f-align: An Open-Source Alignment Tool for LFG f-Structures

Anton Bryl
CNGL, School of Computing,
Dublin City University
abryl@computing.dcu.ie

Josef van Genabith
CNGL, School of Computing,
Dublin City University
josef@computing.dcu.ie

Abstract

Lexical-Functional Grammar (LFG) f-structures (Kaplan and Bresnan, 1982) have attracted some attention in recent years as an intermediate data representation for statistical machine translation. So far, however, there are no alignment tools capable of aligning f-structures directly, and plain word alignment tools are used for this purpose. In this way no use is made of the structural information contained in f-structures. We present the first version of a specialized f-structure alignment open-source software tool.

1 Introduction

The use of LFG f-structures in transfer-based statistical machine translation naturally requires alignment techniques as a prerequisite for transfer rule induction. The existing research (Riezler and Maxwell, 2006; Avramidis and Kuhn, 2009; Graham and van Genabith, 2009) uses general-purpose word-alignment tools such as GIZA++ by Och et al. (1999) for aligning the f-structures. The general-purpose alignment tools, however, take no advantage of the dependencies, which are made explicit in the f-structures, thus actually ignoring a lot of useful information readily available in the f-structure annotated data. This paper focuses on a way of making use of precisely this information.

A relevant algorithm was proposed by Meyers et al. (1998), who represent sentences with trees similar to f-structures. Their algorithm aligns a pair of trees in a recursive bottom-up procedure, ensuring that, somewhat simplifying, if a node A is aligned to a node B, the descendant nodes of A are aligned to the descendant nodes of B. The main reason we did not adapt this algorithm for our task are problems related to generalizing this approach. In the general case an f-structure is not a tree, but a directed acyclic graph (DAG). While the algorithm of Meyers et al. aligns two trees with n nodes and maximum degree d in $O(n^2d^2)$ time, we see no straightforward way of adapting it to DAGs without increasing complexity substantially. Another issue not to be ignored is that the output of f-structure parsers (Kaplan et al., 2002; Cahill et al., 2004) is often fragmented. Unlike incorrect parses, fragmented parses do carry useful information and their exclusion is undesirable.

The extensive existing work on phrase-structure tree alignment, starting with the work by Kaji et al. (1992) and proceeding to a number of more recent approaches (Ambati and Lavie, 2008; Zhechev, 2009), is also not straightforward to reuse, as LFG f-structures represent sentences in a way quite different from phrase-structure trees, in particular having all internal nodes and not only leaves (potentially) lexicalized; not to mention again that, in general, f-structures are graphs, and not necessarily trees.[1]

Therefore we decided to design a new algorithm. For the sake of computational speed, we keep the structure-related part of the algorithm as simple as possible. As measures of "structural closeness" between two nodes we propose to use the best lexical match between their children and the best lexical match between their parents. These measures are, on one hand, simple and efficient to calculate, al-

[1] Phrase-structure trees are used in another layer of LFG, namely in c-structure.

lowing a greedy alignment of two DAGs, irrespective of whether they are fragmented or not, to be built in $O(n^2 d^2) + O(n^2 \ln n)$ time, which is comparable to the complexity of Meyers et al. (1998) algorithm for trees. On the other hand, these measures are sufficient for resolving simple ambiguities, such as several occurrences of the same word in the same sentence (a very frequent situation, if we take function words into account). A similar idea underlies the work by Watanabe et al. (2000) on resolving the alignment ambiguities which arise when using a translation dictionary; their method checks how well the neighbors of a word match the neighbors of each of its candidate counterparts.

We use a very simple bilexical dictionary unit (a plain cooccurrence counter on training bitext) in this version of the software, but nothing prevents using more elaborate dictionary units within the same architecture. The software supports the data formats of two different LFG pasers, namely XLE (Kaplan et al., 2002) and DCU (Cahill et al., 2004).

The evaluation of the tool presented here is inevitably limited. We use sentence-aligned English and German Europarl (Koehn, 2005) and SMULTRON 2.0 (Volk et al., 2009) data. To the best of our knowledge, there is no relevant gold standard, so we produced a small gold standard set ourselves, manually node-aligning 20 f-structure pairs created from SMULTRON data, using the word alignments contained in SMULTRON for reference. It would be also possible to evaluate the method within an SMT system, but the available LFG-based SMT software is currently not accurate enough for the alignment to be reliably evaluated by the translation scores. We also manually examined and analyzed some sentences from the output. The evaluation, even though limited, supports the validity of the core idea of the method and suggests ways of further improvement.

The paper is organized as follows. In Section 2 we explain the algorithm; Section 3 is dedicated to the resulting software tool; Section 4 details the evaluation; in Section 5 we present our conclusion and outline directions for further improvement.

2 The Alignment Algorithm

The algorithm requires each sentence to be represented as a graph (probably disjoint) with a lexical item associated with each node. If an f-structure node is not lexicalized (e.g. it is a `SPEC` node with a `DET` as its child), it is removed and its children are linked directly to its parents.

2.1 Composite Alignment Score

The composite alignment score is defined as a simple scoring formula which makes some use of structural information and incurs reasonable computational cost.

Let $Lex(A, A')$ be a measure of lexical closeness of the words associated with the nodes A and A'. Such measure may either come from a dictionary, as in the work by Meyers et al. (1998), or, as in our experiments, be extracted in some way from the data.

In addition to the lexical score, and on the basis of it, we calculate two supplementary scores:

1. The score of the best lexical match between the children of A and A'. If both nodes have children, the score is calculated as follows:

$$S_c(A, A') = \max_{\substack{B \in C(A),\\ B' \in C(A')}} Lex(B, B'), \qquad (1)$$

where $C(.)$ is the set of children of the node. If only one of the two nodes has children, $S_c(A, A') = 0$; if none of the two nodes has children, $S_c(A, A') = 1$.

2. The score of the best lexical match between the parents of A and A'. If both nodes have parents, the score is calculated as follows:

$$S_p(A, A') = \max_{\substack{B \in P(A),\\ B' \in P(A')}} Lex(B, B'), \qquad (2)$$

where $P(.)$ is the set of parents of the node (all nodes with which the node is connected by incoming edges). If only one of the two nodes has parents, $S_p(A, A') = 0$; if none of the two nodes has parents, $S_p(A, A') = 1$.

To allow for some differences in structure, an *extended* children matching score can be calculated: in this case the best match for each child of A is searched for also among the "grandchildren" (children of children) of A' and vice versa. Matches between the grandchildren of A and the grandchildren of A' are not considered. In the same way, an extended parent matching score can be calculated.

The composite score is a weighted sum of the lexical score and the supplementary scores:

$$S(A,A') = w_c S_c(A,A') + w_p S_p(A,A') + (1 - w_c - w_p) Lex(A,A') \quad (3)$$

The weights w_c and w_p may vary, with larger values corresponding to greater reliance on structural information.

Let us consider aligning two DAGs with at most n lexicalized nodes in each, and with at most d children and at most d parents for each node. It is easy to see that (3) is calculated in $O(d^2)$ time, and therefore the scores for all possible pairs are calculated in $O(n^2 d^2)$ time. Building a greedy alignment using the scores takes $O(n^2 \ln n)$ time, the most costly operation being the sorting procedure; so the overall complexity of a greedy alignment of two f-structures is $O(n^2 d^2) + O(n^2 \ln n)$. Heuristics can be used to further improve the accuracy.

2.2 The Alignment Procedure for a Pair of DAGs

Given two lexicalized DAGs with n and n' nodes the alignment procedure establishes $\min(n, n')$ one-to-one matches, maximizing the product of the composite alignment scores of the matches. It:

- calculates the composite alignment scores for each possible pair (A, A');
- finds a candidate one-to-one match with a greedy algorithm;
- runs a 2-opt heuristic: on each step the heuristic checks whether switching the right-hand sides of any two pairs in the alignment, or replacing one side of any pair with a node not yet aligned to anything, improves the result. The heuristic is repeated until it fails to improve the result further.

2.3 The Resulting Algorithm

The word-alignment of a parallel corpus is performed as follows:

1. The lexical score is calculated, which reflects the frequency of co-occurrence of two words:

$$Lex(A,A') = \frac{\frac{N_{pair}(A,A')}{N_1(A)+1} + \frac{N_{pair}(A,A')}{N_2(A')+1}}{2},$$

where $N_{pair}(A,A')$ is the number of sentence-pairs in which the lexical entry from the node A occurs on the source side, and the lexical entry from A' occurs on the target side, and $N_1(A)$ and $N_2(A')$ are monolingual occurrence counts. Additionally, if A and A' are identical strings consisting only of digits, then $Lex(A,A')$ is set to 1.0.

2. For each pair of DAGs in the corpus the alignment procedure is performed as described in Section 2.2.

3 The Tool: f-align

The algorithm is the basis for version `0.1` of a new open-source tool, `f-align`, which we present here.[2] It is written in C++ and can be compiled both under Linux and under Windows. The tool currently understands two representations of LFG f-structures: XLE parser output (Kaplan et al., 2002) and DCU parser output (Cahill et al., 2004).

The following parameters are passed through the command line: the names of two input and one output directory, the number of structure pairs to align and the input formats (`LFG_XLE` or `LFG_DCU` for each of the input directories), the weights for the composite score calculation, w_c and w_p (0.2 being the default value for both), and the flag for using/not using the extended scores (the default is no). The files in the input directories should be named `0.txt`, `1.txt`, etc. For each structure pair the tool creates a separate output file in the output directory. Each line of an output file contains a pair of aligned nodes, together with the composite alignment score of this alignment. For example, for `LFG_XLE` format:

```
var(11) => var(20) (0.186723)
```

Explicit inclusion of the scores in the output allows the user to apply a quality threshold if required.

4 Evaluation

For this work we prepared a small 20-sentence gold standard and evaluated our menthod against it; in addition, we manually checked some sentences and visualized a typical success case and a typical problem case in Figures (3) and (4). In future we plan to also

[2] The source code can be downloaded from the page `http://www.computing.dcu.ie/~abryl/software.html`.

evaluate the method as a part of an LFG-based SMT system.

During the evaluation we also assessed the speed of the alignment tool. On a desktop PC under Windows Vista with a 3GHz CPU, alignment of 10000 pairs of f-structures takes about 4 minutes when using the extended matching scores, and about 3 minutes 15 seconds when not using them.

4.1 Experimental Data

For our experiments we prepared a dataset composed of two parts:

- 10000 sentence pairs from the German-English part of the Europarl corpus (Koehn, 2005) parsed into f-structures with the English and German treebank-based DCU LFG parsers (Cahill et al., 2004; Rehbein and van Genabith, 2009). For this dataset no gold standard word alignment is available.
- 20 sentence pairs from the SMULTRON 2.0 parallel treebank (Volk et al., 2009) (the first 20 sentence pairs with one-to-one sentence alignment for which both German and English parts were parsed successfully[3]). We used the same DCU LFG f-structure annotation software to annotate the trees from the corpus and to transform them info f-structures. Then the resulting f-structure pairs were manually node-aligned (256 alignments in total), using the word-alignment supplied with the corpus as reference. The sentences were picked from the "economy" sub-corpus of SMULTRON, as we assumed that it is not too different lexically from the Europarl corpus.

With a dataset that small, the automatically acquired dictionary (the $Lex(.,.)$ scores in Section 2.3) is not reliable enough, which allows us to have a close-up look at the effect of considering structural information.

4.2 Comparing Against the Gold Standard

For these experiments we automatically aligned our experimental data (10020 f-structure pairs) as a whole with f-align, and then compared the output for the SMULTRON part against the manually prepared gold standard. The following measures were used:

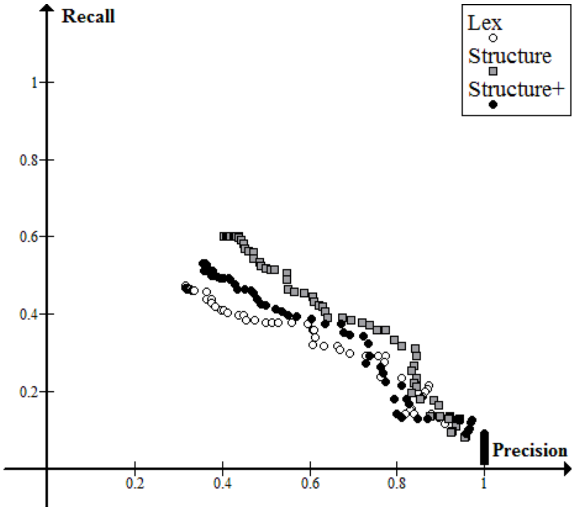


Figure 1: The Effect of Considering Structure.

$$precision = \frac{\left| \begin{matrix} Established \\ alignments \end{matrix} \bigcap \begin{matrix} Gold{-}standard \\ alignments \end{matrix} \right|}{\left| \begin{matrix} Established \\ alignments \end{matrix} \right|}$$

$$recall = \frac{\left| \begin{matrix} Established \\ alignments \end{matrix} \bigcap \begin{matrix} Gold{-}standard \\ alignments \end{matrix} \right|}{\left| \begin{matrix} Gold{-}standard \\ alignments \end{matrix} \right|}$$

$$f\text{-}measure = 2 \times \frac{precision \times recall}{precision + recall}$$

By established alignments we mean those which appear in the output of the tool and have scores higher then the threshold.

4.2.1 The Effect of Considering Structure

In this experiment we calculated precision and recall for different confidence thresholds for the following three configurations:

1. $w_c = w_p = 0$ (Lex);
2. $w_c = w_p = 0.2$ (Structure);
3. $w_c = w_p = 0.2$, extended matching scores used (Structure+).

[3] "Successfully" does not always mean "correctly"; for several long sentences the parses were quite noisy, complicating the alignment task and probably accounting for part of the noisiness of the resulting graphs.

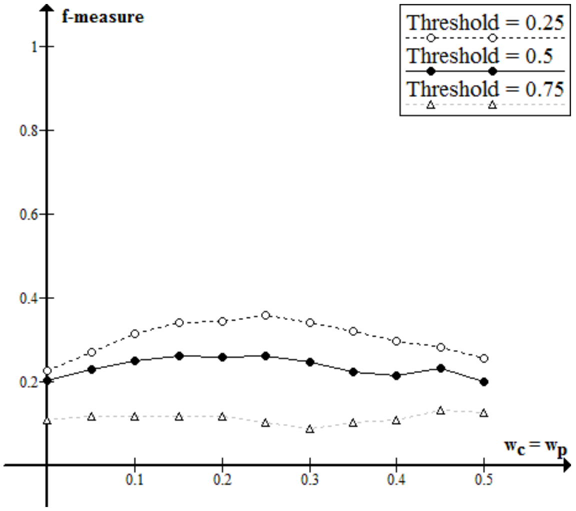


Figure 2: The Effect of the Parameter Values.

The result is presented in Figure 1.[4] Consideration of parents and children in matching brings about a clear improvement of the alignment quality, even though the structural information in the present data is imperfect. The effect of using the extended matching score (that is, using grandparents and grandchildren) is rather negative. In comparison to purely lexical alignment the extended matching score shows better recall only for low precision levels, and is invariably worse than the variant with children and parents only. It seems that the additional flexibility gained by the extended matching score does not repay for the general bluring of the structure caused by it.

4.2.2 The Effect of the Parameter Values

In this experiment we assessed the effect of the parameter values on the f-measure. Confidence thresholds of 0.25, 0.5, 0.75 was used. In all cases $w_c = w_p$. The results are shown in Figure 2. The figure confirms that the parameter values $w_c = w_p = 0.2$, chosen initially based on general considerations[5] are in fact reasonable.

4.3 Manual Examination

For this experiment we automatically aligned the Europarl part of our experimental data (10000 f-structure pairs) with f-align and manually examined the results to explore the effect of considering the structure.

4.3.1 When Structure Helps

There are two most obvious cases where the structure is helpful:

i when the same word occurs several times in the sentence;

ii when one or several rare words (or words in rare senses) occur in the sentence.

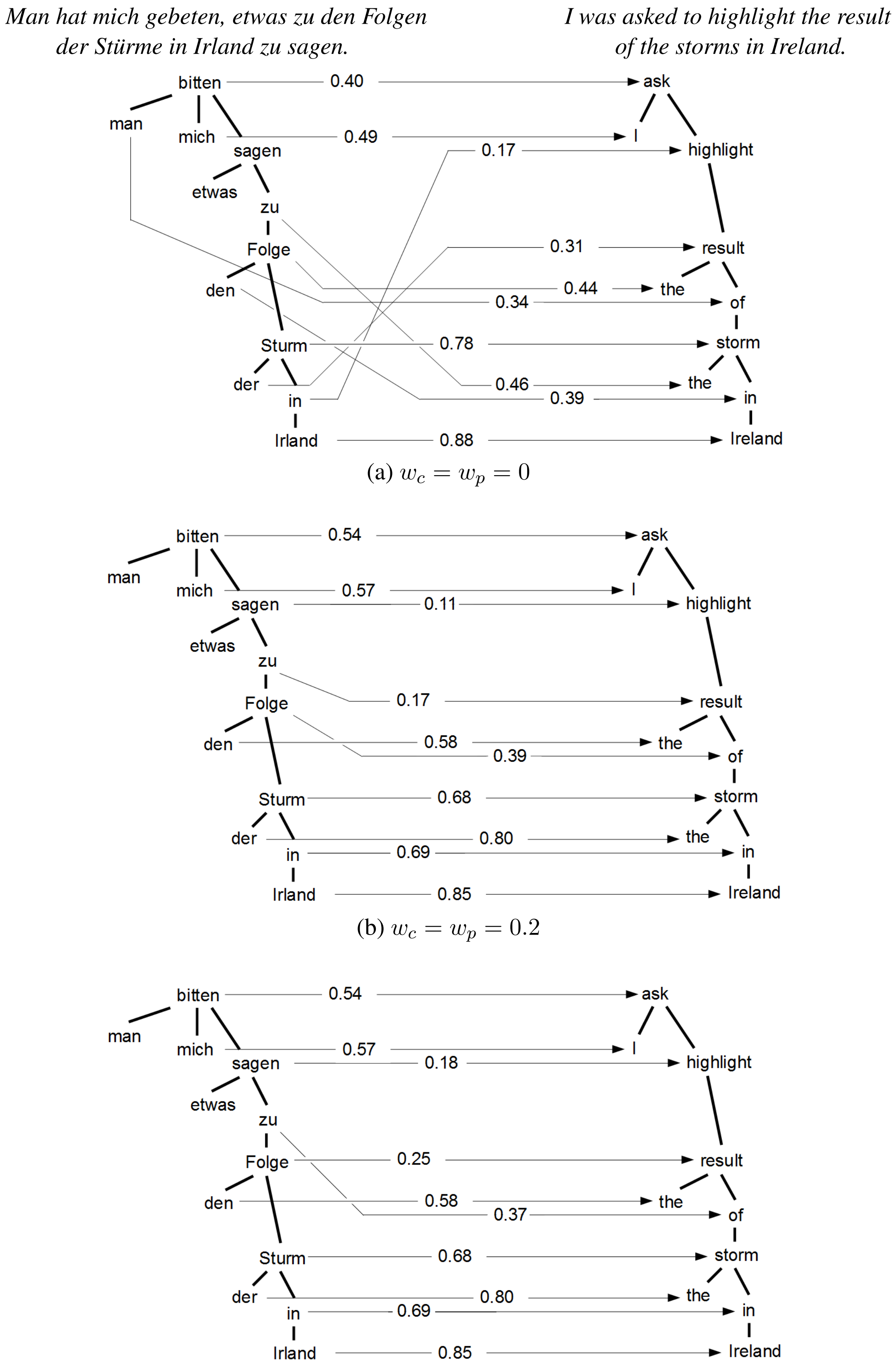
For illustration we picked a Europarl sentence pair which has both these problems (Figure (3)): the English side has *'the'* used twice, and the word *'highlight'* is used to match *'sagen'*, which is not much supported by the co-occurrence counts (Section 2.3). There is also some structural dissimilarity between German and English: *'zu'* (in *'zu den Folgen'*) and *'of'* (in *'of the storms'*) actually have no matches.

Figure (3a) shows the performance of dictionary-only alignment; given the weak dictionary, it is no surprise that only few nodes are aligned correctly. However, with the same dictionary it is possible to perform much better once structural parent and child matching scores are considered: Figure (3b) shows this clearly. The result is easy to explain: the highly scored lexical matches for *'Sturm'* and *'Irland'* forced *'der'* and *'in'* to be matched correctly, while *'bitten'* offered some help to *'sagen'*; but in this last case the score of the match is very low. The dissimilarity in structure caused a predictable problem: *'zu'* was aligned with *'result'*, and *'Folge'* with *'of'*. However, the scores of these incorrect matches are not high; applying the threshold of 0.5 to the scores would provide a reasonable overall alignment with some gaps.

The application of extended matching scores (Figure 3c) allows the algorithm to make another step in the correct direction, though an insufficient one for a practical improvement. *'Folge'* is now aligned with *'result'*, but this correct match has lower score than the incorrect match *'zu'* → *'of'*. This situation, namely low confidence of correct alignments, must be an explanation why the aligner with extended matching score looses recall so fast with the increase of precision (see Figure 1).

[4] Graphs are plotted using Graph 4.3 software tool, `http://www.padowan.dk/graph/`

[5] So as to make the structural information provide a bit less than a half of the composite score.



(a) $w_c = w_p = 0$

(b) $w_c = w_p = 0.2$

(c) $w_c = w_p = 0.2$, extended matching scores used

Figure 3: When Structure Helps: the correctly aligned words help their children and parents to find the correct matches.

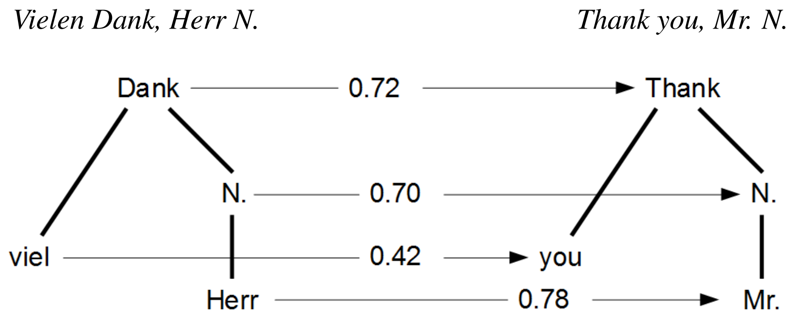


Figure 4: When Structure Misleads: over-aligning fixed expressions. The word-to-word alignment between *viel* and *you* is in fact wrong, but the structural information gives strong support to it, so the score of this alignment becomes quite high. $w_c = w_p = 0.2$

4.3.2 When Structure Misleads

At least in one rather rare case the structural similarity is misleading, namely when two structurally similar fixed expressions are aligned. In this case the method aligns them word-by-word, and the matching scores, due to the structural matches, are not too low. An example is given in Figure 4. The alignment between *viel* and *you* is almost inevitable in one-to-one alignment, but at least it is desirable to decrease the score of the incorrect match, which is boosted by structural similarity. This can probably be achieved if, following Meyers et al. (1998), we construct a dictionary not only for the words, but also for grammatical functions (labels on the edges of the graph); *viel* and *you* have unrelated grammatical functions (`quant` and `obj` respectively in the DCU parser output), and this mismatch, if properly considered, could decrease the matching score to some extent.

5 Conclusions and Future Work

In this paper we presented a specialized method for aligning LFG f-structures, and presented a software tool based on this method. Standalone evaluation allows us to draw optimistic conclusions. The results, as well as the manually examined portions of the output suggest that the method is effective enough to resolve simple ambiguities and that it improves the overall alignment accuracy.

There is much space for improvement. First of all, the support of one-to-many and probably many-to-many node alignments is necessary. Another desirable and apparently feasible feature is the capability to detect incorrect parses by measuring the inconsistency between lexical and structural matches; the presence of incorrect parses in the training data does nothing but harms the MT process, and their exclusion (if not correction) could be extremely useful. More elaborate ways of building bilexical dictionaries may also be incorporated, as the current method is obviously very simple. Another direction of work is to perform more thorough evaluation, in particular by using the method for rule extraction in a transfer-based SMT system.

It is straightforward to extend the applicability of the tool to other kinds of dependency graphs, such as e.g. that provided by the Malt dependency parser (Nivre et al., 2007).

Acknowledgements

This research is supported by the Science Foundation Ireland (Grant 07/CE/I1142) as part of the Centre for Next Generation Localisation (CNGL) at Dublin City University.

The authors thank Dr. Özlem Çetinoğlu for assistance with LFG parsers.

References

Vamshi Ambati and Alon Lavie. 2008. Improving syntax driven translation models by re-structuring divergent and non-isomorphic parse tree structures. In *Student Research Symposium at the The Eighth Conference of the Association for Machine Translation in the Americas (AMTA)*.

Eleftherios Avramidis and Jonas Kuhn. 2009. Exploiting xle's finite state interface in lfg-based statistical machine translation. In *In Proceedings of the 14th International Lexical Functional Grammar Conference (LFG'09)*.

Aoife Cahill, Michael Burke, Josef Van Genabith, and Andy Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage pcfg-based lfg approximations. In *In Proceedings of the 42nd Meeting of the ACL*, pages 320–327.

Yvette Graham and Josef van Genabith. 2009. An open source rule induction tool for transfer-based SMT. *The Prague Bulletin of Mathematical Linguistics*, Special Issue: Open Source Tools for Machine Translation, 91:37–46.

Hiroyuki Kaji, Yuuko Kaji, and Yasutsugu Kaji. 1992. Learning translation templates from bilingual text. In *Proceedings of the 14th conference on Computational linguistics*, pages 672–678, Morristown, NJ, USA. Association for Computational Linguistics.

Ronald Kaplan and Joan Bresnan. 1982. Lexical functional grammar, a formal system for grammatical represenation. *The Mental Representation of Grammatical Relations*, pages 173–281.

Ronald M. Kaplan, Tracy Holloway King, and John T. Maxwell III. 2002. Adapting existing grammars: the XLE experience. In *Proceedings of COLING02, Workshop on Grammar Engineering and Evaluation*.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the tenth Machine Translation Summit (MT Summit X)*, pages 79–86.

Adam Meyers, Roman Yangarber, Ralph Grishman, Catherine Macleod, and Antonio Moreno-Sandvval. 1998. Deriving transfer rules from dominance-preserving alignments. In *In Proceedings of COLING-ACL98*, pages 843–847.

Joakim Nivre, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13:95–135.

Franz Josef Och, Christoph Tillmann, and Hermann Ney. 1999. Improved alignment models for statistical machine translation. In *Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing (EMNLP99)*, pages 20–28.

Ines Rehbein and Josef van Genabith. 2009. Automatic acquisition of lfg resources for german - as good as it gets. In *In Proceedings of the 14th International Lexical Functional Grammar Conference (LFG'09)*, pages 320–327.

Stefan Riezler and John T. Maxwell, III. 2006. Grammatical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-ACL)*, pages 248–255.

Martin Volk, Torsten Marek, and Yvonne Samuelsson. 2009. SMULTRON (version 2.0) The Stockholm MULtilingual parallel TReebank. www.cl.uzh.ch/research/paralleltreebanks_en.html. An English-German-Swedish parallel Treebank with sub-sentential alignments.

Hideo Watanabe, Sadao Kurohashi, and Eiji Aramaki. 2000. Finding structural correspondences from bilingual parsed corpus for corpus-based translation. In *Proceedings of the 18th conference on Computational linguistics*, volume 2, pages 906–912.

Ventsislav Zhechev. 2009. Unsupervised generation of parallel treebanks through sub-tree alignment. In *The Prague Bulletin of Mathematical Linguistics, 91*, pages 89–98.