Representation and parsing of multiword expressions

Current trends

Edited by

Yannick Parmentier

Jakub Waszczuk

Phraseology and Multiword Expressions 3

language science press

ISSN: 2625-3127

Representation and parsing of multiword expressions

Current trends

Edited by

Yannick Parmentier

Jakub Waszczuk

language
science
press

Parmentier , Yannick & Jakub Waszczuk (ed.). 2019. *Representation and parsing of multiword expressions*: *Current trends* (Phraseology and Multiword Expressions 3). Berlin: Language Science Press.

ISBN: 978-3-96110-145-0 (Digital)
978-3-96110-146-7 (Hardcover)

ISSN: 2625-3127
DOI:10.5281/zenodo.2579017
Source code available from www.github.com/langsci/202
Collaborative reading: paperhive.org/documents/remote?type=langsci&id=202

Cover and concept of design: Ulrike Harbort
Fonts: Linux Libertine, Libertinus Math, Arimo, DejaVu Sans Mono
Typesetting software: XƎLATEX

Language Science Press
Unter den Linden 6
10099 Berlin, Germany
langsci-press.org

Storage and cataloguing done by FU Berlin

Contents

Preface

Yannick Parmentier
University of Orléans
University of Lorraine

Jakub Waszczuk
University of Tours
University of Düusseldorf

In this introductory chapter, we first present the topic and context of this volume. We then summarize its contributions, which have been collected through an open call for submissions and a peer-reviewing process.

1 Introduction

While Multiword Expressions (MWEs), i.e. sequences of words with some unpredictable properties such as *to count somebody in* or *to take a haircut*, have been attracting attention for a long time because of these idiosyncratic properties which go beyond word boundaries, they remain a challenge for both linguistic theories and natural language (NL) applications.

Indeed, most of these theories and applications admit an (explicit or implicit) division of language phenomena into clear-cut levels: (i) tokens (indivisible text units, roughly words), (ii) morphology (properties of words e.g. number, gender, etc.), (iii) syntax (structural links between words, e.g. number/gender agreement), (iv) semantics (meaning of words and sentences). However, human languages frequently show a high degree of ambiguity and fuzziness with respect to this layer-oriented model. In particular, MWEs are placed on the frontier between these levels due to their idiosyncratic properties on the one hand, and their morphological, syntactic and semantic variations on the other hand. For instance, their meaning is often non-compositional as in *to take a haircut* (i.e. *to suffer a*

Yannick Parmentier & Jakub Waszczuk. 2019. Preface. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, iii–ix. Berlin: Language Science Press. DOI:10.5281/zenodo.2579031

serious financial loss), although they admit some syntactic variation similarly to many other expressions (*take/takes/have taken/has taken/took a serious/70% haircut*). Strictly layer-oriented language models fail to reflect this specificity, and thus yield erroneous text processing results (e.g. word-to-word translations of idioms). Although the quantitative importance of MWEs is well known (they cover up to 30% of all words in human language utterances, and are much more numerous in lexicons than single words), the achievements in their formal representation and automatic processing are still largely unsatisfactory.

In this context, an international and multilingual consortium of researchers recently took part in the European PARSEME COST Action[1] (2013–2017), which aimed at better understanding the nature of MWEs in order to improve their support in natural language applications. Two main challenges were considered: LINGUISTIC PRECISION (how to account for the highly heterogeneous nature of MWEs in linguistic resources and treatments?) and COMPUTATIONAL EFFICIENCY (how to deal with MWEs' idiosyncratic properties within reliable applications?).

To contribute to meeting these two challenges, PARSEME was based on four Working Groups (WGs):

- WG1 focused on the Grammar/Lexicon interface and the design of interoperable MWE lexicons,
- WG2 aimed at developing parsing techniques for MWEs,
- WG3 studied hybrid (e.g. symbolic and/or statistical) NL applications dealing with MWEs (e.g. MWE detection, machine translation, etc.),
- WG4 was concerned with the annotation of MWEs within treebanks.

This book has been created within WG2. It consists of contributions related to the definition, representation and parsing of MWEs. These contributions were collected via an open call for chapters. Each Chapter proposal was reviewed by 2 members of the editorial board. Out of this reviewing, 10 proposals were selected. They reflect current trends in the representation and processing of MWEs. They cover various *categories* of MWEs such as verbal, adverbial and nominal MWEs, various *linguistic frameworks* (e.g. tree-based and unification-based grammars), various *languages* including English, French, Modern Greek, Hebrew, Norwegian), and various *applications* (namely MWE detection, parsing, automatic translation) using both symbolic and statistical approaches.

[1]http://www.cost.eu/COST_Actions/ict/IC1207

2 Outline of the book

The book is organized as follows.

Part 1: MWE representations

The first part of the volume (Chapters 1 to 5) is dedicated to the study of MWE properties and representations.

In Chapter 1, Lichte et al. (2019 [this volume]) discuss the representation of MWEs within lexicalised formalisms. In particular, they show how the eXtensible MetaGrammar (XMG2) formalism offers a natural encoding of MWEs, which allows us to account for the fact that irregularities exhibited by MWEs are a matter of scale rather than binary properties.

In Chapter 2, Sheinfux et al. (2019 [this volume]) study a specific type of MWEs (namely verbal MWEs), focusing mostly on Hebrew, and show that unlike what previous work suggests, flexibility of verbal MWEs is not a discrete concept but rather a continuous property. They propose a new classification of MWEs which is based on semantic notions.

In Chapter 3, Dyvik et al. (2019 [this volume]) present the analysis of MWEs in an LFG grammar for Norwegian, NorGram, which is used in the construction of NorGramBank, a treebank of parsed sentences. The chapter describes how classes of MWEs are analysed by means of LFG templates, which capture the lexical and syntactic properties of MWEs in a succinct way.

In Chapter 4, Markantonatou et al. (2019 [this volume]) present a grammar of Modern Greek in the LFG formalism. Their grammar has been implemented with the Xerox Linguistic Engine (XLE), a grammar editor which also includes a parsing engine. In their Chapter, the authors pay a particular attention to the use of a pre-processor to detect and annotate MWEs prior to parsing.

In Chapter 5, Angelov (2019 [this volume]) presents the Grammatical Framework, a description language for developing NLP multilingual resources, and its application to some classes of MWEs. In particular, the author shows how to define MWE-aware multilingual grammars, which can be used for instance for in-domain machine translation.

Part 2: MWE parsing

The second part of the volume (Chapters 6 to 8) focuses on MWE parsing, that is, on the automatic construction of deep representations of the syntax of MWEs. Two main approaches to parsing coexist: the data-driven approach aims at extracting syntactic information from corpora using Machine Learning techniques and is discussed in Chapter 6. The knowledge-based approach relies on the encoding of linguistic properties of MWEs within lexical entries, which are used by a parsing algorithm to compute the expected syntactic structure. The impact of MWE detection on such parsing algorithms is discussed in Chapters 7 (for a categorial parser) and 8 (for an attachment-rule-based parser).

In Chapter 6, Constant et al. (2019 [this volume]) give a detailed overview of various ways to extend statistical parsing with MWE identification, either during parsing or as a pre- or post-processing step. These extensions are compared and their evaluation discussed.

In Chapter 7, de Lhoneux et al. (2019 [this volume]) extend a CCG parsing architecture for English with a module for detecting MWEs and pre-process them. The effect of this pre-processing is evaluated in terms of parsing accuracy when (i) the parser is trained on pre-processed data (so-called training effect) and (ii) the parser uses information from pre-processed data (so-called parsing effect).

In Chapter 8, Foufi et al. (2019 [this volume]) investigate the extension of a knowledge-based parser with collocation identification. They apply this extension to the description of MWEs for various languages (including English and Greek), and show how it improves parsing efficiency in terms of percentages of complete analyses.

Part 3: Multilingual NL applications for MWEs

Finally, in the third part of the volume (Chapters 9 and 10), multilingual MWE acquisition techniques are presented.

In Chapter 9, Semmar et al. (2019 [this volume]) present three techniques for word alignment between parallel corpora and their application to MWEs. The bilingual MWE lexicons built using these techniques are then evaluated according to their effect on phrase-based statistical machine translation. The authors empirically show that MWE-aware lexicons improve translation quality.

Finally, in Chapter 10, Jacquet et al. (2019 [this volume]) present an architecture which allows for the identification of multiword entities (organizations, medical terms, etc.) within large collections of texts, together with the linking of monolingual variants of a given multiword entity, and of groups of variants accross multiple languages. Their architecture is evaluated against data from *Wikipedia*.

3 Acknowledgments

We are grateful to the COST framework of the European Union for their support for the PARSEME Action.

We would like to warmly thank Agata Savary and Adam Przepiórkowski, respectively chair and vice-chair of PARSEME, for their commitment to this action. They made it a dynamic environment, where researchers can have fruitful discussions and exchange ideas, leading to long-term collaborations.

We are grateful to Manfred Sailer, who, as a member of the editorial board of the *Phraseology and Multiword Expressions* series, accompanied us throughout the publication process.

We would like to thank the reviewers of this volume:

- Doug Arnold, University of Essex, UK
- Gosse Bouma, University of Groningen, the Netherlands
- Svetla Koeva, Bulgarian Academy of Sciences, Bulgaria
- Cvetana Krstev, University of Belgrade, Serbia
- Ana R. Luís, University of Coimbra, Portugal
- Stella Markantonatou, Institute for Language and Speech Processing/Athena RIC, Greece
- Petya Osenova, Bulgarian Academy of Sciences, Bulgaria
- Carla Parra Escartín, Dublin City University, ADAPT Centre, Ireland
- Victoria Rosén, University of Bergen, Norway
- Michael Rosner, University of Malta, Malta
- Manfred Sailer, University of Franfkurt-am-Main, Germany
- Agata Savary, University of Tours, Blois, France
- Veronika Vincze, University of Szeged, Hungary
- Shuly Wintner, University of Haifa, Israel

We are grateful for their valuable evaluations, comments and feedback, and to the proofreaders for their thorough work. Without their help, this book would not exist.

Special thanks go to Language Science Press (especially Sebastian Nordhoff and Stefan Müller for their continuous help and their engagement in the promotion of high-quality peer-reviewed open-access publication).

Yannick Parmentier and Jakub Waszczuk, Feb. 2019

References

Angelov, Krasimir. 2019. Multiword expressions in multilingual applications within the Grammatical Framework. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 129–148. Berlin: Language Science Press.

Constant, Mathieu, Gülşen Eryiğit, Carlos Ramisch, Mike Rosner & Gerold Schneider. 2019. Statistical mwe-aware parsing. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 149–184. Berlin: Language Science Press.

de Lhoneux, Miryam, Omri Abend & Mark Steedman. 2019. Investigating the effect of automatic MWE recognition on CCG parsing. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 185–217. Berlin: Language Science Press.

Dyvik, Helge, Gyri Smørdal Losnegaard & Victoria Rosén. 2019. Multiword expressions in an LFG grammar for Norwegian. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 69–107. Berlin: Language Science Press.

Foufi, Vasiliki, Luka Nerima & Eric Wehrli. 2019. Multilingual parsing and MWE detection. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 219–240. Berlin: Language Science Press.

Jacquet, Guillaume, Maud Ehrmann, Jakub Piskorski, Hristo Tanev & Ralf Steinberger. 2019. Cross-lingual linking of multi-word entities and language-dependent learning of multi-word entity patterns. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 271–300. Berlin: Language Science Press.

Lichte, Timm, Simon Petitjean, Agata Savary & Jakub Waszczuk. 2019. Lexical encoding formats for multi-word expressions: the challenge of "irregular" regularities. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 1–33. Berlin: Language Science Press.

Markantonatou, Stella, Niki Samaridi & Panagiotis Minos. 2019. Issues in parsing MWEs in an LFG/XLE framework. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 109–127. Berlin: Language Science Press.

Semmar, Nasredine, Christophe Servan, Meriama Laib, Dhouha Bouamor & Morgane Marchand. 2019. Extracting and aligning multiword expressions from parallel corpora. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 241–270. Berlin: Language Science Press.

Sheinfux, Livnat Herzig, Tali Arad Greshler, Nurit Melnik & Shuly Wintner. 2019. Verbal multiword expressions: idiomaticity and flexibility. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 35–68. Berlin: Language Science Press.

Chapter 1

Statistical MWE-aware parsing

Mathieu Constant
ATILF UMR 7118, Université de Lorraine/CNRS

Gülşen Eryiğit
Istanbul Technical University

Carlos Ramisch
Aix-Marseille Université

Mike Rosner
University of Malta

Gerold Schneider
University of Konstanz and University of Zurich

This chapter aims at presenting different strategies that have been designed to incorporate multiword expression (MWE) identification in the process of syntactic parsing using statistical approaches. We discuss MWE representation in treebanks, pipeline and joint orchestrations, the integration of external lexicons and the evaluation of MWE-aware parsers, concluding with our suggestions for future research.

1 Introduction

Supervised *statistical parsing* is nowadays an important and challenging field of *natural language processing (NLP)*. It consists in predicting the most probable syntactic structure of a new sentence, given a statistical model that has been trained on a *treebank*, that is, a syntactically annotated corpus. Since the seminal works of Nivre & Nilsson (2004) for dependency parsing and Arun & Keller (2005)

Mathieu Constant, Gülşen Eryiğit, Carlos Ramisch, Mike Rosner & Gerold Schneider. 2019. Statistical MWE-aware parsing. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 1–29. Berlin: Language Science Press. DOI:10.5281/zenodo.2579043

for constituency parsing, a new research line has emerged: incorporating the analysis of *multiword expressions (MWEs)* in such parsers. The main objective of this chapter is to present different approaches that have been developed and evaluated for statistical MWE-aware parsing systems.

The design of MWE-aware parsers must address the following questions: How are MWEs represented in combination with syntactic trees? When is MWE identification performed with respect to parsing? What algorithms and machine learning techniques are to be used for the two tasks? How can external lexical resources be integrated to improve MWE coverage? How are systems evaluated?

Answering the question about MWE *representation* is fundamental as it enables the definition of a system's output. Hence, it influences the design of datasets used for training and testing, including treebanks, as shown in Section 3.

The *orchestration* issue is also crucial in order to position MWE identification with respect to parsing: should it be performed before, during, or after it? The answer is not straightforward as it might depend on the type of MWE (Eryiğit et al. 2011). Orchestration also implies determining how the two components interact. For instance, in pipeline strategies (before or after) discussed in Section 4, should the intermediate input/output be computed using MWE concatenation strategies or MWE substitution ones? Joint strategies (during) discussed in Section 5 alongside n-best strategies, might involve different methods like adapting a grammatical formalism for constituency parsing (Green et al. 2013) or concatenating arc labels in dependency parsing (Vincze et al. 2013).

Concerning *algorithms* and machine learning, most techniques use workaround approaches by adapting the MWE-aware representation to existing representations directly exploitable by off-the-shelf tools (Nasr et al. 2015). Nonetheless, new parsing algorithms have been recently proposed that include specific handling of MWEs, notably when using joint strategies (Nivre 2014).

The integration of *exogenous lexical knowledge* in the system, discussed in Section 6, is non-trivial but potentially helpful. Indeed, supervised systems are trained on datasets of limited size. Therefore, one drawback of such systems is the limited coverage in terms of MWEs. One possible solution consists in integrating knowledge coming from large-scale MWE lexicons, either manually built and/or validated (Candito & Constant 2014) or automatically acquired (Schneider 2012).

The last issue concerns *evaluation*: what is the impact of MWE identification on syntactic parsing and vice-versa? What types of measure are adequate to quantify this impact? We try to answer these questions in Section 7.

The outline of this chapter is as follows. First, we briefly explain some basic concepts and terms in statistical parsing in Section 2. Then, each section addresses the questions above. We conclude in Section 8 by providing a summary of the current research in statistical MWE-aware parsing and presenting pointers that, in our opinion, may lead to significant advances in the field in the future.

2 Statistical parsing

Parsing, also referred as syntactic analysis, is the process of assigning a syntactic structure to a given input sentence. The analysis is aimed at producing a valid syntactic tree conforming to a hand-written or automatically induced language grammar. With the emergence of manually annotated datasets (i.e. treebanks) and machine learning techniques, statistical parsing (Collins 1996; Charniak 2000) has become the dominant approach in the parsing literature.

Statistical parsing aims at selecting the most probable parse tree from the set of all possible parse trees for a given sentence. These data-driven parsing models may be basically grouped under generative or discriminative approaches. *Generative* parsing models generally rely on a grammatical formalism whereas *discriminative* ones are usually performed without any underlying grammar. There exist also joint approaches where a discriminative model is used to rerank the top n candidates of a generative parser.

Constituency and dependency formalisms are the two most common parsing formalisms used in statistical parsing. Figure 1 and Figure 4 each provide constituency and dependency parse tree samples for the sentence *The prime minister made a few good decisions.*

In the *constituency formalism*, a sentence is regarded as being composed of phrases and parsing is the task of determining the underlying *phrase structure*. For example, a statistical generative constituency parser aims to assign probabilities to a parse tree by combining the probabilities of each of its sub-phrases. In the *dependency formalism*, parsing is defined as correctly determining the dependency relations between words of an input sentence. More precisely, the aim of dependency parsing is to correctly determine the dependent-head relationships between words and also the type of these relationships such as subject, object, predicate. Dependency parsing is nowadays strikingly more popular than constituency parsing and attracts the attention of an ever-growing community in NLP. Furthermore, most existing MWE-aware parsers are developed in the dependency framework. Therefore, in this chapter, we focus mainly on different orchestration scenarios applied for different *statistical dependency parsing* approaches.

The two commonly used approaches for statistical dependency parsing in the literature are transition-based (Yamada & Matsumoto 2003; Nivre et al. 2007) and graph-based (Eisner 1996; McDonald et al. 2006; Nakagawa 2007). *Transition-based* approaches treat the dependency parsing task as the determination of parsing actions (such as push/pop operations in a shift-reduce parser) by the use of a machine learning classifier. *Graph-based* approaches treat parsing as finding the most likely path within a graph, such as the highest-scoring directed spanning tree in a complete graph. Most MWE-aware parsing strategies are adaptations of standard parsers experimenting with various models of orchestration concerning the scheduling of MWE identification with respect to syntactic analysis.

MWEs pose *challenges* for all areas of NLP, and statistical parsing is not an exception. An MWE may be *ambiguous* among accidental co-occurrence, literal, and idiomatic uses. The possible surface forms of an MWE *vary*, especially due to morphological variations which may become radical in morphologically rich languages. MWE components do not have to appear in consecutive locations within a sentence and it is hard to correctly identify a *discontinuous* MWE by ignoring the intervening words. The syntactic *non-compositionality* of MWEs may result in irregular parse trees. The ambiguous, discontinuous, non-compositional and variable nature of MWEs needs to be carefully handled during parsing in order to produce a valid syntactic structure. Additionally, annotated datasets (treebanks) are crucial resources for the training of data-driven statistical parsers. The *scarcity and limited size* of MWE-annotated treebanks is a great challenge faced by MWE-aware parsing.

3 MWE representations in treebanks

The choice of an appropriate MWE representation is crucial, with strong consequences on the format of treebanks. Representational choices that have affected existing treebanks in this way range from words-with-spaces – e.g., the French Treebank (Candito & Crabbé 2009) – to the use of special MWE syntactic relations – e.g., the Universal Dependencies project (Nivre et al. 2016). Some treebanks may not even contain MWE representations at all, while others may have sophisticated multi-layer representations (Bejček et al. 2012).

The number and variety of available MWE-aware treebanks is growing (Rosén et al. 2015). They do not necessarily cover the same kinds of MWEs. They often belong to the constituency or the dependency frameworks, but some can also be compatible with different types of grammatical formalisms, like lexical functional grammar (Dyvik et al. 2016). To narrow down the scope of this section, we focus on MWE representations in relation to treebanks *that are useful to or that have been used in statistical MWE-aware parsing*.

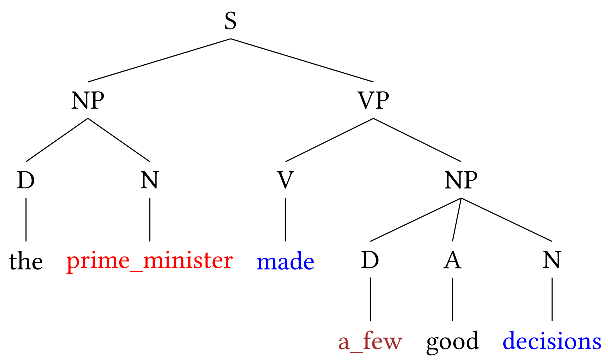


Figure 1: Constituency MWE-aware tree with words-with-spaces representation

3.1 No representation at all

The simplest and most obvious MWE representation is not to consider MWEs at all, only considering separate word tokens. While such a treatment is simplistic, it also has a number of advantages. First and foremost, it is easy to operationalize: no distinction is necessary between single words in combination and MWEs. MWEs include a variety of phenomena: compound nouns, technical terms, multiword entities, light-verb constructions, phrasal verbs, idioms, and proverbs. In general they are partly non-compositional, but due to this characteristic they also border on or overlap with collocations, which are an inherently gradient phenomenon. Not representing MWEs can thus be seen as a tacit assumption that all forms of MWEs are gradient.

Statistical parsers were conceived to improve parsing performance by modeling lexical interactions (Gross 1984; Sinclair 1991; Collins 1999). As MWEs are a subclass of collocations, the statistical attraction between the participating words is typically very strong and errors are therefore much rarer. Statistical parsers generally perform better on relations that are semantically expected (as e.g., in selectional preferences), so performance on verb complements for example is much higher than on verb adjuncts.

3.2 Words-with-spaces representation

A simple representation consists in considering MWEs as single nodes of the syntactic tree (Sag et al. 2002), such as in the strategy adopted in the LFG/XLE parser described in Chapter 4. This *words-with-spaces* representation implies that MWEs have an atomic interpretation. In the constituency framework, the MWE forms are leafs. Their parent nodes correspond to their parts-of-speech (POS) category, as shown in Figure 1. For instance, *prime minister* has a noun parent node and *a few* has a determiner parent node. A concrete example where MWEs are represented this way is the first version of the French treebank distributed for parsing (Candito & Crabbé 2009). In the dependency framework, the MWE node has the same linguistic attributes as a single word token: POS tag, lemma and morphological features. For instance, *hot dogs* would be a noun in plural, whose lemma is *hot dog*. Such representations imply that MWEs have been pre-identified and represented as word-with-space tokens before parsing. Moreover, they have several drawbacks in terms of linguistic expressiveness. First, discontinuous MWEs like the light-verb construction *make decisions* in Figure 1 cannot be represented this way. Then, the semantic processing of semi-compositional MWEs might be problematic as the internal syntactic structure is impossible to retrieve.

3.3 Chunking representations

Another way of representing MWEs uses *chunking*. Chunks are a polysemous concept, but its two meanings are related. On the one hand, chunks are seen as psycholinguistic units that are partly or fully lexicalized, that is, stored as one entity in the mental lexicon (Miller 1956; Pawley & Syder 1983; Tomasello 1998; Wray 2008). On the other hand, they are the concrete output of applying finite-state technology to obtain base-NPs and verb groups deterministically. While the psycholinguistic and the computational concepts are related, the latter has the drawback that chunks need to be continuous.

Black et al. (1991) pointed out that dependency grammars are particularly suited to model chunks and parse between heads of chunks. In fact, chunks are close to Tesnière's original conception of nucleus, which is typically not a single word (Tesnière 1959). Some dependency parsers following this scheme exist, for example Schneider (2008). Nivre (2014) has proposed a transition-based parser that performs MWE merging as it syntactically parses a sentence. This operation can be seen as MWE chunking.

A standard way of representing chunks in tagging systems is the IOB annotation scheme (Ramshaw & Marcus 1995).[1] Such representations have been successfully adapted to named entity recognition (Tjong Kim Sang 2002) and MWE identification (Vincze et al. 2011; Constant et al. 2012). For MWEs, there are variants covering continuous MWEs (Blunsom & Baldwin 2006) and gappy

[1]Tokens are tagged as B for *begin*, I for *inside* and O for *outside* a chunk.

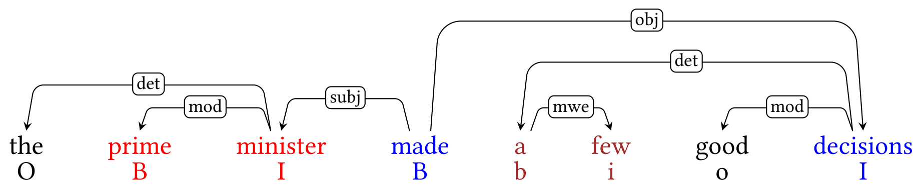


Figure 2: Chunking-based representation with IOB tags (Schneider et al. 2014)

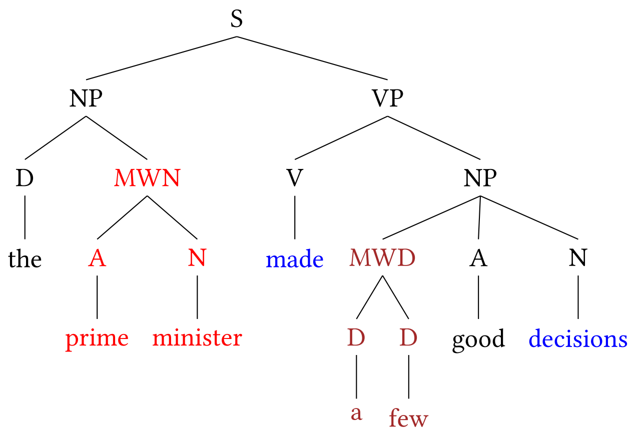


Figure 3: Flat constituency subtree representation (Green et al. 2011)

ones (Schneider et al. 2014). For instance, Schneider et al. (2014) use a 6-tag set (with additional lowercased tags in order to emphasize nested MWE structures) to represent MWEs enabling 1-level nesting, as shown in Figure 2. Such representations can be used in treebanks for training pipeline MWE-aware systems (Section 4) and joint MWE-aware parsers (Section 5).

3.4 Subtree representations

Another way of representing MWEs is to annotate them as *subtrees* made of several nodes of the syntactic tree. Many treebanks using such representations can be found in Rosén et al. (2015). Several types of subtree MWE representations were proposed in treebanks, according to the language, MWE type and syntactic formalism.

For processing purposes, words-with-spaces representations have often been automatically converted into flat subtrees. In the constituency framework, an

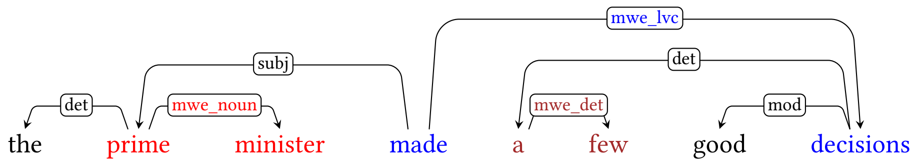


Figure 4: Flat head-initial dependency subtree representation

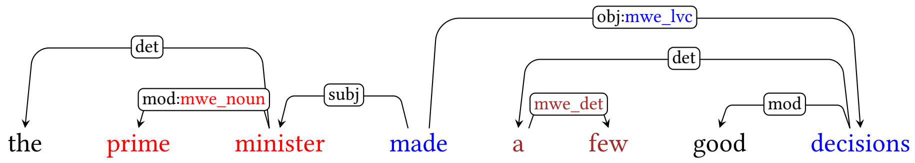


Figure 5: Structured dependency subtree representation with extended labels

MWE is considered as a special constituent with a given POS tag. MWE components are leaves of the MWE subtree, as shown in Figure 3.[2] There exist different variants for constituency treebanks (Głowińska & Przepiórkowski 2010). This representation has been used by Arun & Keller (2005) and Green et al. (2011), especially for compounds. In the dependency framework, flat subtrees can be either head-initial, that is, the root of the subtree is the first token (Nivre et al. 2004; Seddah et al. 2013), or head-final, with the root being the last token of the MWE (Eryiğit et al. 2011). All other MWE component tokens depend on this arbitrarily defined head, as shown in Figure 4. This representation is used, for example, in the Universal Dependencies treebanks (Nivre et al. 2016).

Flat subtree representations have a disadvantage: the internal syntactic structure of MWEs, required for semi-fixed MWEs in particular, is lost, like for words-with-spaces representation. To retain the internal syntactic structure as well as the MWE status, some authors propose representing an MWE with its syntactic subtree, where arc labels are extended with MWE tags, as shown in Figure 5. This kind of representation has been used, for instance, for annotating light-verb constructions (Vincze et al. 2013) and continuous MWEs (Candito & Constant 2014).

Candito & Constant (2014) adopt a hybrid representation scheme to distinguish regular from irregular MWEs. Regular MWEs have a regular syntactic

[2]MWE-related symbols `MWN` and `MWD` respectively stand for multiword noun and determiner.

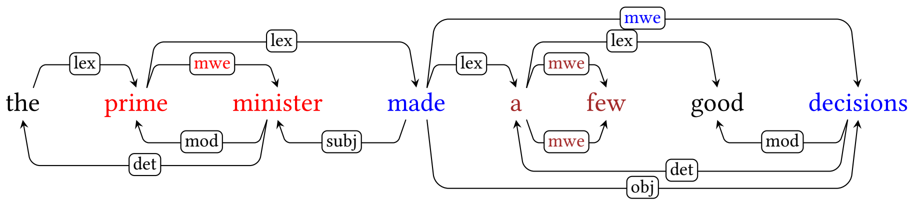


Figure 6: Representation on two distinct layers (Constant et al. 2016)

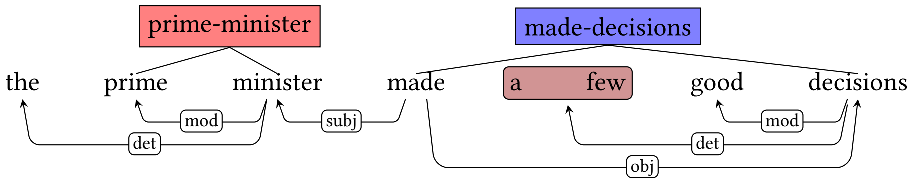


Figure 7: Representation on factorized lexical and syntactic layers (Constant & Nivre 2016)

structure[3] whereas they display semantic irregularity. They are represented with structured MWE subtrees, as in Figure 5. Irregular MWEs display an irregular syntactic structure (e.g., *by and large* is the coordination of a preposition and an adjective) and therefore cannot be analysed syntactically in a compositional way. They are represented with flat subtrees, as in Figure 4.

3.5 Multilayer representations

One of the most interesting MWE representations combined with (deep) syntactic analysis is the one used in the Prague Dependency Treebank (Bejček et al. 2012). It combines three different analysis layers in the form of trees: morphological (*m*-layer), syntactic (*a*-layer) and "semantic" ones (*t*-layer). Nodes of one layer can be linked to nodes of another layer to model the interleaving of the different types of analysis. MWEs are represented on the t-layer and are associated with MWE entries of a lexicon. To our knowledge, there is unfortunately no statistical parser outputting such combined structures.

Though less linguistically expressive, other multilayer representations have been proposed on top of a combined lexical and syntactic parser. The proposal of Constant et al. (2016) is to have two distinct layers for representing lexical and syntactic analysis in the form of dependency trees. The two layers share the same

[3]The distinction between irregular and regular MWEs is arbitrary, being defined by a manually-built set of POS patterns.

nodes, that correspond to the tokens, as shown in Figure 6. The syntactic layer represents the syntactic structure in the dependency framework. The lexical layer represents the lexical segmentation in the form of a tree. Arcs in MWE subtrees have a special label *mwe*. For instance, the MWE *prime minister* corresponds to a subtree whose root is *prime* and which is composed of an *mwe* arc from *prime* to *minister*. In order to form a unique tree for the lexical layer, lexical units are sequentially related via arcs labeled *lex*. For instance, the MWE *prime minister* is linked to the following lexical unit *made decisions*. This dual representation has several advantages. First, syntactic and lexical analyses are explicitly separated. In the case of regular MWEs, there is a clear distinction between the syntactic and the semantic status (regular syntactic structure vs. irregular semantics). In addition, the representation enables not only nested MWEs to be annotated (e.g., *a few* in *made a few good decisions*) but also fully overlapping expressions (e.g., the noun compound *rain check* inside the light verb construction *to take a rain check*). On the down side, irregular MWEs are duplicated on the two layers because there is no possible compositional syntactic analysis (e.g., *a few*). Additionally, arcs linking lexical units could be made implicit, as they can straightforwardly be computed from their positions in the sequence.

Constant & Nivre (2016) correct the main drawbacks of the previous two-layer representation by making it more compact and more factorized. The representation is still composed of two layers, but the lexical layer is a forest of constituent-like trees representing complex lexical units like MWEs, as shown in Figure 7. Here, the discontinuous MWE *made decisions* is represented by a tree whose root corresponds to a new lexical node having linguistic attributes like any token: a form (*made decisions*), a lemma (*make decision*), a POS tag (verb) and morphological features (past tense). It is straightforward to elegantly represent embedded and fully overlapping MWEs, as lexical units are trees. Irregular MWEs like *a few* and simple words are called syntactic nodes. The syntactic layer is a dependency tree over such nodes. Therefore, irregular MWE nodes and simple word nodes are shared by the two layers. For example, there is a *det* arc from *decisions* to *a few*, as it is compositionally modified by the complex determiner. This representation is not without some limitations: the lexical layer cannot represent an MWE that strictly requires a graph (and not a tree). For instance, it is impossible to represent the coordinated MWEs *had shower* and *had bath* in the sentence *John had*$_{1,2}$ *a shower*$_1$ *then a bath*$_2$.

4 Pipeline approaches

A minimal processing pipeline consists of a collection of two processes arranged in a chain so that the output of the first process is the input of the other. Thus, a processing pipeline for statistical MWE parsing involves two processes, one to identify the MWEs in the input sentence, and another for parsing the sentence into one or more structures that include the MWEs. The question that we address in this section concerns the order in which these two processes are arranged, and there are clearly two possibilities referred to as preprocessing (Section 4.1), and postprocessing (Section 4.2).

4.1 Preprocessing approaches

Preprocessing means that the MWE identification task takes place before parsing. For the parser to benefit from this, a decision must be made about how to represent MWEs in the input. As discussed earlier, there are different approaches, the most important of which employ concatenation (Section 4.1.1), or substitution (Section 4.1.2) operations, as discussed in the following sections.

4.1.1 Concatenation approach

A widely used pipeline approach to statistical MWE-aware parsing is to have a *re-tokenization* phase before parsing. It consists in first pre-identifying MWEs, then concatenating their components in one single token, and finally applying a syntactic parser trained on a treebank where MWEs have a words-with-spaces representation (Section 3.2). Note that this approach is limited to continuous MWEs.

For example, given the input token sequence *The prime minister made a few good decisions*, the MWEs *prime minister* and *a few* are first pre-identified. Each of them is then merged by concatenating its components into a single token. The sequence is retokenized as *The prime_minister made a_few good decisions* and is then parsed. This approach has the advantage of reducing the token-count of the sentence and hence reducing the search space of the parser. However, it may not be realistic to recognize some types of MWEs without access to morpho-syntactic information.

Seminal studies on gold MWE identification performed before either constituency parsing (Arun & Keller 2005) or dependency parsing (Nivre et al. 2004; Eryiğit et al. 2011) showed that it may have a great impact on parsing accuracy. Other studies confirmed that more realistic MWE pre-identification actually helps parsing. Korkontzelos & Manandhar (2010) evaluated MWE pre-identification using Wordnet 3.0 for lexicon lookup before shallow parsing. The set of MWEs was limited to two-word continuous compound nominals, proper names, and adjective-noun constructions. The authors showed that the approach improves shallow parsing accuracy. For instance, without MWE pre-identification, *he threw the fire wheel up into the air* is erroneously parsed as: *(he) (threw) (the fire) (wheel up) (into) (the air)*, whereas with MWE pre-identification the result is: *(he) (threw) (the fire_wheel) (up) (into) (the air)*. Cafferkey et al. (2007) carried out similar experiments with a probabilistic constituency parser. MWEs were automatically identified by applying a named entity recognizer and list of prepositional MWEs. A slight but statistically significant improvement was observed. We should note that in the above studies, MWE identification itself was not evaluated.

The SPMRL shared task (Seddah et al. 2013) had a special track dedicated to MWE-aware parsing in French. The provided treebank included continuous MWE annotations represented as flat subtrees (Figure 4). All but one competing team did not develop special treatments for MWEs. The winning team was the only one to have a preprocessing stage to identify MWEs using a tagger based on linear conditional random fields (Constant, Candito, et al. 2013). The tagger model also incorporated features based on an MWE lexicon (Section 6.3).

4.1.2 Substitution approach

Another approach is to use substitution: whenever an MWE from the lexicon matches, it is replaced by its head word. Such approach is employed by Weeds et al. (2007) for technical terms (Section 6.2), and by Schneider (2008) on all chunks. In a typical substitution approach, for example, the term *natural language processing* would be replaced by *processing* before parsing.

The advantage of keeping the lexical head is that resources taking lexical relations into account, such as bi-lexical disambiguation (Collins 1999), can use the lexical information. Thus, potential sparsity problems are reduced in comparison to the concatenation approach. For example, the prepositional phrase attachment ambiguity in *We help users with natural language processing* can be resolved properly, even if *natural language processing* is unseen in the training data. As long as *processing* exists in the training corpus, the ambiguity can be solved because the combination *help-with-processing* is more likely than *user-with-processing*.

The potential drawbacks of this approach are that, on the one hand, strings may be ambiguous, and on the other hand non-compositionality may affect the results. Ambiguous strings are illustrated below: while the first sentence of each example is an MWE, the second is accidental cooccurrence. The last example involves light verbs, for which Tu & Roth (2011) use token-wise disambiguation, as ambiguity is relatively frequent.

(1) a. I saw her, and *by the way* she went there on foot
    b. I recognized her *by the way* she walks

(2) a. In *natural language processing*, humans are also challenged
    b. In *natural language processing* can be difficult

(3) a. The politician *took* a strong *position* on the issue
    b. The soldier *took* a vanguard *position* on the mountain top

Non-compositionality may lead to situations in which the head is semantically so different that attachment preferences are also affected.

(4) a. I saw the road with the *torch light*
    b. I saw the road with the *traffic light*

If the MWE *traffic light* is reduced to *light*, the chances are that the prepositional phrase is erroneously attached to the verb, as *see-with-light* is likely. If *traffic light* is treated as an MWE, bi-lexical disambiguation can only profit if very large annotated resources exist. Unless a backoff method to treat MWE components is included, the increased data sparseness may easily lead to worse results.

4.2 Postprocessing approach

In this section, we present approaches where parsing precedes MWE processing. We make a distinction between MWE identification and discovery. We define *identification* as the process of recognizing MWEs in context, that is, as tokens inside running text. On the other hand, *discovery* aims at creating a lexicon of MWE types from the corpus. This lexicon can later be used to guide MWE identification and parsing. In this section, we describe approaches for identification after parsing (Section4.2.1) and for discovery after parsing (Section 4.2.2), focusing on works in which the result of discovery was later employed for identification.

4.2.1 Post-parsing MWE identification

Identifying MWEs after syntactic parsing is a natural approach to MWE-aware parsing as an MWE generally constitutes a syntactic constituent. In the dependency framework, there is usually a path continuously linking the MWE components in the syntactic tree. As a consequence, pre-parsing is particularly relevant for detecting discontinuous MWEs, that is, MWEs that include alien elements, by employing adapted lexicon lookup methods. In Figure 7, the MWE *made decisions* is discontinuous. As there is an object arc from *made* to *decisions*, the two words are *syntactically* adjacent. A matching procedure taking the syntactic structure into account can therefore be beneficial for MWE identification. Furthermore, MWEs can have different syntactic variants. For instance, *a* ***decision*** *was* ***made*** *by John* is the passive voice variant of *John made a decision*. The detection of such syntactic variants obviously benefits from the result of syntactic parsing.

Fazly et al. (2009) identify verb-noun expressions in a parsed text based on a list of 60 candidate expressions. First, they identify candidate occurrences of the expressions using rules based on syntactic annotations and lexical values. Then, they discriminate MWEs from literal expressions using different methods. One is based on the assumption that a verbal MWE expression has *fewer syntactic variants* than its literal counterparts, giving rise to the heuristic that canonical forms are idiomatic (e.g., *pull one's weight*) and non-canonical variants are literal (e.g., *pull a weight, pull the weights*). Another method compared the distributional contexts of co-occurring verb-object pairs to two sets of gold-standard contexts: one for idiomatic readings and another one for literal readings.

Nagy T. & Vincze (2014) compare the use of parsers and of a syntax-based pipeline approach to identify verb particle constructions in English. English off-the-shelf parsers usually have a specific syntactic arc label to identify occurrences of verb-particle constructions. Nonetheless, such parsers tend to get good precision but low recall, as they do not use dedicated features for this task. The pipeline method developed in this paper uses a standard parser to identify a first set of candidates. This set is subsequently enlarged using other syntactic relations. A classifier is then applied in order to decide whether they are verb-particle constructions or not. They show a significant gain in terms of recall and F-score with respect to standard parsers on the Wiki50 corpus (Vincze et al. 2011).

4.2.2 Post-parsing MWE discovery

This section discusses the discovery of new MWEs after parsing. This is particularly useful for the creation of resources that can be used for MWE-aware parsing (Section 6). For instance, such lexicon of newly discovered MWEs can be subsequently used for MWE pre-identification at the next cycle of processing. Seretan (2011) has shown that discovery based on parsed corpora provides considerably cleaner results than those relying on shallow analysis (e.g., POS-tagged corpora). Chapter 8 discusses the integration of resources built with the help of MWE discovery into a language-independent symbolic parser.

Since the literature in MWE discovery is huge, we focus on two studies that represent a sample of this type of approach. Lehmann & Schneider (2011) and Ronan & Schneider (2015) used automatically parsed data for discovering MWEs of different types, including idiomatic verb + prepositional phrase (PP) combinations and light-verb constructions in English. These cases involved the use of different collocation extraction scores.

Table 1: Top-ranked verb-object + preposition-noun tuples, using the the O/E score (Lehmann & Schneider 2011)

verb	object	prep	desc. noun	T-score	O/E
send	*shiver*	*down*	*spine*	5.74456	2.21477×10^8
tap	*esc*	*for*	*escape*	6.40312	2.1134×10^8
separate	*shield*	*from*	*plate*	6.78233	2.33384×10^7
refer	*gentleman*	*to*	*reply*	8.24621	7.8143×10^6
obtain	*property*	*by*	*deception*	5.2915	7.60043×10^6
ask	*secretary*	*for*	*affairs*	6.40312	5.01529×10^6
kill	*bird*	*with*	*stone*	5.38516	3.37917×10^6
add	*insult*	*to*	*injury*	6.08276	2.21769×10^6
throw	*caution*	*to*	*wind*	5.09902	2.03157×10^6
refer	*friend*	*to*	*reply*	7.54983	1.36298×10^6
report	*loss*	*on*	*turnover*	7.14142	1.34742×10^6

For discovering Verb-PP idioms the O/E score was used, combined with filters including T-score and Yule's K (which estimates the degree of non-modifiability of a candidate). Table 1 reproduces the results of discovery, sorting the candidates by descending O/E score. Among the top-ranked candidates, many are genuine idioms (e.g., *to kill two birds with one stone*).

For discovering light-verb constructions, the t-score was used together with a number of filters including WordNet and NomBank lookup (Ronan & Schneider 2015). An example of analysis is shown in Figure 8, showing a precision and recall plot by candidate list length. The vertical axis shows precision and recall, respectively, the horizontal axis (which is logarithmic) gives the cutoff in the ranked list of candidates to be included in the evaluation. For the cutoff at 20, the reported candidates for *give*+object, precision is 100%, while recall is 10%. At rank 2560, about 88% of all instances in the gold standard were found.

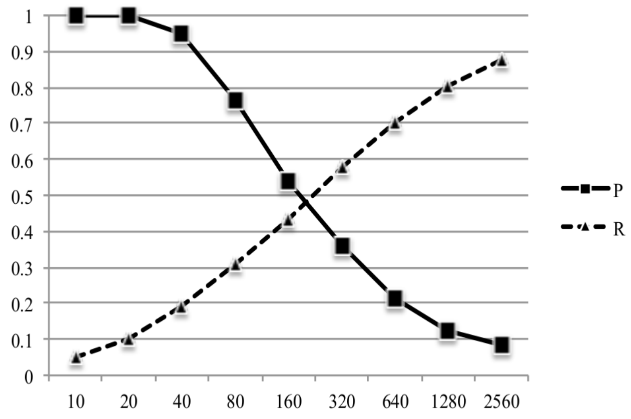


Figure 8: Precision vs. recall curve of the light verb *give* in the British National Corpus, using t-score (Ronan & Schneider 2015)

5 Joint approaches

Joint approaches perform parsing and MWE identification simultaneously. Since syntactic and lexical-semantic information are complementary, both processes can help each other if performed together. In such systems, MWE lexical-semantic segmentation is often seen as a by-product of syntactic analysis, or vice-versa.

Some MWEs require quite sophisticated syntactic information to be recognized, such as subcategorization frames and phrase structure. Joint approaches favor delaying the decision as to whether a given combination is an MWE to the parser, where this information is available. In other words, the system has access to the right information at the right moment.

Parser evaluation scores are often reported on standard test sets, where MWEs have been manually pre-identified (gold). Jointly performing MWE identification and parsing is more realistic than parsing pre-annotated test sets, where MWEs are often represented as words with spaces (Figure 1). Indeed, when moving from standard test sets to real texts, gold MWE identification is not necessarily available. It may be hard to use a pipeline approach (Section 4) if the target MWEs are ambiguous or discontinuous.

On the downside, parsers that perform both syntactic analysis and MWE identification simultaneously are harder to design. First, ambiguity is increased, often by a larger number of labels and/or parsing decisions that are possible at a given moment. It is crucial, for such systems, to have coherent MWE annotations in treebanks, datasets that are large enough, and features that generalize well.

We classify such approaches according to the degree of "MWE-awareness" of the parser. In shallow approaches, the parser generates *n*-best solutions without putting any particular emphasis on MWEs, then uses MWE information for reranking (Section 5.1). The majority of joint approaches add MWE information to training and test treebanks, and then use off-the-shelf parsers enriched with dedicated MWE features (Section 5.2). We also present fully MWE-aware parsers that take them into account in the parsing algorithm itself (Section 5.3).

5.1 N-best and reranking approaches

One possible orchestration solution is to consider MWE identification as a retokenization problem, as described in Subsection 4.1.1. In *n*-best approaches, however, the text is first segmented into tokens in a non-deterministic way, considering *several possible segmentations*. Usually, the output of such non-deterministic tokenizer is a lattice containing all possible segmentation paths for a sentence (Sagot & Boullier 2005). This representation is particularly suited for ambiguous irregular constructions, that could be considered as MWEs or as accidental cooccurrence, depending on the context. The parser then must take this ambiguous segmentation and uses simple parsing models to disambiguate the input and generate a parse tree (Nasr et al. 2011).

An *n*-best MWE identifier is used by Constant, Le Roux & Sigogne (2013), producing a lattice of possible segmentations. Then, a PCFG-LA parser is used to disambiguate the possible readings. The authors test two variants. First, they consider that MWEs in the lattice are single nodes (words with spaces). Thus, different segmentation possibilities in the lattice are represented by paths with different lengths. Second, they consider that MWE components are individual nodes tagged using an IOB scheme, like in Figure 2. The latter obtains better performance because all possible paths in the lattice have the same length, resulting in more accurate parsing scores.

Conversely, the parser can use the same kind of approach and also generate *n*-best parsing trees. A reranker can then use MWE-aware features, among others, to choose the highest scoring tree. Constant et al. (2012), for instance, use a deterministic tokenizer but output *n*-best MWE-aware syntactic trees using the Berkeley constituency parser. Then, they use a discriminative reranker to choose the correct parse tree that includes MWE features.

These are considered joint approaches because, even though MWE segmentation and parsing are independent processes, one needs to be aware of the format of the input/output of the other. For example, the parser has to be able to process lattices as input, provided by the non-deterministic MWE identifier.

5.2 Treebank modification approaches

In Section 3, we discussed several ways to represent MWEs in treebanks. Standard statistical parsers trained on such treebanks will be inherently aware of MWEs, provided that they can handle the particular MWE representation in that treebank. For example, if MWEs are represented as subtrees (Figure 5), then there is no need to explicitly handle MWEs (Nivre et al. 2016). This subsection covers MWE-aware parsing studies in which the learning and parsing algorithms *remain unchanged* with respect to their standard version.

Approaches discussed in this section face several challenges. First, most of the time MWEs are either *absent* from treebanks, or the available representation requires *adaptations* in order to be usable by the parser. Second, parsers learned from MWE-annotated treebanks often require *extra features* to take MWEs into account properly. Third, these features may suffer from data *sparseness*, as individual MWEs may not occur often enough in limited-size treebanks.[4]

In this subsection, we present approaches that tackle the challenges posed by MWEs by:

- adding or modifying the MWE representation in the treebanks, and/or
- adding MWE-dedicated features to the parsing model.

The last challenge, related to data sparseness and domain adaptation, is tackled by integrating external resources in the parser, as discussed in Section 6.

In constituency parsing, several parsers, MWE representations and feature sets have been tested, especially on continuous MWEs in the French Treebank. Constant, Le Roux & Sigogne (2013) experiment with two implementations of a PCFG-LA parser, using a representation similar to the one of Green et al. (2011) and a variant similar to IOB encoding.

When MWE annotation is absent, a reasonably straightforward solution is to automatically project an MWE lexicon on the treebank before training the parser. For instance, Kato et al. (2016) project a lexicon of compound function words (e.g., *a number of*) onto the English Ontonotes constituency treebank. Syntactic trees are modified to take MWEs into account. Constituents are then automatically transformed into dependencies and a standard first-order graph-based parser is learned. While the training data is modified, no MWE features are added to the model.

[4]Some MWE categories may never occur (e.g., colloquial idioms) because many existing treebanks cover a single register (e.g., newspapers).

Early experiments on MWE-aware dependency parsing compared two representation variants: MWEs as subtrees or as words with spaces (Nivre & Nilsson 2004). The results indicated that the subtree representation (joint approach) is worse than parsing MWEs as words with spaces (pipeline approach). However, these results were obtained assuming gold MWE segmentation.

Vincze et al. (2013) were among the first to use a dependency parser to perform realistic MWE identification. They focus on light-verb constructions (LVCs) in Hungarian. They first perform an automatic matching of two annotation layers in the Szeged Treebank: syntactic dependencies and LVCs. As a result, the dependency link between a light verb and a predicative noun (e.g., OBJ) is suffixed with a LVC tag, whereas regular verb-argument links remain unchanged, like in Figure 5. An off-the-shelf parser is used to predict the syntactic structure of sentences, including LVC links. Given that Hungarian is a relatively free word-order language, LVCs often involve long-distance dependencies. When compared with a classifier baseline, the parser performs slightly worse on continuous LVC instances (F1=81% vs. 82.8%) but considerably better on discontinuous LVCs (F1=64% vs. 60%).

Treebanks containing MWEs as words with spaces pose problems when converted into subtrees. When splitting an MWE, one needs to manually or semi-automatically assign POS tags, lemmas and morphological features to the individual MWE components. Additionally, the internal syntactic structure must be inferred. Since it is difficult to automate this task, the internal syntactic structure of decomposed MWEs is often underspecified using flat head-initial subtrees (Seddah et al. 2013), head-initial (Nivre et al. 2016) or head-final chained subtrees (Eryiğit et al. 2011), as detailed in Section 3.4. Eryiğit et al. (2011) compare parsing and MWE identification accuracy on different treebank representations for different MWE types. Their original treebank includes MWEs as words with spaces, which are semi-automatically transformed into subtrees. Contrary to previous conclusions (Nivre & Nilsson 2004), results indicate that subtrees may be a more suitable solution for some MWE types, specially when looking at MWE-aware parsing evaluation metrics (Section 7). In this study, the words-with-spaces representation is shown to have a harming effect on the types where it increases lexical sparsity, such as in Turkish light-verb constructions.

Candito & Constant (2014) explore several orchestrations for combining syntactic parsing and continuous MWE identification in French, distinguishing syntactically regular from irregular multiword constructions. In particular, they experimented with an off-the-shelf graph-based parser that was learned from an MWE-aware treebank where the subtrees representing regular and irregular expressions have their usual labels suffixed by the POS of the MWE, as shown in

Figure 5. They showed on-par results with different pipeline variants.

Nasr et al. (2015) focus on ambiguous compound grammatical words in French of the form ADV+*que* and *de*+DET. While these represent a limited scope, such constructions are pervasive and hard to identify without access to syntactic information, because its component words can co-occur by chance. For instance, the two sentences below have the same sequences of POS and similar lexical units, but the first one contains an MWE whereas the second one does not:

(5) *Je chante* ***bien que*** *je sois triste*

I	sing	well	that	I	am	sad

'I sing even though I am sad'

(6) *Je pense* ***bien que*** *je suis triste*

I	think	well	that	I	am	sad

'Indeed, I think that I am sad'

In order to deal with these constructions, the training treebank is modified similarly to Candito & Constant (2014), splitting MWEs originally represented as words with spaces into two tokens linked by a special dependency. For example, since *bien que* functions as a conjunction, the conjunction *que* becomes the head, modified by the adverb *bien*. Using a standard graph-based dependency parser, the authors evaluate the identification of the target MWEs on a dedicated dataset. As described in Section 6.3, the use of subcategorization frame information for verbs, coming from an external lexicon, improves the results.

5.3 MWE-aware parsing models

The models discussed up to now have the advantage of being simple and fast to deploy. Provided that the training treebank contains MWEs in a suitable representation (which can be manually or automatically converted), the parsing algorithm itself does not need to be changed to accommodate MWEs. These approaches achieve reasonably good results, specially if compared to MWE systems based on purely sequential models. However, they often use language-specific or treebank-specific workarounds and are not always generalizable. Therefore, some recent contributions focus on designing parsing models that are truly aware of MWEs in the model, with promising results.

In the framework of constituency parsing, Green et al. (2011) propose and evaluate an MWE-aware parser based on tree substitution grammars (TSGs). This work was latter extended, comparing the TSG with a PCFG model enriched with a factorized lexicon (Green et al. 2013). The authors apply these models to MWE-rich treebanks for French and Arabic, showing gains for both parsing and MWE identification. The authors state that TSGs are more powerful than PCFGs, being able to store lexicalized tree fragments. They are therefore more suitable for idiomatic MWEs, whose particular syntactic analysis requires larger contexts to be predicted.

Along the same lines, Le Roux et al. (2014) design a joint parsing and MWE identification model based on dual decomposition. In this work, however, a specialized sequence model performs lexical segmentation of MWEs. The MWE identification module uses conditional random fields, while the parsing module uses a PCFG-LA also including MWE identification, using the approach of Green et al. (2013). Both models are combined using penalty vectors that are updated in an iterative way. In other words, until reaching consensus on MWE identification, the MWE identifier and parser analyse the input sentence. If the systems do not agree, they are penalized in proportion to the difference between the given solution and the average solution. This model reaches impressive performance on the French treebank, reaching an MWE identification F-score of up to 82.4% on the test set.

Constant & Nivre (2016) propose a new dependency parsing system that jointly performs syntactic analysis and lexical segmentation (including MWE identification). The authors design and evaluate a transition-based parser using two synchronized stacks: one for syntactic parsing and another for lexical segmentation. The synchronization of both stacks is guaranteed by a unique Push transition which pushes the first element of the buffer on both stacks. The parser models MWE-dedicated transitions $\textsc{Merge}_N$ and $\textsc{Merge}_F$, which respectively create new merged lexical nodes for regular MWEs and lexico-syntactic nodes for fixed MWEs. An additional Complete transition marks that a given lexical node has been fully parsed (while being potentially implicit). This approach obtains results that compare with or exceed state-of-the-art performance on French and English MWE-rich treebanks. Finally, the authors show that lexical information can guide parsing, leading to slightly better syntactic trees. The converse assumption does not seem to hold, though, as adding syntactic information to a purely lexical parser tends to slightly degrade its performance.

6 Integration of lexical resources

Lexical resources are large-scale repositories of information typically about simple words, more rarely about MWEs. They can play different roles with respect to statistical MWE-aware parsing, and in this section we discuss three of them. We show how lexical information can help in general to resolve parsing ambiguities (Section 6.1). Then, we focus on the availability of lexical information within pipeline approaches (Section 6.2). Finally, we shift the emphasis to the effect of lexical resources on MWE identification rather than on parsing itself (Section 6.3).

6.1 General integration of lexical resources in statistical parsers

Statistical parsers have several drawbacks due to the limited size of available gold standard treebanks used for training. Many words in the datasets are infrequent, which makes it very difficult to learn relevant (lexical) regularities. In addition, when parsing an unseen text, some words are simply absent from the training dataset, which negatively impacts parsing accuracy. Experiments with different solutions have been undertaken within the parsing community, notably by incorporating external resources mostly (but not only) learned automatically from large raw corpora.

The use of word clusters is one method to deal with the lexical sparsity issue. Clusters (e.g., Brown clusters), consist of groups of words occurring in the same context. Replacing words by clusters or using clusters as features has each been shown to improve parsing accuracy (Koo et al. 2008; Candito & Seddah 2010). Pairs of words that co-occur frequently in large corpora tend to be related syntactically. The provision of information about such lexical affinities to the parser has been shown to usefully support syntactic attachment decisions. Lexical affinities might be integrated using either soft constraints (Bansal & Klein 2011; Mirroshandel et al. 2012) or hard ones (Mirroshandel & Nasr 2016). The deep learning revolution has opened new perspectives to help handle lexical sparsity, as words are represented as continuous space vectors (i.e., word embeddings) learned from large corpora. Words having similar syntactic behaviors have vectors that are geometrically close to each other (Durrett & Klein 2015; Dyer et al. 2015).

The use of external lexicons has also turned out to be of great interest, notably for dependency parsing. For instance, Candito et al. (2010) successfully use the MELt tagger (Denis & Sagot 2012), thereby incorporating features based on a large-scale morphological lexicon. The integration of hard constraints based on syntactic lexicons was also shown to have a positive impact (Mirroshandel et al. 2013).

6.2 MWE resources help parsing

We now give examples of MWE lexical resource integration using a pipeline approach (Section 4) in which MWEs are replaced by their syntactic heads. We do so on two levels: general NP chunking and technical terms.

On the chunking level, replacing chunks with their head words reduces parsing complexity considerably. According to experiments carried out by Prins (2005), parsing performance also increases slightly. However, experiments on technical terms have not confirmed this hypothesis. In other words, replacing chunks with their heads does not necessarily lead to improved results in other settings.

Weeds et al. (2007) used a substitution approach (Section 4.1.2) for term identification in the domain of biomedical research, where gene and protein names in particular are often MWEs. Because taggers, unless they are trained on the domain, perform very poorly, they report better results when replacing technical terms with their head, using a large lexicon of domain terms.[5]

A comparable example is the situation in which a sentence such as … *he did not see the traffic_N light_V* is POS-tagged incorrectly (*light_V* instead of *light_N*). Here, a pipeline substitution approach relying on an MWE lexicon can clearly improve results. This improvement is passed on to the subsequent parsing step. When domain-adapted taggers are available, though, the advantages of the substitution approach tend to disappear. The performance of adapted taggers is often comparable or slightly higher than that of the substitution approach, as tagging accuracy of technical terms increases. In short, sometimes it is better to adapt statistical models (in this case, a domain-adapted tagger) rather than using lexical resources (in this case, an MWE gazetteer of the domain).

Schneider (2014) conducted an experiment using LT-TTT2, an off-the-shelf rule-based named entity recognizer (Grover 2008) on the standard evaluation suite GREVAL (Carroll et al. 2003) with the same approach of replacing multiword named entities by the head of the MWE. The performance of the substitution approach was slightly worse than when leaving the MWE unchanged. Also this experiment did confirm that statistically motivated resources are usually better than purely lexical resources.

6.3 Lexical resources help MWE-aware parsing

Having discussed the effect of lexical resources on parsing accuracy, we now turn to two different ways to use them as a source of features for dependency parsers, to help MWE identification as well as parsing accuracy.

[5]Such a lexicon is often referred to as *gazetteer*.

The first is to use MWE lexicons to alleviate the low coverage of MWEs in the training dataset. The idea is to perform an MWE pre-segmentation of the input text by lexicon lookup. The pre-segmentation, encoded in an IOB-like format, is then used as source of features during MWE-aware parsing, either in the parser itself for joint approaches (Candito & Constant 2014), or in the MWE tagger applied before parsing in pipeline approaches (Constant et al. 2012; Constant, Candito, et al. 2013).

One advantage of using soft constraints like features is their ability to handle ambiguous MWEs. Let us take the sequence *up to*, which can be either a complex preposition (*no more than*) or an accidental co-occurrence (*look up to the sky*). A naive segmentation will systematically consider it to be an MWE, independently of the context. However, a better decision can be made taking the context (i.e., the set of other features) into account. Using a joint approach on the French treebank, Candito & Constant (2014) managed to gain around 4 points in terms of tagged MWE identification F-score using such lexicon-based features: F1 = 74.5 (with) vs. F1 = 70.7 (without). We should recall, however, that their approach is limited to continuous MWEs.

A second method proposed by Nasr et al. (2015) is to incorporate subcategorization frame information, derived from a syntactic lexicon, as features in a joint parser. This was used to improve the resolution of ambiguities between grammatical compound MWEs and accidental co-occurrences. An example is the French sequence *bien que* which is either a multiword conjunction (*although*) or an adverb (*well*) followed by a relative conjunction (*that*), as exemplified in Subsection 5.2. This ambiguity may be resolved using information about the verb in the syntactic neighborhood. The authors included specific features indicating whether a given verb accepts a given complement: *manger (to eat)* −QUE −DE, *penser (to think)* +QUE −DE, *boire (to drink)* −QUE −DE, *parler (to speak)* −QUE +DE. In particular, they show for French that there is a 1-point gain in F-score, 85.24 (without) vs. 86.41 (with), for MWEs of the form ADV+*que* (ADV+*that*). The effect is spectacular for compounds of the form *de*+DET, that display a 15-point gain : 75.00 (without) vs. 84.67 (with).

7 Evaluation

Evaluating a syntactic parser generally consists in comparing the output to reference (gold-standard) parses from a manually labeled treebank. In the case of constituency parsing, a constituent is treated as correct if there exists a constituent in the gold standard parse with the same labels, starting and ending points. These parsers are traditionally evaluated through precision, recall and F-score (Black et al. 1991; Sekine & Collins 1997).

In standard dependency parsing with single-head constraint[6], the number of dependencies produced by a parser is equal to the number of total dependencies in the gold-standard parse tree. Common metrics to evaluate these parsers include the percentage of tokens with correct head, called *unlabelled attachment score* (UAS), and the percentage of tokens with correct head *and* dependency label, called *labeled attachment score* (LAS) (Buchholz & Marsi 2006; Nilsson et al. 2007).

The evaluation of MWE-aware parsers and the evaluation of whether or not MWE pre-identification helps improving the parsing quality should be carefully carried out. As stated in previous sections, in most works where MWE identification is realized before parsing, the MWEs are merged into single tokens. As a result, the common metrics for parsing evaluation given above become problematic for measuring the impact of MWE identification on parsing performance (Eryiğit et al. 2011). For example, in dependency parsing, the concatenation of MWEs into single units decrements the total number of evaluated dependencies. It is thus possible to obtain different scores without actually changing the quality of the parser, but simply the representation of the results. Instead of UAS and LAS metrics, the attachment scores on the surrounding structures, namely UAS_{surr} and LAS_{surr} (i.e., the accuracy on the dependency relations excluding the ones between MWE elements) are more appropriate for extrinsic evaluation of the impact of MWE identification on parsing. Similar considerations apply to constituency parsing.

Figure 9 provides two example sequences for the phenomena discussed above; one containing a continuous MWE (on the left side) and another one containing a non-continuous MWE (on the right side). The dependency trees in this figure provide the gold standard unlabeled dependency relations for both examples. Correctly predicted dependencies are presented with check marks (✓) over the relations, whereas the wrongly predicted dependencies are presented with a cross mark (✗). The continuous MWE of the left side sequence consists of three tokens (w_4, w_5 and w_6). In other words, the two dependency relations of the overall sequence belong to the relations between MWE elements. The non-continuous MWE of the right side sequence consists of two tokens (w_3 and w_6).

The first examples of each column ("(A)" and "(E)") show the success of a dependency parser without any prior MWE identification process. In the remaining settings, an MWE identifier is run over the given sequence before parsing. Both

[6]Each dependent node has at most one head in the produced dependency tree.

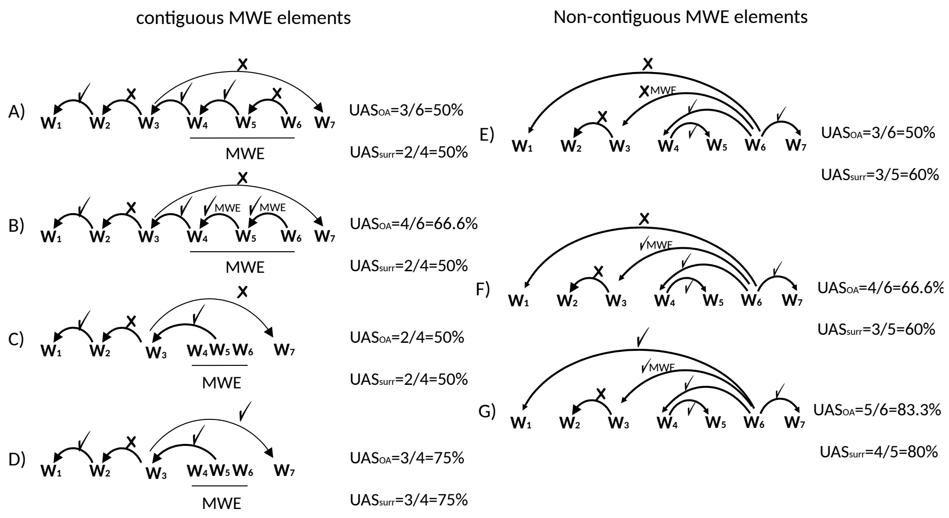


Figure 9: Extrinsic evaluation examples of the impact of MWE identification on dependency parsing performances

the overall unlabeled accuracy UAS_{OA} and the accuracy of the surrounding structures UAS_{surr} are provided next to the trees. Examples "(B)", "(C)" and "(D)" show the correctly detected relations by applying an MWE identifier prior to the syntactic parsing. In "(C)" and "(D)", the detected MWE is combined into a single unit ($w_4w_5w_6$) whereas in "(B)", the detected MWE is represented as a subtree.

In "(A)", "(B)" and "(C)", although the parser success does not change on detecting the syntactic dependencies, UAS_{OA} is affected by the total number of evaluated dependencies, whereas UAS_{surr} remains stable, as expected. In "(D)", MWE identification helps the parser to detect one more dependency relation, which is reflected in UAS_{surr}. Similarly, in "(F)", the pre-identification of "w_3 - w_6" MWE has no impact on the parser's performance. Although this can be directly observed by UAS_{surr} (60%), UAS_{OA} mistakenly gives the impression of an improvement in parsing performance (50% $\Longrightarrow$ 66.6%). This is because in this setting (second column of Figure 9) UAS_{OA} evaluates the performance of MWE pre-identification and dependency parsing as a whole. In "(G)", the parser performs better after MWE identification, which is again reflected in the surrounding structure evaluation.

Although UAS_{surr} and LAS_{surr} are valuable scores for measuring the impact of identifying different MWE types on parsing performance, they are troublesome with automatic MWE identification, when gold-standard MWE segmentation is not available. Then, erroneous MWE identification would degrade parsing scores on the surrounding dependencies. An alternative solution is to detach the concatenated MWE components (if any) into a dependency or constituency subtree (Candito & Constant 2014; Eryiğit et al. 2011). This way, the standard evaluation scores UAS and LAS are still applicable in all different orchestration scenarios, for both continuous and non-continuous MWEs, successfully assessing the performance of joint syntactic parsing and MWE identification as a whole.

8 Conclusions

In this chapter, we elaborated upon several approaches for combining MWE processing with statistical parsing to yield statistical MWE-aware parsing. These approaches depend on different parameters such as MWE representation, orchestration and external resource integration. First of all, the selected MWE representation combined with syntactic analysis have a strong impact on the system implementation, since the more elaborated and hence more linguistically expressive the representation is, the more complex the computational system has to be. Representations vary from simple words with spaces to multilayer structures. The timing of MWE identification with respect to syntactic parsing, namely orchestration, is a crucial feature that needs to be carefully taken into account when designing a statistical MWE-aware parser, as the best choice partly depends on MWE type under consideration. MWE identification may be performed before, after, or during parsing. The first two were discussed under the rubric "pipeline" approaches in Section 4; the third, under "joint" approaches, in Section 5. Last, we showed that the use of external resources is another important feature that is required to handle the sparsity problem, not only to support syntactic attachment decisions, but also MWE identification.

Although it is difficult to draw hard and fast conclusions, it seems that further investigation of dedicated MWE-aware parsing models is called for. Such models can benefit from joint modeling of closely related tasks, with information from one layer helping to disambiguate the other. Joint approaches seem to offer a very promising line of research, as has been shown for other NLP tasks: e.g., joint POS tagging and parsing (Bohnet et al. 2013), joint syntactic and semantic parsing (Henderson et al. 2013). Such approaches are now becoming prominent in NLP alongside the deep learning revolution. In fact, most joint approaches to statistical MWE-aware parsing are not truly joint, as they consist of workaround solutions. We saw how many studies investigated the use of off-the-shelf parsers by modifying training data, thus making the datasets MWE-aware. Truly joint systems are rarer, requiring the use of specific grammatical formalisms for constituency parsing or the development of new dependency parsing mechanisms dedicated to MWE identification.

As a consequence, there is much ground for future work. However, special emphasis should be given to the development of MWE-rich treebanks. Not only are these resources lacking for many languages, but also the representation and covered MWE types vary considerably among different resources. We believe that the development of new MWE-aware parsing models and resources would enable satisfactory solutions for this hard problem. Such solutions could then be further integrated into downstream applications, taking a significant step towards semantic processing of MWEs, and thus of a key element of language itself.

Acknowledgements

This work was supported by the PARSEME action (COST IC1207). It was also partially funded by the French National Research Agency (ANR) through the PARSEME-FR project (ANR-14-CERA-0001) and by a TUBITAK 1001 grant (no: 112E276).

References

Arun, Abhishek & Frank Keller. 2005. Lexicalization in crosslinguistic probabilistic parsing: the case of French. In *Proceedings of the 43rd annual meeting of the Association for Computational Linguistics (ACL'05)*, 306–313. Ann Arbor, Michigan: Association for Computational Linguistics. http://aclweb.org/anthology/P05-1038.

Bansal, Mohit & Dan Klein. 2011. Web-scale features for full-scale parsing. In *Proceedings of the 49th annual meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT'11)*, 693–702. Portland, Oregon. http://www.aclweb.org/anthology/P11-1070.

Bejček, Eduard, Jarmila Panevová, Jan Popelka, Pavel Straňák, Magda Ševčíková, Jan Štěpánek & Zdeněk Žabokrtský. 2012. Prague Dependency Treebank 2.5–a revisited version of PDT 2.0. In *Proc. of COLING 2012*, 231–246. Bombay, India.

Black, E., S. Abney, S. Flickenger, C. Gdaniec, C. Grishman, P. Harrison, D. Hindle, R. Ingria, F. Jelinek, J. Klavans, M. Liberman, M. Marcus, S. Roukos, B. Santorini & T. Strzalkowski. 1991. Procedure for quantitatively comparing the syntactic coverage of English grammars. In *Proceedings of the workshop on speech and natural language*, 306–311. Pacific Grove, California: Association for Computational Linguistics. https://doi.org/10.3115/112405.112467.

Blunsom, Phil & Timothy Baldwin. 2006. Multilingual Deep Lexical Acquisition for HPSGs via Supertagging. In *Proceedings of the 2006 conference on empirical methods in natural language processing (EMNLP 2006)*, 164–171. Sydney.

Bohnet, Bernd, Joakim Nivre, Igor Boguslavsky, Richard Farkas, Filip Ginter & Jan Hajic. 2013. Joint Morphological and Syntactic Analysis for Richly Inflected Languages. *Transactions of the Association for Computational Linguistics (TACL)* 1. 415–428.

Buchholz, Sabine & Erwin Marsi. 2006. CoNLL-X Shared Task on Multilingual Dependency Parsing. In *Proceedings of the tenth conference on computational natural language learning (CoNLL-X 2006)*, 149–164. New York, NY.

Cafferkey, Conor, Deirdre Hogan & Josef van Genabith. 2007. Multi-word units in treebank-based probabilistic parsing and generation. In *Proceedings of the international conference on recent advances in natural language processing (RANLP 2007)*. Borovets, Bulgaria.

Candito, Marie & Matthieu Constant. 2014. Strategies for contiguous multiword expression analysis and dependency parsing. In *Proceedings of the 52nd annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, 743–753. Baltimore, Maryland: Association for Computational Linguistics. http://aclweb.org/anthology/P14-1070.

Candito, Marie & Benoît Crabbé. 2009. Improving generative statistical parsing with semi-supervised word clustering. In *Proc. of the 11th international conference on parsing technologies (IWPT'09)*, 138–141. Paris, France: Association for Computational Linguistics. http://www.aclweb.org/anthology/W09-3821.

Candito, Marie, Joakim Nivre, Pascal Denis & Enrique Henestroza Anguiano. 2010. Benchmarking of Statistical Dependency Parsers for French. In *Proceedings of COLING 2010, 23rd international conference on computational linguistics, posters volume*, 108–116. Beijing, China.

Candito, Marie & Djamé Seddah. 2010. Parsing Word Clusters. In *Proceedings of the NAACL HLT 2010 first workshop on statistical parsing of morphologically-rich languages*, 76–84. Los Angeles, California.

Carroll, John, Guido Minnen & Edward Briscoe. 2003. Parser evaluation: using a grammatical relation annotation scheme. In Anne Abeillé (ed.), *Treebanks: building and using parsed corpora*, 299–316. Dordrecht: Kluwer.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, 132–139. Seattle, Washington.

Collins, Michael. 1996. A new statistical parser based on bigram lexical dependencies. In *Proceedings of the 34th annual meeting of the Association for Computational Linguistics (ACL 1996)*, 184–191. Santa Cruz, California.

Collins, Michael. 1999. *Head-driven statistical models for natural language parsing*. Philadelphia, PA: University of Pennsylvania Ph.D. thesis.

Constant, Matthieu, Marie Candito & Djamé Seddah. 2013. The LIGM-Alpage architecture for the SPMRL 2013 shared task: Multiword expression analysis and dependency parsing. In *Proceedings of the fourth workshop on Statistical Parsing of Morphologically-Rich Languages*, 46–52. Seattle, Washington, USA: Association for Computational Linguistics. http://aclweb.org/anthology/W13-4905.

Constant, Matthieu, Joseph Le Roux & Anthony Sigogne. 2013. Combining compound recognition and PCFG-LA parsing with word lattices and conditional random fields. *ACM Transaction on Speech and Language Processing (TSLP), Special Issue on MWEs* 10(3).

Constant, Matthieu, Joseph Le Roux & Nadi Tomeh. 2016. Deep lexical segmentation and syntactic parsing in the Easy-First dependency framework. In *Proceedings of the 15th annual conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2016)*, 1095–1101. San Diego, California.

Constant, Matthieu & Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, 161–171. Berlin, Germany: Association for Computational Linguistics. http://www.aclweb.org/anthology/P16-1016.

Constant, Matthieu, Anthony Sigogne & Patrick Watrin. 2012. Discriminative strategies to integrate multiword expression recognition and parsing. In *Proceedings of the 50th annual meeting of the Association for Computational Linguistics: Long papers - volume 1* (ACL '12), 204–212. Jeju Island, Korea: Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=2390524.2390554.

Denis, Pascal & Benoît Sagot. 2012. Coupling an Annotated Corpus and a Lexicon for State-of-the-art POS Tagging. *Language Resources and Evaluation* 46(4). 721–736.

Durrett, Greg & Dan Klein. 2015. Neural CRF parsing. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th*

International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 302–312. Beijing.

Dyer, Chris, Miguel Ballesteros, Wang Ling, Austin Matthews & Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proc. of ACL 2015*, 334–343. Beijing.

Dyvik, Helge, Paul Meurer, Victoria Rosén, Koenraad De Smedt, Petter Haugereid, Gyri Smørdal Losnegaard, Gunn Inger Lyse & Martha Thunes. 2016. NorGramBank: A 'Deep' Treebank for Norwegian. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the tenth international conference on language resources and evaluation (LREC 2016)*, 3555–3562. Portorož, Slovenia. http://www.lrec-conf.org/proceedings/lrec2016/summaries/943.html.

Eisner, Jason M. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of the 16th conference on computational linguistics (ACL 1996)*, 340–345. Santa Cruz, California.

Eryiğit, Gülşen, Tugay İlbay & Ozan Arkan Can. 2011. Multiword expressions in statistical dependency parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, 45–55. Dublin, Ireland.

Fazly, Afsaneh, Paul Cook & Suzanne Stevenson. 2009. Unsupervised type and token identification of idiomatic expressions. *Computational Linguistics* 35(1). 61–103.

Głowińska, Katarzyna & Adam Przepiórkowski. 2010. The Design of Syntactic Annotation Levels in the National Corpus of Polish. In *Proc. of the seventh international conference on language resources and evaluation (LREC 2010)*. Valletta, Malta.

Green, Spence, Marie-Catherine de Marneffe, John Bauer & Christopher D. Manning. 2011. Multiword expression identification with tree substitution grammars: A Parsing tour de force with French. In *Proc. of EMNLP 2011*, 725–735. Edinburgh.

Green, Spence, Marie-Catherine de Marneffe & Christopher D. Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics* 39(1). 195–227.

Gross, Maurice. 1984. Lexicon-grammar and the syntactic analysis of French. In *Proceedings of the 10th International Conference on Computational Linguistics and 22nd Annual Meeting of the Association for Computational Linguistics*, 275–282. Stanford, California, USA: Association for Computational Linguistics.

Grover, Claire. 2008. *LT-TTT2: Example pipelines documentation*. Tech. rep. Edinburgh Language Technology Group.

Henderson, James, Paola Merlo, Ivan Titov & Gabriele Musillo. 2013. Multilingual Joint Parsing of Syntactic and Semantic Dependencies with a Latent Variable Model. *Computational Linguistics* 39(4). 949–998.

Kato, Akihiko, Hiroyuki Shindo & Yuji Matsumoto. 2016. Construction of an English Dependency Corpus incorporating Compound Function Words. In *Proceedings of the tenth international conference on language resources and evaluation (LREC 2016)*. Portorož, Slovenia: European Language Resources Association (ELRA).

Koo, Terry, Xavier Carreras & Michael Collins. 2008. Simple Semi-supervised Dependency Parsing. In *Proceedings of ACL-08: HLT*, 595–603. Columbus, Ohio.

Korkontzelos, Ioannis & Suresh Manandhar. 2010. Can recognising multiword expressions improve shallow parsing? In *Proc. of the 11th annual conference of the north american chapter of the association for computational linguistics: human language technologies NAACL/HLT 2010*, 636–644. Los Angeles, California.

Le Roux, Joseph, Antoine Rozenknop & Matthieu Constant. 2014. Syntactic parsing and compound recognition via dual decomposition: Application to French. In *Proc. of COLING 2014*. Dublin, Ireland.

Lehmann, Hans Martin & Gerold Schneider. 2011. A large-scale investigation of verb-attached prepositional phrases. In S. Hoffmann, P. Rayson & G. Leech (eds.), *Studies in variation, contacts and change in english, volume 6: methodological and historical dimensions of corpus linguistics*. Helsinki: Varieng.

McDonald, Ryan, Kevin Lerman & Fernando Pereira. 2006. Multilingual Dependency Analysis with a Two-Stage Discriminative Parser. In *Proceedings of the tenth conference on computational natural language learning (CoNLL-X)*, 216–220. New York City: Association for Computational Linguistics. http://www.aclweb.org/anthology/W/W06/W06-2932.

Miller, George Armitage. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *Psychological Review* 63. 81–97.

Mirroshandel, Seyed Abolghasem & Alexis Nasr. 2016. Integrating selectional constraints and subcategorization frames in a dependency parser. *Computional Linguistics* 42(1). 55–90.

Mirroshandel, Seyed Abolghasem, Alexis Nasr & Joseph Le Roux. 2012. Semi-supervised dependency parsing using lexical affinities. In *Proceedings of the 50th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, 777–785. Jeju Island, Korea: Association for Computational Linguistics. http://aclweb.org/anthology/P12-1082.

Mirroshandel, Seyed Abolghasem, Alexis Nasr & Benoît Sagot. 2013. Enforcing subcategorization constraints in a parser using sub-parses recombining. In *Proceedings of the 2013 conference of the North American chapter of the Association for Computational Linguistics: Human Language Technologies*, 239–247. Atlanta, Georgia: Association for Computational Linguistics.

Nagy T., István & Veronika Vincze. 2014. VPCTagger: Detecting verb-particle constructions with syntax-based methods. In *Proceedings of the EACL 2014 workshop on MWEs*, 17–25. Gothenburg.

Nakagawa, Tetsuji. 2007. Multilingual dependency parsing using global features. In *Proceedings of the CoNLL shared task session of emnlp-conll 2007*, 952–956. Prague, Czech Republic: Association for Computational Linguistics. http://www.aclweb.org/anthology/D/D07/D07-1100.

Nasr, Alexis, Frederic Bechet, Jean-Francois Rey, Benoit Favre & Joseph Le Roux. 2011. MACAON: An NLP tool suite for processing word lattices. In *Proceedings of ACL 2011 Demonstrations*. Portland, Oregon.

Nasr, Alexis, Carlos Ramisch, José Deulofeu & André Valli. 2015. Joint dependency parsing and multiword expression tokenisation. In *53rd annual meeting of the Association for Computational Linguistics*, 1116–1126. Beijing, China.

Nilsson, Jens, Sebastian Riedel & Deniz Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of EMNLP/CoNLL 2007 CoNLL shared tasks session*, 915–932.

Nivre, Joakim. 2014. *Transition-based parsing with multiword expressions*. Athens.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajič, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty & Dan Zeman. 2016. Universal Dependencies v1: A Multilingual Treebank Collection. In *Proceedings of the 10th international conference on language resources and evaluation (LREC 2016)*. Portorož, Slovenia.

Nivre, Joakim, Johan Hall & Jens Nilsson. 2004. Memory-based dependency parsing. In *Proceedings of CoNLL 2004*, 49–56. Boston, Massachusetts.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülsen Eryigit, Sandra Kübler, Svetoslav Marinov & Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering* 13(02). 95–135.

Nivre, Joakim & Jens Nilsson. 2004. Multiword units in syntactic parsing. *Proceedings of Methodologies and Evaluation of Multiword Units in Real-World Applications (MEMURA)*.

Pawley, Andrew & Frances Hodgetts Syder. 1983. Two Puzzles for Linguistic Theory: Native-like selection and native-like fluency. In J. C. Richards & R. W. Schmidt (eds.), *Language and communication*, 191–226. London: Longman.

Prins, Robbert. 2005. *Finite-state pre-processing for natural language analysis*. Behavioral & Cognitive Neurosciences (BCN) research school, University of Groningen dissertation.

Ramshaw, Lance A. & Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of the 3rd acl workshop on very large corpora*, 82–94.

Ronan, Patricia & Gerold Schneider. 2015. Determining light verb constructions in contemporary British and Irish English. *International Journal of Corpus Linguistics* 20(3). 326–354.

Rosén, Victoria, Gyri Smørdal Losnegaard, Koenraad De Smedt, Eduard Bejček, Agata Savary, Adam Przepiórkowski, Petya Osenova & Verginica Barbu Mititelu. 2015. A Survey of Multiword Expressions in Treebanks. In *Proc. of 14th international workshop on treebanks and linguistic theories (TLT 2015)*, 179–193. Warsaw, Poland.

Sag, Ivan, Timothy Baldwin, Francis Bond, Ann Copestake & Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Lecture Notes in Computer Science. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing*, vol. 2276, 1–15. Springer.

Sagot, Benoît & Pierre Boullier. 2005. From raw corpus to word lattices: Robust pre-parsing processing with SxPipe. *Archives of Control Sciences* 15(4). 653–662.

Schneider, Gerold. 2008. *Hybrid long-distance functional dependency parsing*. Institute of Computational Linguistics, University of Zurich Doctoral Thesis.

Schneider, Gerold. 2012. Using semantic resources to improve a syntactic dependency parser. In *Proceedings of semantic relations II workshop (SEM-II) at LREC 2012*, 67–76. Istanbul, Turkey.

Schneider, Gerold. 2014. Improving PP attachment in a hybrid dependency parser using semantic, distributional, and lexical resources. In *Second PARSEME meeting*. Athens, Greece.

Schneider, Nathan, Spencer Onuffer, Nora Kazour, Emily Danchik, Michael T. Mordowanec, Henrietta Conrad & Noah A. Smith. 2014. Comprehensive annotation of multiword expressions in a social web corpus. In *Proceedings of the ninth international conference on language resources and evaluation (LREC 2014)*, 456–461. Reykyavik.

Seddah, Djamé, Reut Tsarfaty, Sandra Kübler, Marie Candito, Jinho Choi, Richárd Farkas, Jennifer Foster, Iakes Goenaga, Koldo Gojenola, Yoav Goldberg, Spence Green, Nizar Habash, Marco Kuhlmann, Wolfgang Maier, Joakim Nivre, Adam Przepiorkowski, Ryan Roth, Wolfgang Seeker, Yannick Versley, Veronika Vincze, Marcin Woliński, Alina Wróblewska & Eric Villemonte de la Clérgerie. 2013. Overview of the SPMRL 2013 Shared Task: A Cross-Framework Evaluation of Parsing Morphologically Rich Languages. In *Proceedings of the fourth international workshop on statistical parsing of morphologically-rich languages (SPRML IV)*. Seattle, WA.

Sekine, Satoshi & Michael Collins. 1997. *EVALB bracket scoring program*. http://www.%20cs.%20nyu.%20edu/cs/projects/proteus/evalb.

Seretan, Violeta. 2011. *Syntax-based collocation extraction*. Vol. 44 (Text, Speech and Language Technology). Dordrecht: Springer.

Sinclair, John. 1991. *Corpus, concordance, collocation*. Oxford: OUP.

Tesnière, Lucien. 1959. *Eléments de syntaxe structurale*. Klincksieck.

Tjong Kim Sang, Erik F. 2002. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *Proceedings of the 6th conference on natural language learning (CoNLL) - volume 20*, 1–4. Taipei, Taiwan.

Tomasello, Michael. 1998. Cognitive linguistics. In W. Bechtel & G. Graham (eds.), *A companion to cognitive science*. Basil Blackwell.

Tu, Yuancheng & Dan Roth. 2011. Learning English light verb constructions: contextual or statistical. In *Proceedings of the acl 2011 workshop on MWEs*, 31–39. Portland, OR.

Vincze, Veronika, István Nagy T. & Gábor Berend. 2011. Multiword expressions and named entities in the Wiki50 corpus. In *Proceedings of the international conference Recent Advances in Natural Language Processing 2011*, 289–295. Hissar, Bulgaria: Association for Computational Linguistics.

Vincze, Veronika, János Zsibrita & István Nagy T. 2013. Dependency parsing for identifying Hungarian light verb constructions. In *Proceedings of the sixth international joint conference on natural language processing (IJCNLP)*, 207–215. Nagoya, Japan: Asian Federation of Natural Language Processing. http://aclweb.org/anthology/I13-1024.

Weeds, Julie, James Dowdall, Gerold Schneider, Bill Keller & David Weir. 2007. Using distributional similarity to organise biomedical terminology. In Fidelia Ibekwe-SanJuan, Anne Condamines & M. Teresa Cabré Castellví (eds.), *Application-driven terminology engineering*. Amsterdam/Philadelphia: Benjamins.

Wray, Alison. 2008. *Formulaic language: Pushing the boundaries*. Oxford University Press.

Yamada, Hiroyasu & Yuji Matsumoto. 2003. Statistical dependency analysis with support vector machines. In *Proceedings of the international workshop on parsing technologies (IWPT)*, vol. 3, 195–206. Nancy, France.

Chapter 2

Investigating the effect of automatic MWE recognition on CCG parsing

Miryam de Lhoneux
Uppsala University

Omri Abend
Hebrew University of Jerusalem

Mark Steedman
University of Edinburgh

The objective of this work is to find out whether or not information about Multiword Expressions (MWEs) can improve parsing with Combinatory Categorial Grammar (CCG). Inspired by studies that have shown the benefit of using information about MWEs for parsing, we transform the representation of some MWEs in CCGbank by collapsing them to one token. In contrast with those studies, we use information about MWEs obtained automatically, in order to find out if automatic MWE recognition can be used to help parsing. We look at two different effects that such an approach can lead to. Training on the transformed data can help improve parsing accuracy. We call this a training effect. Transformed data can help the parser in its decisions. We call this a parsing effect. Our model significantly outperforms the baseline model on the transformed gold standard, which indicates that there is a training effect. Our model performs significantly better on the transformed gold standard when the transformation is done before parsing as opposed to after parsing which indicates that there is a parsing effect. We show that these results can lead to improved performance on the non-transformed standard benchmark although we fail to show that it does so significantly. We conclude that despite the limited settings (our transformation algorithm is only able to deal with MWEs that do not cross constituent boundaries), there are noticeable improvements from using MWEs in parsing. We discuss ways in which the incorporation of MWEs into parsing can be improved and hypothesize that this will lead

Miryam de Lhoneux, Omri Abend & Mark Steedman. 2019. Investigating the effect of automatic MWE recognition on CCG parsing. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 31–60. Berlin: Language Science Press. DOI:10.5281/zenodo.2579045

to more substantial results. We obtain different results with recognisers that detect different types of MWE and therefore emphasise the need to experiment with different recognisers. In this way, we can find out what types of MWEs this method is best suited to.

1 Introduction

1.1 Motivation

Multiword Expressions (henceforth MWE(s)) are increasingly receiving attention in NLP. They represent a wide variety of phenomena with different properties but are generally agreed to be a group of multiple lexemes which have some level of idiomaticity or irregularity (Sag et al. 2002). They represent varied phenomena but, due to this irregularity, they are all generally considered a problem for NLP tasks and they are often a problem for syntactic parsing.

Recent research is showing that information about MWEs can help the syntactic parsing task (Nivre & Nilsson 2004; Korkontzelos & Manandhar 2010) and inversely, information about syntactic analysis helps MWE identification (Green et al. 2013; Weller & Heid 2010; Martens & Vandeghinste 2010). Working on either of the tasks by using information from the other has thus proven to be a useful thing to do and adding MWE information to the syntactic parsing task has proven useful in that it has helped increase parsing accuracy. Work on adding MWE information to syntactic parsing so far has been restricted to certain types of MWEs (multiword nouns, numerical expressions and compound function words in Nivre & Nilsson (2004), compound nominals, proper names and adjective-noun constructions in Korkontzelos & Manandhar (2010)) and hence leaves room for improvement.

Combinatory Categorial Grammar (henceforth CCG) is a strongly lexicalized formalism that is increasingly being used for parsing in NLP applications because of its computational and linguistic properties and because CCG parsers perform relatively well on the Penn Treebank (PTB). To give just a few examples, CCG is used in Machine Translation (e.g. Birch et al. 2007), sentence realization (e.g White 2006), semantic parsing and language acquisition (e.g. Krishnamurthy & Mitchell 2012), open-domain question answering and entailment (e.g. Lewis & Steedman 2013).

For these reasons, CCG parsing is an ideal framework to carry on the work on the interaction between syntactic parsing and MWEs and because CCG is a lexicalized formalism and thus encodes a lot of information in the lexicon, it would be useful to work on it by providing it with information about MWEs.

1.2 Aims

No work so far has tried to use MWE information to improve CCG parsing which is what we intend to do in this work. Different approaches to using MWE information for improving syntactic parsing have been conducted so far with different syntactic models. We conduct one of them which will be argued to be far from ideal but a necessary first step useful to build a sound baseline. The approach we pursue consists in altering training and test data, i.e. transforming the representation of MWEs so that they form one lexical item in them (and hence retokenize the sentence). We experiment with different MWE recognition methods so as to find out if the approach works better with certain types of MWEs than with others.

The two research questions we therefore try to answer are first whether or not we can improve CCG parsing with MWEs and second whether or not applying the same transformation approach to different types of MWEs can lead to different results.

1.3 Overview of the chapter

We give an overview of the background literature to further support our motivations and elaborate on the research questions in Section 2. We then explain and motivate the methodology we propose to use in order to answer the research questions in Section 3. We present our experiments and results in Section 4. We conclude from our study and propose avenues of research in Section 5.

2 Background

2.1 Multiword expressions

MWEs is an umbrella term that has been used to characterize a wide variety of phenomena. The most commonly acknowledged definition of this term since Sag et al. (2002) is that it is a group of multiple lexemes which have some level of idiomaticity or irregularity. The multiple lexemes in a MWE are called MWE units in the remainder of this chapter for convenience. This idiomaticity may be lexico-syntactic such as in the unusual coordination of a preposition and an adjective in 'by and large'. It may be semantic such as in the idiom 'kick the bucket' in which the meaning of the whole is not dividable into the meaning of the parts. It may be pragmatic such as in 'good morning' which has a meaning attached to the situation in which it is said. Finally, it may be statistical such as the collocation 'strong coffee' in which both units occur more frequently than expected.

Different MWE types present different properties. They vary in flexibility: words may appear between the units of a flexible collocation (strong home-made coffee for example) but not between the units of a lexically fixed figurative expression such as 'it's raining cats and dogs'. They also vary in compositionality: 'Strong coffee' is fully compositional whereas 'kick the bucket' is not and 'spill the beans' is semantically decomposable, i.e. the meaning of the whole is not predictable from the meaning of the parts but can be decomposed into its parts: if 'spill' is interpreted as 'reveal' and 'the beans' as 'the secret' (Nunberg et al. 1994). Despite these varied properties they are all generally agreed to be hard to deal with in NLP applications (Sag et al. 2002) and the importance of dealing with them properly has been increasing over the past decade. As described at length in Kim's (2008) thesis, 'dealing with' MWEs consists in developing systems and models for various kinds of tasks. For syntactic analysis, it is important to identify them in text and extract them to a dictionary. For semantic understanding, it is important to measure their compositionality, classify and interpret them.

2.2 Combinatory Categorial Grammar

inline]Subsection missing here Combinatory Categorial Grammar (Steedman 2000) is a strongly lexicalized grammar formalism which is currently gaining popularity in the NLP community.

CCG was built with the intent of being linguistically aware as well as computationally tractable partly as a reaction to transformationalist ideas which were predominant in formal grammars at the time. It differs from the latter mainly in having one component including syntactic and semantic information instead of having separate modules for each in the grammar. Similarly, instead of having a large amount of rules and a lexicon as is the case in traditional grammars, it has a small set of universal rules and a lexicon which encodes most syntactic information. For the sentence 'John buys shares', a traditional grammar has information in the lexicon: that 'John' is an NP, that 'buys' is a verb, that 'shares' is an NP, and in the grammar: that a V and an NP form a VP and that an NP and a VP form a sentence S, as in Figure 1. By contrast, for the same sentence, CCG has information in its lexicon that 'John' is an NP, that 'shares' is an NP and that 'buys' first takes an NP to its right then an NP to its left to form a sentence S, as in Figure 2.

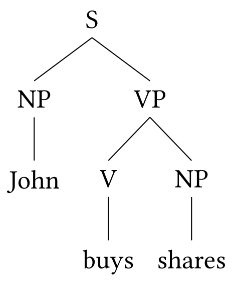
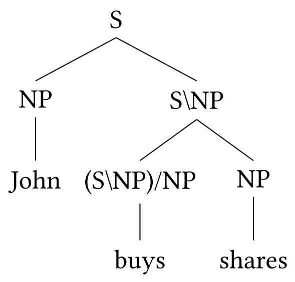


Figure 1: PTB-style tree



Fig[redacted]e 2: CCG tree

Without going into too much detail about how this works, lexical categories work either as functor or as argument and a set of combinatory rules allow them to combine. For example, the category (S\NP)/NP works as a functor that takes an NP to the right (indicated by the forward slash followed by an NP). The category of ‘buys’ therefore can combine with the category of ‘shares’ to result in the category S\NP which in turns takes an NP argument to the left (indicated by the backslash followed by an NP), which it finds in the category of ‘John’ to form a sentence S.

This grammar architecture allows CCG to deal elegantly with long-range dependencies. Instead of adding a level of representation in the form of a trace as in Figure 3, the grammar has universal rules which allow the combination of lexical items, as shown in Figure 4. This has computational advantages and linguistic plausibility: linguistics is increasingly adopting a view of grammar where syntax and the lexicon are two modules that are not completely separate in the grammar but instead interact with each other. It is a tenet of the recently emerging framework of Construction Grammar (Hoffmann & Trousdale 2013). These linguistic and computational properties have made it a widely used framework across NLP research.

2.2.1 CCG parsing

The first efficient statistical model was the generative model built by Hockenmaier & Steedman (2002) and extended to a discriminative model by Clark & Curran (2007). Both use CCGbank (Hockenmaier & Steedman 2007), a CCG converted version of the PTB. Both models perform close to state-of-the-art although with simpler statistical models which is argued by the authors to be the result of having a more expressive grammar than the PCFGs used by state-of-the-art parsers.

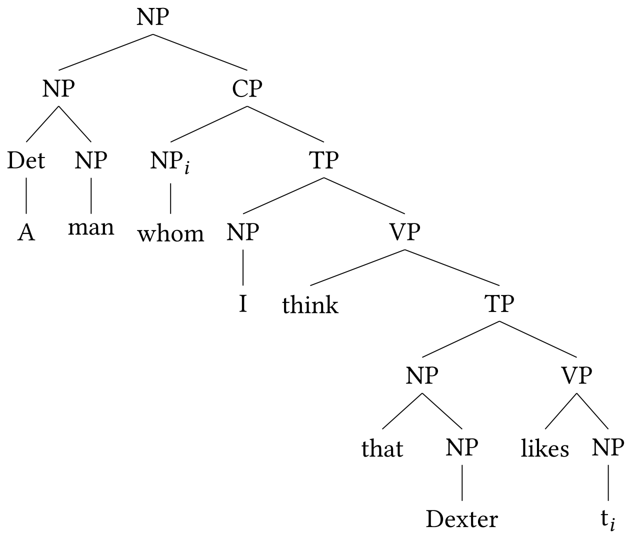


Figure 3: PTB-style tree

2.3 Syntactic parsing and multiword expressions

As mentioned in Section 2.1, the identification of MWEs is important for syntactic analysis. Because they have unusual properties, however, their analysis can be quite problematic. The question of how to deal with MWEs for syntactic parsing has been raised by many researchers. It has been approached in different ways. Researchers working with precision grammars such as HPSG for example have accommodated the lexical entries for MWEs in the lexicon so that MWEs are not a problem for parsing. Researchers on data-induced grammars have accommodated the testing and/or training data before parsing. Recent research has proposed to both change the lexicon and the parsing algorithm. A last recent approach is to learn MWE representations and dependency trees jointly. We briefly describe each of these approaches in turn. We describe the second approach in more details than the other two because this is the one we use in this work for reasons explained in 3.1.

2.3.1 Transforming the lexicon

Different types of lexical entries have been proposed for MWEs in the grammar. A lot of research proposes to simply analyse all MWEs as 'words-with-spaces', i.e. group the MWE units together in the syntactic analysis. This analysis has been argued against by many. Sag et al. (2002) have suggested sophisticated

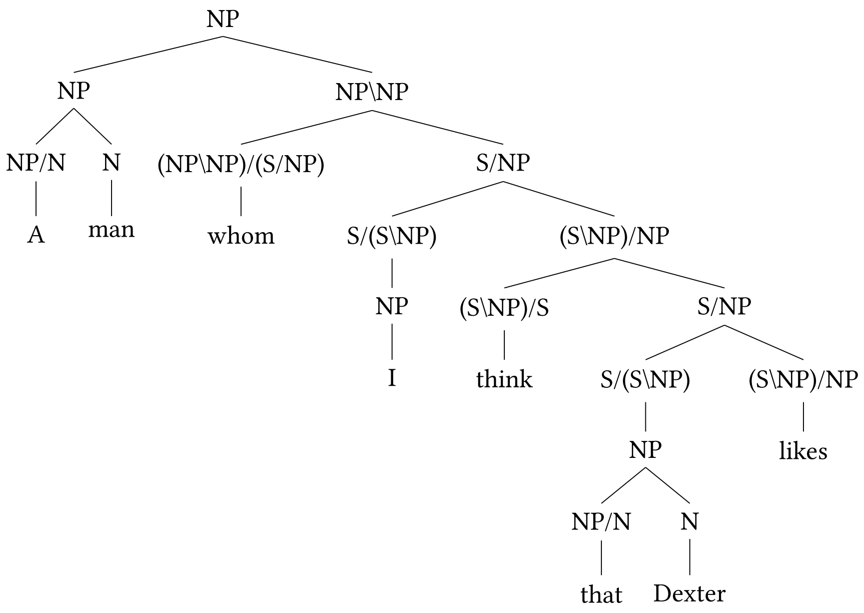


Figure 4: CCG tree

ways of representing the different MWE types in a grammar, which have been partly implemented within the framework of the precision grammar HPSG, as described by Copestake et al. (2002). Zhang et al. (2006) established that MWEs are a tremendous source of parse failures when parsing with a precision grammar such as HPSG and henceforth proposed a way of using this information to identify new MWEs and enrich a lexicon: they suggested using parse failures to predict the existence of a MWE.

2.3.2 Transforming the data

Since the seminal work of Nivre & Nilsson (2004), research has shown that treating MWEs as one token or a 'word-with-spaces' in test and/or in training data before parsing and/or training leads to an improvement in parsing accuracy. Nivre & Nilsson (2004) have shown that to be true for deterministic dependency parsing and Korkontzelos & Manandhar (2010) have shown that to be true for shallow parsing.

The approaches adopted in these two papers are quite different and we describe each in turn.

2.3.2.1 Transforming training and test data

Nivre & Nilsson (2004) created two versions of a treebank, one in which MWEs are annotated as if compositional and one in which they are joined as one lexical item. They show that training a parser on the second version of the treebank leads to a better parsing accuracy. They use a corpus with manual MWE annotation to create both versions of the treebank and hence simulate 'perfect' MWE recognition. MWE annotation, however, only consists in a few MWE types so it is not comprehensive. They report improvement in parsing accuracy of the MWEs themselves but also of their surrounding syntactic structure. They opened the gate for improving syntactic parsing with MWE information but left many questions unanswered. For example, the question of whether or not their results port to other syntactic parsing models, whether or not the full potential they obtained with 'perfect' recognition of MWEs can be obtained with an automatic recogniser and whether or not this potential can be increased when recognising other types of MWEs. Some of these questions have been partially addressed since then. Constant et al. (2012) have shown that with an automatic recogniser, the parsing accuracy improvement is not as dramatic as predicted by Nivre & Nilsson (2004). Eryiğit et al. (2011) found out that in the case of a morphologically rich language (e.g., Turkish), the approach works with some types of MWEs but not with others.

2.3.2.2 Transforming test data

Korkontzelos & Manandhar (2010) reported similar parsing accuracy improvements for shallow parsing, showing that Nivre & Nilsson (2004)'s results do seem to port to at least one other parsing model. Their technique is, however, quite different. They created a corpus containing a large number of pre-selected MWEs (randomly chosen from WordNet) and converted it to a version in which the MWE units are collapsed to one lexical item. They POS-tag the two versions of the corpus before parsing each. They subsequently analyse the differences in output. In order to do so, they randomly select a sample of output from both parsed corpora and build a taxonomy of changes they observe from one to the other. For each class in the taxonomy, they determine whether the change in output led to increased accuracy, decreased accuracy or did not change the accuracy. They automatically classify the rest of the output data and observe an overall increase in accuracy. Their work not only confirms the results obtained from previous work but also provides an insightful qualitative analysis of changes obtained with their method. They believe the improvement in accuracy is partly due to the fact that the parsing model backs off to POS-tags for rare and unseen words. When MWE units are collapsed to one token, that token is not known by the parser but it still gets assigned a sensible POS-tag because the POS-tagger uses contextual information.

2.3.3 Transforming the lexicon and the parsing algorithm

A lot of work has shown that although MWE information improves syntactic parsing, the reverse is also true: syntactic analysis improves MWE identification. Green et al. (2013) successfully tuned a parser for MWE identification, Weller & Heid (2010) and Martens & Vandeghinste (2010) showed that using parsed corpora for MWE identification is beneficial. These findings led Seretan (2013) to propose that neither accommodating the grammar with MWE information, nor recognising MWEs in raw text as a help to parsing are appropriate ways of dealing with the issue of MWEs in syntactic parsing because neither approach takes advantage of the fact that MWE information and syntactic analysis are mutually informative. She proposes instead to have a MWE lexicon and to deal with potential MWEs during parsing.

2.3.4 Joint learning of MWE identification and parsing

Based on the same observation that the tasks of MWE identification and parsing can inform each other, Constant & Nivre (2016) propose to learn both jointly. They use corpora that both have dependency and MWE annotations and modify the parsing algorithm so as to learn both representations jointly. They show that this approach is effective.

2.3.5 Advantages and caveats of the different approaches

All of these researchers have shown the importance of MWEs for syntactic parsing but all of the approaches presented have caveats. Research on HPSG seems to have found the most sophisticated methods of dealing with MWEs but parsing with precision grammars is known to be much less robust (Zhang & Kordoni 2008) than parsing with data-induced grammars which make it a suboptimal solution for practical parsing. Learning MWE representations and syntactic parsing jointly is probably the most promising approach but it requires a lot of manual work since it requires a corpus that is annotated both with MWEs and dependency trees. As far as other solutions are concerned, they are often very much limited by the type of MWEs that have been dealt with. All other solutions presented as a matter of fact concentrate on a few types of MWEs. However, as argued by Kim (2008), because of the different but interrelated properties of MWEs, it is neither appropriate to try and generalise from MWEs and find a single representation which works for all types, nor is it appropriate to deal with each MWE type at a time. An approach for improving syntactic parsing on all MWE types is still lacking and previous approaches leave the question of whether the results can be reproduced with different types of MWEs unanswered.

2.4 Research questions and objectives

In Section 2.3, it was said that MWE identification information improves syntactic parsing although current approaches to doing so leave room for improvement. Trying to improve syntactic parsing with MWE information therefore looks like a promising avenue of research. In Section 2.2, arguments for working with CCG parsing were put forward.

Very little attention has, however, been given to MWEs in CCG parsing. Constable & Curran (2009) modified CCGbank to have a better representation of verb-particle constructions but did not report any parsing accuracy improvement. No work has tried to establish whether CCG parsing accuracy could be improved by adding information about MWEs which is what we intend to do in this work. Our aim is twofold: we want to find out whether or not MWE information can improve CCG parsing and we wish to find out if using methods that have been used for a restricted set of types of MWEs can be extended to different types of MWEs.

3 Methodology

3.1 Approach

As explained in the last section, in this work we concentrate on an approach that consists in transforming the representation of MWEs in treebanks. In Section 2.3.2, two different versions of this kind of approach have been described. In the first, manual annotation is used to create two versions of the treebank. This left questions unanswered, two of which being whether or not the approach can work with automatic recognition and whether or not the approach can work with different types of MWEs. In this work, we conduct the type of approach described by Nivre & Nilsson (2004) using an automatic recogniser to answer the first of these questions in the context of CCG parsing. We also experiment with the recogniser by using different versions of it to answer the second of these two questions. This approach is especially interesting in that, as has been shown in Schneider et al. (2014) who attempted a comprehensive annotation of MWEs in a corpus, even manual annotation of MWEs is a difficult task and experimenting with different MWE recognisers could lead to interesting results.

Our approach therefore involves transforming both training and test data. Transforming the training data can help the parsing model learn more sensible representations of language. For example in the tree for ‘part of speech’, ‘of speech’ is considered a modifier of ‘part’ as in Figure 5 which does not make much sense. Instead, grouping the three lexical items as in Figure 6 gives a better representation of this group of words.

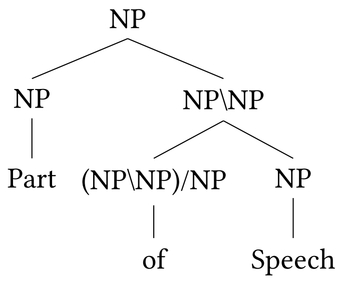


Figure 5: Traditional tree for ‘Part of Speech’

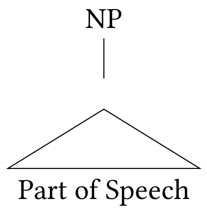


Figure 6: Tree for ‘Part of Speech’ where tokens are grouped

Transforming the test data by for example collapsing the three lexical items ‘part’, ‘of’ and ‘speech’ to one token ‘part+of+speech’ can help the parser make sensible decisions locally by telling it to consider the three words as one. For example, if this token is followed by a coordinator, the parser knows that coordinating one of the units is not a possibility. In the sentence ‘it gives part+of+speech and lemma information’, the parser cannot coordinate ‘speech’ with ‘lemma’ which would be a possibility otherwise. Transforming MWEs in training and test data leads to two different effects of adding MWE information to the syntactic parsing pipeline and it is best if we can differentiate both in the experiments. We call the first type of effect *training effect* and the second *parsing effect*. We repeat the definition of these two effects below.

Training effect: the parser learns more sensible representations of MWEs and its units.

Parsing effect: the parser is helped locally in its decision by considering the MWE units as one unit.

3.2 Parsing model

As mentioned in Section 2.2, both generative and discriminative models exist for parsing with CCG. There are different generative models with different properties. We chose to use StatOpenCCG, developed by Christodoulopoulos (2008) and recently further expanded by Deoskar et al. (2014) because of its ease of use, flexibility and fast training. The expansion of the parser by Deoskar et al. (2014) is particularly well suited to our purposes: it was extended so that it works better on unknown lexical items. Joining lexical items to one token will increase the sparsity of the data and being able to deal with unknown data is therefore a concern for our approach. More particularly, the model proposed by Christodoulopoulos (2008) and Deoskar et al. (2014) is based on one of Hockenmaier (2003)'s models called LexCat which conditions probabilities on lexical categories. Deoskar et al. (2014) make use of this LexCat model instead of the fully lexicalized model which conditions it on words precisely so that the parser is better equipped to deal with unseen lexical items. They introduce a smoothed lexicon to deal with these. They POS-tag the test data in a pre-processing stage and use POS-information to determine the lexical categories of words by using probabilities of lexical categories that appear with each POS-tag of unseen word in the seen data. Because, as mentioned in Section 2.3.2.2, Korkontzelos & Manandhar (2010) have shown that POS-tags assigned automatically to MWEs were useful when parsing, the LexCat model therefore looks ideal for our purposes. We follow Deoskar et al. (2014) in using the C&C tools (Curran et al. 2007) to POS-tag our test data so as to have a model that is comparable with theirs.

3.3 Extending the parsing model with MWE information

As explained in Section 3.1, the objective is first to recognise MWEs in the unlabeled version of CCGbank and then to collapse MWEs to one lexical item in the annotated version of the treebank and in the unlabeled test data. The MWE recognition part is described in Section 3.3.1 and the CCGbank conversion is described in Section 3.3.2.

3.3.1 Recognising MWEs

For MWE recognition, we use a tool developed by Finlayson & Kulkarni (2011). It can be used to build an index of MWEs with information about their probability. It can also be used with a default index which contains all the MWEs and inflections extracted from Wordnet 3.0 and Semcor 1.6 and statistics for each MWE. There are three different tools of interest to us. Simple detectors detect MWEs in text. There is a detector to find proper nouns, one to find all types of MWEs that are in the index, one that finds MWEs that contain only stop words, etc. These simple detectors can also be combined to form a complex detector. There are filters which filter the results of detectors. One for example only accepts MWEs that are continuous, one throws out MWEs which have a score under a certain threshold, one only keeps MWEs under a certain length. The last tool we need is called a 'resolver' and it resolves conflicts when lexical items are assigned to more than one MWE. Conflicts can be resolved in different ways: one resolver picks the leftmost MWE. For example, let us say we have an input sentence that includes 'new york life insurance'. If the MWE index contains 'new york life' and 'life insurance', the resolver will return 'new york life' but will not consider 'insurance' as part of a MWE. Another resolver picks the longest matching MWE. For example, let us say we have an input sentence which contains 'new york stock exchange'. If the MWE index contains 'stock exchange', 'new york' but also 'new york stock exchange', the longest matching resolver will return 'new york stock exchange' as a match.

Let us take the following sentence as an example of input for a resolver:

- Mr. Spoon said the plan is not an attempt to shore up a decline in ad pages in the first nine months of 1989; Newsweek's ad pages totaled 1,620, a drop of 3.2 % from last year, according to Publishers Information Bureau.

The resolver returns a list of its MWEs from left to right. In the case of our example, the output for example looks like this:

- mr._spoon, shore_up, according_to, publishers_information_bureau

The presented protocol can work with any type of MWE recogniser, provided that it is filtered to output only continuous MWEs and resolved so that any word can only appear in one MWE. This library therefore serves our purposes perfectly since it leaves quite a lot of room for experiments. Experiments are described in Section 4.

3.3.2 Transforming the treebank

The algorithm collapses the MWE units to one node when they form a constituent in the tree. The label of the node is the label of that constituent. For example, given the MWE ‘publishers_information_bureau’ and the subtree in Figure 7, the algorithm returns the subtree in Figure 8.

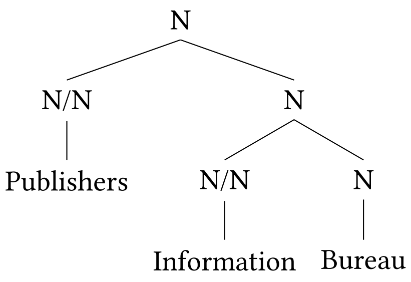
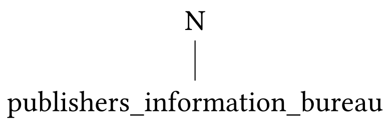


Figure 7: Original subtree



Figur[redacted] 8: Transformed subtree

The algorithm discards MWEs if they do not form a constituent in the tree. An example of tree in which MWE units (e.g., according to) are not siblings in the tree is given in Figure 9. The ideal way in which it should be transformed is given in Figure 10 but attempting to find an algorithm which would work for all non-sibling cases is beyond the scope of this work. We tried our algorithm with a good recogniser, collected statistics and found that 79.5% of the cases (42,309/53,208) were siblings in the tree which we considered a good basis for experimentation. Note, however, that modifying those non-sibling MWEs would make bigger changes to the tree as it would not only remove the lexical categories of MWE units but it would additionally remove the parent category of the MWE units involved and create a new category for the whole MWE. As we will see in Section 4, dealing only with sibling MWEs leads to slight improvements, we hypothesize that an improved algorithm that can deal with non-sibling MWEs can lead to more substantial improvements.

Because, as explained in Section 3.5, we evaluate our method on dependency trees (which can be read off CCG trees), we also need to modify the gold standard dependency trees. Transforming dependency trees involves merging nodes in the graph and changing edges according to the new nodes. When the nodes

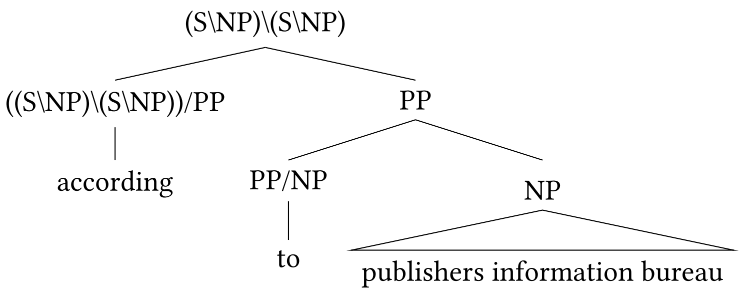


Figure 9: Tree with MWE units that are not siblings

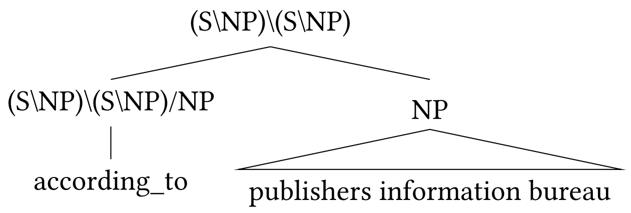


Figure 10: Ideal MWE non-sibling transformation

are merged, edges from the original dependency graph fall into three different categories. Let us take the dependency graph in Figure 11 as an example in which 'Mr. Vinken' is a MWE. There are edges between two units of a MWE such as the one between 'Mr.' and 'Vinken'. We call these *internal edges* for convenience. Our algorithm removes them as shown in Figure 12. There are edges between a MWE unit and another word in the sentence such as the edge between 'Vinken' and 'is'. We call this type of edge a *mediating edge*. In the transformed graph, the whole MWE becomes the node of that incoming or outgoing edge. The edge between 'is' and 'chairman' does not connect any MWE and does not need changing. We call it *external edge*.

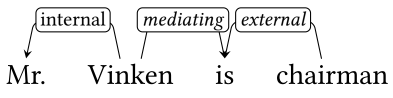


Figure 11: Dependency graph

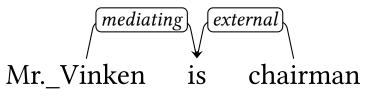


Figure 12: T[redacted]ansformed dependency graph

The downside of this algorithm is that it can create cyclic dependencies between lexical items, i.e. two nodes are connected by two edges going in the opposite direction. We tested the algorithm on the CCGbank and found that in practice this is not a major issue: only 7 cyclic dependencies were created in the ~48,000 sentences.

3.4 Training and parsing

We follow the tradition and use sections 01–22 of CCGbank for training, section 00 for development and section 23 for testing.

Using the same parameters as in Deoskar et al. (2014) to train and parse the test data, we obtain around 87% of correct lexical categories, which is similar to the result they report. This model serves as our baseline and we henceforth call it $model_A$.

For each experiment, we run the MWE recogniser on an unlabeled version of the CCGbank[1]. We then apply our cascaded algorithms described in the previous section to every sentence from the CCGbank. We train the model and parse the test file. We call the transformed treebank $CCGbank_B$ and the model trained on it $model_B$.

3.5 Evaluation

The traditional parsing accuracy metric PARSEVAL has been argued (for example by Clark & Hockenmaier (2002)) to be too harsh on CCG derivation trees because they are always binary, as opposed to PTB-style trees which can have flat constructions with more than one branching node. This binary nature of CCG trees make them prone to having more errors. Consequently evaluation of dependencies has generally been preferred for CCG parsing. As further argued by Clark & Hockenmaier (2002), it also makes sense to use dependencies to evaluate CCG parsing since one of the advantages of CCG over other formalisms is precisely its treatment of long-range dependencies.

In order to evaluate our models, we can thus extract dependencies from the parsed files and compare them with the gold standard. Because we changed the gold standard as compared to $model_A$ (as defined in the previous subsection), however, the results obtained from comparing our parsed files with our gold standard are not directly comparable with the results obtained when applying the same evaluation scheme to $model_A$. Therefore, we cannot directly compare $model_A$ with our $model_B$ s (as defined in the previous subsection) which is essential in answering our research questions. Instead, we have to transform the data of one of the models so as to compare each of the models with the same gold

[1]We created an unlabeled version of CCGbank from the tree leaves to make sure the data is compatible with the trees we work with when transforming trees.

standard. Because we assume that we have created a sensible gold standard with our transformation algorithm, we mainly use gold standard$_B$ for evaluation.

As mentioned in Section 3.1, transforming training and test data can lead to two different effects which can lead to an improved parsing accuracy, i.e. training (the parser learns useful information during training) and parsing effects (the trained parser is helped in its decisions by MWE information) which we would like to differentiate. This can be achieved by conducting different experiments with our existing models. We can assess training effect by testing whether or not $model_B$ can outperform $model_A$ when evaluated on the same gold standard. We can assess parsing effect by testing whether or not $model_A$ can outperform itself when given transformed test data. We discuss these evaluation schemes in Section 3.5.1.

If there is training and/or parsing effect, we can assume that automatic recognition of MWEs can be used to improve syntactic parsing. We can verify this by testing whether or not we can use information from $model_B$ to outperform $model_A$ on gold standard$_A$. We use a second evaluation scheme where we combine information from output from $model_A$ and output from $model_B$ (henceforth called *model combination*) to test this which we discuss in Section 3.5.2.

As mentioned in Section 2.4, we not only want to know whether or not information about MWEs can help CCG parsing but we are also interested in finding out whether or not different types of MWEs impact parsing accuracy in different ways. As will be explained in Section 4, we created different versions of CCGbank$_B$ and different versions of $model_B$ with these. Because we created different gold standard for each of these models, they cannot directly be compared. Instead, comparing how different $model_B$ s can improve $model_A$ is possible by comparing their combination with it against gold standard$_A$. Again then we can use model combination and combine information from output from $model_A$ with information from output from $model_B$. We compare this combined model output against gold standard$_A$ and compare the results when combining $model_A$ with different versions of $model_B$. We discuss how this can be achieved in Section 3.5.3.

3.5.1 Assessing training and parsing effects

Modifying the output from $model_A$ so that it is comparable with the gold standard from $model_B$ is straightforward: we just need to apply the transformation algorithms to the output from $model_A$ with the MWEs found in the test data. We can also test $model_A$ on data transformed before parsing.

3.5.1.1 Parsing effect

Testing whether there is a parsing effect can be done by testing whether or not $model_A$ can perform better on test data transformed before parsing than on test data transformed after parsing. We conduct this evaluation. There is a caveat in this evaluation, however: we are using information from the gold standard in the test data, i.e. we know which MWEs are siblings in the test data. This introduces an artefact which makes the results somewhat difficult to interpret: the transforming before parsing method has sibling information which the transforming after parsing method does not. There can be parsing and sibling effects and the two cannot be decoupled. A way to circumvent this problem is to transform MWEs regardless of their sibling status (i.e. treat all detected MWEs as if they were siblings) and compare the model when we transform before parsing with the model when we transform after parsing. Transforming all MWEs in unlabeled test data is straightforward. As mentioned in Section 3.3.2, we do not have an algorithm to transform MWEs that are not siblings in trees so we cannot transform the output parse trees. However, since we are working only with dependencies for evaluation, it is possible to transform all MWEs in all the dependencies of the sentence. The problem with this evaluation is that the output cannot perform well on gold standard$_B$ because it is not tokenized in the same way and we treat dependencies wrongly tokenized as errors. However, both transforming before and transforming after parsing suffer from the same problem and the comparison between the two is fair.

3.5.1.2 Training effect

Testing whether there is a training effect consists in comparing the results of $model_A$ on transformed data with the results of $model_B$ on transformed data. In this evaluation, the caveat that we are using information from the gold standard in the test data can also be considered problematic because $model_B$ is trained on data with information about siblings. This information is unseen by $model_A$. We therefore test both models on data where only siblings are transformed (called *gold test* for convenience) and on data where all MWEs are transformed (called *fully transformed test* for convenience). Again, the problem with this evaluation is that the output cannot perform well on gold standard$_B$ because it is not tokenized in the same way. Again, however, both models suffer from the same problem and the comparison between the two is fair.

3.5.2 Verifying whether or not automatic recognition of MWEs can improve CCG parsing on the original gold standard

Results which will be discussed in Section 4 seem to indicate that there is both a training and a parsing effect and that $model_B$ performs better than $model_A$ on some dependencies. Our findings support the claim that automatic recognition of MWEs can improve CCG parsing. These results, however, led us to want to verify whether or not $model_B$ can improve the score on the standard evaluation benchmark, i.e. on gold standard A. This involves 'detransforming' the output from $model_B$ and splitting MWEs back into their units. However, by transforming the data, we have lost information about some dependencies in the sentence. We have no *internal edges* (edges between MWE units of the same MWE) and when there is a *mediating edge* (edges between MWE units of any MWE and other words in the sentence) we do not know which MWE unit of the MWE should the incoming or outgoing node of that edge. In Figure 12 reproduced in Figure 13 for convenience, we do not know whether the label between 'is' and 'mr._vinken' should come from 'mr' or 'vinken'. For this reason, we propose to combine information obtained from parsing the test data with our transformed model $model_B$ with information obtained from parsing the test data with the original model $model_A$. We therefore take some dependencies from output_A and some from output_B.

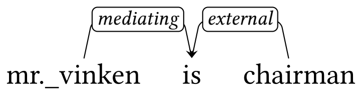


Figure 13: Transformed dependency graph

External edges can be taken from output_B. *Internal edges* do not exist in output_B. Hence we propose to take them from output_A. For *mediating edges*, there are different possibilities. We can take them from output_A and therefore only test whether or not $model_B$ performs better than $model_A$ on *external edges*. We call this combination method *medFromA*. If we want to test $model_B$ on *mediating edges*, we can take *mediating edges* from output_B. For this to work, the model combining algorithm needs to choose one node as the incoming or outgoing node of that edge: in our example, it should either be 'Mr.' or 'Vinken'. We use two additional combination methods. We use one in which the rightmost node is chosen as incoming or outgoing node for *mediating edges* from output_B which we call the *rightmostMed* scheme. We also use one in which the leftmost node is chosen which we call the *leftmostMed* scheme. In order for the model combining algorithm to work, we need to recover information about MWEs and their units and hence to know for each dependency if we are dealing with an *internal*, *external* or *mediating edge*. This can easily be done because MWE and their units are annotated in the unlabeled data[2].

When these models are combined in these three different ways, we have a new combined model that we can compare with $model_A$ on gold standard$_A$. In this case, using 'gold test' data is again problematic. As a matter of fact, if we use output$_B$ as obtained after parsing 'gold test' data, we are using information obtained during the conversion of the gold standard and we are using a parsing pipeline which is not fully automatic. In order to make sure that we can outperform $model_A$ in a fully automatic way, we can use parses of $model_B$ tested on the 'fully transformed test' data set as described in Section 3.5.1, and combine them in the same three ways as described above.

3.5.3 Testing whether or not different MWE types impact the results differently

As mentioned before, we use different MWE recognisers to create different versions of CCGbank$_B$ and hence different versions of $model_B$. Because we created a different gold standard for each, results from different models are not directly comparable. We can, however, convert output parses using the model combination algorithm described in Section 3.5.1 and test each model against gold standard$_A$. In this way, different versions of $model_B$ can be compared.

3.6 Summary of the experimental setup

Figures 14 and 15 summarize our experimental setup.

In Section 2, we motivated our study and identified two research questions: whether or not information about MWEs can improve CCG parsing and whether or not different types of MWEs can influence parsing accuracy in different ways. In this Section, we proposed a methodology for testing this. We refined the first research questions: what we want to find out is whether or not automatic recognition of MWEs can improve CCG parsing. Additionally, we separated it into two further research questions: whether we can observe a parsing effect (the parser is helped in its decisions by transformed data) and/or whether we can observe a training effect (the parser learns something useful). We proposed to use different MWE recognisers to answer the second question. When defining an algorithm for transforming the treebank, however, we could not find a straightforward algorithm to transform MWEs that are not siblings in the tree and decided to settle

[2]Our MWE recogniser joins MWE units of a MWE by a '+' symbol.

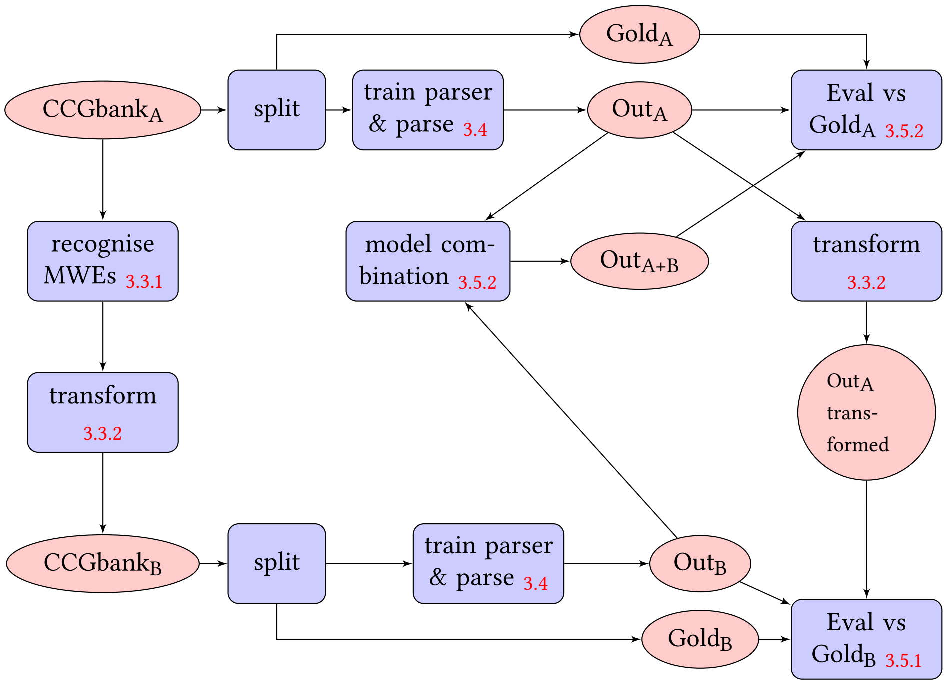


Figure 14: Pipeline of an experiment on one version of one application of MWE recognition to the parsing pipeline with all the evaluation schemes that can be applied to it. The transforming before parsing of *model*$_A$ (see Section 3.5.1) is omitted for clarity and given in Figure 15.

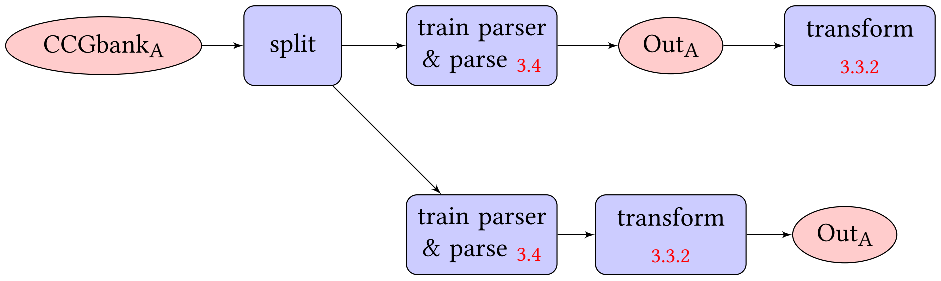


Figure 15: Pipeline of the 'transforming before parsing' against the 'transforming after parsing experiment.'

for an algorithm that only transforms siblings. This led to further complications in the evaluation schemes because it makes it harder to give a fair evaluation of our models. We found ways to circumvent the problems: we proposed different evaluation schemes together with cross-validations. We now turn to the results of our experiments.

4 Results

In this section, we look at each of the research questions in turn. We first describe the different MWE recognisers used for our experiments.

Significance tests. We use a one-tailed randomized shuffling test with 10,000 iterations to assess statistical significance of our best results. We use the software created by Padó (2006) (slightly modified in order to make it a one-tailed test instead of a two-tailed one) for our tests.

4.1 MWE recognition

We use the jMWE library described in Section 3.3.1 with the default index which contains MWEs from Wordnet 3.0 and Semcor 1.6. We use the library's three different tools which were explained in that section. Those tools are detectors which detect MWEs in text, filters which filter through the results of one or more detectors and resolvers which resolve conflicts between MWEs when one word is assigned to more than one MWEs by the detector.

We use the following tools:

- Detectors:
  - Proper Nouns: detects proper nouns, like 'wall street'.
  - Stop words: detects MWEs that only contain stop words, like 'instead of'.
  - Exhaustive: finds all MWEs that are in the index.
- Filters:
  - MoreFreqAsMWE: only keeps MWEs if its units appear more often together than apart in the corpora in which they were collected.
  - ConstrainLength: only keeps MWEs that have 2 units.
- Resolvers:
  - Longest: always picks the longest matching MWEs.
  - Leftmost: picks the MWE that starts earliest in the sentence.

We build 5 different MWE recognisers with different combinations of these tools. This means that the study is by no means exhaustive. Information about our recognisers and statistics about the MWEs they detect are summarised in Table 1. The numbers in column 'ID' denote the recognisers used in the remainder of this section. Similarly, each $model_B$ is denoted by the recogniser which was used to train it as indicated by this number.

Table 1: Description (detector, filter and resolver) of MWE recognisers used and statistics of MWEs collected with them in the treebank

ID	detector	filter	resolver	MWE #	Sibling #	Sibling %
1	Exhaustive	MoreFreqAsMWE	Longest	53,208	42,309	79.51
2	Exhaustive	MoreFreqAsMWE	Leftmost	51,543	21,532	41.85
3	Proper Nouns	no filter	Longest	32,583	28,068	86.14
4	Exhaustive	ConstrainLength	Leftmost	49,587	19,984	40.30
5	Stop words	no filter	Longest	13,623	286	2.09

4.2 Can we improve CCG Parsing accuracy with automatic MWE recognition?

As explained in Section 3.5, we use different evaluation schemes to answer this question. First we evaluate $model_B$ and $model_A$ against gold standard$_B$ and determine whether there is training and/or parsing effects. Then we verify whether we can use $model_B$ to improve over $model_A$ on gold standard$_A$ by using model combination with $model_A$ and $model_B$. We deal with each of these in turn. We test all evaluation schemes on all of our versions of $model_B$. Results fluctuate according to the recognisers as discussed in Section 4.3. We give general remarks about results and report our best results in this Section.

4.2.1 Can representing MWEs as one token introduce a training effect?

In order to find out whether or not training data on an MWE-informed corpus can lead to an improved accuracy, i.e. leads to training effect, we compare the output of $model_B$ against the output of $model_A$ tested on the 'gold test' data. 3 out of our 5 $model_B$ s outperform $model_A$ on unlabeled$_B$, although generally by a slight margin. The best results are obtained by model$_{B_3}$ and are given in Table 2. Model$_B$ significantly outperforms $model_A$ by 0.24% (p = 0.006) which supports the hypothesis that there is indeed a training effect.

Because using gold test data gives $model_B$ an unfair advantage, we also test these models on the 'fully transformed test' data. In this case 3 of our 5 $model_B$

Table 2: Precision (P), recall (R), and F_1-measure of unlabelled dependencies against gold standard B with recogniser 3

model	test data	P	R	F_1
A	gold test	**84.53**	84.76	84.64
B3	gold test	84.48	**85.28**	**84.88**

s outperform *model*$_A$ again although by an even slighter margin. The biggest difference in results is obtained with model$_{B_1}$ and results are given in Table 3. Although the margin is smaller, *model*$_B$ still significantly outperforms *model*$_A$ by 0.15% (p=.047) which shows that there is a training effect.

Table 3: Precision (P), recall (R), and F_1-measure of unlabelled dependencies against gold standard B with recogniser 1

model	test data	P	R	F_1
A	fully transformed test	**73.15**	72.38	72.77
B1	fully transformed test	73.08	**72.74**	**72.92**

4.2.2 Can representing MWEs as one token introduce a parsing effect?

In order to test whether there can be a parsing effect, we compare the output of *model*$_A$ when data are transformed before parsing with *model*$_A$ when data are transformed after parsing. In this case *model*$_B$ always outperforms *model*$_A$. Our best results are shown in Table 4 in which *model*$_B$ highly significantly outperforms *model*$_A$ ($p < 0.0001$).

Table 4: Precision (P), recall (R), and F_1-measure of unlabelled dependencies against gold standard B with recogniser 1 when transforming before parsing uses gold sibling information and only siblings are transformed after parsing

model	transformed	P	R	F_1
A	before parsing	**83.88**	**84.24**	**84.06**
A	after parsing	78.92	79.41	79.17

The problem with these results is that ‘transforming before parsing’ method has gold standard information about siblings which the ‘transforming after parsing’ method does not. In order to cross-validate our result, we transform all MWEs both before and after parsing and compare the results. In this case, $model_A$ when data are transformed before parsing outperforms $model_A$ when data are transformed after parsing only in one of the 5 cases which undermines a little the previous argument about the parsing effects showing that there can also be undesirable effects to transforming test data. It could, however, be partly due to the fact that we transformed non-siblings, which may have triggered errors during parsing. In any case, our best results still show a significant improvement with the ‘transforming before parsing’ method over the ‘transforming after parsing’ method. These results are obtained when using recogniser_3 and are given in Table 5. Model_A transformed before parsing significantly outperforms $model_A$ transformed after parsing by 0.20% (p = 0.008). This indicates that there can be a parsing effect.

Table 5: Precision (P), recall (R), and F_1-measure of unlabelled dependencies against gold standard B with recogniser 3 when all MWEs are considered siblings

model	transformed	P	R	F_1
A	before parsing	**79.83**	79.54	**79.69**
A	after parsing	79.38	**79.60**	79.49

4.2.3 Can we improve the parsing model on the original gold standard?

We now verify if we can also outperform the baseline on the untransformed gold standard. This means testing whether or not $model_B$ improves over $model_A$ on *external edges* and/or on *mediating edges*. We test this by combining dependency edges obtained from $model_A$ and $model_B$. We combine these edges with 3 different methods. *Internal* edges are always taken from $model_A$ and *external edges* are always taken from $model_B$. *Mediating edges* are taken from A in the *medFromA* evaluation and from B in the 2 other cases. In the *rightmostMed* evaluation, the rightmost MWE unit is always chosen as incoming or outgoing node and in the *leftmostMed* evaluation, it is the leftmost MWE unit that is always taken as incoming or outgoing node. Our best results are given in 6 in which $model_B$ only outperforms $model_A$ in the *medFromA* case by 0.13% which is not significant (p > 0.05). This seems to show that $model_B$ may perform better than *model$_A$* on *external edges* but as far as *mediating edges* are concerned, the picture is unclear. If we take the *mediating edge* from B, it seems clearly better to choose the rightmost MWE unit as incoming or outgoing node (which is not surprising since compound nouns are almost always right-headed) but doing so does not seem to be a big help in parsing accuracy. Model$_B$ might perform better than *model$_A$* on *mediating edges* if we had a better mechanism to recover the head word but with our simple method we cannot say whether or not this is the case.

Table 6: Precision (P), recall (R), and F_1-measure of unlabelled dependencies against gold standard A using recogniser 3

model	combination type	P	R	F_1
A		**85.27**	85.02	85.15
A+B3	medFromA	84.89	**85.68**	**85.28**
A+B3	rightmostMed	84.84	85.46	85.15
A+B3	leftmostMed	81.43	82.02	81.72

In this result, *model$_B$* is again helped in the parsing decisions by being told which MWEs are siblings. In order to test whether we can improve on *model$_A$* in a fully automatic manner, we test *model$_B$* on the 'fully transformed test' data which is a version of the test data obtained automatically, i.e. by transforming all MWEs in the text instead of only the siblings. All MWEs are then parsed as a unit. When we combine the models, we have more MWEs than we should have and consequently, more edges are considered to be *mediating* and *internal* edges and less edges are considered to be *external* edges. Hence, we are led to choose edges from *model$_A$* where *model$_A$* is not expected to perform better than *model$_B$*. When combining both models with the *medFromA* method, however, we still outperform *model$_A$* by 0.04% when using recogniser$_3$ showing that *model$_B$* may have learnt something useful although there is no significant evidence for it at this point.

4.3 Does using different MWE recognisers impact parsing accuracy differently?

As explained in Section 3.5.3, the last experiment we conduct is to test our model using different recognisers, combine the output using the model combination algorithm explained in Section 3.5.1 and compare it to gold standard$_A$. This provides a way to compare different versions of our *model$_B$*.

As can be seen in Table 7, different MWE recognition methods seem to make a difference in results. There is a significant difference between our best model (based on recogniser$_3$) and our worst model (based on recogniser$_2$) of .26 (p=.01). Some recognisers lead to decreases in parsing accuracy while others lead to increases. It appears from the table that using a leftmost resolver (a resolver that always chooses the leftmost MWE when there is a conflict) has a bad impact on parsing accuracy. Looking at the different models, it is interesting to note that there is a much lower percentage of MWEs that are siblings in the tree and hence a much lower amount of changes made in the treebank. It is interesting to note that the best model is based on a detector that only detects proper nouns. This seems to show that they are the best candidates for being treated as words-with-spaces. This is not surprising because they are not flexible and never get inflected. For other types of MWE, an analysis as word-with-spaces might not be the most appropriate, as argued by many researchers (Sag et al. 2002) to give just one example, see Section 2).

Table 7: F_1-measure of unlabelled dependencies against gold standard A using different recognisers from the $model_A$ combining method

model	detector type	resolver type	F_1
A			85.15
B1	exhaustive	longest	85.18
B2	exhaustive	leftmost	85.02
B3	Proper Nouns	longest	**85.28**
B4	Length 2	leftmost	85.07
B5	Stop words	longest	85.19

4.4 Summary of our findings

We summarise our findings in Table 8.

Table 2 and 3 are respectively upper and lower bounds on the training effect that can be obtained with our method with these recognisers. Similarly, Table 4 and 5 are respectively upper and lower bounds on the parsing effect that can be obtained. Given that the lower bounds are still significantly above the baselines in both cases, we can conclude that there can be both a training and a parsing effect, and that we can improve CCG parsing with information about MWEs.

Table 8: Summary of our findings

question	answer	tables concerned
Can there be a parsing effect?	yes	Table 2 and 3
Can there be a training effect?	yes	Table 4 and 5
Can we improve parsing on the untransformed gold standard?	not significantly	Table 6
Do different types of MWEs impact the results differently?	yes	Table 7

5 Conclusion

5.1 Contributions

Our main contributions in this work are:

- Improvements on CCG parsing with automatic MWE recognition
- Significant results despite limited settings
- An algorithm to automatically transform MWEs in a treebank
- Techniques for distinguishing training from parsing effects
- Empirical support that there is both training and parsing effects
- Interesting differences in results when using different recognisers

The task we have been trying to improve in this work is the task of syntactic parsing. Adding MWE information to CCG parsing was singled out as a useful direction because it has proven useful in the past with other parsing frameworks and because it seemed an interesting approach to attempt within the framework of a lexicalized grammar. We built on previous work which had shown the benefits of giving information about MWEs to a syntactic parser. It had been shown to work for deterministic dependency parsing, shallow parsing and deterministic constituency parsing but not for statistical constituency parsing. We implemented an existing pipeline which consists in transforming the representation of MWEs in training and test data by collapsing its units to one token and adapted it to our purposes. We gave further evidence supporting these studies and showed that statistical constituency parsing with a lexicalized grammar too can benefit from MWE information. Our study provided further empirical support to the hypothesis that MWE information can improve syntactic parsing by showing that we can improve CCG parsing with information about MWEs.

MWE identification was also identified as a notoriously difficult task although important for many applications because MWEs violate usual compositional rules and can be the source of many errors if not handled properly. We have shown that using an existing automatic recogniser as a source of MWE information was useful which had so far been left a bit unclear in the literature.

Our results have shown small but significant improvements over previous models which is very encouraging given the restricted settings we have worked with. We have as a matter of fact hypothesized that the results were very much limited by the methodology used and have suggested ways of improving the current approach. Our biggest contributions, however, are not in the results we obtained but in the techniques we proposed. The study has proposed novel techniques to improve on previous pipelines. We have proposed an algorithm to automatically transform MWEs in a treebank which can be used with other formalisms although this algorithm is limited to transforming MWEs which form a constituent in the tree. More importantly, we have proposed ways of experimenting with our models in a way that we can distinguish parsing (the parser is helped in its decisions by transformed data) from training effects (the parser learns something useful) in the evaluation and have shown evidence for both. In addition, we have proposed to experiment with different MWE recognisers and study the impact of different MWE recognition methods on parsing accuracy. This is especially interesting in that it is never quite clear in the literature what counts as an MWE. Experimenting with recognisers that detect different types of MWEs can help find out what types of MWEs this method is most suitable for. Our results in this work have shown that collapsing MWE units to one token is most useful for MWEs that are made of proper nouns. It makes intuitive sense that treating them as words with spaces is appropriate since they are not flexible and do not get inflected.

5.2 Future work

We propose the following for future work:

- Extending the transformation algorithm to the non-sibling case
- Testing more MWE recognition methods
- Conducting error analysis

A lot more interesting research can still be done on the interaction between MWE identification and syntactic parsing. Theoretical research has emphasized the need to give different syntactic representations for different types of MWEs but a lot of empirical work is still needed if we want to automatically assign sensible syntactic representations to MWEs. Extending our transformation algorithm to the non-sibling case would allow conducting more extensive experiments. We also believe that testing more recognition methods could lead to interesting discussions where we could find out more about what type of MWE is dealt best with by what method. This could also help discover interesting properties of MWEs. Conducting error analysis could also lead to further insight into why the method is sometimes successful, sometimes less successful.

In the meantime, we believe to have offered new perspectives in the study of the integration between syntactic parsing and MWE identification especially in relation to CCG parsing. We have given encouraging results on a difficult task and suggested ways of improving them. We have given further evidence that the integration of MWE identification with syntactic parsing is a promising and exciting research direction.

References

Birch, Alexandra, Miles Osborne & Philipp Koehn. 2007. CCG supertags in factored statistical machine translation. In *Proceedings of the second workshop on statistical machine translation*, 9–16.

Christodoulopoulos, Christos. 2008. *Creating a natural logic inference system with Combinatory Categorial Grammar*. University of Edinburgh MA thesis.

Clark, Stephen & James R Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics* 4(33). 493–552.

Clark, Stephen & Julia Hockenmaier. 2002. Evaluating a wide-coverage CCG parser. In *Proceedings of the LREC 2002 beyond PARSEVAL workshop*, 60–66.

Constable, James & James Curran. 2009. Integrating verb-particle constructions into CCG parsing. In *Proceedings of the Australasian language technology association workshop 2009*, 114–118. Sydney, Australia. http://aclweb.org/anthology/U09-1017.

Constant, Matthieu & Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, 161–171. Berlin, Germany: Association for Computational Linguistics. http://www.aclweb.org/anthology/P16-1016.

Constant, Matthieu, Anthony Sigogne & Patrick Watrin. 2012. Discriminative strategies to integrate multiword expression recognition and parsing. In *Proceedings of the 50th annual meeting of the Association for Computational Linguistics: Long papers - volume 1* (ACL '12), 204–212. Jeju Island, Korea: Association for Computational Linguistics. http://dl.acm.org/citation.cfm?id=2390524.2390554.

Copestake, Ann, Fabre Lambeau, Aline Villavicencio, Francis Bond, Timothy Baldwin, Ivan A. Sag & Dan Flickinger. 2002. Multiword expressions: linguistic precision and reusability. In *Proceedings of the third international conference on language resources and evaluation (LREC 2002)*, 1941–1947. Las Palmas, Canary Islands, Spain.

Curran, James, Stephen Clark & Johan Bos. 2007. Linguistically motivated large-scale NLP with C&C and boxer. In *Proceedings of the 45th annual meeting of the Association for Computational Linguistics, companion volume, proceedings of the demo and poster sessions*, 33–36. Prague, Czech Republic: Association for Computational Linguistics.

Deoskar, Tejaswini, Christos Christodoulopoulos, Alexandra Birch & Mark Steedman. 2014. Generalizing a strongly lexicalized parser using unlabeled data. In Gosse Bouma & Yannick Parmentier (eds.), *Proceedings of the 14th conference of the European chapter of the Association for Computational Linguistics (EACL'2014)*, 126–134. Gothenburg.

Eryiğit, Gülşen, Tugay İlbay & Ozan Arkan Can. 2011. Multiword expressions in statistical dependency parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages*, 45–55. Dublin, Ireland.

Finlayson, Mark Alan & Nidhi Kulkarni. 2011. Detecting multi-word expressions improves word sense disambiguation. In *Proceedings of the workshop on multiword expressions: from parsing and generation to the real world* (MWE '11), 20–24. Stroudsburg, PA, USA: Association for Computational Linguistics.

Green, Spence, Marie-Catherine de Marneffe & Christopher D. Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics* 39(1). 195–227.

Hockenmaier, Julia & Mark Steedman. 2002. Generative models for statistical parsing with Combinatory Categorial Grammar. In *Proceedings of the 40th annual meeting on Association for Computational Linguistics* (ACL '02), 335–342. Stroudsburg, PA, USA: Association for Computational Linguistics.

Hockenmaier, Julia & Mark Steedman. 2007. CCGbank: A corpus of CCG derivations and dependency structures extracted from the Penn Treebank. *Computational Linguistics* 33(3). 355–396.

Hockenmaier, Julie. 2003. *Data and models for statistical parsing with Combinatory Categorial Grammar*. The University of Edinburgh dissertation.

Hoffmann, Thomas & Graeme Trousdale. 2013. The handbook of Construction Grammar. In Thomas Hoffmann & Graeme Trousdale (eds.), *The oxford handbook of construction grammar* (Oxford Handbooks in Linguistics), chap. Construction Grammar: Introduction, 1–14. Oxford University Press.

Kim, Su Nam. 2008. *Statistical Modeling of Multiword Expressions*. University of Melbourne dissertation.

Korkontzelos, Ioannis & Suresh Manandhar. 2010. Can recognising multiword expressions improve shallow parsing? In *Proc. of the 11th annual conference of the north american chapter of the association for computational linguistics: human language technologies NAACL/HLT 2010*, 636–644. Los Angeles, California.

Krishnamurthy, Jayant & Tom M Mitchell. 2012. Weakly Supervised Training of Semantic Parsers. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP/CoNLL)*, 754–765.

Lewis, Mike & Mark Steedman. 2013. Combining distributional and logical semantics. *Transactions of the Association for Computational Linguistics* 1. 179–192.

Martens, Scott & Vincent Vandeghinste. 2010. An efficient, generic approach to extracting multi-word expressions from dependency trees. In *Proceedings of the Workshop on Multiword Expressions: from Theory to Applications (MWE 2010)*, 84–87. Beijing, China: Association for Computational Linguistics.

Nivre, Joakim & Jens Nilsson. 2004. Multiword units in syntactic parsing. In *Workshop on Methodologies and Evaluation of Multiword Units in Real-World Applications*, 39–46.

Nunberg, Geoffrey, Ivan A. Sag & Thomas Wasow. 1994. Idioms. *Language* 70(3). 491–538.

Padó, Sebastian. 2006. *User's guide to* `sigf`*: Significance testing by approximate randomisation.*

Sag, Ivan, Timothy Baldwin, Francis Bond, Ann Copestake & Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Lecture Notes in Computer Science. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing*, vol. 2276, 1–15. Springer.

Schneider, Nathan, Spencer Onuffer, Nora Kazour, Emily Danchik, Michael T. Mordowanec, Henrietta Conrad & Noah A. Smith. 2014. Comprehensive annotation of multiword expressions in a social web corpus. In *Proceedings of the ninth international conference on language resources and evaluation (LREC 2014)*, 456–461. Reykyavik.

Seretan, Violeta. 2013. On collocations and their interaction with parsing and translation. *Informatics* 1(1). 11–31.

Steedman, Mark. 2000. *The syntactic process*. Cambridge, MA, USA: MIT Press.

Weller, Marion & Ulrich Heid. 2010. Extraction of German multiword expressions from parsed corpora using context features. In Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike

Rosner & Daniel Tapias (eds.), *Proceedings of lrec 2010*, 3195–3201. Valletta: European Language Resources Association.

White, Michael. 2006. Efficient realization of coordinate structures in Combinatory Categorial Grammar. *Research on Language and Computation* 4(1). 39–75.

Zhang, Yi & Valia Kordoni. 2008. Robust parsing with a large HPSG grammar. In *Proceedings of the sixth international language resources and evaluation conference (LREC'08)*, 1888–1893. Marrakech, Morroco.

Zhang, Yi, Valia Kordoni, Aline Villavicencio & Marco Idiart. 2006. Automated multiword expression prediction for grammar engineering. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties* (MWE '06), 36–44. Stroudsburg, PA, USA: Association for Computational Linguistics.

Chapter 3

Multilingual parsing and MWE detection

Vasiliki Foufi
University of Geneva

Luka Nerima
University of Geneva

Eric Wehrli
University of Geneva

Identifying multiword expressions (MWEs) in a sentence in order to ensure their proper processing in subsequent applications, like machine translation, and performing the syntactic analysis of the sentence are interrelated processes. In our approach, priority is given to parsing alternatives involving collocations, and hence collocational information helps the parser through the maze of alternatives, with the aim to lead to substantial improvements in the performance of both tasks (collocation identification and parsing), and in that of a subsequent task (machine translation).

1 Introduction

Multiword expressions (MWEs) are lexical units consisting of more than one word (in the intuitive sense of 'word'). There are several types of MWEs, including idioms (*a frog in the throat*, *break a leg*), fixed phrases (*per se*, *by and large*, *rock'n roll*), noun compounds (*traffic lights*, *cable car*), phrasal verbs (*look up*, *take off*), etc. While easily mastered by native speakers, their detection and/or their interpretation pose a major challenge for computational systems, due in part to their flexible and heterogeneous nature.

Vasiliki Foufi, Luka Nerima & Eric Wehrli. 2019. Multilingual parsing and MWE detection. In Yannick Parmentier & Jakub Waszczuk (eds.), *Representation and parsing of multiword expressions: Current trends*, 61–79. Berlin: Language Science Press. DOI:10.5281/zenodo.2579047

In our research, MWEs are categorized in five subclasses: compounds, discontinuous words, named entities, collocations and idioms. While the first three are expressions of lexical categories (N, V, Adj, etc.) and can therefore be listed along with simple words, collocations and idioms are expressions of phrasal categories (NPs, VPs, etc.). The identification of compounds and named entities can be achieved during the lexical analysis, but the identification of discontinuous words (e.g., particle verbs or phrasal verbs), collocations and idioms requires grammatical data and should be viewed as part of the parsing process.

In this chapter, we will primarily focus on collocations, roughly defined as arbitrary and conventional associations of two words (not counting grammatical words) in a particular grammatical configuration (adjective-noun, noun-noun, verb-object, etc.). Throughout this chapter, we will refer to words belonging to such associations as *content words*. We will argue that the identification of collocations and parsing are interrelated processes – in the sense that one cannot precede the other – and we will show how this has been achieved in the Fips multilingual parser (Wehrli 2007; Wehrli & Nerima 2015).

Section 2 will give a brief review of MWEs and previous work. Section 3 will describe how Fips handles MWEs and the way they are represented in our lexical database. Section 4 will be concerned with the treatment of collocation types which present a fair amount of syntactic flexibility (e.g. verb-object). For instance, verbal collocations may undergo syntactic processes such as passivization, relativization, interrogation and even pronominalization, which can leave the collocation constituents far away from each other and/or reverse their canonical order. Section 5 will present the collocation extraction process, which will be evaluated in Section 6. Finally we will conclude in Section 7.

2 Multiword expressions: A brief review of related work

The standard approach in dealing with MWEs in parsing is to apply a 'words-with-spaces' preprocessing step, which marks the MWEs in the input sentence as units which will later be integrated as single blocks in the parse tree built during analysis (Brun 1998; Zhang & Kordoni 2006). This method is not really adequate for processing collocations. Unlike other expressions that are fixed or semi-fixed, several collocation types do not allow a 'words-with-spaces' treatment because they have a high morphosyntactic flexibility. On the other hand, Alegria et al. (2004); Villavicencio et al. (2007) adopted a compositional approach to the encoding of MWEs, able to capture more morphosyntactically flexible MWEs. Alegria et al. (2004) showed that by using a MWE processor in the preprocessing stage, a significant improvement in the POS tagging precision is obtained. Villavicencio et al. (2007) found that the addition of 21 new MWEs to the lexicon led to a significant increase in the grammar coverage (from 7.1% to 22.7%), without altering the grammar accuracy. However, as argued by many researchers (e.g., Heid 1994; Seretan 2011), collocation identification is best performed on the basis of parsed material. This is due to the fact that collocations are co-occurrences of lexical items in a specific syntactic configuration. For that reason, we have chosen the identification of collocations as soon as possible during parsing. Finkel & Manning (2009) have built a joint model of parsing and named entity recognition, based on a discriminative feature-based constituency parser. They tested their model on the OntoNotes annotated corpus[1] and they achieved a remarkably good performance on both parsing and recognition of named entities. Green et al. (2013) have developed two structured prediction models with the aim to identify arbitrary-length, contiguous MWEs in Arabic and French. The first is based on context-free grammars and the second uses tree substitution grammars, a formalism that can store larger syntactic fragments. They claim that these techniques can be applied to any language for which a syntactic treebank, a MWE list, and a morphological analyzer exist. Nasr et al. (2015) have developed a joint parsing and MWE identification model for the detection and representation of ambiguous complex function words. Constant & Nivre (2016) developed a transition-based parser which combines two factorized substructures: a standard tree representing the syntactic dependencies between the lexical elements of a sentence and a forest of lexical trees including MWEs identified in the sentence.

3 The Fips parser

Fips is a multilingual parser, available for several languages, i.e. French, English, German, Italian, Spanish, Modern Greek, Romanian and Portuguese. It relies on generative grammar concepts and is basically made up of a generic parsing module which can be refined in order to suit the specific needs of a particular language. Fips is a constituent parser that functions as follows: it scans an input string from left to right, without any backtracking. The parsing algorithm, iteratively, performs the following three steps:

- get the next lexical item and project the relevant phrasal category ($X \longrightarrow XP$);
- merge XP with the structure in its left context (the structure already built);

[1]http://www.gabormelli.com/RKB/OntoNotes_Corpus, last accessed 26 February 2019.

- (syntactically) interpret XP, triggering procedures
  - to build predicate-argument structures
  - to create chains linking preposed elements to their trace
  - to find the antecedent of (3rd person) personal pronouns
  - to identify collocations.

The parsing procedure is a one-pass (no pre-processing, no post-processing) scan of the input text, using rules to build up constituent structures and (syntactic) interpretation procedures to determine the dependency relations between constituents (grammatical functions, etc.), including cases of long-distance dependencies. One of the key components of the parser is its lexicon, which contains detailed morphosyntactic and semantic information, selectional properties, valency information, and syntactico-semantic features that are likely to influence the syntactic analysis.

3.1 The Fips lexicon

The lexicon was built manually and contains fine-grained information required by the parser. It is organized as a relational database with four main tables:

words, representing all morphological forms (spellings) of the words of a language, grouped into inflectional paradigms;

lexemes, describing more abstract lexical forms which correspond to the syntactic and semantic readings of a word (a lexeme corresponds roughly to a standard dictionary entry);

collocations, which describe multiword expressions combining two lexical items, not counting function words;

variants, which list all the alternatives written forms for a word, e.g. the written forms of British English vs American English, the spellings introduced by a spelling reform, presence of both literary and modern forms in Greek, etc.

3.2 Representation of MWEs in the lexicon

In the introduction we mentioned that in our research, MWEs are categorized in five subclasses, i.e. compounds, discontinuous words, named entities, collocations and idioms. We will now describe how they are represented in the lexical database.

Compounds and named entities are represented by the same structure as simple words. An entry describes the syntactic and (some) semantic properties of the word: lexical category (POS), type (e.g., common noun, auxiliary verb), subtype, selectional features, argument structure, semantic features, thematic roles, etc. Each entry is associated with the inflectional paradigm of the word, that is all the inflected forms of the word along with the morphological features (number, gender, person, case, etc.). The possible spaces or hyphens of the compounds are processed at the lexical analyzer level in order to distinguish those that are separators from those belonging to the compound.

Discontinuous words, such as particle verbs or phrasal verbs, are represented in the same way as simple words as well, except that the orthographic string contains the bare verb only, the particle being represented separately in a specific field. The benefit of such an approach is that the phrasal verb inherits the inflectional paradigm of the basic verb. For agglutination, a lexical analyzer will detect and separate the particle from the basic verb.

Collocations are defined as associations of two lexical units (not counting function words) in a specific syntactic relation (for instance adjective-noun, verb-object, etc.). A lexical unit can be a word or a collocation. The definition is therefore recursive and enables to encode collocations that have more than two words (Nerima et al. 2010). For instance, the French collocation *tomber en panne d'essence* ('to run out of gas') is composed of the word *tomber* (lit. 'fall') and the collocation *panne d'essence* (lit. 'failure of gas'). Similarly, the English collocation *guaranteed minimum wage* is composed of the word *guaranteed* and the collocation *minimum wage*.

In addition to the two lexical units, a collocation entry encodes the following information: the citation form, the collocation type (i.e., the syntactic relation between its two components), the preposition (if any) and a set of syntactic frozenness constraints.

Some examples of entries are given in (1), (2) and (3).

(1) *ein Schlaglicht werfen* (DE) 'to highlight'
type : verb-direct object
lexeme #1 : *Schlaglicht* 'spotlight', noun-noun collocation
lexeme #2: *werfen* 'throw', _ NP PP verb
preposition : ∅
features :{}

(2) *κινητό τηλέφωνο* (kinitó tiléfono) (MG) 'mobile phone'
type : adjective-noun
lexeme #1 : *κινητό* (kinitó) 'mobile', adjective

lexeme #2 : *τηλέφωνο* (tiléfono) ‘phone’, noun
preposition : ∅
features : {}

(3) *banc de poissons* (FR) ‘shoal of fish’
type : noun-prep-noun
lexeme #1 : *banc* ‘bench’, noun
lexeme #2 : *poisson* ‘fish’, noun
preposition : *de* ‘of’
features : {determiner-less complement, plural complement}

For the time being, we represent idioms as collocations with more restriction features (cannot passivize, no modifiers, etc.). They are, therefore, stored in the same database table. Reducing idioms to collocations with specific features though convenient and appropriate for large classes of idioms is nevertheless not general enough. In particular, it does not allow for the representation of idioms with fixed phrases, such as *to get a foot in the door.*

3.3 Fips and collocations

3.3.1 Collocation identification mechanism

The collocation identification mechanism is integrated in the parser. In the present version of Fips, collocations, if present in the lexicon, are identified in the input sentence during the analysis of that sentence, rather than at the end. In this way, priority is given to parsing alternatives involving collocations, and collocational information helps the parser through the maze of alternatives. To fulfill the goal of interconnecting the parsing procedure and the identification of collocations, we have incorporated the collocation identification mechanism within the constituent attachment procedure (see next section). The Fips parser, like many grammar-based parsers, uses left attachment and right attachment rules to build respectively left subconstituents and right subconstituents. The grammar used for the computational modelling comprises rules and procedures. Attachment rules describe the conditions under which constituents can combine, while procedures compute properties such as long-distance dependencies, agreement, control properties, argument-structure building, and so on.

3.3.2 Treatment of collocations

The identification of compounds and named entities can be achieved during the lexical analysis, but the identification of discontinuous words, collocations and idioms requires grammatical data and are, therefore, part of the parsing process. The identification of a collocation occurs when the second lexical unit of the collocation is attached, either by means of a left attachment rule (e.g., adjective-noun, noun-noun) or by means of a right-attachment rule (e.g., noun-adjective, noun-prep-noun, verb-object), as shown in example (4).

(4) Paul took up a new challenge.
$[_{\text{TP}}$ $[_{\text{DP}}$ Paul$]$$[_{\text{VP}}$ took up $[_{\text{DP}}$ a $[_{\text{NP}}$ $[_{\text{AP}}$ new$]$ challenge$]]]]$

When the parser reads the noun *challenge* and attaches it (along with the prenominal adjective) as complement of the incomplete $[_{\text{DP}}$ a$]$ direct object of the verb *take up*, the identification procedure considers iteratively all the governing nodes of the attached noun and checks whether the association of the lexical head of the governing node and the attached element constitutes an entry in the collocation database. The process stops at the first governing node of a major category (noun, verb or adjective). In our example, going up from *challenge*, the process stops at the verb *take up*. Since *take up - challenge* is an entry in the collocation database and its type (verb-object) corresponds to the syntactic configuration, the identification process succeeds.

In several cases the two constituents of a collocation can be very far apart, or do not appear in the expected order. We will turn to such examples in the next section. To handle them, the identification procedure sketched above must be slightly modified so that not only the attachment of a lexical item triggers the identification process, but also the attachment of the trace of a preposed lexical item. In such a case, the search will consider the antecedent of the trace. This shows, again, that the main advantage provided by a syntactic parser in such a task is its ability to identify collocations even when complex grammatical processes disturb the canonical order of constituents.

4 Detection of collocations in free word-order languages

Just as other types of MWEs, collocations are problematic for NLP because they have to be recognized and treated as a whole, rather than compositionally (Sag et al. 2002). On the other hand, there is no systematic restriction on lexical forms which constitute a collocation, on the order of items in a collocation, or on the number of words that may intervene between these items especially in free word-order languages. In such languages, the direct object of a verbal collocation can be found either before or after the verb, with or without intervening material. This is illustrated in the following examples with the Greek verb-object collocation *κάνω* *έκκληση* (káno éklisi) 'to make an appeal'. In (5a), the direct object follows the verb, while in (5b), it precedes the verb, with several intervening words between them:

(5) a. *Ο Υπουργός Παιδείας έκανε έκκληση στους διοικητικούς υπαλλήλους*
O Ipurgós Pedías *ékane éklisi* stus diikitikús ipalílus
να σταματήσουν την απεργία.
ná stamatísun tín aperyía
'The Minister of Education *made an appeal* to the administrative staff to stop the strike.'

b. *Έκκληση στους διοικητικούς υπαλλήλους να σταματήσουν την*
Éklisi stus diikitikús ipalílus ná stamatísun tín
απεργία έκανε ο Υπουργός Παιδείας.
aperyía *ékane* o Ipurgós Pedías
'*An appeal* to the administrative staff to stop the strike *made* the Minister of Education.'

4.1 Nominal collocations

Modifiers can often be attached within a nominal collocation, separating the two terms. For example, between the constituents of a nominal collocation in the form of adjective-noun, other lexemes may interfere. Table 1 shows a part of the analysis of a sentence where the possessive determiner του (tu) 'his' occurs between the adjective παρθενικό (parthenikó) 'maiden' and the noun ταξίδι (taxídi) 'voyage' of the collocation παρθενικό ταξίδι (parthenikó taxídi) 'maiden voyage'. Note that, for the POS tagset, we opted for the universal tagset (Petrov et al. 2012).

Table 1: Identification of the nominal collocation παρθενικό ταξίδι (parthenikó taxídi) 'maiden voyage'

word	tag	position	collocation
To (to) 'the'	DET	1	
παρθενικό (parthenikó) 'maiden'	ADJ	4	
του (tu) 'his'	PRON	14	
ταξίδι (taxídi) 'voyage'	NOUN	18	παρθενικό ταξίδι 'maiden voyage'

4.2 Verbal collocations

Verb-object collocations may undergo syntactic processes such as passivization, relativization, interrogation and even pronominalization, which can leave the collocation constituents far away from each other and/or reverse their canonical order.

4.2.1 Passive

In passive constructions, the direct object is promoted to the subject position leaving an empty constituent in the direct object position. The detection of a verb-object collocation in a passive sentence is thus triggered by the insertion of the empty constituent in direct object position. The collocation identification procedure checks whether the antecedent of the (empty) direct object and the verb constitute a verb-object collocation. In example (6), the noun απόφαση (apófasi) ‘decision’ of the collocation παίρνω απόφαση (pérno apófasi) ‘to make a decision’ precedes the verb and is in the nominal case, the usual case for subjects.

(6) *Η απόφαση πάρθηκε.*
I apófasi párthike.
‘The decision was made.’

4.2.2 Pronominalization

Another transformation that can affect some collocation types is pronominalization. In such cases, it is important to identify the antecedent of the pronoun which can be found either in the same sentence or in the context. Example (7) illustrates a phrase where the pronoun *it* refers to the noun *money*. Since the pronoun is the subject of the passive form *would be well spent*, it is interpreted as the direct object of the verb and therefore stands for an occurrence of the collocation *to spend money*.

(7) … though where the money would come from, and how to ensure that *it* would be well *spent*, is unclear.

In example (8) and Table 2, both the verb να αναλάβουν (na analávun) ‘to take’ of the verb-object collocation αναλαμβάνω ευθύνη (analamváno efthíni) ‘to take responsibility’ and the pronominalized object τις (tis) ‘them’ are found in another sentence.

(8) *Ας αναλογιστούν τις ευθύνες τους. Να τις αναλάβουν.*
As analogistún tis efthínes tus. Na tis analávun.
'Let them consider their responsibilities. Should they take them.'

Table 2: Identification of a verbal collocation

word	tag	position	collocation
Ας (as) 'Let them'	PRT	1	
αναλογιστούν (analogistún) 'consider'	VERB	4	
τις (tis) 'the'	DET	17	
ευθύνες (efthínes) 'responsibilities'	NOUN	21	
τους (tus) 'their'	PRON	21	
.	PUNC	33	
Να (Na) 'Should'	CONJ	35	
τις (tis) 'them'	PRON	35	
αναλάβουν (analávun) 'take'	VERB	42	αναλαμβάνω την ευθύνη 'take responsibility'
.	PUNC	51	

Example (9) and Table 3 concern French and show again two sentences. Each one of them contains a collocation with the noun *record*: *établir un record* 'to set up a record' in the first one, and *battre un record* 'to break a record' in the second one, where the noun is pronominalized in the form of a clitic pronoun (*le* 'it').

(9) *Ce record a été établi l'été dernier. Paul espère le battre bientôt.*
This record has been set up last summer. Paul hopes *it break* soon.
'This record was set up last summer. Paul hopes to break it soon.'

The parser detects collocations in which the nominal element has been pronominalized thanks to the anaphora resolution component incorporated in Fips (Wehrli & Nerima 2013).

4.2.3 *Wh*-constructions

Our parser can also cope with long-distance dependencies, such as the ones found in wh-questions.[2] In sentence (10) and Table 4, the direct object constituent occurs at the beginning of the sentence. Again, assuming a generative grammar

[2]wh-words are interrogative (or relative) words such as *who*, *what*, *which*, etc. For a general discussion of wh-constructions, see (Chomsky 1977).

Table 3: Identification of verbal collocations, one with pronominalized object

word	tag	position	collocation
Ce	DET	1	
record	NOUN	4	
a	VERB	11	
été	VERB	13	
établi	VERB	17	établir un record
l’	DET	24	
été	NOUN	26	été dernier
dernier	ADJ	30	
.	PUNC	37	
Paul	NOUN	1	
espère	VERB	6	
le	PRON	13	
battre	VERB	16	battre un record
bientôt	ADV	23	
.	PUNC	30	

analysis, we consider that such pre-posed constituents are connected to so-called canonical positions. The fronted element being a direct object, the canonical position is a post-verbal DP position immediately dominated by the VP node. The parser establishes such a link and returns the structure from (10), where [DP e]$_i$ stands for the empty category (the ‘trace’) of the preposed constituent Ποιο ρεκόρ (Pxó rekór) ‘Which record’.

(10) *Ποιο ρεκόρ θέλει να σπάσει ο Μελισσανίδης?*
Pxó rekór théli na spási o Melisanídis
[$_{CP}$ [$_{DP}$ Ποιο ρεκόρ]$_i$] [$_{TP}$ θέλει] [$_{CP}$ να] [$_{TP}$ σπάσει] [$_{VP}$ [$_{DP}$ e]$_i$] [$_{DP}$ ο Μελισσανίδης]
‘Which record does Melissanidis want to break?’

In such cases, the collocation identification process is triggered by the insertion of an empty constituent in the direct object position of the verb. Since the empty constituent is connected to the pre-posed constituent, such examples can be easily treated as a minor variant of the standard case described in Section 3.3.1. All so-called wh-constructions are treated in a similar fashion, that is relative clause and topicalization.

Table 4: Identification of verbal collocation in a wh-question

word	tag	position	collocation
Ποιο (Pio) 'Which'	DET	1	
ρεκόρ (rekór) 'record'	NOUN	6	
θέλει (théli) 'wants'	VERB	12	
να (na) 'to'	CONJ	18	
σπάσει (spási) 'break'	VERB	21	σπάζω το ρεκόρ 'break the record'
ο (o) 'the'	DET	28	
Μελισσανίδης (Melisanídis) 'Melisanidis'	NOUN	30	

4.2.4 *Tough*-movement constructions

In such constructions, the matrix subject is construed as the direct object of the infinitival verb governed by a 'tough' adjective. Following Chomsky's (1977) analysis of such constructions, the parser will hypothesize an abstract wh-operator in the specifier position of the infinitival clause, which is linked to the matrix subject. Like all wh-constituents, the abstract operator will itself be connected to an empty constituent later on in the analysis, giving rise to a chain connecting the subject of the main clause and the direct object position of the infinitival clause. The structure as computed by the parser is given in (11), with the chain marked by the index $_i$.

(11) [$_{\text{TP}}$ [$_{\text{DP}}$ this record]$_i$ seems[$_{\text{AP}}$ difficult[$_{\text{TP}}$ [$_{\text{DP}}$ e]$_i$ to[$_{\text{VP}}$ break[$_{\text{DP}}$ e]$_i$]]]]

4.3 Complex collocations

As observed by Heid (1994), among others, collocations can involve more than two content words. Such complex expressions can be described recursively as collocations of collocations. Our identification procedure has been extended to handle such cases. For example, the Greek noun-noun collocation απεργία πείνας (aperyía pínas) 'hunger strike', which combines with the verb κάνω (káno) 'to do', yields the larger verb-object collocation κάνω απεργία πείνας (káno aperyía pínas) 'to go on hunger strike', where the object is itself a noun-noun collocation. Given the strict left-to-right processing order assumed by the parser, the system will first identify the collocation κάνω απεργία (káno aperyía) 'to go on strike' when attaching the word απεργία (aperyía) 'strike'. Then, reading the last word, πείνας (pínas) 'hunger' (here in genitive case), the parser will identify the collocation απεργία πείνας (aperyía pínas) 'hunger strike'. The search succeeds with the verb κάνω (káno) ‘to do’, and the collocation κάνω απεργία πείνας (káno aperyía pínas) ‘to go on hunger strike’ is identified.

Moreover, the Greek lexical database comprises nominal collocations formed by a simple noun and a collocation or by two collocations. For example, δύναμη πολιτικής προστασίας (dínami politikís prostasías) ‘civil protection force’ is formed by a simple noun, δύναμη (dínami) ‘force’, and a nominal collocation in genitive case, πολιτικής προστασίας (politikís prostasías) ‘of civil protection’. The collocation πυρηνικός σταθμός παραγωγής ενέργειας (pirinikós stathmós paragoyís enéryias) ‘nuclear power station’ is formed by the collocations πυρηνικός σταθμός (pirinikós stathmós) ‘nuclear station’ and παραγωγής ενέργειας (paragoyís enéryias) ‘of energy production’.

5 Collocation extraction

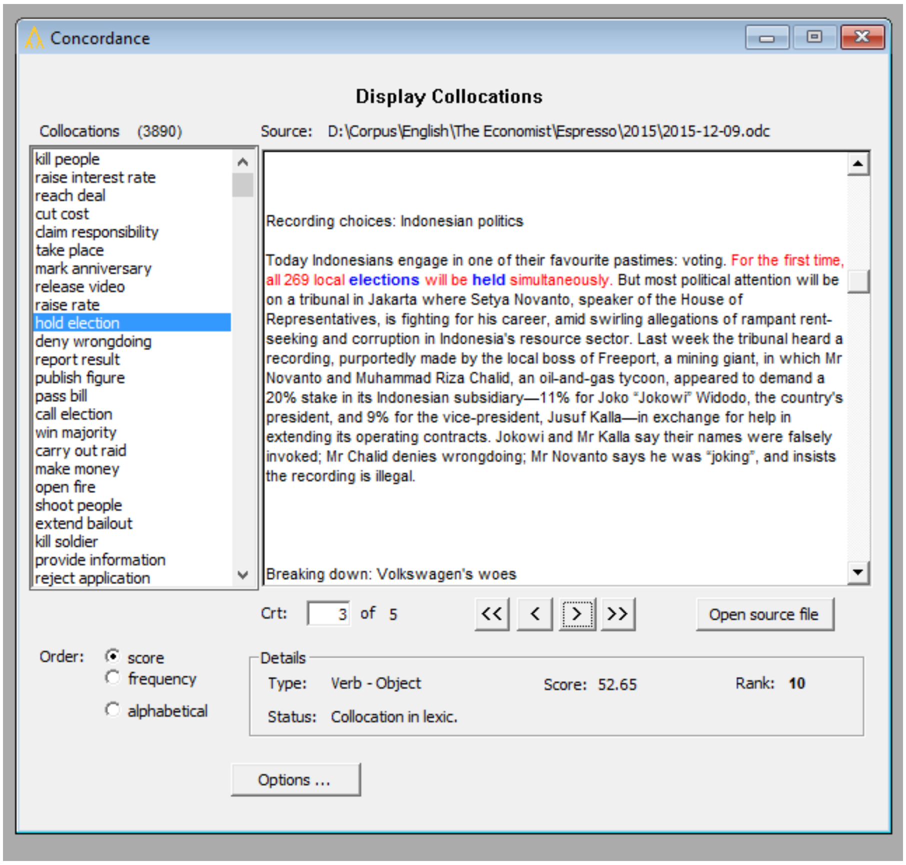
As already mentioned, the parser can only identify collocations that are part of its lexical database. Therefore, it is crucial to have as good a coverage of collocations as possible in the database. To help the linguist/lexicographer in the time-consuming task of inserting collocations, we have designed a collocation extraction tool (Seretan 2011), dubbed FipsCo. Applied to a corpus, FipsCo parses all the sentences, extracting all the pairs of lexical items which co-occur in predefined grammatical configurations (adjective-noun, noun-noun, subject-verb, verb-object, etc.). All those pairs are considered as potential collocations.

Once the corpus has been completely parsed, a statistical filter is used to rank the potential collocations according to their degree of association. By default, we use the log-likelihood ratio measure (LLR), since it was shown to be particularly suited to language data (Dunning 1993). In our extractor, the items of each candidate expression represent base word forms (lemmas) and they are considered in the canonical order implied by the given syntactic configuration (e.g., for a verb-object candidate, the object is postverbal in subject-verb-object (SVO) languages like Greek). Even if the candidate occurs in corpus in different morphosyntactic realizations, its various occurrences are successfully identified as instances of the same type thanks to the syntactic analysis performed by the parser.

Figure 1 displays a list of verb-object collocations extracted from an English corpus taken from the magazine *The Economist*. On the left, candidate collocations are listed and at the same time they are shown in their context.

Our system recognizes a large range of collocation types (more than 30 types), including several nominal and verbal ones. The most frequent types are:

- Adjective-noun, e.g. *nuclear war*;


Concordance
Display Collocations
Collocations (3890)
Source: D:\Corpus\English\The Economist\Espresso\2015\2015-12-09.odc

kill people
raise interest rate
reach deal
cut cost
claim responsibility
take place
mark anniversary
release video
raise rate
hold election
deny wrongdoing
report result
publish figure
pass bill
call election
win majority
carry out raid
make money
open fire
shoot people
extend bailout
kill soldier
provide information
reject application

Recording choices: Indonesian politics

Today Indonesians engage in one of their favourite pastimes: voting. For the first time, all 269 local elections will be held simultaneously. But most political attention will be on a tribunal in Jakarta where Setya Novanto, speaker of the House of Representatives, is fighting for his career, amid swirling allegations of rampant rent-seeking and corruption in Indonesia's resource sector. Last week the tribunal heard a recording, purportedly made by the local boss of Freeport, a mining giant, in which Mr Novanto and Muhammad Riza Chalid, an oil-and-gas tycoon, appeared to demand a 20% stake in its Indonesian subsidiary—11% for Joko "Jokowi" Widodo, the country's president, and 9% for the vice-president, Jusuf Kalla—in exchange for help in extending its operating contracts. Jokowi and Mr Kalla say their names were falsely invoked; Mr Chalid denies wrongdoing; Mr Novanto says he was "joking", and insists the recording is illegal.

Breaking down: Volkswagen's woes

Order: [✓ score] frequency alphabetical

Details
Type: Verb - Object
Score: 52.65
Rank: 10
Status: Collocation in lexic.

Figure 1: Extraction of verb-object collocations

- Noun-noun, e.g. *flower shop*;
- Noun-preposition-noun, e.g. *casco di banane* ('*bunch of bananas*');
- Verb-object where the object is a bare noun, e.g. *take part*;
- Verb-preposition-noun, e.g. *bring to light*;
- Verb-adverb, e.g. *put together*.

Once filtered and ordered by means of standard association measures, the candidate collocations are manually validated and added to the lexical database. The current content of the database for six European languages is shown in Table 5.

Table 5: Number and types of collocations in the Fips lexical database

collocation type	English	French	German	Italian	Spanish	Greek
Adjective-noun	3,049	5,935	490	1,325	1,621	20,131
Noun-noun	5,671	454	2,476	131	66	471
Noun-prep-noun	555	7,846	22	1,246	988	11
Verb-object	850	1,560	197	250	1,098	382
Others	932	2,963	330	209	592	126
Total	11,057	18,758	3,515	3,161	4,365	21,122

6 Evaluation and results

The Fips parser performs well compared to other 'deep' linguistic parsers (Delph-in,[3] ParGram,[4] etc.) in terms of speed. Parsing time depends on two main factors: (i) the type and complexity of the corpus, and (ii) the selected beam size (maximum number of alternatives allowed). By default, Fips runs with a beam size of 40 alternatives, which gives it a speed ranging from 150 to 250 tokens (word, punctuation) per second. At that pace, parsing a one million word corpus takes approximately 2–3 hours. We are going to present the experiments that were performed for Modern Greek and English in order to evaluate the performance of our parser.

6.1 Modern Greek

The evaluation measures the performance of our parser to identify collocations that are lexicalized (i.e. collocations that are present in the lexical database). We also measure the impact of the collocation knowledge on the performance of the parser (in percentage of complete analyses). To achieve the evaluation, we took a small newspaper corpus of about 20,000 words and we manually identified 638 collocations (both nominal and verbal). We ran the parser twice on the corpus: the first time before and the second time after enrichment of the collocation database. On the first run, the parser achieved 43.26% of complete analyses and identified 124 collocations. On the second run, after enrichment of the lexicon,

[3]International consortium developing HPSG grammars and other tools, cf. http://www.delph-in.net/wiki/index.php/Home.

[4]ParGram is an international consortium for the development of LFG-based grammars, see http://pargram.b.uib.no.

the percentage of complete analyses increased to 44.33% and nearly three quarters of the corpus collocations were identified (482/638). Over this small corpus, the parser achieved a 100% precision in the collocation identification task, with a recall of 75.54% and an F-measure of 86%. The collocations that were not identified (156 out of 638) were part of sentences for which the parser did not achieve a complete analysis.

6.2 English

We have also conducted an evaluation over a corpus with approximately 6,000 sentences taken from *The Economist*. The research questions were specifically focused on the statistical significance of ambiguity resolution based on collocation knowledge and on how frequently, in a given corpus, the detection of a collocation helps the parser make the 'right' decision. To answer those questions, we parsed the corpus twice, first with the collocation detection component turned on and then with the component turned off. We then compared the results of both runs. Since it was difficult to compare phrase-structure representations, we used the Fips tagger, that is the Fips parser with part-of-speech output. It is indeed much easier to compare POS-tags than phrase-structures. Tables 6 and 7 illustrate the Fips tagger output for the segment in boldface of the sentence *The researchers estimated* ***the total worldwide labour costs*** *for the iPad at $33, of which China's share was just $8.*

Table 6 gives the results obtained with the collocation detection component turned on, and Table 7 the results obtained with the component turned off.

Table 6: Parser output *with* collocation knowledge

word	tag	position	collocation
the	DET	27	
total	ADJ	31	
worldwide	ADJ	37	
labour	NOUN	47	
costs	**NOUN**	54	labour costs

The sentence segment *the total worldwide labour costs* is displayed in both tables with the words in the first column, the part-of-speech tag in the second column and the position – expressed as position of the first character of each word starting from the beginning of the sentence – in the third column. As we

Table 7: Parser output *without* collocation knowledge

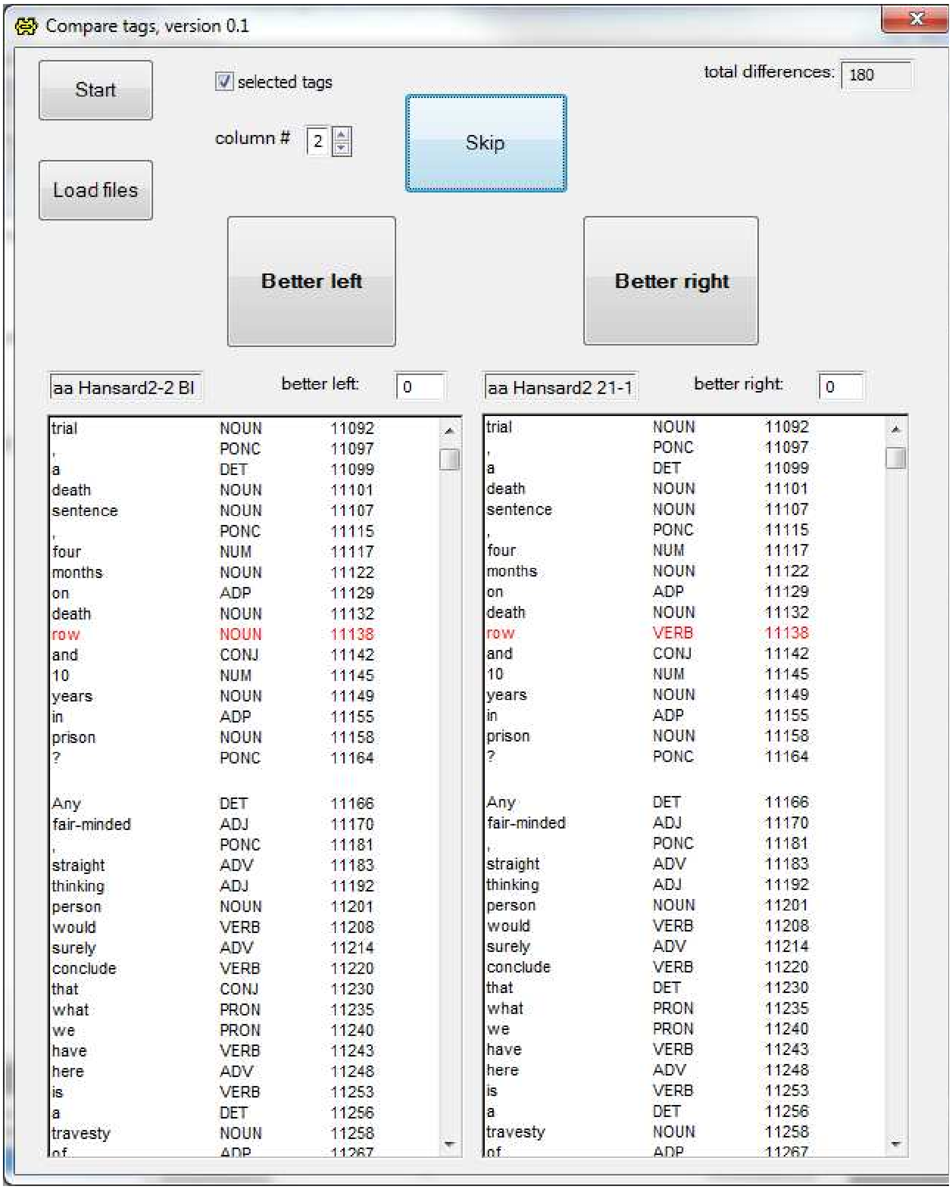
word	tag	position	collocation
the	DET	27	
total	ADJ	31	
worldwide	ADJ	37	
labour	NOUN	47	
costs	**VERB**	54	

can see, the word *costs* is taken as a noun in the first analysis, as a verb in the second. The (correct) choice of a nominal reading in the first analysis is due to the detection of the collocation *labour costs*. In the second run, given the absence of collocational knowledge, the parser opts for the verbal reading. Both output files could easily be manually compared using a specific user interface as illustrated in the screenshot in Figure 2, where POS differences are displayed in red.

Table 8: POS-tagging with and without collocation knowledgeinline]value missing in row 2, col 3?

	with collocations	without collocations
complete analyses	73.41%	72.95%
POS-tag differences	727	
better tags	382	106
number of collocations	1668	-

A summary of the results of the evaluation is given in Table 8. The first line shows the number of complete analyses. Collocational knowledge increases the number of complete analysis by approximately 0.5%, or about 30 sentences for our corpus of 6,000 sentences. 727 tags are different between the two runs. Of those, excluding differences which do not really matter (some words can be analyzed either as predicative adjectives or as adverbs without much semantic differences, etc.), in 382 cases the tags were better in the first run (with collocational knowledge), and 106 cases better in the second run (without collocational knowledge). In other words, collocational knowledge helped the parser make the better decision four times more than it penalized it. Notice finally that 1,668 collocations were detected in the corpus (more than one in four sentences), which clearly stresses the high frequency of this phenomenon in natural language.



☑ selected tags

column # 2

total differences: 180

aa Hansard2-2 BI — better left: 0

aa Hansard2 21-1 — better right: 0

trial	NOUN	11092
,	PONC	11097
a	DET	11099
death	NOUN	11101
sentence	NOUN	11107
,	PONC	11115
four	NUM	11117
months	NOUN	11122
on	ADP	11129
death	NOUN	11132
row	NOUN	11138
and	CONJ	11142
10	NUM	11145
years	NOUN	11149
in	ADP	11155
prison	NOUN	11158
?	PONC	11164
Any	DET	11166
fair-minded	ADJ	11170
,	PONC	11181
straight	ADV	11183
thinking	ADJ	11192
person	NOUN	11201
would	VERB	11208
surely	ADV	11214
conclude	VERB	11220
that	CONJ	11230
what	PRON	11235
we	PRON	11240
have	VERB	11243
here	ADV	11248
is	VERB	11253
a	DET	11256
travesty	NOUN	11258
of	ADP	11267

trial	NOUN	11092
,	PONC	11097
a	DET	11099
death	NOUN	11101
sentence	NOUN	11107
,	PONC	11115
four	NUM	11117
months	NOUN	11122
on	ADP	11129
death	NOUN	11132
row	VERB	11138
and	CONJ	11142
10	NUM	11145
years	NOUN	11149
in	ADP	11155
prison	NOUN	11158
?	PONC	11164
Any	DET	11166
fair-minded	ADJ	11170
,	PONC	11181
straight	ADV	11183
thinking	ADJ	11192
person	NOUN	11201
would	VERB	11208
surely	ADV	11214
conclude	VERB	11220
that	DET	11230
what	PRON	11235
we	PRON	11240
have	VERB	11243
here	ADV	11248
is	VERB	11253
a	DET	11256
travesty	NOUN	11258
of	ADP	11267

Figure 2: The evaluation user interface

7 Conclusion

In this chapter, we have argued in favour of a treatment of collocations, and by extension of all MWEs, fully integrated in the parsing process. The argument is rather simple. On the one hand, we have shown that the identification of collocations must be based on analyzed data, and therefore cannot be performed before parsing. On the other hand, we have also shown that collocation identification can help the parser, for instance to solve lexical as well as syntactic ambiguities, provided that the identification is done before the end of parsing. The solution to this apparent paradox – collocation identification cannot be done before and cannot be done after parsing – is clear: collocation identification must be part of the parsing process and must be performed as early as possible, that is at the time the parser attaches the second constituent of the collocation, or inserts the trace of that constituent.

Abbreviations

Tagset from Petrov et al. (2012).

ADJ	adjective	NUM	numeral
ADP	adposition	PRON	pronoun
ADV	adverb	PRT	particle
CONJ	coordinating conjunction	PUNC	punctuation
DET	determiner	VERB	verb
NOUN	noun	X	other

References

Alegria, Iñaki, Olatz Ansa, Xabier Artola, Nerea Ezeiza, Koldo Gojenola & Ruben Urizar. 2004. Representation and Treatment of Multiword Expressions in Basque. In Takaaki Tanaka, Aline Villavicencio, Francis Bond & Anna Korhonen (eds.), *Second ACL workshop on multiword expressions: Integrating processing*, 48–55. Barcelona, Spain: Association for Computational Linguistics.

Brun, Caroline. 1998. Terminology finite-state preprocessing for computational LFG. In *COLING 1998 volume 1: The 17th international conference on computational linguistics*, 196–200. Montreal, QC.

Chomsky, Noam. 1977. On Wh-movement. In P.W. Culicover, T. Wasw & A. Akmajian (eds.), *Formal syntax*. San Francisco, London: Academic Press.

Constant, Matthieu & Joakim Nivre. 2016. A transition-based system for joint lexical and syntactic analysis. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, 161–171. Berlin, Germany: Association for Computational Linguistics. http://www.aclweb.org/anthology/P16-1016.

Dunning, Ted. 1993. Accurate Methods for the Statistics of Surprise and Coincidence. *Computational Linguistics* 19(1). 61–74. http://dl.acm.org/citation.cfm?id=972450.972454.

Finkel, Jenny Rose & Christopher D. Manning. 2009. Joint Parsing and Named Entity Recognition. In *Proceedings of Human Language Technologies: The 2009 annual conference of the North American chapter of the Association for Computational Linguistics*, 326–334. Boulder, Colorado: Association for Computational Linguistics. http://www.aclweb.org/anthology/N/N09/N09-1037.

Green, Spence, Marie-Catherine de Marneffe & Christopher D. Manning. 2013. Parsing models for identifying multiword expressions. *Computational Linguistics* 39(1). 195–227.

Heid, Ulrich. 1994. On Ways Words Work Together – Topics in Lexical Combinatorics. In Willy Martin et al. (ed.), *Proceedings of the vith Euralex International Congress (EURALEX'94)*, 226–257. Amsterdam. Eingeladener Hauptvortrag.

Nasr, Alexis, Carlos Ramisch, José Deulofeu & André Valli. 2015. Joint dependency parsing and multiword expression tokenisation. In *53rd annual meeting of the Association for Computational Linguistics*, 1116–1126. Beijing, China.

Nerima, Luka, Eric Wehrli & Violeta Seretan. 2010. A Recursive Treatment of Collocations. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, 634–638. Valletta, Malta.

Petrov, Slav, Dipanjan Das & Ryan McDonald. 2012. A universal part-of-speech tagset. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the eighth international Language Resources and Evaluation Conference, LREC 2012*, 2089–2096. Istanbul, Turkey. http://www.lrec-conf.org/proceedings/lrec2012/summaries/274.html.

Sag, Ivan, Timothy Baldwin, Francis Bond, Ann Copestake & Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In *Lecture Notes in Computer Science. Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Text Processing*, vol. 2276, 1–15. Springer.

Seretan, Violeta. 2011. *Syntax-based collocation extraction*. Vol. 44 (Text, Speech and Language Technology). Dordrecht: Springer.

Villavicencio, Aline, Valia Kordoni, Yi Zhang, Marco Idiart & Carlos Ramisch. 2007. Validation and evaluation of automatically acquired multiword expressions for grammar engineering. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 1034–1043. Prague, Czech Republic: Association for Computational Linguistics. http://www.aclweb.org/anthology/D/D07/D07-1110.

Wehrli, Eric. 2007. Fips: A "deep" linguistic multilingual parser. In *Proceedings of the ACL 2007 Workshop on Deep Linguistic Processing*, 120–127. Prague, Czech Republic: Association for Computational Linguistics. http://www.aclweb.org/anthology/W/W07/W07-1216.

Wehrli, Eric & Luka Nerima. 2013. Anaphora resolution, collocations and translation. In J. Monti, R. Mitkov, G. Corpas Pastor & V. Seretan (eds.), *Proceedings of the workshop on multi-word units in machine translation and translation technology (MUMTT'2013)*. Nice.

Wehrli, Eric & Luka Nerima. 2015. The Fips multilingual parser. In N. Gala, R. Rapp & G. Bel-Enquix (eds.), *Language Production Cognition, and the Lexicon, festschrift in honour of Michael Zock*, 473–489. Springer.

Zhang, Yi & Valia Kordoni. 2006. Automated deep lexical acquisition for robust open texts processing. In *Proceedings of the 5th internation conference on Language Resource and Evaluation (LREC-2006)*, 275–280. Genoa, Italy.

Representation and parsing of multiword expressions

Deep parsing is the fundamental process aiming at the representation of the syntactic structure of phrases and sentences. In the traditional methodology this process is based on lexicons and grammars representing roughly properties of words and interactions of words and structures in sentences. Several linguistic frameworks, such as Head-driven Phrase Structure Grammar (HPSG), Lexical Functional Grammar (LFG), Tree Adjoining Grammar (TAG), Combinatory Categorial Grammar (CCG), etc., offer different structures and combining operations for building grammar rules. These already contain mechanisms for expressing properties of Multiword Expressions (MWE), which, however, need improvement in how they account for idiosyncrasies of MWEs on the one hand and their similarities to regular structures on the other hand. This collaborative book constitutes a survey on various attempts at representing and parsing MWEs in the context of linguistic theories and applications.