Supporting LFG Parsing with Dependency Parsing

Adam Przepiórkowski[1,2] and Alina Wróblewska[1]

[1]Institute of Computer Science, Polish Academy of Sciences
[2]University of Warsaw
E-mail: {adamp|alina}@ipipan.waw.pl

Abstract

In order to increase the coverage of the Polish LFG grammar, a novel method of combining grammar-based and data-driven parsers is proposed consisting in 1) augmenting the LFG grammar with so-called FRAGMENT rules which make it possible to obtain substructures for parsable fragments of sentences, 2) composition of such FRAGMENT substructures into a full f-structure on the basis of dependency relations proposed by an independent data-driven parser, with 2a) modification of the internal structure of FRAGMENT substructures only if absolutely necessary for independent (LFG-theoretical) reasons and 2b) modification of dependency relations in accordance with the naming and grammatical conventions of the LFG grammar.

1 Introduction

Dependency parsers trained on large treebanks have obvious advantages over parsers relying on manually-constructed grammars adhering to linguistic formalisms such as Head-driven Phrase Structure Grammar (HPSG; Pollard and Sag 1987, 1994) or Lexical Functional Grammar (LFG; Bresnan 1982, 2001; Dalrymple 2001). Data-driven parsers have good coverage and produce a single (i.e. best, according to some metric) parse for a sentence, while grammar-based parsers – unless they are supported by pre- and post-processing heuristics and/or some data-driven training – usually have poor coverage and often produce numerous parses for the sentences that they do cover. On the other hand, the resulting analyses based on manually-constructed grammars are often much deeper syntactically and may also contain semantic information often missing from data-driven dependency parsers, which concentrate on grammatical functions (subject, object, etc.). Hence, some work has been devoted to combining the two kinds of approaches (see Section 6 below).

The aim of this paper is to propose a novel way of combining grammar-based and data-driven dependency parsing. If the grammar-based parser produces a single parse, nothing needs to be done. Otherwise, there are two possibilities: the

grammar-based parser produces many parses, or it finds none. In both cases, an independently constructed data-driven dependency parser is employed to parse the same sentence and produce a single dependency tree. Trivially, this dependency tree may be used to disambiguate between parses produced by the grammar-based parser. Less trivially, in case the grammar-based parser finds no complete parse but manages to construct representations for fragments of the sentence, the dependency tree may be used to glue these representations together into a single parse. This paper presents an implementation of such a less trivial scenario.

The method proposed here is tested with the Polish LFG grammar, POLFIE (`http://zil.ipipan.waw.pl/LFG`; Patejuk and Przepiórkowski 2012, 2014), which suffers from the usual coverage problems of manually-constructed grammars. The solution consists in applying the following stages of processing. First, the text is parsed with pure POLFIE. Second, unparsed sentences are processed with POLFIE augmented with a so-called FRAGMENT sub-grammar, which makes it possible to construct an artificial parse containing f-structures corresponding to parsed fragments. Finally, such partial f-structures are composed into a single coherent f-structure using dependency relations in the dependency tree found by the data-driven parser.

There are two assumptions behind this procedure that should be made explicit. First, while LFG analyses are represented by c- and f-structures, only f-structures are modified in the current approach, while c-structures are discarded as much less important for further (esp. semantic) processing. Second, the internal structure of sub-f-structures produced by the augmented grammar should – in principle – not be modified. However, some modification is necessary in order to apply well-formedness LFG principles, i.e. completeness, coherence, and uniqueness (see Dalrymple 2001 or Bresnan 2001).

2 LFG-based partial parsing

Lexical Functional Grammar focuses on syntax and its relations with morphology, semantics and pragmatics. The syntactic dimension of a sentence contains two main (parallel) representations – a constituent structure (c-structure; essentially, a context-free phrase structure) and a functional structure (f-structure; a finite set of attribute-value pairs that encode functional properties of a sentence).

Handcrafted LFG grammars are typically implemented within the Xerox Linguistic Environment (XLE; Maxwell III and Kaplan 1993; Crouch *et al.* 2011). Apart from parsing complete sentences, XLE provides a mechanism for parsing possibly incorrect sentences: if a string of tokens cannot be parsed with standard rules of a grammar (i.e. correct c- and f-structures cannot be assigned to this string), it is reparsed with a grammar augmented by so-called FRAGMENT rules, and a sequence of well-formed partial structures specified by the augmented grammar is produced, as in Figure 1. The final set of such "FRAGMENT parses" is selected in a way that minimises the number of partial structures (i.e. parses of larger

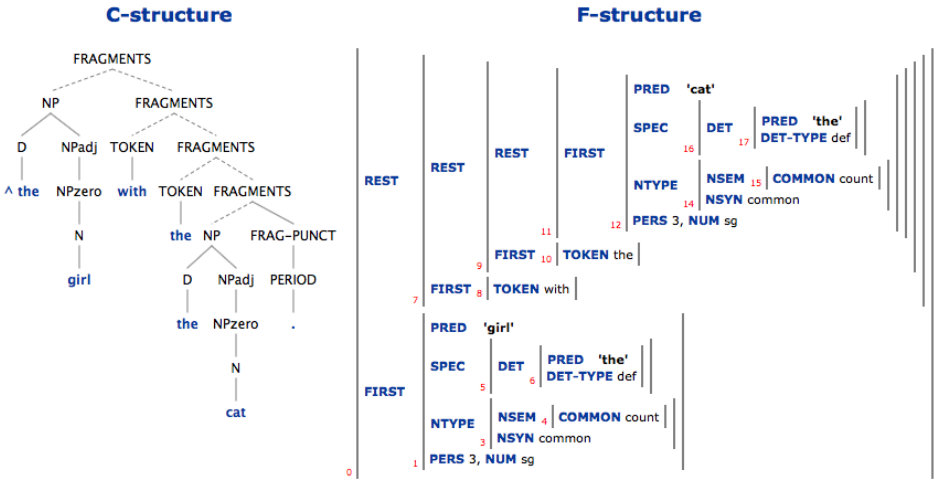

Figure 1: The FRAGMENT LFG analysis of 'The girl with the the cat.'

chunks are preferred to those of smaller chunks). Furthermore, it is possible that some words cannot be recognised and interpreted morphosyntactically, since they are misspelled or are not represented in the lexicon. Such words, as well as words which are not covered by any grammatical rules in the current parse, are encoded as unparsed TOKENs in f-structures. The FRAGMENT parses are assigned the root category FRAGMENTS. They are encoded with FRAGMENT and TOKEN nodes in c-structures and FIRST and REST functions in f-structures.

This view of FRAGMENT parsing seems to be language-independent: FRAGMENTS are combined in a linear manner consistent with the order of tokens – the sub-f-structure of the FRAGMENT which is first on the list is a value of the attribute FIRST and the sub-f-structure of all other FRAGMENTS on the list is a value of the attribute REST. However, what kinds of phrases or tokens can constitute FRAGMENTS is decided by the designers of the grammar, so the resulting FRAGMENT grammar is to some extent language-dependent.

For English, the combination of full and fragment parsing techniques allows for achieving 100% grammar coverage on unseen data (cf. Riezler *et al.* 2002).

3 Augmented grammar

The Polish LFG grammar POLFIE consists of 65 large rules (i.e. with disjunctive right-hand sides), which compile into a finite-state automaton with 479 states and 1041 arcs. There are no guessers – neither for the output of the morphological analyser (i.e. all analyses output by the morphological analyser build the lexicon for a particular sentence), nor for words unrecognised by the analyser. Since writing grammar rules is a very-time consuming process, there are still many constructions that are not defined yet in the grammar. POLFIE covers only about 40% of a representative corpus of Polish (cf. the initial row in Table 1).

Table 1: Test coverage on 20K sentences from the manually annotated part of National Corpus of Polish (http://nkjp.pl/; Przepiórkowski *et al.* 2012) parsed with POLFIE

grammar	parsed sentences	unparsed sentences	errors and time out
POLFIE	8364 (42%)	8181 (41%)	3455 (17%)
FRAGMENT grammar	11349 (57%)	0	8648 (43%)
FRAGMENT grammar with OT-marks and pruning	18909 (95%)	0	1091 (5%)

In order to increase the coverage of the Polish LFG grammar, the technique of partial parsing described in the previous section is applied (cf. the penultimate row in Table 1). The procedure of partial parsing makes it possible to parse a larger number of sentences which otherwise receive no analysis. However, in contrast to English, where partial parsing is used to parse incorrect sentences, Polish sentences with FRAGMENT parses are not necessarily incorrect. Many of them are well-formed but contain linguistic phenomena for which POLFIE rules have not been defined yet. FRAGMENT analyses of such sentences are candidates for improvement with the proposed method.

A quantitative analysis shows that the augmented grammar produces a huge number of analyses for some sentences. In order to limit the number of parses the augmented grammar is extended with some optimality marks.[1] Furthermore, in order to reduce the number of memory and time out errors, the XLE mechanism of pruning c-structures before processing f-structure constraints is employed.[2] The statistics of parsing with the extended version of the LFG grammar is presented in the final row in Table 1.

4 Recomposition of FRAGMENT sub-f-structures

The recomposition method only applies to LFG analyses with the root category FRAGMENTS. Since only f-structures are currently the subject of modification, the set of all LFG analyses output for a sentence is restricted to the set of analyses with unique f-structures. The main idea behind the method consists in the recomposition of FRAGMENT substructures in f-structures in accordance with dependency relations between their highest PRED attributes.

[1] Optimality theory marks (i.e. OT-marks) are preference and dispreference marks which are used to rank grammar rules, templates, and lexical entries. The most preferable grammar rules and templates can be applied and the most preferable lexical entries can be selected, e.g. in order to resolve ambiguity. While there are no OT-marks in the publicly avaliable POLFIE version used in the current experiments, we augmented the FRAGMENT rules with some OT-marks and defined their ranking.

[2] XLE documentation on pruning: http://www2.parc.com/isl/groups/nltt/xle/doc/xle.html#SEC14J.

A FRAGMENT f-structure consists of sub-f-structures connected with attributes FIRST and REST (see Figure 2). New relations between these sub-f-structures (see Figure 3) are determined on the basis of the corresponding dependency structure (see Figure 4).

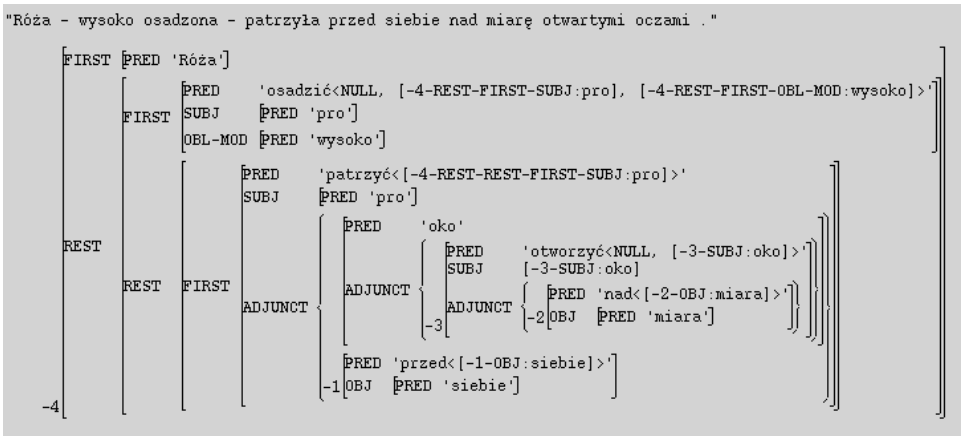

Figure 2: The FRAGMENT parse of *Róża – wysoko osadzona – patrzyła przed siebie nad miarę otwartymi oczami.* (Eng. 'Rose – placed highly – was looking straight ahead with her eyes open too widely.')

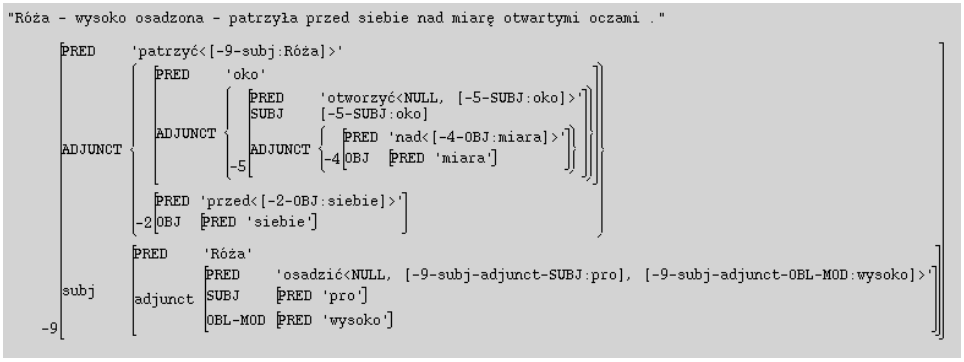

Figure 3: The recomposed f-structure of the FRAGMENT f-structure of Figure 2; dependency labels used as new attributes are written in lower case

Dependency structures are generated with an external data-driven dependency parser MATE (Bohnet, 2010) trained on the Polish dependency treebank (Wróblewska, 2014). Although the parsing quality of short and simple sentences with manually annotated tokens is relatively high (87.2 LAS and 92.7 UAS), the parsing quality of complex sentences with semi-automatically annotated tokens is significantly lower (70.3 LAS and 76.0 UAS) – see Wróblewska 2014 for details. As dependency trees may contain errors, the internal structures of rule-based sub-f-structures are not modified when they disagree with the dependency tree, unless

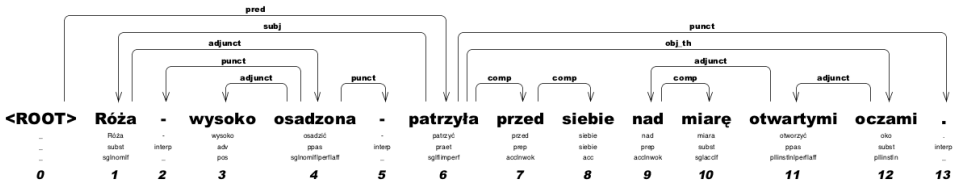

Figure 4: Dependency tree of *Róża – wysoko osadzona – patrzyła przed siebie nad miarę otwartymi oczami.* (Eng. 'Rose – placed highly – was looking straight ahead with her eyes open too widely.')

such a modification is essential for constructing a coherent f-structure for the whole sentence.

An essential modification of sub-f-structures includes removal of sub-f-structure for a pro-drop pronoun with the SUBJ (or OBJ) function, if one of FRAGMENT substructures is annotated as SUBJ (or OBJ) in the modified f-structure, in order to avoid f-structures with double subject. An example of an incoherent f-structure is given in Figure 5.

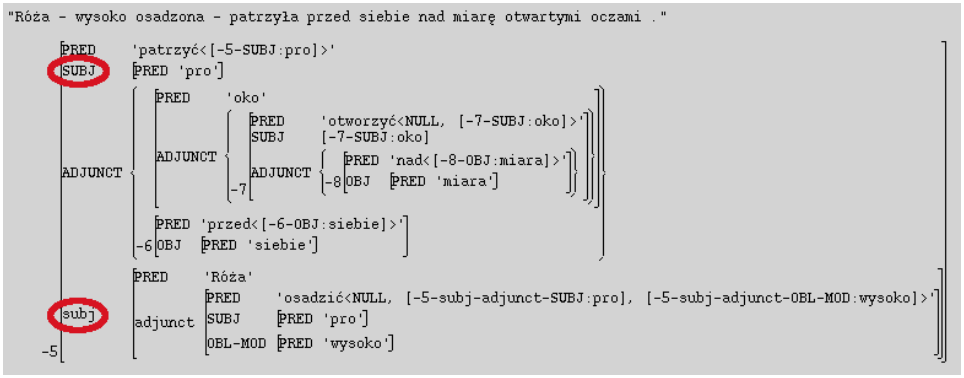

Figure 5: The incoherent f-structure glued from sub-f-structures in Figure 2

Rules converting dependency relations into an f-structure must convert dependency labels into f-structure attributes, e.g. – trivially – the *subj* dependency label is converted into the SUBJ attribute. Less trivially, as some linguistic phenomena (e.g. passive voice, analytical predicative constructions with *to*[3]) are treated differently in POLFIE and by the dependency parser, conversion rules must also perform some regular restructuring.

In passive constructions encoded in POLFIE f-structures, the participle is annotated as an XCOMP-PRED dependent of the auxiliary verb *zostać*. By contrast, in dependency structures, the participle functions as the governor of the auxiliary *zostać*. For example, the sentence *Wyznaczone zostaną również miejsca*

[3]The predicative *to* is a governor of an auxiliary verb form in dependency trees. In f-structures, in turn, the auxiliary is not encoded as a function with the f-structure value but as a value of the attribute TENSE incorporated into the f-structure of *to*.

parkingowe. (Eng. ‘Parking spaces will also be designated.’), when parsed by the LFG grammar augmented with FRAGMENT rules, receives the f-structure given in Figure 6. In order to glue the two FRAGMENT sub-f-structures, the same

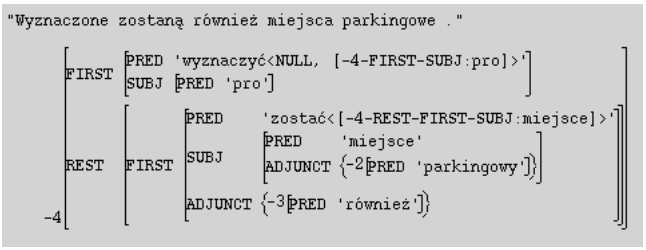

Figure 6: The FRAGMENT parse of *Wyznaczone zostaną również miejsca parkingowe.* (Eng. ‘Parking spaces will also be designated.’)

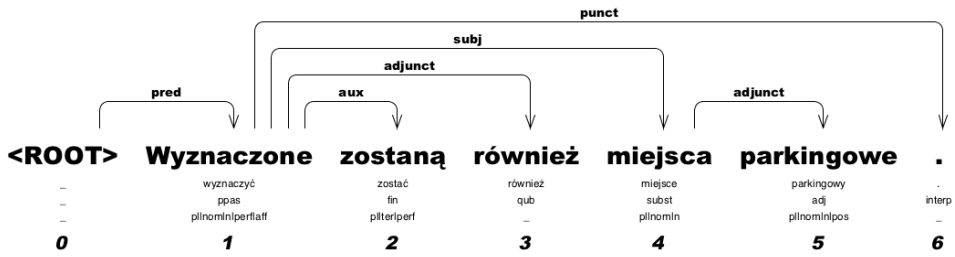

Figure 7: Dependency tree of *Wyznaczone zostaną również miejsca parkingowe.* (Eng. ‘Parking spaces will also be designated.’)

sentence is parsed with the dependency parser, and the resulting dependency tree (see Figure 7) is inspected for the presence of the two words corresponding to the top PRED values of the two fragments: *wyznaczone* ‘designated’ (cf. `PRED 'wyznaczyć<...>'` in Figure 6) and the auxiliary *zostaną* (`PRED 'zostać<...>'` in that figure). These two words are connected by an arc with the *aux* (auxiliary verb) dependency label, the governor’s part of speech is *ppas* (passive adjectival participle), and the dependent’s lemma is ZOSTAĆ, so this is a passive construction according to the dependency tree, and it is translated into the LFG analysis of passive adopted in POLFIE; the resulting f-structure is given in Figure 8.

If it is not possible to match any dependency label to an appropriate LFG grammatical function, the dependency label is left without any modification. The properly converted labels are marked with the asterisk (see XCOMP-PRED in Figure 8), in order to retain information about newly introduced attributes, not generated by the rules of the LFG grammar.

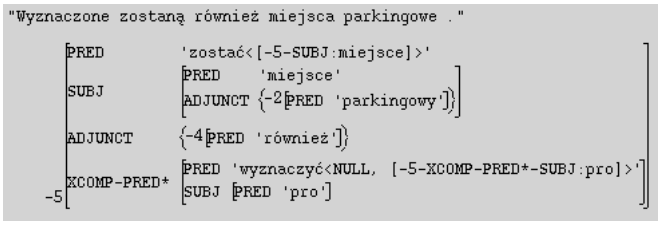

Figure 8: The recomposed f-structure of the FRAGMENT f-structure of Figure 6

5 Evaluation

There is no gold standard data that could be used for the evaluation of the proposed gluing procedure. In this very preliminary evaluation, 30 FRAGMENT analyses were randomly selected from the set of all those FRAGMENT analyses which have only one solution. Sub-f-structures of these analyses were then manually glued into proper f-structures, and the resulting 30 f-structures were used as the gold standard.

As assumed, FRAGMENT analyses with only one solution were generated mostly for relatively short sentences. There are 6.9 tokens per sentence on average in the resulting test set. These sentences were not covered by the POLFIE grammar for various reasons, e.g. wrong punctuation, lack of rules for predicate-less sentences, reported speech, unconventional word order, or eliptical constructions.

In order to evaluate glued f-structures, some metrics inspired by dependency parsing metrics are defined: UAS – the percentage of FIRST sub-f-structures that are correctly unified with the final f-structure, LAS – the percentage of FIRST sub-f-structures that are correctly unified with the final f-structure as a value of a correct grammatical function, and LA – the percentage of FIRST sub-f-structures that are incorporated as a value of a correct grammatical function. Tested against the set of 30 manually composed f-structures, the following results obtain: 79.45% UAS, 72.6% LAS, and 78.08% LA.

Additionally, the choice of grammatical functions was evaluated using the usual measures: precision, recall, and f-measure (see Table 2). The repertoire of evaluated grammatical functions is not representative since many functions do not appear in test data. On the one hand, it is because of the simplicity of test data. On the other hand, it is because of relatively good treatment of some grammatical functions (e.g. XCOMP, POSS) in POLFIE; these functions only appear inside of sub-f-structures.

The results indicate that many sub-f-structures are glued with a correct grammatical function. In particular, sub-f-structures that function as the sentence predicate (i.e. the value of PRED for the whole sentence) or are values of ADJUNCT and OBJ attributes are correctly identified in most cases. The results also indicate what should be improved in the proposed gluing procedure.

Table 2: Precision, recall and f-measure of individual grammatical functions gluing sub-f-structures in the test FRAGMENT f-structures

grammatical function	number of occurrences	precision	recall	f-score
ADJUNCT	21	90%	86%	0.88
ADJUNCT-QT	9	100%	22%	0.36
SUBJ	3	33%	66%	0.44
OBJ	2	66%	100%	0.80
OBL-GEN	2	100%	50%	0.66
COMP	1	100%	100%	1.00
sentence predicate	27	96%	88%	0.92
coordination conjunct	6	54%	100%	0.71

6 State of the art

The idea of using analyses of one type to improve analyses of other types is not new. There are some approaches employing LFG to improve dependency parsing or dependency parsing to improve LFG parsing. In the approach by Øvrelid *et al.* (2009), the output of a grammar-driven LFG parser is encoded as additional features in the data-driven dependency parsing models. Çetinoğlu *et al.* (2010) train a dependency parser on LFG-inspired dependency trees generated either with 'LFG constituency parsing pipeline' or 'LFG dependency parsing pipeline'. Çetinoğlu *et al.* (2013) in turn propose a dependency-based sentence simplification approach. The simplification consists in deleting erroneous parts from unparsed sentences (the erroneous parts are identified on the basis of dependency structures of considered sentences). Sentences that fail to have a complete analysis in their original form are simplified this way and parsed with XLE, in the hope of receiving a coherent – even if incomplete – analysis.

Furthermore, Sagae *et al.* (2007) use a dependency parser to restrict the search space of a more complex HPSG parser. Output of a statistical dependency parser serves as constraints (hard or soft) to improve the HPSG parsing. The HPSG parser produces parse trees that strictly conform to the output of the dependency parser (hard dependency constrains). Some dependency structures do not conform to HPSG schema used in parsing. Predetermined dependencies are therefore treated as soft constraints that do not prohibit schema applications but penalise the log-likelihood of partial parse trees created by schema application that violate the dependency constraints.

To the best of our knowledge, using clues from a simple dependency parser to recompose deeply-parsed fragments of a sentence not completely analysable by a deep parser, is a novel contribution of this work.

7 Future work

While the results reported above are quite promising, there is still room for improvement. First, not all FRAGMENT analyses could be converted into proper f-structures. Some of them contain strings of tokens that could not be analysed as correct phrases by POLFIE rules and are annotated as TOKENs. In the worst case the entire sentence is annotated as a TOKEN. FRAGMENT f-structures with TOKENs are currently not modified, but as they might correspond to proper sentences, it would be useful to develop a procedure of modifying them. Second, it should be verified whether it is possible to disambiguate multiple solutions which are output for a sentence based on a dependency tree of this sentence. Finally, another possibility to investigate is to provide a dependency parser with a sentence partially parsed by XLE. Then, the initial configuration of a transition-based parser could correspond to the set of relations in a FRAGMENT f-structure,[4] or – in the graph-based approach – arcs of a directed graph corresponding to relations of a FRAGMENT f-structure could be initially scored high so that they are selected to build the final dependency tree.

Acknowledgements

We are grateful to the anonymous reviewers for comments which led to multiple improvements in the form and contents of this paper. Work reported here has been partially financed by the Polish Ministry of Science and Higher Education within the CLARIN ERIC programme 2015–2016 (`http://clarin.eu/`).

References

Bohnet, B. (2010). Very High Accuracy and Fast Dependency Parsing is not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING 2010, pages 89–97.

Bresnan, J., editor (1982). *The Mental Representation of Grammatical Relations*. MIT Press Series on Cognitive Theory and Mental Representation. The MIT Press, Cambridge, MA.

Bresnan, J. (2001). *Lexical-Functional Syntax*. Blackwell Textbooks in Linguistics. Blackwell, Malden, MA.

Çetinoğlu, O., Foster, J., Nivre, J., Hogan, D., Cahill, A., and van Genabith, J. (2010). LFG Without C-Structures. In *Proceedings of the 9th International Workshop on Treebanks and Linguistic Theories*, pages 43–54.

Çetinoğlu, O., Zarrieß, S., and Kuhn, J. (2013). Dependency-based sentence simplification for increasing deep LFG parsing coverage. In *Proceedings of the LFG13 Conference*, pages 191–211.

[4] As suggested in one of the reviews.

Crouch, D., Dalrymple, M., Kaplan, R., King, T., Maxwell, J., and Newman, P. (2011). *XLE Documentation*. Palo Alto Research Center (PARC), Palo Alto, CA.

Dalrymple, M. (2001). *Lexical Functional Grammar*. Academic Press, San Diego, CA.

Maxwell III, J. T. and Kaplan, R. M. (1993). The Interface between Phrasal and Functional Constraints. *Computational Linguistics*, **19**(4), 571–590.

Øvrelid, L., Kuhn, J., and Spreyer, K. (2009). Improving Data-Driven Dependency Parsing Using Large-Scale LFG Grammars. In *Proceedings of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (Conference Short Papers)*, pages 37–40.

Patejuk, A. and Przepiórkowski, A. (2012). Towards an LFG parser for Polish: An exercise in parasitic grammar development. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012*, pages 3849–3852, Istanbul, Turkey. ELRA.

Patejuk, A. and Przepiórkowski, A. (2014). Synergistic development of grammatical resources: A valence dictionary, an LFG grammar, and an LFG structure bank for Polish. In V. Henrich, E. Hinrichs, D. de Kok, P. Osenova, and A. Przepiórkowski, editors, *Proceedings of the Thirteenth International Workshop on Treebanks and Linguistic Theories (TLT 13)*, pages 113–126, Tübingen, Germany. Department of Linguistics (SfS), University of Tübingen.

Pollard, C. and Sag, I. A. (1987). *Information-Based Syntax and Semantics, Volume 1: Fundamentals*. Number 13 in CSLI Lecture Notes. CSLI Publications, Stanford, CA.

Pollard, C. and Sag, I. A. (1994). *Head-driven Phrase Structure Grammar*. Chicago University Press / CSLI Publications, Chicago, IL.

Przepiórkowski, A., Bańko, M., Górski, R. L., and Lewandowska-Tomaszczyk, B., editors (2012). *Narodowy Korpus Języka Polskiego*. Wydawnictwo Naukowe PWN, Warsaw.

Riezler, S., King, T. H., Kaplan, R. M., Crouch, R., Maxwell, J. T., and Johnson, M. (2002). Parsing the Wall Street Journal using a Lexical-Functional Grammar and Discriminative Estimation Techniques. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 271–278.

Sagae, K., Miyao, Y., and Tsujii, J. (2007). HPSG Parsing with Shallow Dependency Constraints. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 624–631.

Wróblewska, A. (2014). *Polish Dependency Parser Trained on an Automatically Induced Dependency Bank*. Ph. D. dissertation, Institute of Computer Science, Polish Academy of Sciences, Warsaw.