the detailed syntactic structures necessary for subsequent semantic analysis. The f-structure encodes grammatical functions as well as semantically relevant features like tense and number. The c-structure, in conjunction with the ϕ-mapping, provides the information on linear precedence necessary for semantic scope and anaphora resolution. In this paper, we present a system in which a stochastic LFG-like grammar of English provides the input to the semantic processing. The LFG-like grammar uses stochastic methods to create a c-structure and a proto f-structure. A set of ordered rewrite rules augments and reconfigures the proto f-structure to add more information to the stochastic output, thereby creating true LFG f-structures with all of the features that the semantics requires. Evaluation of the resulting derived f-structures and of the semantic representations based on them indicates that the stochastic LFG-like grammar can be used to produce input to the semantics. These grammars combine the advantages of LFG structures, e.g. the explicit encoding of grammatical functions, with the advantages of stochastic systems, e.g. providing connected parses in the face of less-than-ideal input.

1 Introduction

LFG c(onstituent)-structure and f(unctional)-structure analyses provide the detailed syntactic structures necessary for subsequent semantic analysis (Dalrymple, 1999, 2001). The f-structure encodes grammatical functions as well as semantically relevant features like tense and number. The c-structure, in conjunction with the ϕ-mapping, provides the information on linear precedence necessary for semantic scope and anaphora resolution.

LFG has also proven an excellent theory for use in computational linguistics due to its computational and mathematical tractability (Maxwell and Kaplan, 1989, 1993, 1996). Cross-linguistic theoretical and implementational work has resulted in large-scale LFG grammars for typologically varied languages (Butt et al., 1999, 2002). These LFG grammars face two challenges. First, some constructions may be outside the scope of the grammar. This can arise when the input is ungrammatical, e.g. contains typos, or when a construction is not covered by the grammar, e.g. the construction is too computationally costly or too rare to warrant inclusion in the grammar. Second, even highly efficient LFG implementations can be significantly slower than state-of-the-art stochastic parsers.

In this paper, we present a project where the output of a stochastic LFG-like grammar of English (Cahill et al., 2008) serves as input to the XFR semantic representation (Crouch and King, 2006), mapping f-structures into semantic representations. The XFR semantics is used for meaning-sensitive applications such as question answering (Bobrow et al., 2007) and search, expecting as input well-formed

[†]We thank Josef van Genabith and Jennifer Foster from Dublin City University for providing the initial development data and the Natural Language Theory and Technology group at PARC for providing the XLE LFG grammar and the XFR ordered rewrite system.

LFG c- and f-structures as created by the English ParGram grammar (referred to here as the XLE grammar) which runs on the XLE LFG parser (Crouch et al., 2009). The stochastic LFG-like grammar, created at Dublin City University and referred to here as the DCU grammar, uses stochastic methods to create a c-structure and a proto f-structure (Cahill et al., 2002). These proto f-structures do not necessarily obey LFG's completeness and coherence conditions, especially when long-distance dependencies are involved, and do not have all of the f-structure features that the XLE grammar provides.

Therefore, we augmented and reconfigured the output of the DCU grammar and created a set of rewrite rules that add the information needed to obtain true LFG f-structures with all of the features that the semantics requires (Hautli, 2009). For most open class items, general rules could be used; other lexical items, such as pronouns and determiners, required more specific, lexicalized rules to create the appropriate f-structure facts.

The project results suggest that the proto LFG structures of stochastic grammars such as the DCU grammar can be used for meaning-sensitive applications. They combine the advantages of LFG structures, e.g. the explicit encoding of grammatical functions in f-structure, in conjunction with the advantages of stochastic systems, e.g. providing connected parses in the face of less-than-ideal input. The initial results also suggest that stochastic grammars producing the proto LFG structures can be used when no XLE LFG grammar is available but a treebank of the language is: in such situations, it can be faster to create a stochastic grammar instead of a rule-based one (Cahill et al., 2005). When an XLE LFG grammar does exist, the DCU grammar can be used in conjunction with the XLE LFG grammar to replace it in out-of-coverage sentences. The XLE LFG grammars produce distinctive fragment parses when sentences are out of coverage (Riezler et al., 2002); these can be replaced by the DCU proto f-structures to provide spanning c- and f-structures.

Section 2 provides an overview of the rule-based XLE grammar and the DCU stochastic LFG-like grammar. The rewrite rules that apply to the DCU structures to create XLE-style f-structures are described in section 3, with its evaluation following in section 4. Section 5 discusses the approach and points to future work.

2 The Grammars

2.1 The English XLE Grammar

XLE is an efficient rule-based grammar development platform, developed by the Palo Alto Research Center (PARC). It consists of cutting-edge algorithms for parsing and generating Lexical-Functional Grammars, along with a user interface for writing and debugging such grammars (Crouch et al., 2009). The platform is also used in the ParGram project (Butt et al., 1999, 2002) for the development of parsers for several languages including Arabic, Chinese, German, French, Norwegian, Turkish, Urdu, and Welsh.

The English XLE LFG grammar is designed to handle well-edited English text (e.g. newspaper text, technical manuals) and is part of a larger system that maps text to an Abstract Knowledge Representation (AKR) (Bobrow et al., 2007) via the XLE parser and the XFR ordered rewrite system. The output of the system is used for applications such as search, question-answering (Bobrow et al., 2007), and redaction (Bier et al., 2009). The basic system pipeline is shown in Figure 1.

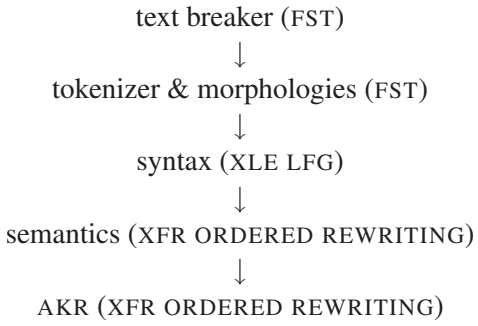

Figure 1: Pipeline architecture of the XLE-based system

The text is broken into sentences and words, using finite-state transducers (FST). The morphology analyzes each word and passes the information on to the broad-coverage XLE LFG grammar, which outputs c- and f-structure. These are then processed by the semantic and AKR rewrite rules.

Like all LFG grammars, the output of the syntax is a tree (c-structure) encoding linear order and constituency and an attribute-value matrix (f-structure) encoding predicate argument structure and semantically important features such as number and tense. These structures are more articulated than those usually found in LFG textbooks and papers because they contain all the features needed by subsequent processing and applications. Sample XLE c- and f-structures for *The boys hopped.* are shown in Figure 2.[1]

XLE outputs a packed representation of all possible solutions which allows subsequent processing to choose between different analyses for ambiguous sentences. In order for the grammar to be robust, XLE uses Optimality Theory marks (OT marks) in the syntax rules to indicate which analyses are dispreferred (Frank et al., 1998). In addition, the grammar can produce well-formed fragments if there is no analysis that spans the entire input (Riezler et al., 2002). The combination of these capabilities makes XLE robust in the face of ill-formed input and shortfalls in the coverage of the grammar.

[1]The XLE f-structures generally encode standard theoretical LFG f-structure features. The one exception to this are the CHECK features which are used primarily grammar-internally to constrain the application of specific syntactic constructions or to provide information useful for debugging. By convention, the names of these features begin with an underscore. As will be seen in §3.2, for the purposes of this project, the only CHECK feature of importance is the _SUBCAT-FRAME feature which the XFR semantics uses for lexical look-up.

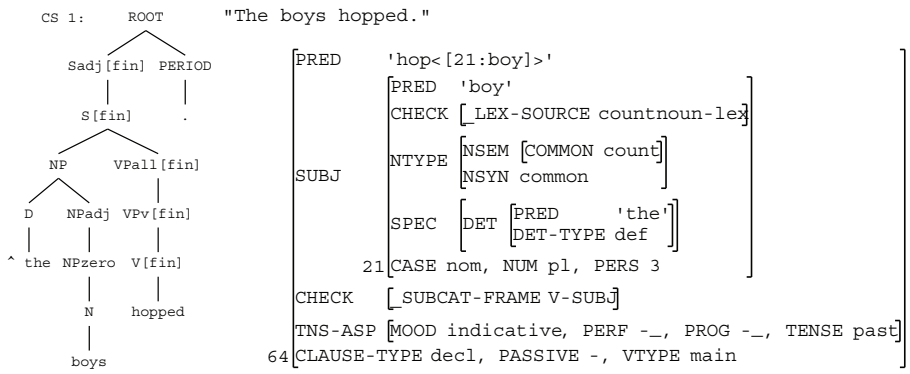

Figure 2: XLE output for *The boys hopped.*

On top of the syntactic c- and f-structure output, a semantic representation is derived using the XFR rewriting system. The semantic XFR system consists of rules that rewrite the syntactic structure to a semantic one, using external resources to replace words with concepts and grammatical functions with semantic roles (Crouch and King, 2006). In this paper, the XFR rewrite system is further used to rewrite DCU f-strctures to XLE-like f-structures (§3).

2.2 The DCU LFG-like parser

Extensive efforts at Dublin City University have resulted in the development of an automatic treebank annotation algorithm which annotates Penn-Treebank style trees (Marcus et al., 1994) with LFG f-structure information (Cahill, 2004). The annotated treebank can be used as a training resource for stochastic versions of unification and constraint-based grammars and for the automatic extraction of such resources (Cahill and McCarthy, 2002). The treebank is annotated such that solving the annotated functional equations produces LFG-like f-structures. The annotations describe what are called "proto-f-structures", which

- encode basic predicate-argument-modifier structures;
- may be partial or unconnected (i.e. in rare cases a sentence may be associated with two or more unconnected f-structure fragments rather than a single f-structure);
- may not encode some re-entrancies, e.g. in the case of *wh-* or other movement or distribution phenomena (e.g. of subjects into VP coordinate structures) (Cahill and McCarthy, 2002).

The basis of the annotation algorithm are treebank trees. These can either be Penn-II Treebank trees or, for novel text where there is no treebanked analysis, the output of a statistical parser such as that of Charniak (Charniak, 2000) or Bikel

(Bikel, 2002). After obtaining the tree, the nodes in the tree are annotated with f-structure equations. An example for *Boys hopped.* with the annotations is shown in Figure 3.

Unannotated tree:
(S (NP (NNS Boys)) (VP (VBD hopped)) (. .)

Annotated DCU **tree**:
```
(S
    (NP [up-subj=down]
          (NNS boys [up-pred='boy', up-num=pl,up-pers=3]))
    (VP [down-stmt_type=decl]
          (VBD hopped [up-pred='hop',up-tense=past])
   (. .))
```

Figure 3: Annotated DCU tree representation for *Boys hopped.*

After the annotation, all equations are percolated up the tree and unified at the topmost node. This process results in the f-structure in Figure 4.

$$\begin{bmatrix} \text{subj} & \begin{bmatrix} \text{num} & \text{pl}, \\ \text{pers} & 3, \\ \text{pred} & \text{boy} \end{bmatrix} \\ -1 & \begin{bmatrix} \text{pred} & \text{hop}, \\ \text{stmt\_type} & \text{declarative}, \\ \text{tense} & \text{past} \end{bmatrix} \end{bmatrix}$$

Figure 4: DCU f-structure for *Boys hopped.*

Evaluating the DCU annotation algorithm against existing gold standards shows that it can outperform hand-crafted, wide-coverage constraint grammars. The current DCU system achieves an f-score of 82.73 against the PARC 700 Dependency Bank (King et al., 2003), compared to 80.55% for the hand-crafted XLE LFG parsing system (Cahill et al., 2008). However, there are two issues with the f-structures produced by the DCU gramamrs. First, the PARC 700 Dependency Bank has a reduced feature set and contains only a subset of the features that are found in the very detailed ParGram f-structures. Therefore, the XFR semantics would fail due to missing f-structure features. Second, many of the features are present in the DCU f-structures in a different form than those of the XLE ones, and so they must be reformatted in order for the semantics to process them. Both of these problems will be illustrated in the next section.

To summarize, with the DCU LFG-like grammar and the XLE grammar, we have two different approaches to obtaining LFG analyses. On the one hand, the rule-based XLE grammar has very detailed feature structures, but faces coverage issues. On the other hand, the stochastic DCU grammar has the drawback of a less detailed f-structure, but with more connected parses. In this project we aimed to combine the advantages of both approaches.

3 Hybridization with XFR Augmentation Rules

The reasons for hybridizing the XLE-based system (Figure 1) are two-fold. In the case of English, the language used in this project, the stochastic grammar can be used in place of the rule-based grammar for out-of-coverage sentences, thereby supplying more connected input to the semantics. In the case of other languages, if no rule-based grammar is available but a treebank of the language is, it can be used to create a stochastic grammar for that language (Cahill et al., 2005).

3.1 The Overall Architecture

To produce the full, detailed f-structures needed by the semantics, we apply XFR rules to map DCU proto f-structures to XLE-style f-structures. The XFR ordered rewrite rules consume a set of input facts and replace it with another set of facts (§3.2). These rules can create a link between the stochastic DCU grammar and the rule-based XLE grammar output. The system using the DCU output as input for the XLE semantics is shown in the pipeline in Figure 5.

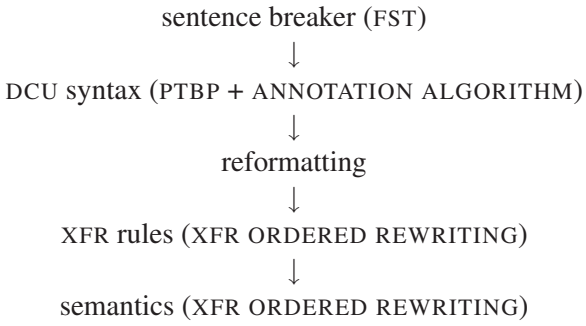

Figure 5: Hybridized pipeline

First, a sentence breaker splits running text into sentences, which are then processed by a probabilistic treebank based parser (PTBP) and annotated by the DCU annotation algorithm (§2.2). The DCU proto f-structure output is then reformatted by a script in order to be compatible with the input format expected by the XFR system. After that, the XFR ordered rewrite rules are applied to create XLE-style f-structures. In the final step, the rewritten f-structures are fed into the XFR semantic rules.

3.2 The XFR Rules

Input to the system is a set of facts representing the f-structures obtained by the DCU parser and the output is a set of rewritten facts representing the full f-structures that are fed into the XFR semantic system. The XFR system operates on a source f-structure and transforms it incrementally into the target structure. The order of the rules is important because each rule has the potential to change the set of input

facts that the subsequent rules will encounter: rules can prevent following rules from applying by removing facts that they would otherwise have applied to; they can also enable the application of later rules by introducing facts that these rules require. See the XLE documentation (Crouch et al., 2009) for details of the XFR system rule notation.

The rewriting works as follows: if a set of f-structure features (or part of an f-structure) matches the left-hand side of a rule, then the rule applies to produce the features on the right-hand side of the rule. A XFR rule which rewrites the DCU proto f-structure for the subject `boys` is shown in Figure 6.

input:
$$\begin{bmatrix} \text{subj} & \begin{bmatrix} \text{pred} & \text{boy} \\ \text{num} & \text{pl} \\ \text{pers} & 3 \end{bmatrix} \end{bmatrix}$$

XFR rule:

```
subj(%X,%Subj), pred(%Subj,%Pred), num(%Subj,%Num), pers(%Subj,3)
==>
SUBJ(%X,%Subj), PRED(%Subj,%Pred),
NUM(%Subj,%Num), PERS(%Subj,3),
NTYPE(%Subj,%Ntype), NSYN(%Ntype,common),
NSEM(%Ntype,%Nsem), COMMON(%Nsem,count).
```

output:
$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'boy'} \\ \text{NTYPE} & \begin{bmatrix} \text{NSEM} \begin{bmatrix} \text{COMMON} & \text{count} \end{bmatrix} \\ \text{NSYN} & \text{common} \end{bmatrix} \\ \text{NUM} & \text{pl} \\ \text{PERS} & 3 \end{bmatrix} \end{bmatrix}$$

Figure 6: Rewriting of the noun *boys*

The XFR rule in Figure 6 works as follows. The material before the arrow (==>) contains the input f-structure facts which must be matched for the rule to apply. The material after the arrow contains the f-structure facts created by the application of the rule. Forms beginning with a percent sign (`%`) are variables. For example, in Figure 6, the variable `%X` is the f-structure which contains a `subj`; that `subj` is then referred to by the variable `%Subj`. This `%Subj` f-structure must have `pred` attribute with value `%Pred` and a `num` attribute with value `%Num` in order for the XFR rule to match.

Given the input f-structure in Figure 6, the left-hand side of the rule goes through the list of XFR facts and matches with the `subj` fact, whose `pred` argument has the value `boy` and also matches the subject's `num` and `pers` attributes with their values. The rule rewrites these facts to those on the right-hand side of

the rule, resulting in the output f-structure shown in Figure 6. This is a very simple example of an ordered rewrite rule but the principle remains the same for more complicated constructions.

In total, the XFR system mapping from DCU to XLE f-structures consists of 162 rewrite rules.

In the remainder of this section we discuss several classes of issues that arose: the correction of core predicate-argument structures which did not conform to the XLE analysis or which were simply incorrect; the addition of default values which are necessary for the semantics; and the lexicalization of rules to provide features for particular predicates.

3.2.1 Core Predicate-Argument Structure

Of primary importance was correcting syntactic constructions where the core predicate-argument structure provided in the DCU f-structure differed from that in the XLE, and often in the theoretical LFG, analysis.

There were a few places where the original DCU analysis did not capture functional control as a re-entrant f-structure. For example, in the DCU analysis of sentences like *This seems to be a post-1990 problem.*, what traditional LFG analysis would consider a functionally controlled subject was represented only once within the f-structure, as the subject of the matrix verb. In order to correctly represent functional control, the identity relation between controller and infinitival subject is done by creating a re-entrant f-structure for the controlled SUBJ under the XCOMP. The creation of functional control structures is shown in Figure 7.

```
[ subj  [num sg, pred pro, pron_form this]
  xcomp [ xcomp [ adjunct {1 [adegree positive, pred post-1990]}
                  spec    [det [pred a]]
                  num sg, pers 3, pred problem ]
          pred be, to_inf + ]
 -1 [pred seem, stmt_type declarative, tense pres] ]
```

```
[ PRED    'seem<[-1-SUBJ:this], [-1-XCOMP:be]>'
  SUBJ    [ PRED  'this'
            NTYPE [NSYN pronoun]
            DEIXIS proximal, NUM sg, PERS 3, PRON-TYPE demon ]
  CHECK   [_SUBCAT-FRAME V-SUBJ]
  TNS-ASP [MOOD indicative, PERF -_, PROG -_, TENSE pres]
  XCOMP   [ PRED  'be<[-1-SUBJ:this], [-1-XCOMP-XCOMP:problem]>'
            SUBJ  [-1-SUBJ:this]
            XCOMP [ PRED    'problem<[-1-SUBJ:this]>'
                    SUBJ    [-1-SUBJ:this]
                    ADJUNCT {1 [ PRED 'post-1990'
                                 ADV-TYPE vpadv, DEGREE positive ]}
                    SPEC    [DET [ PRED     'a'
                                   DET-TYPE indef ]]
                    NUM sg, PERS 3 ]
            PASSIVE -, VTYPE main ]
 -1 [CLAUSE-TYPE decl, PASSIVE -, VTYPE main] ]
```

Figure 7: Predicate-Argument Structure: Creation of functional control for *This seems to be a post-1990 problem.*

A second set of phenomena where the core predicate argument structure had to be changed was imperatives and certain participial constructions. These structures lack subjects in the original DCU f-structures. This occurs because the f-structures produced by the DCU parser are not subject to the LFG completeness requirement whereby all arguments of a predicate must be present in the f-structure, even if they are not realized in the c-structure. These constructions were identified and the appropriate subject information was provided. An example is provided in Figure 8.

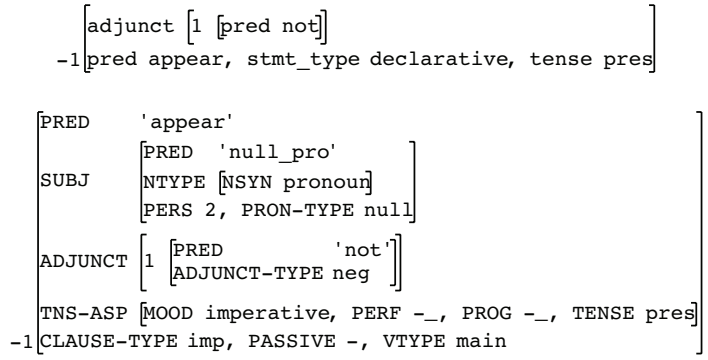

Figure 8: Predicate-Argument Structure: Insertion of a subject for *Do not appear*

3.2.2 Adding Default Values: Verbs

The addition of default features for f-structures was one of the most important tasks of the XFR mapping. This was particularly the case for verbs which govern many semantically-relevant features such as tense, mood, and aspect (TNS-ASP) and subcategorization frame information. All of these are used by the semantics rules and so must be present in the input f-structure.

The features such as `TENSE`, `VTYPE`, `MOOD` and `PASSIVE` posed a challenge, because of feature sparseness in the DCU structures. For example, the DCU grammars provide a `passive +` feature when the verb is passive, but no `passive -` feature when the verb is active. In the augmentation rules, these missing features and values are provided by judicious use of rules inserting default values.

The system also adds subcategorization features. The XLE grammar's verb lexicon has almost 9,800 verb stems with an average of 2.8 subcategorization frames each.[2] Subcategorization features are essential for the semantic lexical look-up that aids in mapping the verb's arguments to thematic roles in the semantics. The features dealing with subcategorization include the core argument structure of the PRED and a feature encoding the subcategorization frame name, which is not part of the DCU f-structures.

[2]Most of the frames were obtained from electronic dictionaries or manually. See O'Donovan et al. (2005) for ways to bootstrap creation of such lexical resources from treebanks.

An example for the sentence *He pushes it.* is shown in Figure 9. Notice the insertion of negative values for PASSIVE, PERF, and PROG as well as insertion of the _SUBCAT-FRAME feature.

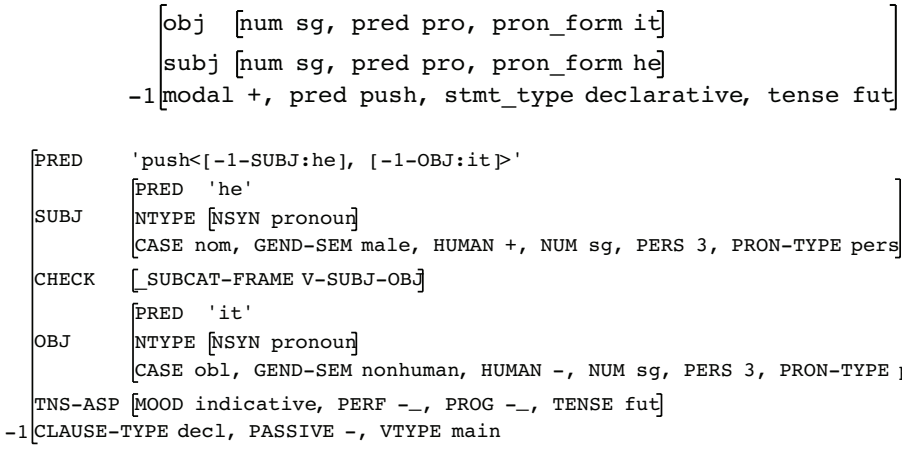

Figure 9: Default Values: Augmentation of verb-related features for *He pushes it.*

3.2.3 Lexicalization: Nouns and Pronouns

In some cases, more specific lexically-based information had to be provided in the f-structure. This arose when the semantics depends on f-structure features which are present in the XLE structures but not the DCU ones and which are not predictable from the general syntactic configuration.

For example, in the DCU grammar, proper nouns are correctly identified as such, but are not categorized by type. The morphology used in the XLE parser types many proper nouns (e.g. locations (*Detroit*, *France*), organizations (*IBM*, *Congress*), people (*Mary*, *Smith*), and gender for first names (*Mary* vs. *John*)). Such information is valuable for the semantic interpretation, especially for anaphora resolution and more accurate concept look-up. For this project, we extracted this information from the morphology and incorporated it into the XFR rules.

Similarly, many time-related nouns, such as months, days, and seasons, which the semantics expects to have identified with special f-structure date/time features, were lexicalized in the XFR rules, as the DCU output did not distinguish them from other nouns. However, all other nouns are accounted for by a general rule for modifying and inserting common noun features and rewritten accordingly.

An example for the rewriting of proper nouns (i.e. *Masha*) and time expressions (i.e. *fall*) is shown in Figure 10.

As a final example of lexicalization, personal, possessive, demonstrative, interrogative, and relative pronouns have to be mapped individually based on the lexical item due to the lack of relevant features on the DCU side. Examples of rewritten

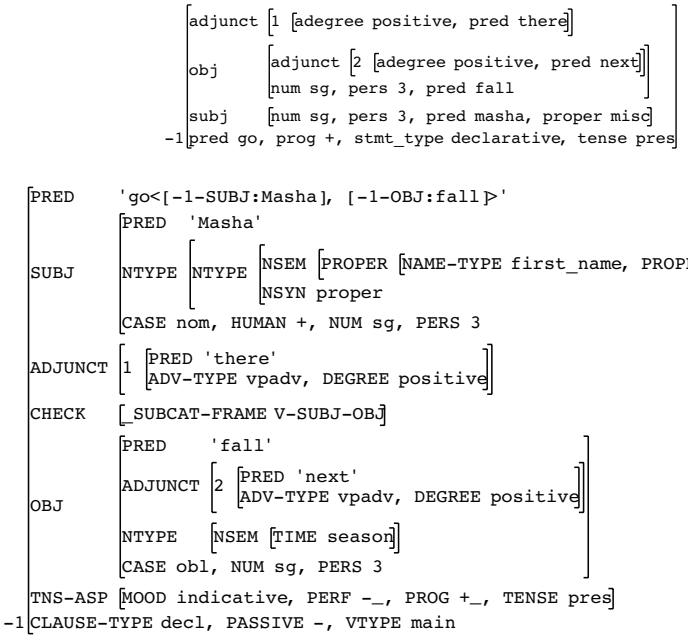

Figure 10: Lexicalization: Rewriting of proper nouns and time expressions for *Masha is going there next fall.*

pronouns were shown in Figure 9 for the personal pronouns *he* and *it*. In particular, note the addition of the PRON-TYPE, PERS, GEND-SEM, and HUMAN features.[3]

4 Development and Evaluation

To develop the XFR augmentation rules that map from the DCU proto f-structures to XLE-style f-structures, we created a testsuite of 430 sentences which covered core and some peripheral syntactic phenomena in English and which was based on the testsuites used by the XLE grammar developers to test syntactic coverage. For example, the testsuite covers syntactic phenomena such as extraposition, extraction, gerunds, sentential subjects and different clause types (declarative, interrogative and imperative sentences).

A schematic overview of the system development and evaluation is provided in Figure 11. The development and evaluation sentences were parsed by the XLE parser to obtain full f-structures. The same sentences were parsed by the DCU parser, creating proto f-structures. In the next step, augmentation ordered rewriting

[3]The XLE f-structures contain the nominative form of the pronoun as the PRED instead of the more commonly accepted PRED 'pro' familiar from the theoretical LFG literature and seen in the input DCU f-structure in Figure 9. This choice of PRED value for pronouns is independent of the issues in this paper, other than the fact that the XFR rewrite rules must be able to alter the PRED values correctly in order to create the structures that the XFR semantics expects and input.

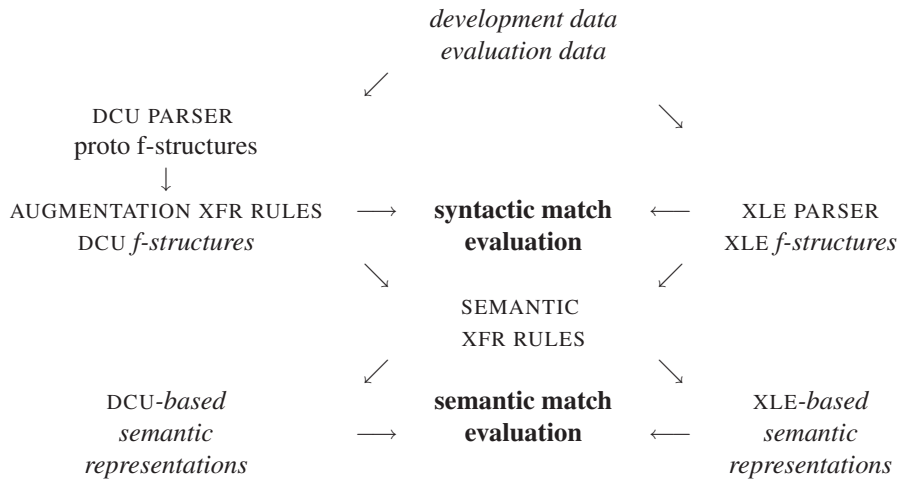

Figure 11: Overview of development (and evaluation)

rules reformatted and rewrote the DCU f-structures. The resulting f-structures were compared to those produced by the XLE parser using XLE's triples match (Crouch et al., 2009). The same process was done at the level of the semantic representations.

4.1 Evaluation Measures

To compare the f-structures and semantic representations, we use the standard evaluation measures *f-score, precision* and *recall*, which compare the features of the representations to be evaluated in relation to the features of a (often gold standard) reference set.[4] In this project, precision measures how many features in the transferred DCU f-structures are correct, whereas recall focuses on the completeness of the transferred DCU structures. Here precision is usually higher than recall; this is true for f-structures and for semantic representations. However, recall is the more important measure here as it shows how complete the transferred DCU structures are in relation to the original XLE structures.

4.2 Syntactic F-structure Evaluation

The f-structures in the transferred DCU f-structures were compared to those in the XLE f-structures. The results are shown below for declaratives, interrogatives, and imperatives.

[4]Precision and recall are widely used to evaluate the output of natural language processing systems. When a set of "test" items Y (rewritten DCU f-structures) is compared to a set of "reference" items X (original XLE f-structures), precision (X|Y) = $\frac{|X \cap Y|}{Y}$ and recall (Y|X)= $\frac{|X \cap Y|}{X}$ (Melamed et al., 2003) are measures to compare the output. The f-score is a weighted average of precision and recall.

F-Structure Matching of Declaratives The matching of declaratives varies quite a bit, depending on how many proper nouns are included in the sentence.[5] Expressions like *Eiffel Tower* and *President George W. Bush* initially get a proper noun feature from the DCU parser, but no information as to what kind of proper noun they are (e.g. location, person, etc.). In order to insert this information, one would have to list every item in the rewrite rules, something which cannot be done in the general case for unknown text. This is why sentences with proper nouns usually have a lower matching score.

Another difference between the DCU and XLE grammars is that the DCU parser always treats hyphenated forms as single units. For example, in the noun phrase *a high-interest loan*, the head noun *loan* has a single modifer *high-interest* in the DCU analysis. This loses certain semantic relationships which are needed for the semantic matching.

precision	recall	f-score
72.63	65.61	**68.94**

Table 1: Matching results for declaratives with proper nouns

If we consider sentences that do not contain proper nouns (e.g. *I'll go.* or *He laughed every third year.*), the results are as in Table 2. Note the significantly higher scores.

precision	recall	f-score
87.13	82.67	**84.84**

Table 2: Matching results for indicatives without proper nouns

Many clauses have a perfect f-score of 100, but this is countered by issues concerning coordination and the correct assignment of adjuncts in other sentences.

F-Structure Matching of Interrogatives A major issue is the DCU parsing of interrogative clauses. The training data for the DCU grammar is a corpus from the Wall Street Journal, which does not contain many matrix interrogatives. Due to the lack of training data, interrogatives are often analyzed incorrectly, e.g. the subject of the sentence is often analyzed as an object. Any mismatch in grammatical functions is a serious issue for the semantic processing. Judge et al. (2006) propose a method to add more interrogatives to the training data to alleviate this problems by building a QuestionBank. This bank consists of a corpus of 4,000 annotated questions used to train parsers in question answering technology and the evaluation of question parsing. Unfortunately, this DCU parser option was not available for this project.

[5]The analysis of proper nouns is particularly important for the semantics because of the applications it is used in. As such, the testsuites contain examples of proper nouns in order to assure that they are being correctly processed by the syntax and the semantics.

Thirty-one sentences of the development data were interrogative sentences. Matching the rewritten interrogatives against the original XLE interrogatives gives the results in Table 3. The reason for the relatively low matching figures is that these clauses get incorrect analyses due to the lack of interrogative sentences in the DCU training data.

precision	recall	f-score
45.17	43.15	**44.13**

Table 3: Matching results for interrogatives

F-Structure Matching of Imperatives As with interrogatives, imperatives are relatively rare in the Wall Street Journal corpus used to train the DCU parser. Our development set contains 25 imperatives. The figures for the matching of imperatives are higher than those for interrogatives, as more features could be added by the XFR rules and the analyses in general are closer to the XLE f-structures.

precision	recall	f-score
61.83	49.10	**54.74**

Table 4: Matching results for imperatives

4.3 Semantic Representation Evaluation

In order to determine whether enough information is included in the f-structures for them to be input to the semantic representations used by meaning-sensitive applications, we prepared an evaluation of semantic representations of 66 queries and answers, chosen for their coverage of phenomena of interest to the semantics (e.g. negation, anaphora).[6] These were parsed by both the DCU and XLE parser, then the DCU f-structures were processed by the XFR rules. Both XLE and DCU f-structures were used as input to the semantic system, and the resulting semantic representations were compared against each other. An example for a passage and query (and answer) is the following:

(1) P: Although Mary likes vegetables she eats them raw.
Q; Does Mary like vegetables?
A: Yes

The matching figures for the semantic representations of the passage and the query are shown in Table 5.

[6]These passage-query-answer pairs came from the regression sets (de Paiva and King, 2008) used in developing the question answering system. By using a regression set which was designed for an application that uses the XFR semantics, any changes in overall system performance could be more easily detected.

precision	recall	f-score
64.04	60.27	**62.10**

Table 5: Query-passage pair match results for the semantic representation

The overall figures for the semantic matching are lower than those in §4.2 for the syntactic matching due to the large number of interrogatives. However, on the positive side, some of the passage sentences are significantly longer than those in the development set, which had been chosen as representative of isolated syntactic phenomena. The ability to correctly process these data provides evidence that the hybrid system works on data which combines simple syntactic constructions into the complex sentences found in naturally occurring text.

5 Future Work and Discussion

Given the initial positive results of the project, the next step is to build a fully integrated hybrid DCU-XLE system (Figure 5) that can be run over large corpora and compare the results with those of the standard XLE system (Figure 1). Of particular importance is the behavior of the hybrid DCU-XLE system in application contexts. Having such a system raises some issues that were unimportant in the initial project. We address two of these here. We first discuss the issues arising from the different treatment of ambiguity in the two systems. We then discuss efficiency: back-of-the-envelope calculations show that the two systems should be roughly similar in efficiency, but this remains to be tested empirically.

5.1 Ambiguity

The XLE LFG grammar can efficiently produce multiple analyses for a given sentence (Maxwell and Kaplan, 1991). A maximum entropy model is applied to the output of the grammar to rank the parses (Riezler et al., 2002) and an n-best subset of the parses is then passed to the semantics. The more parses that are passed forward, the more processing that the semantics and AKR rules must perform, although the impact of this is mitigated by the ability of the XFR system to operate on the packed structures produced by the XLE grammar (Crouch, 2005). In fact, the XFR system uses the same packing mechanism and code that the XLE parser does. For meaning sensitive applications, the n-best, instead of the single best, parses are used in order to increase the chances that the correct parse is available.

On the DCU side, our system used the single parse produced by the DCU grammar. In theory, it would be possible to obtain ranked output from the DCU parser, e.g. by taking the n-best trees produced by the PTBG. In order for the semantics to operate on them efficiently, these parses would have to be packed. However, packing unpacked input can be difficult and inefficient. As such, the hybrid DCU-XLE approach seems best suited for applications and situations where a single parse

provides sufficient information. Search, as opposed to question-answering, is one possible application of this type.

5.2 Efficiency

The efficiency of the hybrid DCU-XLE approach was not systematically explored. The XLE system can process sentences in documents with an average of ∼20 words per sentence (e.g. Penn Treebank WSJ sentences) at less than a second from text to semantic output. Half of the time is spent on the syntax (i.e. creating the f-structure).[7] The exact percentage of time spent on the parsing step depends on how many parses are passed forward to the semantics rules: when more parses are passed forward, the processing by the XFR rules slows.

XLE has a number of performance variables that can be set to trade speed for accuracy (Crouch et al., 2009). The one-second-a-sentence results use relative aggressive settings with the result that ∼1.1% of the sentences time out or run out of memory.

This version of the XLE grammar uses c-structure chart pruning to trim the context-free c-structure forest before unification (Crouch et al., 2009). C-structure pruning eliminates a subtree if there is another subtree analysis available and if the subtree is significantly less probable than the most-probable subtree. The chart pruner uses a simple stochastic CFG model where the probability of a tree is the product of the probabilities of each of the rules used to form the tree. The probability of a rule is basically the number of times that that form of the rule occurs in the training data divided by the number of times the rule's category occurs in the training data, plus a smoothing term. If a subtree's probability is lower than the best probability by a given factor, then the subtree is pruned. This approach ensures that there is always at least one tree and that only highly improbable subtrees are eliminated. The resulting c-structure forest is often still very large, but it is often significantly smaller than the original one. Using c-structure pruning speeds the XLE parser by ∼40% for English, while maintaining accuracy.

The DCU parser runs with a similar level of efficiency and hence should not significantly change the speed of the overall system. In parsing the British National Corpus (BNC) (Wagner et al., 2007), which has an average sentence length of 18 words, the PTBG, annotation, and unification took an average of 1.48 seconds per sentence.[8] This longer per-sentence parse time is somewhat misleading because the parser in the DCU project in Wagner et al. (2007) was configured to provide analyses for all sentences, no matter how long, complex, or grammatical; if the occasional missed analysis is acceptable for a given application, more efficient processing settings can be used.

[7] Within the XLE LFG parser, the syntax time is roughly divided as: morphology (including the textbreaker and tokenizer) (4%), lexicon (6%), chart (25%), unifier (55%), completer (4%), solver (6%).

[8] Extremely long sentences take much longer to parse, as is also the case for the XLE parser.

The XFR rules used to map from the DCU output to the semantics input are relatively few in number and add a negligible amount of time to the processing.

5.3 Conclusion

This paper reported on a project to use a stochastic parser (the probabilistic DCU parser) that produces proto f-structures as the input to a semantic parser, in the place of a rule-based LFG parser (the XLE parser). The f-structures were augmented using a set of ordered rewrite XFR rules similar to the rules that create semantic structures. When evaluating the DCU-based system against XLE output on the f-structure level, the results are promising in that the DCU-based f-structures can be used by the semantics to produce well-formed semantic structures. This provides the opportunity to build hybrid systems using different grammar versions depending on their ability to parse the input data. The disadvantages of the DCU parser, which assigns fewer features to the proto f-structures, can be overcome by the XFR rules providing full LFG structures with detailed syntactic and semantic features identical to those produced by XLE LFG grammar.

As more researchers wish to build meaning-sensitive applications on top of ParGram-style XLE grammars, our work suggests that hybrid systems can be built using DCU grammars for the syntactic processing step (e.g. for Spanish for which there is a DCU ParGram grammar but no XLE one).

References

Bier, Eric, Chow, Richard, Gollé, Philippe, King, Tracy Holloway and Staddon, Jessica. 2009. The Rules of Redaction: Identify, Protect, Review (and Repeat). In *IEEE Volume on Access Control*, pages 46–53, IEEE Computer and Reliability Societies.

Bikel, Daniel M. 2002. Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine. In *The Proceedings of HLT 2002*.

Bobrow, Daniel G., Cheslow, Bob, Condoravdi, Cleo, Karttunen, Lauri, King, Tracy Holloway, Nairn, Rowan, de Paiva, Valeria, Price, Charlotte and Zaenen, Annie. 2007. PARC's Bridge and Question Answering System. In *Grammar Engineering Across Frameworks*, pages 46–66, CSLI Publications.

Butt, Miriam, Dyvik, Helge, King, Tracy Holloway, Masuichi, Hiroshi and Rohrer, Christian. 2002. The Parallel Grammar Project. In *Proceedings of COLING2002, Workshop on Grammar Engineering and Evaluation*, pages 1–7.

Butt, Miriam, King, Tracy Holloway, Niño, María-Eugenia and Segond, Frédérique. 1999. *A Grammar Writer's Cookbook*. CSLI Publications.

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

Cahill, Aoife, Burke, Michael, O'Donovan, Ruth, Riezler, Stefan, van Genabith, Josef and Way, Andy. 2008. Wide-Coverage Deep Statistical Parsing Using Automatic Dependency Structure Annotation. *Computational Linguistics* 34(1), 81–124.

Cahill, Aoife, Forst, Martin, Burke, Michael, McCarthy, Mairéad, O'Donovan, Ruth, Rohrer, Christian, van Genabith, Josef and Way, Andy. 2005. Treebank-Based Acquisition of Multilingual Unification Grammar Resources. *Journal of Research on Language and Computation; Special Issue on "Shared Representations in Multilingual Grammar Engineering"* pages 247–279.

Cahill, Aoife and McCarthy, Mairéad. 2002. Automatic Annotation of the Penn Treebank with LFG F-Structure Information. In *Proceedings of the LREC Workshop on Linguistic Knowledge Acquistion and Representation: Bootstrapping Annotated Language Data*, LREC Workshop on Linguistic Knowledge Acquisiton and Representation.

Cahill, Aoife, McCarthy, Mairéad, van Genabith, Josef and Way, Andy. 2002. Parsing with PCFGs and Automatic F-Structure Annotation. In *LFG02 Proceedings*, pages 76–95, CSLI On-line Publications.

Charniak, Eugene. 2000. A Maximum Entropy-inspired Parser. In *Proceedings of the 1st Meeting of the North American Chapter of the Association for Computational Linguistics*, pages 132–139.

Crouch, Dick. 2005. Packed Rewriting for Mapping Semantics to KR. In *Proceedings of the International Workshop on Computational Semantics*.

Crouch, Dick, Dalrymple, Mary, Kaplan, Ron, King, Tracy, Maxwell, John and Newman, Paula. 2009. XLE Documentation, http://www2.parc.com/isl/groups/nltt/xle/doc/.

Crouch, Richard and King, Tracy Holloway. 2006. Semantics via F-structure Rewriting. In *LFG06 Proceedings*, CSLI On-line Publications.

Dalrymple, Mary (ed.). 1999. *Semantics and Syntax in Lexical Functional Grammar*. The MIT Press.

Dalrymple, Mary. 2001. *Lexical Functional Grammar (Syntax and Semantics)*. Academic Press.

de Paiva, Valeria and King, Tracy Holloway. 2008. Designing Testsuites for Grammar-based Systems in Applications. In *Proceedings of the Coling 2008 Workshop on Grammar Engineering Across Frameworks*, pages 49–56.

Frank, Anette, King, Tracy Holloway, Kuhn, Jonas and III, John T. Maxwell. 1998. Optimality Theory Style Constraint Ranking in Large-scale LFG Grammars. In *LFG98 Proceedings*, CSLI On-line Publications.

Hautli, Annette. 2009. *Adapting Stochastic Output for Rule-Based Semantics*. Masters Thesis, University of Konstanz.

Judge, John, Cahill, Aoife and van Genabith, Josef. 2006. QuestionBank: Creating a Corpus of Parse-Annotated Questions. In *Proceedings of the Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*, pages 497–504.

King, Tracy Holloway, Crouch, Richard, Riezler, Stefan, Dalrymple, Mary and Kaplan, Ron. 2003. The PARC700 Dependency Bank. In *Proceedings of the EACL03: 4th International Workshop on Linguistically Interpreted Corpora (LINC-03)*.

Marcus, Mitchell, Kim, Grace, Marcinkiewicz, Mary Ann, MacIntyre, Robert, Bies, Ann, Ferguson, Mark, Katz, Karen and Schasberger, Britta. 1994. The Penn Treebank: Annotative Predicate Argument Structure. In *ARPA Human Language Technology Workshop*.

Maxwell, John T. and Kaplan, Ronald M. 1989. An Overview of Disjunctive Constraint Satisfaction. In *Proceedings of the International Workshop on Parsing Technologies*, pages 18–27.

Maxwell, John T. and Kaplan, Ronald M. 1991. A Method for Disjunctive Constraint Satisfaction. *Current Issues in Parsing Technologies* .

Maxwell, John T. and Kaplan, Ronald M. 1993. The Interface between Phrasal and Functional Constraints. *Computational Linguistics* 19, 571–590.

Maxwell, John T. and Kaplan, Ronald M. 1996. Unification-based Parsers that Automatically Take Advantage of Context Freeness. In *LFG96 Conference*.

Melamed, I. Dan, Green, Ryan and Turian, Joseph P. 2003. Precision and Recall of Machine Translation. In *Proceedings of HLT/NAACL*.

O'Donovan, Ruth, Burke, Michael, Cahill, Aoife, van Genabith, Josef and Way, Andy. 2005. Large-Scale Induction and Evaluation of Lexical Resources from the Penn-II and Penn-III Treebanks. *Computational Linguistics* 31, 329–366.

Riezler, Stefan, King, Tracy Holloway, Kaplan, Ron, Crouch, Dick, III, John T. Maxwell and Johnson, Mark. 2002. Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 271–278.

Wagner, Joachim, Seddah, Djamé, Foster, Jennifer and van Genabith, Josef. 2007. C-Structures and F-Structures for the British National Corpus. In *LFG07 Proceedings*, CSLI On-line Publications.