Treebank-Based Acquisition of LFG Parsing Resources for French

Natalie Schluter, Josef van Genabith

National Centre for Language Technology
School of Computing, Dublin City University, Dublin, Ireland
nschluter@computing.dcu.ie, josef@computing.dcu.ie

Abstract

Motivated by the expense in time and other resources to produce hand-crafted grammars, there has been increased interest in automatically obtained wide-coverage grammars from treebanks for natural language processing. In particular, recent years have seen the growth in interest in automatically obtained deep resources that can represent information absent from simple CFG-type structured treebanks and which are considered to produce more language-neutral linguistic representations, such as dependency syntactic trees. As is often the case in early pioneering work on natural language processing, English has provided the focus of first efforts towards acquiring deep-grammar resources, followed by successful treatments of, for example, German, Japanese, Chinese and Spanish. However, no comparable large-scale automatically acquired deep-grammar resources have been obtained for French to date. The goal of this paper is to present the application of treebank-based language acquisition to the case of French. We show that with modest changes to the established parsing architectures, encouraging results can be obtained for French, with a best dependency structure f-score of 86.73%.

1. Introduction

Large-scale hand-crafted grammars are notoriously expensive to produce and maintain. As a result, there has been increased interest in automatically obtained wide-coverage grammars from treebanks for natural language processing. In particular, recent years have seen the growth in interest in automatically obtained deep resources that can represent information absent from simple CFG-type structured treebanks and which are considered to produce more language-neutral linguistic representations, such as dependency syntactic trees. As is often the case in early pioneering work on natural language processing, English has provided the focus of first efforts towards acquiring deep-grammar parsing resources (see for example, (Cahill et al., 2002; Cahill et al., 2004; Hockenmaier and Steedman, 2002; Miyao and Tsujii, 2002), followed by the successful treatments of, for example, German (Cahill, 2004; Hockenmaier, 2006), Japanese (Yoshida, 2005), Chinese (Guo et al., 2007) and Spanish (Chrupała and van Genabith, 2006a). However, no comparable large-scale automatically acquired deep-grammar resources have been obtained for French to date.

The goal of this paper is to present the application of treebank-based language acquisition to the case of French. We show that with modest changes to the established parsing architectures, encouraging results can be obtained for French, with an overall best dependency structure f-score of 86.73%.

We begin by presenting the linguistic framework within which we are conducting our research, Lexical Functional Grammar (Section 2.). We then give an overview of the adopted approach (Section 3.) and introduce the treebank resource, the Modified French Treebank (Section 4.). Following this, we discuss the adaption of the established acquisition method to French and present and discuss results obtained through the adapted method.

2. Lexical Functional Grammar

Lexical-Functional Grammar is a constraint based theory of language, whose basic syntactic architecture distiguishes two levels of representation : *c-structure* (constituent structure) and *f-structure* (functional structure) —c-structures correspond to traditional constituent tree representations, and f-structures to traditional dependency representations encoded in the form of an attribute value matrix.[1]

Consider, for example, the following sentence.

(1) John helped Mary

Sentence (1) would have the c-structure shown to the left in Figure 1, which corresponds to the f-structure shown to the middle in the same figure.

F-structures are minimal solutions to sets of functional equations such as $(f \;\; a) = v$, where f is an f-structure, a is an attribute, and v is the value taken by that attribute, possibly another f-structure.

These two levels of representation (f-structure and c-structure), for a given phrase, are explicitly related by a structural mapping, called the *f-description*, often denoted by ϕ, which maps c-structure nodes to f-structure nodes.

In the LFG framework, this mapping is given by functional annotations inserted into the c-structure tree, as in Figure 1 on the right.

The metavariables $\uparrow$ and $\downarrow$ refer to the f-structure of the mother node and that of the node itself, respectively. So that if node n is annotated $\uparrow=\downarrow$, then n's f-structure is mapped to the same f-structure as n's mother's f-structure. Also, if n has the annotation $\uparrow$OBJ$=\downarrow$, this means that the f-structure associated with n is mapped to the value of the mother's f-structure OBJ attribute. LFG also has equations for members of sets, such as $\downarrow\in\uparrow$ADJ, which states that the node's f-structure is mapped to an element of the mother's ADJ attribute.

An example of an annotated rule that is used in the derivation of the annotated tree in Figure 1 is the following.

$$S \longrightarrow \begin{array}{cc} \text{NP} & \text{VP} \\ \uparrow \text{SUBJ} =\downarrow & \uparrow=\downarrow \end{array}$$

[1] A detailed introduction to LFG may be found in, for example, (Dalrymple, 2001).

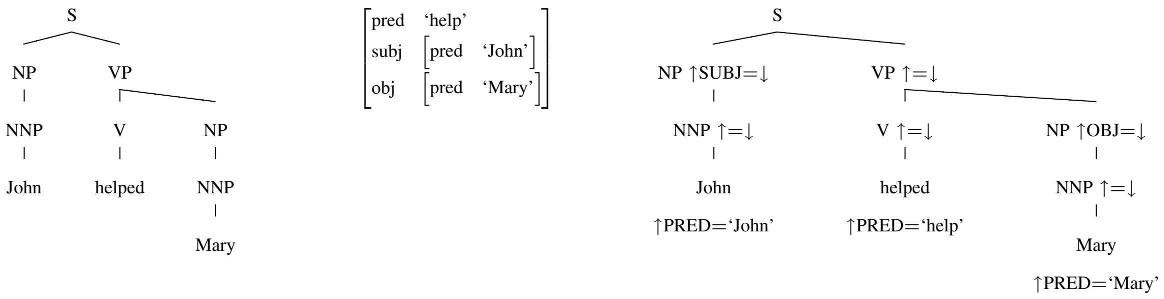

Figure 1: C-structure (left) and F-structure (middle) and Annotated C-structure for *John helped Mary*

3. Grammar Acquisition Methodology

3.1. Initial Stages

The technology for treebank-based acquisition of multilingual LFG probabilistic parsing resources is based on the English model (Cahill et al., 2004) and adapted to the language in question. There are six major stages in initiating this process, the basic input for which is a CFG-type treebank. These stages are as follows. (The adaption of these stages for French is the subject of this paper and will be discussed in the remainder.)

1. **Annotation Algorithm Application.** The treebank is automatically augmented with f-structure equations by means of an f-structure annotation algorithm.

2. **Satisfiability Verification.** These f-structure equations are extracted and sent to a constraint solver to be verified for consistency.

3. **Subcategorisation Frame Extraction.** A lexicon with subcategorisation frames are extracted from f-structures and these subcategorisation frames are associated with relative frequences conditioned on lemmata.

4. **Long-Distance Dependency Extraction.** If the treebank contained empty nodes and co-indexation to represent long-distance dependencies at the c-structure, the f-structure annotation algorithm would annotate these, and they would appear as long-distance dependencies at the f-structure level also. All long-distance dependencies are extracted from f-structures and associated with relative frequencies conditioned on the local (communicative) function (eg. *topic*, *focus*) of the reentrant f-structure in question.

3.2. Parsing Architectures

Given the resources produced in Section 3.1., there are two probabilistic parsing architectures, called the pipeline and the integrated architectures, as shown in Figure 3.2.

In the pipeline architecture, a probabilistic parser is trained on the treebank in its non-f-structure augmented format. The annotation algorithm then traverses parser output trees. The equations from these are extracted and sent to the constraint solver to verify satisfiability.

On the other hand, in the integrated architecture, the probabilistic parser is trained on the f-structure equation augmented treebank, and the f-structure equations from the output of this parser are sent to the constraint solver.

(Chrupała and van Genabith, 2006b) introduce a machine learning component into the pipeline architecture for those annotation algorithms that are based on treebank trees that incorporate some basic function labeling for, for example, subject, object, and so forth, the best out of the tested algorithm implementations being the support vector machine implementation of (Chang and Lin, 2001). This corresponds to an approximation of the basic Pipeline architecture.

In this paper, we introduce the analogue of this machine learning component for f-structure equations, which corresponds to an approximation of the basic Integrated architecture.

Probabilistic parser output trees do not contain empty nodes indicating long-distance dependencies. Therefore, under both architectures, long-distance dependencies are resolved, using the extracted subcategorisation and long-distance dependency information, at the f-structure level of representation.

4. The Modified French Treebank

As explained in Section 3., the approach for treebank-based acquisition of LFG parsing resources takes as input a treebank. For French, the treebank adopted is the Modified French Treebank (MFT) (Schluter and van Genabith, 2007).

The MFT is a cleaner and more consistent resource derived from restructured and modified trees of the Paris 7 Treebank (P7T) (Abeillé et al., 2004), consisting of Le Monde newspaper excerpts and annotated for CFG-type phrase structure. The MFT comprises the first half of the functionally annotated version of the P7T, for a total of 4739 sentences. The structures of the MFT differ to those of the P7T in several important aspects including the introduction of linguistic analyses for untreated phenomena in P7T guidelines, the completion of function tagging and the introduction of path function tags, and so forth, for which examples may be found in (Schluter and van Genabith, 2007). Also, it's construction has lead to improved statistical parsing of French for statistical parsers trained on MFT data, supporting the notion that the good quality of comparatively few training instances far outweighs the gains of five times the

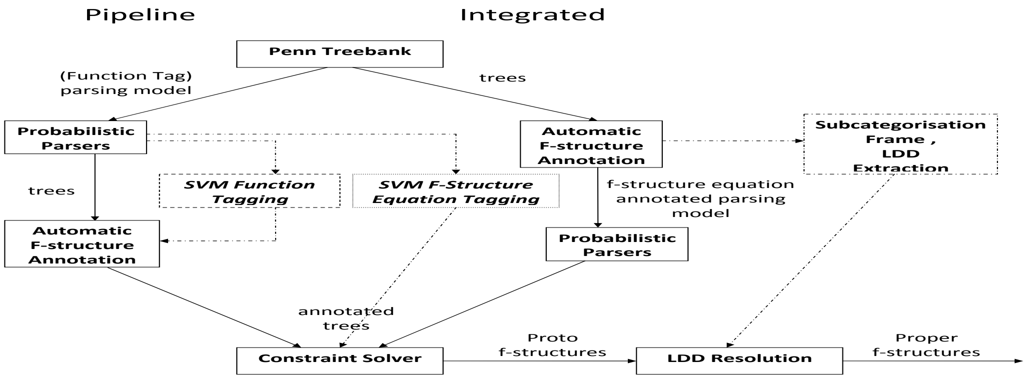
Pipeline
Integrated
Penn Treebank
(Function Tag) parsing model
trees
Probabilistic Parsers
Automatic F-structure Annotation
Subcategorisation Frame , LDD Extraction
trees
SVM Function Tagging
SVM F-Structure Equation Tagging
f-structure equation annotated parsing model
Automatic F-structure Annotation
Probabilistic Parsers
annotated trees
Constraint Solver
Proto f-structures
LDD Resolution
Proper f-structures

Figure 2: Overview of treebank-based LFG parsing architectures.

training data of relatively poor quality training instances from the P7T (Schluter and van Genabith, 2007).

4.1. Path Function Labels of the MFT

One important change carried out to the annotation scheme of the P7T by (Schluter and van Genabith, 2007) is the extension of the function tag set to include function path tags. Consider the sentence in Example (2), taken directly from the MFT, whose tree structure is given in Figure 3.

(2) C'est [...] l'URSS [...] qui se trouve prise
It is [...] the USSR [...] who herself finds taken
'It is the USSR that finds itself trapped'[2]

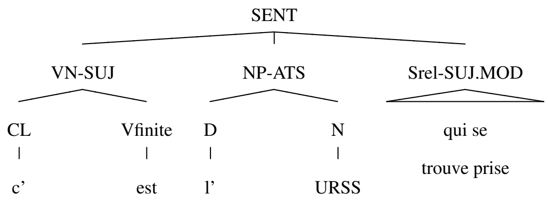

Figure 3: MFT representation of example (2).

In this example, the Srel constituent takes the functional path tag SUJ.MOD, representing the fact that Srel has the function MOD and is dependent on the constituent whose function is SUJ. Also, (Schluter and van Genabith, 2007) carry out the completion of the functional annotation of their derived subset of P7T sentences. The MFT now contains 30,399 functional tags, whereas the P7T only contained 23,772 (in the relevant subset). The completion and extension of the function tag annotation for the MFT was a crucial prerequisite of the f-structure annotation algorithm's LFG Conversion Module module (see Section 5.2.).

4.2. Data Partition

For this research, the MFT sentences were randomly distributed among the training set (3800 sentences), test set (430 sentences), and development set (509 sentences).

4.3. Parsing Results

The parsers employed in this research are a simple PCFG, BitPar (Schmid, 2004), and a lexicalised probabilistic parser—Bikel's implementation of Collins' parser (Bikel, 2002). Perfect POS tag mode parsing results (for parsing without MFT function labels), given in Table 1 show that Bikel's implementation of Collins' parser performs better (with a labelled f-score improvement of +6.82%) than BitPar for our small training set—a difference that is statistically significant with respect to precision and recall.[3]

parser	precision	recall	f-score
BitPar	77.67	78.4	78.03
Bikel	83.73	86.0	84.85

Table 1: Perfect POS tag mode basic CFG parsing results.

4.4. Clitic Constituents and MFT Function Labels

The adaption of the original approach to treebank-based LFG grammar acquisition to other languages is based on both linguistic and data-specific considerations. In terms of data-specific considerations, several important differences between MFT data structures and Penn-II data structures (on which the original technology is based) require special attention. This concerns the lack of empty nodes and coindexation in the MFT, as well as the lexicalised treatment of some personal pronouns (clitic pronouns).

MFT-norm and MFT-fct. The MFT inherited a relatively flat annotation from the P7T, including a lexicalised treatment for clitic pronouns. As such, clitic pronouns are not the head of any constituent, but are enclosed in the verb kernel (VN) with the verb lemma to which they are attached. Also, the function label for clitics is on the VN in which they are enclosed. F-structure equations may not be labelled in this manner in LFG; the predicate node cannot

[2] Sentence 8151, file flmf3_08000_08499ep.xd.cat.xml.

[3] Cf. (Schluter and van Genabith, 2007). Note that in this paper, all parsing results are obtained with parsers in perfect POS tag mode. Also, all CFG parsing will be evaluated using the PARSEVAL metric.

also be the value of the subject. In our parsing experiments for both the pipeline and integrated architectures, we have enclosed clitics in their own constituents and propagated the function labels down to this level. Figure 4 provides the transformed main VN constituent for the MFT tree appearing in Figure 3. The CLP transformed version of the MFT without function labels will be denoted by MFT-norm, and with function labels, by MFT-fct.

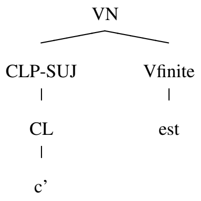

Figure 4: Clitic transformation for example (2).

MFT-comm. Unlike the Penn-II treebank, the MFT does not have any communicative function annotation such as *topic* or *focus*. Rather, whenever local c-structural constituents are not locally dependent on the constituent head, there is a function path indicating where this dependency is resolved (Section 4.1.). Therefore, we have had to derive a complementary treebank to implement the already adopted architectures of other languages–this version of the treebank has no function paths, and introduces the communicative function tags. This is carried out automatically, using the (f-structure) automatically annotated version of the treebank. We also carry out experiments using the original data set labels. The MFT-fct with communicative function annotation will be denoted by MFT-comm.

Parsing results for the MFT-norm, the MFT-fct, and the MFT-comm, are reported in Table 2. We notice in particular that Bikel's parser outperforms BitPar on all treebank versions. However, the quality of these parsed structures will be put into question in the light of the results on the derived f-structures in Section 6.

treebank version	parser	precision	recall	f-score
MFT-norm	BitPar	77.6	78.25	77.92
	Bikel	84.21	86.41	85.31
MFT-fct	BitPar	68.84	69.96	69.39
	Bikel	74.40	70.45	72.37
MFT-comm	BitPar	69.33	70.41	69.87
	Bikel	70.49	74.54	72.46

Table 2: Parsing results for three derived MFT versions.

5. An F-structure Annotation Algorithm for French

The goal of the f-structure annotation algorithm is to traverse the trees of the MFT, augmenting its nodes with LFG dependency annotations as in Figure 1, in order to derive the dependency representation given by the attribute value matrix in the middle of the same figure. The modules of the f-structure annotation algorithm developed for the MFT differ somewhat from those adopted for other languages. Figure 5 shows the structure of the annotation algorithm designed for French and the MFT.

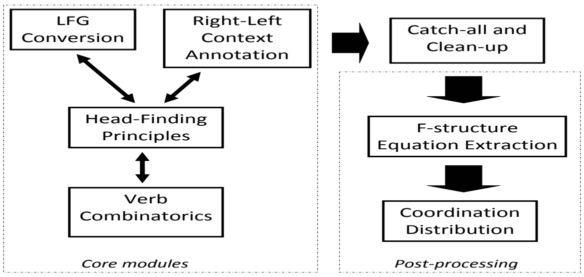

Figure 5: Modules of the F-structure Annotation Algorithm and Post-Annotation Algorithm Processing for French.

Because of the richness in morphological information inherited from the P7T as well as the extension and completion of the function tag annotation, our f-structure annotation algorithm relies less on equation decision heuristics than on simple translation of information already present in MFT trees; this is reflected in the elaboration of the lexical macros (Section 5.1.) and the LFG Conversion Module (Section 5.2.). In addition, to reflect the current linguistic analyses of verbs adopted in LFG, we provide a monoclausal treatment of compound verbs, resulting in the introduction of a Verb Combinatorics Module (Section 5.3.). Also, similarly to previous versions of annotation algorithms, our f-structure annotation algorithm employs left-right context principles (Section 5.4.) and a simple catch-all and clean-up module (Section 5.5.).

For a given tree, the annotation algorithm iterates over the list of n non-terminal nodes, T, which are in some order that maintains the dominance relation (root node downwards). For each non-terminal node $T(i), 1 \leq i \leq n$, $T(i)$'s daughters are annotated as follows. If $T(i)$ is a verb phrase, send $T(i)$ to the Verb Combinatorics Module. Otherwise, if $T(i)$'s head is a verb phrase, send $T(i)$ to the LFG Conversion Module. Otherwise, send $T(i)$ to the Left-Right Context Annotation Module. For the annotation of any leaf node, directly use the lexical macros. Once the list of nodes has been exhausted, it is sent to the Catch-all and Clean-up Module.

Following the application of the annotation algorithm, the functional equations are extracted from the trees, and sent to post-processing for distribution over coordination (Section 5.6.).

We will denote the f-structure annotated versions of the MFT-fct and MFT-comm by A-MFT-fct and A-MFT-comm, respectively.

5.1. Lexical Macros

The automatic f-structure annotation of terminal nodes of trees builds on the rich morphological and lemma information of the MFT, which is directly inherited from the P7T, from which it is derived. Morphological encodings are directly translated into lexical features, and lemma information provides predicate values during f-structure annotation.

5.2. LFG Conversion Module

MFT functional tags in are directly translated into LFG function equations with respect to the constituent under consideration. Table 3 gives a selection of simplified function macros adopted for the f-structure annotation algorithm. We note, in particular, the translation of the function path tags (such as SUJ.MOD), which permit the creation of complete LDD-resolved f-structures, rather than only proto-f-structures, unlike annotation algorithms designed for the other languages.

MFT Functional Tag	LFG F-structure Equation
SUJ	$\uparrow\text{SUBJ}=\downarrow$
DE-OBJ	$\uparrow\text{DE\_OBJ}=\downarrow$
SUJ.MOD	$\downarrow\in\uparrow\text{SUBJ:ADJUNCT}$
ATO	$\uparrow\text{XCOMP}=\downarrow$, $\uparrow\text{OBJ}=\downarrow\text{SUBJ}$

Table 3: MFT Function Tag Macros.

5.3. Verb Combinatorics Module

To produce f-structures similar to those produced by the current ParGram XLE systems (Butt et. al, 1999), in our f-structure analyses for the French annotation algorithm, we have made the move away from the multi-clausal treatment of compound tenses, elaborating a verb combinatorics module for a mono-clausal analysis.

The verb combinatorics module involves the provision of annotations for the various (finite number of) composed tenses with and without coordinated verbal parts. The analysis may be carried out over several levels of representation, depending on the presence of coordinated tenses. A verb in a compound tense is said to be made up of a verb complex, which may include some tensed component, an auxiliary (which may coincide with the tensed component), one or more past participles, and/or an infinitive. The features associated with a verb complex at the f-structure level of representation are factivity, passivity, auxiliary verb, tense, mood, (super-)perfectivity, infiniteness, and whether the verb is a participial or not.

5.4. Left-Right Context Annotation

Following the model of the annotation algorithm for English, the annotation algorithm for French makes use of a Left-Right Context Annotation Module which proceeds by first determining the head of a constituent with respect to head-finding rules provided by the authors, then by consulting annotation charts developed through the the analysis of the 85% most frequent rule types in the corpus. The charts encode information on how to annotate CFG node types to the left or right of the constituent head. This is combined with the simple rule that any argument annotation may be only be used once and therefore is given to the first compatible tag found. Such annotation principles support simple maintenance and development of this basic annotation module.

5.5. Catch-all and Clean-up Module

The catch-all and clean-up module handles any special annotation that was not covered earlier, and repairs any overgeneralisations in annotation. For instance, the statement type (declarative, interrogative, etc.) of the sentence is annotated in this module.

5.6. Coordination Distribution

Within the framework of LFG, individual conjuncts of an instance of coordination are generated by a production rule which places them as sisters among the right-hand side of the rule. The left-hand side is the type of coordination in question. Moreover, coordination at the c-structure level is isomorphic to the corresponding representation at the f-structure level (without considering shared dependencies within control structures or long-distance dependencies).

In the cases of verb phrase or non-constituent coordination, coordinates may share arguments or a predicate (including the predicate's lexical attributes). In such cases, the shared predicate or arguments are distributed among the coordinates at the f-structure level (Dalrymple, 2001). Distribution must be carried out for verb phrase (shared subject), predicate, argument-cluster, gapping, and right-node raising types of coordination, and is accounted for as a post-processing step among equations extracted from f-structure annotated MFT trees.

We have developed simple heuristics to detect these types of coordination and carry out the appropriate distribution action. Discussion of these heuristics must be omitted here due to space constraints. Such distribution is essential for long-distance dependencies through coordinated structures, control verb phrases with coordinated structures as verbal arguments (Burke, 2006), as well as for the extraction of lexical resources (O'Donovan et al., 2005). Previous versions of automatically acquired LFG parsing resources do not carry out any distribution over conjuncts and pay the price in performance. In Section 6., we present results on the performance of the annotation algorithm with and without this post-processing step. It turns out that distributing over coordination leads to statistically significant improvements on the performance of the annotation algorithm.

5.7. Construction of a Gold Standard

Functional equations are extracted and sent to a constraint solver to verify consistency. For completion of the 430 sentence gold standard, all consistent f-structures from the test set were hand verified and corrected, if necessary, twice by one of the authors. They were then compiled into dependency triples following (Crouch et al., 2002).

5.8. Evaluation

Currently the f-structure annotation algorithm associates 98.40% of the MFT's 4739 trees with a complete and connected f-structure. Also, as expected, on gold MFT trees, the annotation algorithm scores high ($>$ 96%), due to its extensive use of MFT macros for the direct translation of treebank information into f-structure equations. Scores with and without coordination distribution are reported in Table 4, for all f-structure features as well as for those paths that terminate in predicates (preds-only).

We notice that there is a drop in performance when the coordination distribution post-processing step is not carried out. In particular, recall drops from more than 3.3% in both

coord dist	features	precision	recall	f-score
no	all	99.10	96.48	97.77
	preds only	98.57	96.38	97.46
yes	all	99.42	99.8	99.61
	preds only	99.49	99.77	99.63

Table 4: F-structure annotation algorithm performance.

preds-only and full f-structures. This is clearly a statistically significant difference in recall (p-value < 2.2E-16 for both), and may partially explain the high score of this annotation algorithm with regards to previous versions.

6. Parsing Basic F-Structures

Parsing experiments, for the MFT-fct and A-MFT-fct, by the two basic pipeline and integrated architectures were carried out and extracted the f-structures evaluated. The results are presented in Tables 5 and 6.

coord dist	features	parser	precision	recall	f-score
no	all	BitPar	89.61	80.92	85.04
	all	Bikel	91.63	68.20	78.20
	preds	BitPar	78.93	72.24	75.44
	preds	Bikel	82.16	61.99	70.66
yes	all	BitPar	89.25	79.39	84.04
	all	Bikel	91.27	67.82	77.82
	preds	BitPar	79.10	71.22	74.96
	preds	Bikel	82.37	61.82	70.63

Table 5: MFT-fct pipeline architecture dependency evaluation.

coord dist	features	parser	precision	recall	f-score
no	all	BitPar	89.89	64.57	75.15
	all	Bikel	89.12	47.63	62.08
	preds	BitPar	78.71	58.35	67.02
	preds	Bikel	77.39	42.51	54.87
yes	all	BitPar	89.39	64.59	74.99
	all	Bikel	88.55	47.21	61.59
	preds	BitPar	78.81	58.62	67.23
	preds	Bikel	77.42	42.25	54.67

Table 6: A-MFT-fct integrated architecture dependency evaluation.

We make three important observations from these results. Firstly, the pipeline architecture consistently outperforms the integrated architecture. Secondly, we notice that though Bikel's parser outperformed BitPar in terms of c-structure labeled bracketing scores (Table 1), BitPar structures preserve better dependency relations between lemmata: the results show that f-structures extracted from BitPar parse trees are generally of better quality. Finally, we observe that coordination distribution post-processing almost never entails a boost in scores for parser output extracted f-structures. This is probably due to the difficulty in obtaining good parsing of coordinate structures in the first place, which may be compounded by the "explosion" of phrasal category tag set. Indeed there are 42 different constituent tags paired with up to 27 different function tags in MFT-fct for the parser to recognise.

7. Using Generic Machine Learning Techniques for Annotation

In the previous section we saw that higher PARSEVAL labeled bracketing scores did not necessarily correspond to better f-structures. The question remained, however, as to whether an explosion of the treebank phrasal category tag set was at fault. In the pipeline architecture we must train parsers on the regular MFT tag set with function labels and in the integrated architecture, parsers are trained on f-structure annotated MFT-norm trees.

Moreover, general results could perhaps be boosted if we could use parse trees from MFT-norm that were found to have an f-score of over 10% higher than the annotated treebank version parse trees.

(Chrupała and van Genabith, 2006b) introduced an approach to side-stepping this tag set explosion, enlisting machine learning of function tag information of the CAST3LB treebank. Several algorithms were tested, with support vector machines performing best. We therefore applied this method to the French case. In addition, we considered the application of this method to directly learning f-structure equations from annotated MFT trees.

As in (Chrupała and van Genabith, 2006b), we use the (Chang and Lin, 2001) implementation of SVM; we also use exactly the same feature information. The PARSEVAL evaluation results for the SVM approximations of MFT-fct (SVM-MFT-fct) and A-MFT-fct (SVM-A-MFT-fct) are presented in Table 7.

treebank version	parser	precision	recall	f-score
SVM-MFT-fct	BitPar	71.50	72.10	71.80
	Bikel	79.00	81.08	80.03
SVM-A-MFT-fct	BitPar	74.55	75.17	74.86
	Bikel	79.73	81.83	80.77

Table 7: PARSEVAL results for the SVM approximations.

We observe that while BitPar makes modest gains in f-score (around +2.5%) using this method, Bikel's parser remarkably gains over 5-7% in f-score. Corresponding advances in f-structure evaluation can be observed in Tables 8 and 9 for Bikel's parser for both parsing architectures, whereas BitPar only makes modest gains in the approximation of the pipeline architecture while quality decreases in the approximation of the integrated architecture. Interestingly, this suggests that BitPar is less influenced by larger constituent tag sets than Bikel's parser.

We also remark that the pipeline approximation outperforms the integrated approximation. Two possible explanations for this discrepancy are immediately evident. Firstly, the MFT-fct has manual (and therefore, probably, better quality) function tag annotation, to be learned by the SVM method. Moreover, there are much fewer function tags to be learned compared to f-structure annotations (27 function tag types as opposed to 69 f-structure annotation types), which makes learning f-structure equations the harder of the two tasks.

Under this method, the SVM approximation of the pipeline architecture, using Bikel's parser produces the highest f-score of 86.61% for all features and 81.35% for predicate paths.

coord dist	features	parser	precision	recall	f-score
no	all	BitPar	91.38	81.21	85.99
	all	Bikel	94.42	79.62	86.39
	preds	BitPar	82.89	74.37	78.41
	preds	Bikel	87.99	74.92	80.93
yes	all	BitPar	90.86	79.77	84.95
	all	Bikel	94.01	80.30	86.61
	preds	BitPar	83.00	73.27	77.84
	preds	Bikel	87.98	75.65	81.35

Table 8: SVM approximation of the pipeline architecture performance.

coord dist	features	parser	precision	recall	f-score
no	all	BitPar	91.97	58.59	71.58
	all	Bikel	93.26	57.61	71.22
	preds	BitPar	83.65	54.32	65.87
	preds	Bikel	87.20	54.14	66.80
yes	all	BitPar	91.56	57.26	70.46
	all	Bikel	93.09	58.96	72.20
	preds	BitPar	83.87	53.16	65.08
	preds	Bikel	87.51	55.45	67.88

Table 9: SVM approximation of the integrated architecture performance.

We also observe that the coordination distribution post-processing is only beneficial to Bikel's parser output under both approximation architectures. This suggests that Bikel's parser is better able to recognise coordination structures than BitPar on MFT-norm trees.

8. Long Distance Dependency Resolution

There are several types of grammatical features that may appear in f-structures. Of exceptional value are the communicative attributes topic, focus, and topic-rel, which represent the communicative organisation a given phrase. We will call those f-structures that are the values of these communicative attributes, communicative f-structures. Long distance dependencies (LDD) in LFG are those non-local f-structural dependencies between non-communicative f-structures and their re-entrancies as communicative f-structures. Therefore, under this linguistic framework, long-distance dependencies are resolved at the f-structural level of representation and not in c-structures. In standard LFG, this is carried out by means of functional uncertainty (FU) equations.

FUs are regular expressions which denote the set of proposed possible paths in an f-structure between a source communicative f-structure and a target non-communicative f-structure. For example, the equation $\uparrow$ topic $=\uparrow$ comp*comp reflects the fact that a topic f-structure may be resolved with a comp f-structure along some non-null path of comp attributes. Among these proposed possible paths, the only possible ones are those that maintain the principles of completeness and coherence in LFG with regards to f-structures. That is, the target's local predicate must subcategorise for the argument in question, and this argument must not already be filled.

A technique for the automatic resolution of long-distance dependencies, by detecting finite approximations of FUs, in English was first outlined by (Cahill et al., 2004). A finite approximation of an FU is a single path, rather than a set of possible paths. It is this technique for long-distance dependency resolution that we test for the French case, with slight adaptations due to the particularities of the MFT treebank encoding schemes.

In Section 4.4., we observed that the MFT has function path tags that are directly translated into f-structure equations. Moreover, it has no communicative function tags. These are added in the translation of MFT's function tags. So, the f-structures generated in the previous sections, using MFT-fct or SVM-MFT-fct (for the pipeline architecture), or A-MFT-fct or SVM-A-MFT-fct (for the integrated architecture), as for the treebank version, are in fact full f-structures. No further long distance dependency resolution is needed. The question remained, however, as to how efficiently the function path tags or functional uncertainty approximations are being recovered in during the parsing or prediction phase. We can test this, by rerunning our experiments on MFT-comm or A-MFT-comm.

The technique starts with the MFT-fct version of the MFT. This treebank includes path function tags that are exploited for simple annotation of long-distance re-entrancies at the f-structural level. These f-structures are complete f-structures; moreover, they are of high quality as the f-structure annotation algorithm performs with an f-score that is close to perfect. Subcategorisation information and long-distance dependency paths for each predicate are extracted from these f-structures and stored with their associated relative frequencies from treebank f-structures.[4] Armed with this information the resolution algorithm works for MFT-comm or A-MFT-comm derived f-structures as follows.

Given an f-structure of type $GF \in \{$focus,topic,topic-rel$\}$, the proposed possible paths associated with GF are retrieved. The list of paths is pruned with regards to the principle of coherence, to obtain a list of possible paths for GF. Each possible path, p, has a relative frequency, $P(p|GF)$. The end (or target) of the path p is in the f-structure of the predicate l. The principle of completeness requires that the predicate l subcategorise for the target of p. Therefore, the subcategorisation information for l is retrieved, each member s associated with a relative frequency $P(s|l)$. The path with the highest ranking $P(p|GF) \times P(s|l)$ is the resolution of the long-distance dependency for GF.

The results of the technique are given in Table 10. All scores are reported with coordination distribution.

We observe that the scores for BitPar are all poorer than in

[4] More on the types of subcategory information that are stored may be found in (Cahill et al., 2004).

the approach for MFT function tag-based LDD (on MFT-fct and A-MFT-fct). However, Bikel's parser in the pipeline architecture sometimes achieves slight gains from separating the process of LDD resolution.

treebank	coord dist	parser	precision	recall	f-score
pipeline	all	BitPar	90.44	77.00	83.18
	all	Bikel	91.67	66.81	77.29
	preds	BitPar	80.15	69.21	74.28
	preds	Bikel	82.46	60.71	69.93
integrated	all	BitPar	89.43	64.00	74.61
	all	Bikel	88.76	47.86	62.18
	preds	BitPar	78.72	58.15	66.89
	preds	Bikel	77.63	42.84	55.21
SVM pipeline	all	BitPar	90.33	80.82	85.31
	all	Bikel	93.62	80.78	86.73
	preds	BitPar	82.26	73.89	77.85
	preds	Bikel	87.40	75.71	81.13
SVM integrated	all	BitPar	91.27	57.43	70.50
	all	Bikel	92.37	58.95	71.97
	preds	BitPar	83.31	53.10	64.86
	preds	Bikel	86.59	55.17	67.39

Table 10: Parsing with LDD resolution.

9. Conclusions and Future Work

In this paper, we have shown that the techniques applied to other languages for treebank-based grammar acquisition may successfully be adapted to the French case. We have also acquired evidence that Bikel's parser is more sensitive to larger tag sets than BitPar.

An avenue for future work concerns further adaption of the LDD resolution for French. Indeed, not all path functions in the MFT-fct correspond to communicative local features. The phenomenon of *de*-phrase extraction from NPs provides the exception. Future research on treebank-based grammar acquisition for French should work towards an account of this phenomenon.

This research was supported by Science Foundation Ireland GramLab grant 04/IN/I527.

10. References

A. Abeillé, F. Toussenel, and M. Chéradame. 2004. Corpus le monde: Annotations en constituants. guide pour les correcteurs. Technical report, LLF and UFRL and Université Paris 7.

D. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the 2nd International Conference on Human Language Technology Research*, San Francisco.

M. Burke. 2006. *Automatic Treebank Annotation for the Acquisition of LFG Resources*. Ph.D. thesis, School of Computing, Dublin City University, Dublin 9, Ireland.

A. Cahill, M. McCarthy, J. van Genabith, and A. Way. 2002. Automatic annotation of the penn treebank with lfg f-structure information. In A. Lenci, S. Montemagni, and V. Pirelli, editors, *Proceedings of the LREC 2002 workshop on Linguistic Knowledge Acquisition and Representation*, Paris. ELRA.

A. Cahill, M. Burke, R. O'Donovan, J. van Genabith, and A. Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *Proc. of ACL04*, pages 21–26, Barcelona, Spain.

A. Cahill. 2004. *Parsing with Automaticaly Acquired, Wide-Coverage, Robust, Probabalistic LFG Approximations*. Ph.D. thesis, School of Computing, Dublin City University, Dublin 9, Ireland.

C. Chang and C. Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

G. Chrupała and J. van Genabith. 2006a. Improving treebank-based automatic lfg induction for spanish. In *Proc. of LFG06*, Konstanz, Germany.

G. Chrupała and J. van Genabith. 2006b. Using machine-learning to assign function labels to parser output for spanish. In *Proc. of COLING/ACL 2006 Main Conference Poster Sessions*, Sydney, Australia.

R. Crouch, R. Kaplan, T. Holloway King, and S. Riezler. 2002. A comparison of evaluation metrics for a broad coverage parser. In *Proc. of the LREC Workshop: Beyond PARSEVAL–Towards Improved Evaluation Measures for Parsing Systems*, pages 67–74, Las Palmas, Spain.

M. Dalrymple. 2001. *Lexical Functional Grammar*, volume 34 of *Syntax and Semantics*. Academic Press, San Diego.

Y. Guo, J. van Genabith, and H. Wang. 2007. Treebank-based acquisition of lfg resources for chinese. In *Proc. of LFG07*, Stanford, CA.

J. Hockenmaier and M. Steedman. 2002. Acquiring compact lexicalized grammars from a cleaner treebank. In *Proc. of the Third International Converence on Language Resources and Evaluation*, Las Palmas, Spain.

J. Hockenmaier. 2006. Creating a ccgbank and a wide-coverage ccg lexicon for german. In *Proc. of ACL/COLING 2006*, Sydney, Australia.

Y. Miyao and J. Tsujii. 2002. Maximum entropy estimation for feature forests. In *Proc. of HLT 2002*.

R. O'Donovan, M. Burke, A. Cahill, J. van Genabith, and A. Way. 2005. Large-scale induction and evaluation of lexical resources from the penn-ii and penn-iii treebanks. *Computational Linguistics*, 31:329–366.

N. Schluter and J. van Genabith. 2007. Preparing, restructuring, and augmenting a french treebank: Lexicalised parsers or coherent treebanks? In *Proc. of the PACLING 2007*, Melbourne, Australia.

H. Schmid. 2004. Efficient parsing of highly ambiguous context-free grammars with bit vectors. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland.

K. Yoshida. 2005. Corpus-oriented development of japanese hpsg parsers. In *Proc. of the 43rd ACL Student Research Workshop*.