Chapter 7

MWE-Finder: Querying for multiword expressions in large Dutch text corpora

Jan Odijk[a], Martin Kroon[a,b], Sheean Spoel[a], Ben Bonfil[a] & Tijmen Baarda[a]

[a]Utrecht University [b]Leiden University

We present MWE-Finder, an application that enables a user to search for multiword expressions (MWEs) in large Dutch text corpora. Components of many MWEs in Dutch can occur in multiple forms, need not be adjacent, and can occur in multiple orders (such MWEs are called *flexible*). Searching for such flexible MWEs is difficult and cannot be done reliably with most search applications. What is needed is a search engine that takes into account the grammatical configuration of the MWE. MWE-Finder is therefore embedded in GrETEL, a treebank search application for Dutch. A user can enter an example of a MWE in a specific canonical form, after which the system searches for sentences in which the MWE occurs, using queries generated automatically from the canonical form. We will describe in detail how the queries for this MWE are derived from the canonical form. The MWE can also be selected from a list of approximately 10k canonical forms for Dutch MWEs that MWE-Finder offers. We will show that MWE-Finder also offers facilities to find examples with unexpected modifiers or determiners on components of the MWE, and that it will yield statistics on the arguments, modifiers and determiners that occur with the MWE and its components.

1 Introduction

Many multiword expressions (MWEs) are flexible in the sense that their components can have different forms, can occur in different orders, or may not be contiguous, with other words appearing between elements of the MWE. This

Jan Odijk, Martin Kroon, Sheean Spoel, Ben Bonfil & Tijmen Baarda. 2024. MWE-Finder: Querying for multiword expressions in large Dutch text corpora. In Voula Giouli & Verginica Barbu Mititelu (eds.), *Multiword expressions in lexical resources: Linguistic, lexicographic, and computational perspectives*, 229–267. Berlin: Language Science Press. DOI: 10.5281/zenodo.10998643

makes searching for such MWEs in large text corpora difficult. What is needed is a search system that can take all this flexibility into account.

In this chapter we present such a system, called MWE-Finder. This system is specific for the Dutch language, but many aspects of the design of the system are not specific to Dutch or the specific parser used, as we will describe in Section 5.

We made a system for Dutch because this language exhibits flexibility in a wide range of MWEs. This is especially true for verbal MWEs (including proverbs), but also for certain nominal and adpositional MWEs. Searching for Dutch MWEs is thus an excellent and challenging test case for MWE-Finder. In addition, an excellent parser is available for Dutch, Alpino (van Noord 2006), which is also fully integrated in a treebank query application, GrETEL (Augustinus et al. 2017).

MWE-Finder enables a user to find occurrences of a multiword expression in a large Dutch text corpus. MWE-Finder is intended as a tool for any linguist or lexicographer interested in research into MWEs, in particular *flexible* MWEs.

MWE-Finder can be used to address the task of MWE *identification* in the sense of Constant et al. (2017): by using MWE-Finder a researcher can find occurrences of a given MWE easily and in a more reliable way than with other search applications. This will stimulate research into individual MWEs, their variants and their properties, and their frequencies, thereby facilitating research into MWEs in general. The system also creates a good basis for software to automatically annotate large text corpora for MWEs, which not only may be beneficial for linguistic research but also for a variety of natural language processing tools dealing with MWEs.

MWE-Finder uses the DUtch CAnonicalised Multiword Expressions lexical resource (DUCAME) to suggest MWEs to the user. This is a resource containing more than 10,000 MWEs for the Dutch language in a canonical form.

The organisation of this chapter is as follows. We begin with a brief introduction of the notion multiword expression (Section 2). The DUCAME resource is described in more detail in Section 3. MWE-Finder is presented in Section 4. In Section 5 we discuss the potential for extending MWE-Finder to other languages and other parsers. We will end with conclusions (Section 6) and plans for future work (Section 7).

2 Multiword expressions

A MWE is a word combination with linguistic properties that cannot be predicted from the properties of the individual words or the way they have been combined

by the rules of grammar (Odijk 2013b).[1] A word combination can, for example, have an unpredictable meaning (*de boeken neerleggen*, lit. 'to put down the books', meaning 'to declare oneself bankrupt'), an unpredictabe form (e.g. *ter plaatse* 'on location', with idiosyncratic use of *ter* and *e*-suffix on the noun), or it can have only limited usage (e.g. *met vriendelijke groet* 'kind regards', used as the closing of a letter). In a translation context, it can have an unpredictable translation (*dikke darm* lit. 'thick intestine', 'large intestine'), etc.

Note that it is not always easy to determine whether a combination of words is a MWE, because we do not always know the exact properties of the individual component words or what the grammar rules of a language are exactly. So this may require a substantial amount of research.

Words of a MWE need not always be fixed. This can be illustrated with the Dutch MWE *de boeken neerleggen* 'to declare oneself bankrupt'. The verb *neerleggen* in (1) can occur in all of its inflectional variants (e.g., past participle in (1a), infinitive in (1b), and past tense singular in (1c) and (1d)), and with the separable particle *neer* attached to it (1a, 1b) or separated (1c, 1d). MWEs do not necessarily consist of words that are adjacent, and the words making up a MWE need not always occur in the same order. This expression allows a canonical order with contiguous elements (as in (1a)), but it also allows other words to intervene between its components (as in (1b)), as well as permutations of its component words (as in (1c)), and combinations of permutations and intervention by other words that are not components of the MWE (as in (1d)):

(1) a. Saab heeft gisteren *de boeken neergelegd.*
Saab has yesterday the books down.laid
'Saab declared itself bankrupt yesterday.'

b. Ik dacht dat Saab gisteren *de boeken* wilde *neerleggen.*
I thought that Saab yesterday the books wanted down.lay
'I thought Saab wanted to declare itself bankrupt yesterday.'

c. Saab *legde de boeken neer.*
Saab laid the books down
'Saab declared itself bankrupt.'

d. Saab *legde* gisteren *de boeken neer.*
Saab laid yesterday the books down
'Saab declared itself bankrupt yesterday.'

[1] For a similar but slightly different definition see Sag et al. (2002).

In addition, certain MWEs allow for (and require) controlled variation in lexical item choice, e.g. in expressions containing bound anaphora, where the possessive pronoun varies depending on the subject, as in (2), exactly as in the English expression *to lose one's temper*.

(2) a. Ik *verloor mijn* / **jouw geduld.*
I lost my / *your patience
'I lost my temper.'

b. Jij *verloor* **mijn* / *jouw geduld.*
You lost *my / your patience
'You lost your temper.'

Of course, not every MWE allows all of these options, and not all permutations of the components of a MWE are well-formed (e.g. one cannot have **Saab heeft neergelegd boeken de.* lit. 'Saab has downlaid books the.').

In Dutch, even proverbs, which have no variable parts, are flexible, because the finite verb occupies a different position in main clauses (3a) than in subordinate clauses (3b), and adverbial modifiers modifying the whole proverb may split the words of the proverb (3c):

(3) Flexibility of proverbs in Dutch:

a. *De appel valt niet ver van de boom.*
the apple falls not far from the tree
'The apple never falls far from the tree.'

b. Hij zegt dat *de appel niet ver van de boom valt.*
he says that the apple not far from the tree falls
'He says that the apple never falls far from the tree.'

c. *De appel valt* immers *niet ver van de boom.*
The apple falls after all not far from the tree
'After all, the apple never falls far from the tree.'

This flexible nature of such MWEs makes it difficult to reliably search for such expressions in text corpora. Standard search engines such as Google do not enable the user to systematically search for different word forms of the same lemma. Search applications for Dutch such as OpenSoNaR (van de Camp et al. 2017, de Does et al. 2017) or Nederlab (Brugman et al. 2016) can do this, but it is difficult to formulate a query allowing different orders and interspersed irrelevant words, and the results of such a query will be unreliable. At best, one will find all

instances but at the same time also many cases where all the component words occur but do not make up a MWE. One should be able to search for flexible MWEs in such a way that their grammatical structure is taken into account. This can be done in a treebank, and MWE-Finder enables searching for MWEs in a treebank.

MWEs can contain multiple content words,[2] but can also contain only a single content word and one or more function words, and can even consist completely of function words. We will not focus here on some classes of MWEs that consist of a content word and one or more function words, such as verbs with obligatory bound reflexive pronouns, verbs with separable particles, verbs with prepositional complements headed by a specific preposition, and combinations thereof, as illustrated in (4), in which the MWE consists of the verb *trekken*, a separable particle (*op*), an idiosyncratically selected adposition (*aan*) and a reflexive pronoun (*zich*):

(4) Hij heeft *zich* altijd *aan* zijn vriend *op* kunnen *trekken*.
he has himself always to his friend up can pull
'He has always received support from his friend.'

Such MWEs are already fully dealt with by the grammar used in MWE-Finder.

3 The DUCAME MWE resource

The DUCAME lexical resource is available[3] and consists of a reworked version of the DuELME database (Grégoire 2009, 2010, Odijk 2013a) and a new list of MWEs composed by one of the authors on the basis of publicly available sources, which include Stoett (1923), Onze Taal,[4] VRT website,[5] Lassy-Small treebank (van Noord et al. 2013), and own collection. DUCAME contains more than 10,000 unique MWEs (many more than DUELME, which had around 5,000).

DUCAME is unique in that it has all the MWEs in a canonical form as described in more detail below. The MWEs also have annotations on properties of their parts. These annotations are based mostly on native speaker intuitions of the developers and have not been tested against large text corpora. MWE-Finder enables carrying out such tests.

[2] *Content word* is defined here as a word belonging to any of the syntactic categories noun, verb, adjective, or adverb.

[3] https://surfdrive.surf.nl/files/index.php/s/2Maw8O0QTPH0oBP

[4] https://onzetaal.nl/schatkamer/lezen/uitdrukkingen and https://onzetaal.nl/zoekresultaten?in=advice&zoek=uitdrukking

[5] https://vrttaal.net/taaladvies-taalkwestie/vaste-uitdrukkingen

Traditional dictionaries usually include a MWE by providing an example sentence, but it is very difficult for humans and nearly impossible for software to derive the general properties of the MWE from such an example. What is needed is a canonical form from which the properties of the MWE are easy to derive automatically. In addition, the canonical forms should be a well-formed expression of Dutch and should be parsable by automatic parsers.

For single words the canonical form is called the *lemma*, i.e. a specific form of an inflectional paradigm that is used as headword in traditional dictionaries. One can adopt this usage for the head of MWEs as well, and that works fine for many MWEs. However, it does not always work for a MWE with a verb as its head. In Dutch, the lemma of a verb is identical to the infinitive, but several problems arise when one tries to use the infinitive as the lemma for the head of a verbal MWE: first, no overt subjects can appear with an infinitive, so a MWE with an overt subject and an infinitive is an ill-formed expression:[6]

(5) a. * *De laatste loodjes het zwaarst wegen.*
the last lead.DIM.PL the heaviest weigh
'The tail end is the most difficult.'

b. * *De schellen* iemand *van de ogen vallen.*
the scales someone from the eyes fall
'His eyes are opened.'

Furthermore, though the subject must be absent, it is present implicitly and interpreted as an animate actor. If the subject of a MWE is not animate, using the MWE with an infinitival head as the canonical form gives infelicitous results:

(6) a. ? iemand *de keel uithangen*
someone the throat outhang
'for something to bore someone'

b. ? iemand *niet kunnen bommen*
someone not can care
'for someone not to care about something'

In order to avoid these problems and at the same time have a canonical form with an infinitive, the canonical forms in this resource are all finite sentences with a form of the future tense auxiliary verb *zullen* 'will' as its main verb, as in (7). These are all well-formed sentences that can in principle be parsed by a parser.

[6] DIM stands for diminutive, PL for plural.

(7) a. *De laatste loodjes* zullen *het zwaarst wegen.*
the last lead.DIM.PL will the heaviest weigh
'The tail end will be the most difficult.'

b. *De schellen* zullen iemand *van de ogen vallen.*
the scales will someone from the eyes fall
'His eyes will be opened.'

c. Iets zal iemand *de keel uithangen.*
something will someone the throat out.hang
'Something will bore someone.'

d. Iets zal iemand *niet kunnen bommen.*
something will someone not can care
'Someone will not care about something.'

By default, the canonical forms in DUCAME must be interpreted as allowing for the head of the MWE to be modified by determiners and/or other modifiers; a component of the MWE that is not its head cannot be modified by determiners and/or other modifiers individually unless these are themselves components of the MWE. Similarly, it is assumed that only the head of the MWE can occur in different inflectional forms, while other parts of the MWE cannot. Of course, there are many exceptions to this, and these are indicated in DUCAME by means of annotations. The annotations allowed are given in Table 1.

Table 1: Notational devices for annotating a canonical form. The code + can also be combined with * or *!* (in any order).

notation	interpretation
**word*	*word* is modifiable/determinable
+*word*	*word* is inflectable
=*word*	*word* must occur in the MWE as given
!*word*	*word* is not modifiable/determinable
dd:[*word*]	*word* must be a definite determiner
⟨*text*⟩	*text* is interpreted as a freely replaceable argument
0*word*	*word* is not part of the MWE

Arguments of the MWE that can be freely replaced by arbitrary phrases are represented by the indefinite pronouns *iemand* 'someone', *iets* 'something', and *ergens* 'somewhere', where this is possible. One can also use combinations such

as *iemand|iets* or *iets|iemand*, which are to be interpreted as allowing either but most likely with the first alternative. If such words must occur in the MWE as such (i.e. cannot be freely replaced), they can be preceded by the annotation =, as in (8).

(8) Iemand zal *voor =iets tussen* iets *zitten.*
someone will for something between something sit
'Someone will be a factor in something.'

The system of pronouns in natural languages in general and in Dutch in particular is in many respects somewhat arbitrary. So, *iemand* implies a human argument, whereas *iets* implies a nonhuman argument. A distinction between animate and inanimate nonhuman arguments does not exist in the Dutch pronominal system, nor does one between objects and events. For many phrase types there are no pronouns at all, e.g. for adjectival, adverbial and clausal phrases.[7] Nevertheless, the use of the existing pronouns is easy and rather natural for humans, and the missing pronouns are covered by a special annotation in which an arbitrary phrase surrounded by angled brackets ⟨...⟩ is interpreted as a freely replaceable argument, as in (9).

(9) Iemand zal ⟨makkelijk⟩ *in de omgang zijn.*
someone will easy in the interaction be
'Someone will be ⟨easy⟩-going, ⟨easy⟩ to deal with.'

Bound pronouns such as reflexives and possessive pronouns are represented by the third person singular forms (*zich*, *zichzelf*, *zijn*). If such forms do not vary, one can precede them by the annotation =, as in the expression *op =zich* lit. 'on REFL', 'in itself'. There is (currently) no convention or annotation to specify the antecedent of such bound anaphors.

It sometimes is necessary to include a word in a canonical form to create a natural utterance even if this word does not belong to the MWE. Such words can be preceded by the code *0*. This very often occurs in MWEs that have an indefinite subject, which prefer the presence of *er*, as in (10a), and in MWEs that are or contain negative polarity items and that require the presence of a licensing element such as a negative adverb (*niet* 'not'), determiner (*geen* 'no') or pronoun (e.g. *niemand* 'nobody'), e.g., in the MWE with canonical form *dd:[die] vlieger zal 0niet opgaan*. In (10b) the negative adverb *niet* cannot be absent, but it is arguably not part of the MWE, as shown by (10c) in which the negative pronoun *niemand*

[7] Sometimes it is possible to use pronouns for noun phrases to refer to these but there are no pronouns that can actually replace them.

in the main clause is the licensing element for the negative polarity MWE in the subordinate clause.[8]

(10) a. 0Er zal iets *op het spel staan.*
there will something on the game stand
'Something will be at stake.'

b. Die *vlieger* zal *(niet) *opgaan.*
that kite will not up.go
'That won't wash.'

c. Niemand denkt dat die *vlieger opgaat.*
nobody thinks that that kite up.goes
'Nobody thinks that that will wash.'

4 MWE-Finder

MWE-Finder enables a user to search for occurrences of a MWE in a treebank based on an example MWE in the canonical form as described in Section 3. It is embedded in GrETEL, an existing web application for searching Dutch treebanks (Augustinus et al. 2012, 2017, Odijk et al. 2018). The distinguishing feature of GrETEL is its query-by-example feature. In its regular search mode, it leads the user through a number of steps to get from an example sentence to search results and analysis of the search results:

1. Example: A user can enter a natural language example that illustrates the construction they are interested in.

2. Parse: The Alpino parser (Bouma et al. 2001, van der Beek et al. 2002) parses the example sentence.

3. Matrix: The user indicates which words of this example are crucial for the construction, and how each word should be generalised from. Based on this the parse tree of the example sentence is transformed into an XPath query.

4. Treebanks: The user can select one or more treebanks to search in.

5. Results: The XPath query is applied to the selected treebank(s) and the results are provided as a list of sentences with matches.

[8]The notation *(...) means that leaving out the parts between the round brackets yields ill-formedness; the notation (*...) means that including the part between the brackets leads to ill-formedness.

6. *Analysis:* The results can be further analysed in a graphical interface to a pivot table for properties of the nodes in the query in combination with any available metadata.

A second important feature of GrETEL is that one can upload one's own text corpus, which is then automatically parsed and made available as a treebank to search in.

MWE-Finder is part of version 5 of the web application GrETEL, available in a first version since the end of 2022.[9] Thanks to this integration, MWE-Finder has access to all GrETEL features, and supports all treebanks that are included in GrETEL as well as the possibility of uploading one's own text corpora. In Sections 4.1 through 4.4 we describe the user interface and the query generation process of MWE-Finder, as well as a number of changes we had to make in GrETEL's backend. In Section 4.2 we illustrate the use of MWE-Finder by means of a concrete example.

4.1 User interface

MWE-Finder partially mimics the structure of GrETEL's main search functionality. It distinguishes the following steps: *Canonical Form* (cf. GrETEL's *Example* step), *Treebanks*, *Results*, and *Analysis*. It currently lacks the *Parse* step and the *Matrix* step.

MWE-Finder enables the user to enter a MWE example, just like GrETEL, though it must be in the canonical form as described in Section 3. The user thereby implicitly formulates a hypothesis about the properties of this MWE. The annotations on the example specify how the system should generalise from this example, so these annotations can be seen as a different way of implementing the *Matrix* step.

The MWEs contained within DUCAME have been included in a drop-down list and are directly searchable within the MWE-Finder. The user can also enter a new MWE, provided that it complies with the conventions for MWE canonical forms (Figure 1).

As a concrete example, suppose that the user selects the canonical form (11):

(11) Iemand zal *de kat uit de boom kijken.*
someone will the cat from the tree watch.
'Someone will wait and see.'

[9] https://gretel5.hum.uu.nl

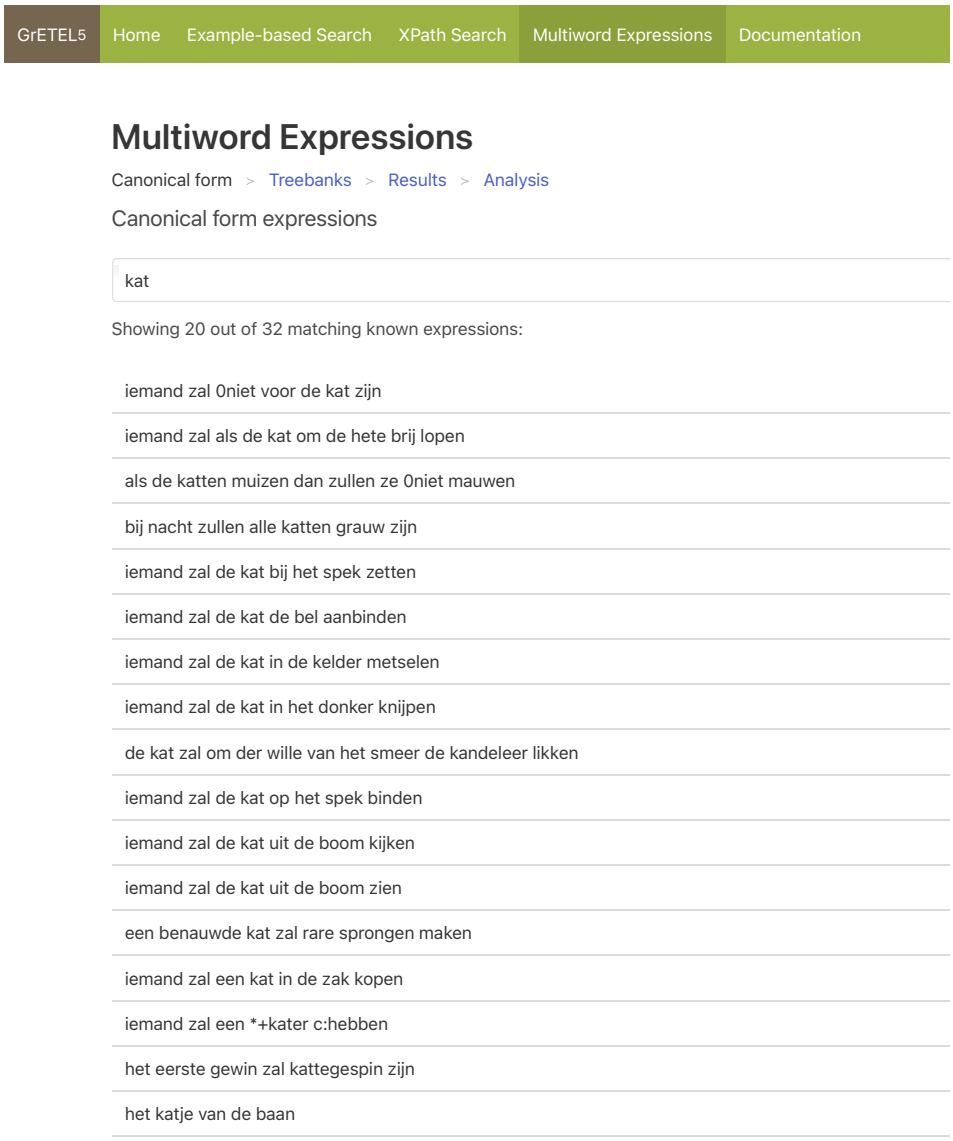

Figure 1: The first step is to choose a MWE from the DUCAME list of canonical forms or to provide a new MWE.

After the MWE has been selected or entered, the system automatically generates three queries to search for occurrences of this MWE in a treebank. They correspond to different levels of agreement between the MWE and the sentences of the corpora. These are the *major lemma query*, the *near-miss query*, and the *MWE query*.[10] The query generation process is explained in detail in Section 4.3.

Next, the user can select the treebank or treebanks that the query should be applied to. Once selected, the application switches to the *Results* view where query results are displayed as they arrive from the server. In that view, the user can also switch between the different queries for the chosen MWE or choose to exclude results of finer-grained queries. It is also possible to inspect or manually change the automatically generated XPath queries and retrieve new results (Figures 2 and 3).

In the *Results* view, users can also look at the parse trees for results or toggle extra context (one preceding sentence, one following sentence) to better analyse the occurrences found, just like in GrETEL.

Finally, there is the analysis step, which is identical to the one in GrETEL. For a MWE, one would like to analyse the result set in ways that cannot be achieved by GrETEL's standard analysis component. We are working on a special analysis step for MWEs, in which the system gathers statistics on the components of the MWE, the arguments of the MWE (their grammatical relations and syntactic categories, and their heads), the argument frames[11] that occur with the MWE, and about modifiers and determiners for the MWE as a whole and for each of its components. It does this for the results of the MWE query, for the results of the near-miss query, and for the difference between the near-miss query and the MWE query. We have an initial version available but at the time of writing it has not been integrated yet in the actual application.

4.2 Illustration

We illustrate the use of MWE-Finder with a specific example. Suppose we want to investigate the use of the MWE *de dans ontspringen* 'to get off scot-free'. The canonical form as listed in DUCAME (version 1) is in (12):

[10]Note that MWE-Finder can identify potential occurrences of a MWE in a treebank. It cannot determine for an expression that is ambiguous between a literal and an idiomatic reading which of these alternative readings is applicable in a specific sentence.

[11]With *argument frame* we mean a list of (extended relation, syntactic category) pairs for the arguments that the MWE occurs with, where an extended relation is a sequence of grammatical relations. For example, in *Marie brak Piets hart.* lit. 'Marie broke Piet's heart.', the argument frame is [(su, NP), (obj1/det, NP)], i.e., it combines with two arguments, a subject NP and a NP functioning as the determiner of the direct object.

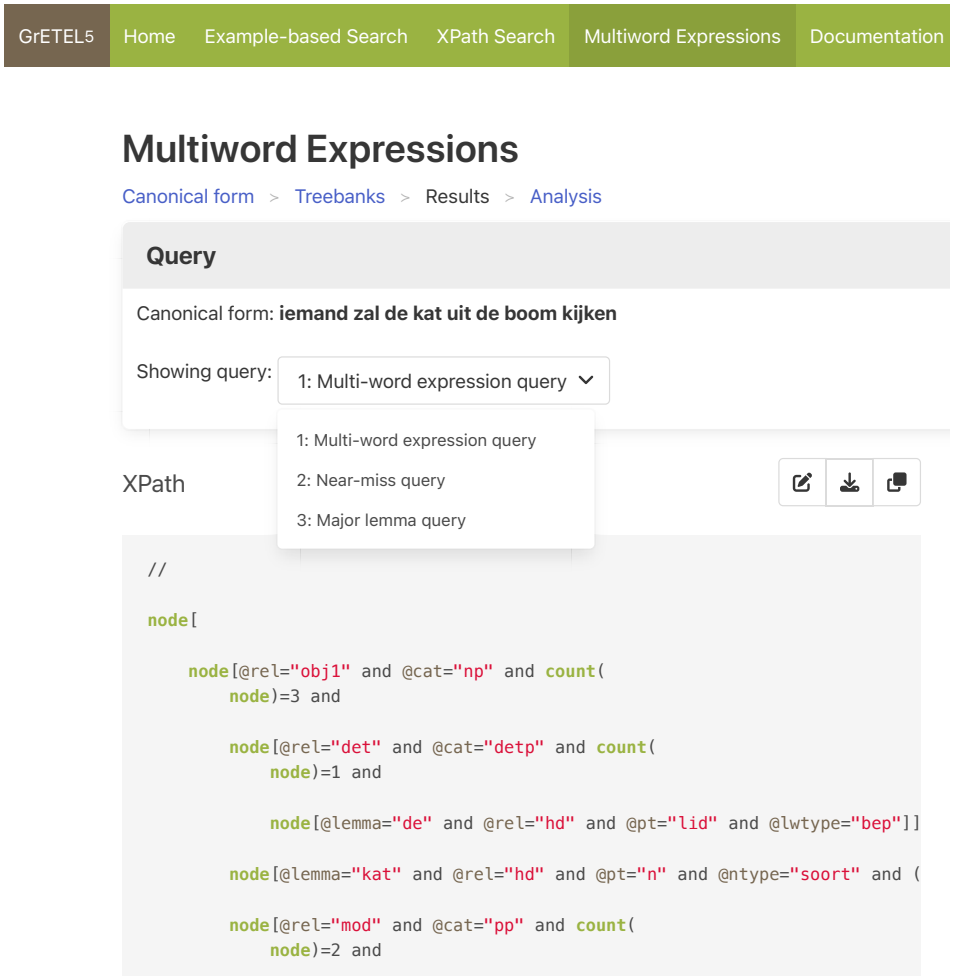

Figure 2: After selecting the treebanks to search in, the results come in and the user can switch between the three queries that are created based on the selected MWE.

Results: 12 Previous Next

#	ID	Component	Sentence
1	ep-02-05-14.data.dz:334	Year 2002	Wij moeten de kat uit de boom kijken en zien hoe de biotechnologieën zich ontwikkelen en op welke manier zij ingrijpen in de natuur van de mens .
2	ep-02-10-23.data.dz:1784	Year 2002	De leiders van Europa , Tony Blair en de Deense regering uitgezonderd , geloven dat alles zichzelf van binnenuit zal oplossen , als de diplomaten maar genoeg praatjes verkopen , als we de kat uit de boom kijken en kritiek op de VS uiten , in de hoop dat de terroristen niet toeslaan in een grote Europese stad .
3	ep-05-06-23.data.dz:375	Year 2005	De Britse premier kan nog de kat uit de boom kijken , maar de voorzitter van de Raad kan daarmee niet volstaan .
4	ep-06-05-31.data.dz:888	Year 2006	Er zijn geen nationale debatten gevoerd ; er was geen steun van de Europese instellingen , zeker niet van de Raad , die na de negatieve uitslagen van de referenda in Frankrijk en Nederland het proces stopzette en de kat uit de boom wilde kijken .

Figure 3: A sample of the results for the MWE *Iemand zal de kat uit de boom kijken.* 'Someone will wait and see.' for the Europarl corpus (Koehn 2005), part of LASSY Groot (van Noord 2008).

(12) Iemand zal de *dans ontspringen.*
someone will the dance escape
'Someone will get off scot-free.'

This canonical form is parsed by the parser in MWE-Finder, resulting in the syntactic structure in Figure 4. In this figure, we omit most attribute value pairs on each node, because there are too many to represent.

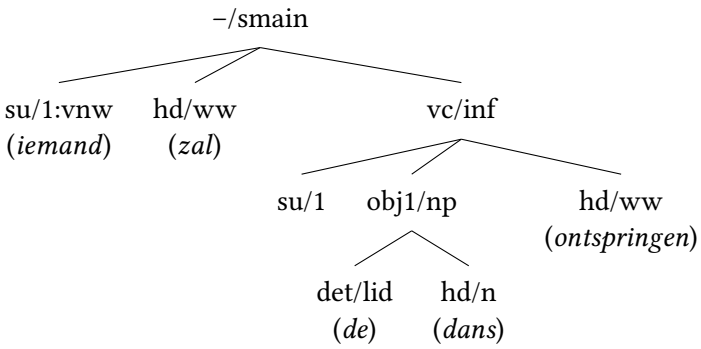

Figure 4: Syntactic structure of *Iemand zal de dans ontspringen.*

The query generation process, described in detail in Section 4.3, first converts this syntactic structure into one or more reduced syntactic structures for the MWE. For this example, there is just one such structure (see Figure 5).

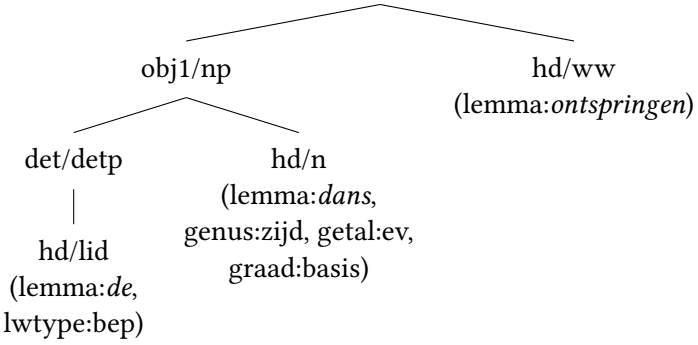

Figure 5: MWE structure of *Iemand zal de dans ontspringen.*

From this the MWE query is generated, shown in Figure 6.

```
//node[
      node[@rel="obj1" and @cat="np" and count(node)=2 and
          node[@rel="det" and @cat="detp" and count(node)=1 and
              node[@lemma="de" and @rel="hd" and @pt="lid" and
                   @lwtype="bep"]
              ] and
          node[@lemma="dans" and @rel="hd" and @pt="n" and
               @ntype="soort" and (@genus="zijd" or @getal="mv") and
               @getal="ev" and @graad="basis"]
          ] and
      node[@lemma="ontspringen" and @rel="hd" and @pt="ww"]
      ]
```

Figure 6: The MWE query for *de dans ontspringen.*

When we apply this query to the Mediargus treebank,[12] MWE-Finder finds 1158 hits in over 103 million sentences.

The near-miss query is given in Figure 7. It finds 1271 hits in the Mediargus treebank.

[12] A large treebank with Flemish newspaper text created by Kris Heylen from KU Leuven in 2009.

```
//node[
      node[@rel="obj1" and @cat="np" and
          node[@lemma="dans" and @rel="hd" and @pt="n" and
               @ntype="soort" and (@genus="zijd" or @getal="mv")]
          ] and
      node[@lemma="ontspringen" and @rel="hd" and @pt="ww"]
      ]
```

Figure 7: The near-miss query for *de dans ontspringen*.

If we exclude the results of the MWE query, which is an option offered by MWE-Finder, we quickly see in the 131 remaining hits that *de dans ontspringen* occurs in variants not predicted by the canonical form that we started with. We list some examples of phrases that the word *dans* occurs with:

different determiners:	None, *die* 'that', *zijn* 'his';
adjectival modifiers:	*gerechtelijke* 'judicial', *fiscale* 'fiscal', *politieke* 'political';
PP modifiers:	*van de bedreigden* 'of the threatened ones', *van de sociale verkiezingen* 'of the social elections'.

We also see that the NP headed by *dans* can be the object of two different coordinated verbs, which is possible (we hypothesise) because the verb *ontspringen* is used in its literal meaning in the MWE (i.e., a meaning that it also has outside of this MWE):

(13) Wie de politieke *dans* gaat leiden of *ontspringen* ...
who the political dance goes lead or escape ...
'Who will lead or escape the political unpleasant event ...'

All of this clearly suggests that the canonical form that we started with was too strict. We must allow for modification of the MWE component *dans*,[13] the article

[13]This word appears to have a metaphorical meaning in this MWE, meaning something like 'unpleasant event'.

de is not a component of the MWE,[14] and ideally we should indicate somehow that the verb *ontspringen* is used in its literal meaning.[15] A better canonical form for this MWE would be *iemand zal 0de *dans ontspringen*, which explicitly allows modification of *dans*, and explicitly states that the determiner *de* is not a component of the MWE. Indeed, the MWE query derived from this canonical form finds 1271 hits, the same number as the near-miss query for the original canonical form. In this way, we can improve upon an initial canonical form mainly based on native speaker intuitions by systematically taking into account corpus data. MWE-Finder makes this possible in a very efficient and user friendly way.

Lastly, the major lemma query (see Figure 8) finds 1309 hits. If we exclude the results of the near-miss query, we have to inspect 38 examples. These are mostly valid instances of the MWE *de dans ontspringen* that have been wrongly parsed by Alpino, but we also find a variant of the MWE, viz. (14) for which we can now add a canonical form to DUCAME.

(14) Iemand zal *aan* de *dans ontspringen.*
someone will to the dance escape
'Someone will get off scot-free.'

In this way, a linguist or lexicographer can easily and efficiently investigate the properties of Dutch MWEs, and improve the description of Dutch MWEs. This process will be even more efficient as soon as the dedicated analysis options have become available.

```
//node[@lemma="dans" and @pt="n"]
/ancestor::alpino_ds/node[@cat="top" and
 descendant::node[@lemma="ontspringen" and @pt="ww"]]
```

Figure 8: The major lemma query for *de dans ontspringen.*

4.3 Query generation

Query generation by MWE-Finder involves multiple aspects. In Section 4.3.1 we list and characterise the queries generated. In Section 4.3.2 we describe which

[14] Absence of a determiner is generally ill-formed, but this is due to the normal rules of the Dutch grammar, viz. that a singular count noun requires a determiner. This should not be described as a property of the MWE. There are examples in the treebank in which *dans* occurs without a determiner, but these are all examples from headlines which obey a different grammar.

[15] A newer version of DUCAME, not described here, has this option.

grammatical properties from the parse of the canonical form end up in the query. Section 4.3.3 lists multiple variants of the MWE structure that must be taken into account. Section 4.3.4 explains how MWE-Finder deals with left-right order. Finally, in Section 4.3.5 we describe the limitations of the approach taken.

4.3.1 Queries

The system processes an input example and interprets it as a canonical form for a MWE: it extracts the annotations and stores them in a data structure, parses the canonical form (with annotations removed) using the Alpino parser, processes any annotations on the canonical form, and then creates three queries: the *major lemma query*, the *near-miss query*, and the *MWE query*. These queries are then applied to a treebank offered by the GrETEL application and selected by the user.

The major lemma query searches for sentences in which at least the lemmas of the so-called major words of the MWE occur (in any grammatical configuration). Major words are the content words if there are at least two in the MWE, and content and function words if there is at most one content word in the MWE. The query yields a superset of the results of both other queries. This query is applied to the full treebank, making use of indexes on the treebank to speed up the process. The major lemma query yields a list of syntactic structures, and can be used to identify the MWE in a grammatical configuration that was not expected at all, to retrieve occurrences of the MWE in sentences that Alpino parsed incorrectly, or to retrieve occurrences of the MWE for which MWE-Finder did not generate the correct other two queries on the basis of the canonical form. The syntactic structures in the output of the major lemma query are adapted in ways described below. The near-miss query and the MWE query are applied to the modified output of the major lemma query.

The near-miss query searches for sentences in which the lemmas of the major words of the MWE occur in the grammatical configuration derived from the canonical form. It can find potential examples of the MWE that deviate from the canonical form provided by showing differences in forms, arguments, modification and determination. It yields a superset of the MWE query results and can be used to fine-tune the hypothesis on the MWE as encoded in the canonical form supplied by the user.

The MWE query finds sentences in which the MWE occurs. This query takes into account the hypothesis on the MWE implied by the canonical form and its annotations supplied by the user.

4.3.2 Grammatical properties

The parse tree for the canonical form contains grammatical properties for each word.[16] These include attributes for the part of speech (*pt*), for the grammatical relation the word has in the structure (*rel)*, for the lemma of the word (*lemma*), for the actual form of the word in the utterance (*word*), and for other grammatical properties, among which we distinguish three classes:

Subcategorisation properties: properties to specify a subcategory of the part of speech, e.g. is a pronoun a demonstrative pronoun or a relative pronoun, is an adposition a preposition or a postposition, is a conjunction a coordinate conjunction or a subordinate conjunction, etc.

Interpretable properties: properties that have an influence on the meaning of the utterance, e.g. is a noun singular or plural, what is the mood of the verb, what is the tense of a finite verb, etc.

Purely grammatical properties: e.g. the person and number of a finite verb, the inflectional form of an adjective, the case of a pronoun, etc.

For the inflectable words in a MWE the query will contain a condition on the lemma of the word, its part of speech and any relevant subcategorisation properties. For the uninflectable words it is tempting to formulate the condition in terms of the *word* property, but that would be ill-considered for a variety of reasons. The most important and principled reason has to do with purely grammatical properties such as (structural) *case* or inflectional properties of adjectives. The case of a direct object of a MWE component is not part of the MWE, since the word may occur in different case forms depending on the syntactic configuration, e.g. a phrase may be in nominative case when it has been turned into a subject as a result of passivisation. In MWEs consisting of an adjective and a noun the adjective gets its normal inflectional variants in plural and in definite noun phrases, as illustrated in (15):[17]

(15) a. een *vrolijk*-(*e) *Fransje*
a gay-E Frans.DIM
'a gay spark'

[16] These properties include the so-called D-COI properties (Van Eynde 2005) and various Alpino-specific properties.

[17] E stands for the adjectival *e*-suffix. *Frans* is a common Dutch name.

b. *vrolijk*-*(e) *Fransjes*
gay-E Frans.DIM.PL
'gay sparks'
c. dit *vrolijk*-*(e) *Fransje*
this gay-E Frans.DIM
'this gay spark'

A second reason for not formulating the part of the query for uninflectable words in terms of the *word* attribute is that the value of the *word* attribute is how the word actually appears in the text, including capitalisation, missing or extra diacritics, and spelling errors.

Instead, the relevant part of the query is defined in terms of lemma, part of speech, subcategorisation properties and interpretable properties.

4.3.3 Creating modified variants

Creating the query for the canonical MWE is nontrivial, since the algorithm for it must take into account all the conventions for and the annotations on the canonical form provided for the MWE. For reasons described below, modification of the structure is often required. In many cases it is necessary to generate a query that takes into account multiple variants. These variants are required in part due to properties of the Dutch language, in part due to specific properties of the structures that Alpino yields, and in part due to the difficulty of parsing natural language utterances in general. We list a few examples.

4.3.3.1 Single word phrases

For phrases that consist of a single word Alpino yields structures with a node for the word but not for the phrase.[18] This is in accordance with conventions that have been agreed upon in the consortia that have developed treebanks for Dutch (Hoekstra et al. 2003, van Noord et al. 2011), but it is a very unfortunate feature for querying, because it requires a complication or even duplication of most of the queries (Van Eynde et al. 2016: 106–107, Odijk et al. 2017, Odijk 2022); in MWE-Finder this feature is mitigated by expanding the structures of the example MWE and the structures in the major lemma query results to contain a phrasal node above single word phrases (also illustrated in Figure 10).

[18] Alpino is, despite what is stated on https://www.let.rug.nl/vannoord/alp/Alpino/, *not* a dependency parser. It is a parser that yields constituent structures with explicitly labelled dependencies. See also Odijk et al. (2017: 283–285).

4.3.3.2 Bare indexed phrases

For words and phrases that play multiple roles in an utterance Alpino yields separate nodes for each role. One of these is a node for the whole phrase (the *antecedent*), whereas the other nodes are nodes with just the property of the grammatical relation and an index attribute (*bare index nodes*). The bare index nodes are coindexed with the antecedent (have the same value for the *index* attribute). This is used for wh-movement, control of the subject of an infinitival clause, for subject and object raising, for object to subject movement in passives, and for various kinds of ellipsis. In MWE-Finder bare index nodes are replaced by their antecedent (though their *rel* attribute is retained), both in the major lemma query output structures and in the structure of the example MWE. This is essential for dealing with passivised MWEs where the object has become the subject (see item below), with raising of subject MWE components, as in (16a), and for wh-movement of MWE-components, as in (16b):

(16) a. *De laatste loodjes* zullen *het zwaarst wegen.*
the last lead.DIM.PL will the heaviest weigh
'The tail end will be the most difficult.'

b. Wiens *hart* heeft zij *gebroken*?
whose heart has she broken
'Whose heart did she break?'

The changes made for single word phrases and bare index node expansion are illustrated in Figure 9 (original parse tree) and Figure 10 (parse tree after single word phrase and bare index node expansion).

4.3.3.3 Passivisation

In passivised variants, several changes occur:

- The direct object, if there is one, is turned into a subject;[19]
- the subject is left out or turned into a phrase headed by the adposition *door* 'by';
- the verb takes on the past participle form;
- a passive auxiliary (*worden* 'be' or *zijn* 'have been') can be introduced.

[19] In Dutch it is sometimes possible to passivise an intransitive verb or a transitive verb without an object, e.g. *er wordt gedanst* 'there is dancing', *er wordt gefietst* 'people are cycling', *er wordt gebouwd* 'something is being built/there is construction going on', prompting a dummy subject *er* 'there' (cf. Broekhuis et al. 2020).

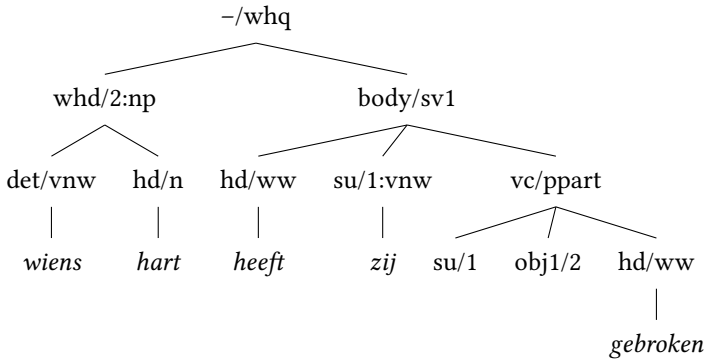

Figure 9: Parse tree of example (16b) *Wiens hart heeft zij gebroken?*. The notation *rel/i:cat* specifies a node with relation *rel*, syntactic category *cat* and index *i*. Not all nodes have an index. Bare index nodes have an index but do not dominate lexical material and have no syntactic category; here *su/1* and *obj1/2*.

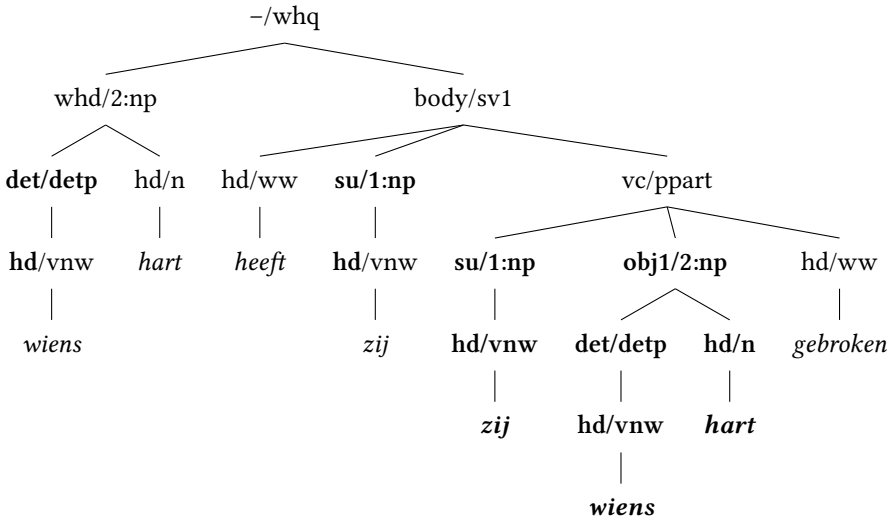

Figure 10: Parse tree of example (16b) *Wiens hart heeft zij gebroken?* after single word phrase expansion and bare index node expansion. The pronouns *wiens* and *zij* are now dominated by a phrasal node. The bare index nodes for the subject and the direct object of the participial phrase have been replaced by their antecedents. Changes are in bold face.

Example (17) illustrates this:

(17) a. *De boeken* werden door Saab *neergelegd.*
the books were by Saab down.laid
'Saab declared itself bankrupt.'

b. Er werd *met de pet naar gegooid.*
there was with the cap to thrown
'People were mucking around.'

Passive forms of MWEs can be dealt with as follows: free argument subjects are simply not part of the query, since they are not needed at all for identifying a MWE (see also below, under Subjects). Since the object bare index phrase has been replaced by its antecedent, it's easy to check whether the direct object matches the requirements (in example (17a), whether the direct object equals *de boeken*). Any verb form is accepted by the MWE query, so the past participle also matches. This leaves only the cases where a MWE with a fixed subject can be passivised: in these cases this subject must be replaced by a phrase headed by the adposition *door* in the query. Note that this also accounts for impersonal passives, of which (17b) is an example.

4.3.3.4 Definite pronouns as complements to an adposition

The definite pronouns *het* 'it', *dit* 'this', and *dat* 'dat' as a complement to a preposition are ill-formed or infelicitous. Instead of these, Dutch uses the corresponding R-pronouns (*er*, *hier*, *daar*), with the postpositional variant of the adposition. The R-pronouns precede the adposition and do in fact not have to be adjacent to it:

(18) a. * Hij *paste een mouw aan het.*
he fitted a sleeve on it

b. Hij *paste er een mouw aan.*
he fitted it a sleeve on
'He found a solution for it.'

For these cases, the query must only allow for the postpositional form of the adposition (e.g. *met* is turned into *mee*); the rest is taken care of by the Alpino grammar itself.

However, if the R-pronoun is adjacent to the adposition, it must be written as one word with the adposition according to the official Dutch spelling rules.[20]

[20]https://en.wikipedia.org/wiki/Dutch_orthography.

The simplest way to analyse this is to assume that this is a low-level orthographic convention (grounded in phonological considerations), so that it can be mostly ignored in the syntax (see Rosetta 1994: 115–116 for such an analysis).[21] But traditional grammar and Alpino deal with these words consisting of an R-pronoun and an adposition (e.g. *eraan* 'on it') as independent words with their own part of speech code, so a variant query is generated to cover these cases.

4.3.3.5 Sentential complements to an adposition

The Dutch language does not allow sentential complements to an adposition. This is illustrated in (19a), which has a NP as a complement to the adposition and is well-formed, vs. (19b), which has a subordinate clause as a complement to an adposition and is ill-formed.

(19) a. Hij *liep* *tegen* veel problemen *aan*.
he walked against many problems on
'He had to face many problems.'

b. * Hij *liep* *tegen* dat hij ziek was *aan*.
he walked against that he ill was on

c. Hij *liep* er *tegen* *aan* dat hij ziek was.
he walked it against on that he ill was
'He had to face the fact that he was ill.'

Instead, the adposition must have the R-pronoun *er* 'it' as a complement and changes into a postposition, and the sentential complement is added at the end of the clause, as in (19c). For each query that contains a free argument to an adposition, a variant taking this into account is generated. Also here, an additional variant is generated to cover the cases where *er* and the adposition are written as a single word.

4.3.3.6 Subjects

Subjects can be absent in Dutch utterances in imperative clauses, in cases of topic drop, and in impersonal passives. This is also the case for subjects of MWEs, unless the subject is or contains a fixed component of the MWE. This is illustrated in (20a) for imperatives, in (20b) for topic drop and in (17b) for impersonal passives, repeated here as (20c):

[21] The operation must be syntactic in nature because the R-pronoun and the adposition must only be written as a single word when the R-pronoun is a complement to the adposition.

(20) a. *Gooi er* niet *met de pet naar*!
throw it not with the cap towards
'Don't muck around!'

b. *Staat in de sterren geschreven.*
stands in the stars written
'That is bound to happen.'

c. Er werd *met de pet naar gegooid.*
there was with the cap to thrown
'People were mucking around.'

These are accounted for by not including any condition on the subject in the query if it is not and does not contain a fixed component of the MWE. Nonfinite clauses have no overt subject but in most cases they do have a bare index node subject in Alpino structures, so these are not relevant here.

4.3.3.7 Relativisation of MWE-parts

Components of a MWE can sometimes be relativised, especially in the case of support verb constructions, but this is certainly not always the case. Example (21a) contains an example where this is possible (for the MWE *(een) poging wagen* 'to make an attempt'), example (21b) shows an example where this is not possible (for the MWE *de plaat poetsen* 'to bolt'):[22]

(21) a. De *poging* die hij *gewaagd* had was hopeloos.
the attempt that he dared had was hopeless
'The attempt that he had made was hopeless.'

b. # *De plaat* die hij *gepoetst* had was mooi.
the plate that he polished had was beautiful
'The plate that he polished was beautiful.'

MWE-Finder replaces a relative pronoun by its antecedent. As its antecedent it takes the NP that it is contained in with the exclusion of the relative clause[23] but with the addition of an abstract dummy modifier. The antecedent of the relative pronoun *die* 'that' is therefore *de dummy poging* in (21a), and *de dummy plaat* in (21b). The presence of the dummy modifier now ensures that relativisation is only allowed when the component of the MWE can be modified (which is the case for *poging* in *een poging wagen*, but not for *plaat* in *de plaat poetsen*).

[22]The symbol # means that the idiomatic reading is not possible.
[23]This is done to avoid an infinite recursion.

4.3.3.8 NP PP sequences

In expressions in which a noun phrase (NP) is immediately followed by an adpositional phrase (PP) the PP can be a sibling or a child of the NP. Alpino resolves this ambiguity sometimes by selecting the PP as child option, sometimes by selecting the PP as a sibling option. The choice is dependent on several factors, among which the nature of the complement in the PP. In (22) the PP *van iets* is analysed as a child of the NP node dominating *de schuld*, while in (23) the (discontinuous) PP *daar ... van* is a sibling of the NP *de schuld*. We indicated this by means of square brackets in these examples.

(22) Iemand zal iemand [*de schuld* [*van* iets]] *geven.*
someone will someone the blame of something give
'Someone will put the blame for something on someone.'

(23) Iemand zal iemand [daar] [*de schuld*] [*van*] *geven.*
someone will someone there the blame of give
'Someone will put the blame for that on someone.'

For this reason, an alternative structure is generated for nodes headed by a verb that contain an NP which in turn contains a PP. In this alternative structure, the PP is a sibling of the NP.

It is not enough to generate this alternative only for the structure of the MWE that the query is derived from. It must also be applied to the structure of each sentence queried, i.e. for PPs that can be part of the MWE. For example, for the variants *Iemand zal iemand daar van de schuld geven* (with a space between *daar* and *van*) and *Iemand zal iemand van iets de schuld geven*, the PPs *daar van* and *van iets* are analysed as modifiers of the immediately preceding *iemand*, which would lead to a mismatch with the query for the expression *iemand van iets de schuld geven*, as indicated in (24).

(24) a. Iemand zal [iemand [daar *van*]] de *schuld geven.*
Someone will [someone [there of]] the blame give.
'Someone will put the blame for something on someone.'

b. Iemand zal [iemand [*van* iets]] de *schuld geven.*
Someone will [someone [of something]] the blame give.
'Someone will put the blame for something on someone.'

4.3.3.9 Adpositional phrases

Adpositional phrases to a verb can get different analyses in Alpino: as a predicative complement, as a locational-directional complement, as an adpositional complement, as a modifier, or as a secondary predicate. The choice is in part dependent on the verb that selects them, but, in the case of ambiguities, also dependent on the disambiguation strategy of Alpino (van Noord 2006), for which it is not easy to predict which selection is made. For PPs dependent on a verb the query is therefore relaxed to accept any of these grammatical relations.

4.3.3.10 Secondary predicates

Modifiers in a clause with a verb cluster are always analysed as modifiers of the deepest embedded verb. However, secondary predicates are always analysed as modifiers of the least embedded verb. If an expression such as (25) with the secondary predicate *als een ketter* is embedded under an auxiliary verb such as *hebben*, as in (26), the phrase *als een ketter* is analysed by Alpino as a modifier to the verb *heeft*, and the MWE will not be found:

(25) Hij *rookt als een ketter.*
he smokes like a heretic
'He smokes like a chimney.'

(26) Hij heeft altijd *gerookt als een ketter.*
he has always smoked as a heretic
'He has always smoked like a chimney.'

In order to avoid this problem, a special operation is applied to move the secondary predicate to become a modifier of the deepest embedded verb, both in the structures that lead to the query and in the structures of the sentences being queried.

4.3.4 Left-right order

The queries that are generated generally do not check for the left-right order of the components of the MWE, its arguments or modifiers. This is desired since the order of these elements is in most cases not a property of the MWE but follows from the grammar of the language. For this reason MWE-Finder can easily identify the different expressions in (1) as instantiations of the same MWE. Dutch has words that in some cases must be used as a preposition (preceding its complement) and in other cases as a postposition (following its complement), but

even this does not require conditions on order since the distinction is marked by a grammatical feature. Thus, MWE-Finder, without restrictions on left-right order, will correctly not identify (27) as containing the MWE *op de klippen lopen* 'to fail', though it will identify (28) as such:

(27) Dat zal de klippen op lopen.
that will the cliffs on walk
'That will walk onto the cliffs.' (Not: 'That will fail.')

(28) Dat zal *op de klippen lopen.*
that will on the cliffs walk
'That will walk on the cliffs.' And: 'That will fail.'

There surely are some MWEs in which the left-right order is a property of the MWE, especially in coordinate structures, e.g. as in (29), but at the time of writing we did not yet implement such restrictions.

(29) a. dag en nacht
day and night
'during night and day'
b. # nacht en dag
c. dames en heren
ladies and gentlemen
'ladies and gentlemen'
d. # heren en dames

There are also restrictions on left-right order that hold for MWEs but not for literal constructions. For example, *de plaat* in (30) can not be clause-initial under the idiomatic reading though it can be under the literal reading:

(30) # De plaat heeft hij niet gepoetst.
the plate has he not polished
'He did not polish the plate.' (Not: 'He bolted.')

We did not yet implement such restrictions. We believe that many such restrictions can be dealt with systematically but whether that turns out to be the case still remains to be investigated.

4.3.5 Limitations

MWE-Finder is fully dependent on the syntactic structures generated by the Alpino parser. If Alpino cannot parse a sentence correctly, MWE-Finder will not be able to identify any MWE in it. This is one of the reasons why MWE-Finder includes the major lemma query: this query will find sentences in which the MWE occurs even if Alpino cannot parse it correctly, so a researcher still has data to work with.[24] However, this query will also find many sentences in which the MWE does not occur, so it will require more manual work by the researcher. The amount of work is significantly reduced by the option to select the results of a query minus the results of a stricter query, as we showed in Section 4.2. We aim to reduce the amount of manual work required even more by providing statistics on the results and the results minus the results of the other two queries in the dedicated MWE analysis step. In particular, it will provide statistics on the grammatical relation between the lemmas of the major words. However, at the time of writing this has not been integrated in the online version yet.

Alpino may analyse a sentence incorrectly for a wide variety of reasons. One possibility is that the sentence contains a construction that Alpino cannot handle. For example, in the sentence *Hoe goed Afrikaanse muzikanten ook zijn, aan de bak komen ze nauwelijks.* 'Good though African musicians may be, they hardly get jobs.',[25] Alpino can only correctly parse the part *Hoe goed Afrikaanse muzikanten ook zijn*, but it cannot connect it to the rest of the sentence, and as a consequence the MWE *aan de bak komen* 'to get a job, get a turn' is incorrectly not identified in this sentence.

MWE-Finder also currently fails to find a MWE if Alpino does not know a word and cannot correctly guess its properties. For example, the word *velen* can be a verb ('tolerate') or a pronoun ('many persons'). As a verb it can only occur in the expression *iets (niet) kunnen velen* 'not be able to stand something'. Alpino does not know this verb and analyses (31) incorrectly as consisting of a full main clause *hij kan dat niet* 'he cannot do that' followed by single word phrase headed by the pronoun *velen* 'many persons'. Similarly, Alpino does know the verb *smeren*, but only in the sense of 'to butter'. MWE-Finder can therefore find *smeerde 'm* in (32) when looking for instances of the MWE *'m smeren* 'to bolt', because it looks for instances of the verb *smeren* with an object *'m* 'him'. The problem is that the verb *smeren* 'to butter' forms its perfect tense with the auxiliary verb *hebben*, while the verb *smeren* in the MWE *'m smeren* forms its perfect tense with the auxiliary

[24] Assuming Alpino can at least lemmatise all major words correctly.

[25] Twente News Corpus (Ordelman et al. 2007), component ad1999, sentence with identifier ad19990108.data.dz:1831 in GrETEL.

verb *zijn*, as illustrated in (33). The result is that Alpino cannot correctly analyse (33), and MWE-Finder cannot identify it as an occurrence of the MWE *'m smeren*.

(31) Hij *kan* dat niet *velen*.
he can that not stand
'He can't stand it.'

(32) Hij *smeerde 'm*.
he buttered him
'He bolted.'

(33) Hij is *'m gesmeerd*.
he is him buttered
'He has bolted.'

4.4 Changes under the hood

Under the hood, the backend of GrETEL was largely rewritten to make it more flexible. The existing PHP backend of GrETEL 4 was migrated to Python in combination with the Django framework for web applications,[26] which gives us better support for asynchronous tasks and better run-time resource management. This allowed us to improve performance and to better support large corpora and complex queries. The existing Angular frontend of GrETEL 4 was modified to communicate with the new backend and expanded with a new functionality for the MWE-Finder.[27]

Support for large text corpora is important in the context of MWEs, because word frequencies in natural language have a Zipfian distribution, so that most of the words occurring in the MWEs have very low frequencies. GrETEL 5 still takes a considerable amount of time to search entire corpora, but does so in the background and will cache the counts and results for further usage. We have prepared several existing large corpora for usage in GrETEL, including LASSY Groot (van Noord 2008),[28] which includes the 500-million-word SoNaR corpus (Oostdijk et al. 2013), a Wikipedia dump, and the TwNC, a multifaceted Dutch news corpus (Ordelman et al. 2007). GrETEL 5 ships with import scripts for these corpora.

The principles of the existing search mechanism of GrETEL have been retained in GrETEL 5, and they also largely form the basis of how the MWE-Finder is integrated into the application, but with certain deviations. In GrETEL, the corpora

[26] https://www.djangoproject.com/
[27] https://angular.io/
[28] https://taalmaterialen.ivdnt.org/download/tstc-lassy-groot-corpus/

are stored in XML format as they were parsed by the Alpino parser[29] in databases of BaseX (Grün 2010),[30] a database system for XML documents. GrETEL translates queries created by the user into XQuery/XPath queries that can be executed by BaseX. This search process is relatively slow compared to other search methods, but searching syntactical structures is not possible using common search methods such as simple full text search.

The most important deviation entails that when searching for a MWE, the BaseX databases are always searched using the major lemma query, while the other two queries are executed with the search results of the major lemma query as its basis. The main reason for this is that MWE queries result in complex nested XPath expressions which are not fully optimised by BaseX's query planner.

On the contrary, a major lemma query contains only a handful of content words and makes good use of the indices that BaseX creates for XML attributes. This means that results for a major lemma query can be efficiently retrieved. The results of the major lemma query, which contain all potential matches for the requested MWE, can then be reused for resolving the other more specific queries.

Another reason for searching MWEs based on the major lemma query is that it is necessary to manipulate the Alpino parse trees of the corpora before the other two queries can be run. Those additional manipulations are needed because of the considerations detailed in Section 4.3.3. Such processing steps would be too expensive computationally to run on entire corpora, and are instead run on the result set of the relevant major lemma query. The latter is of a substantially smaller scale. These processing steps are carried out in-memory using the `lxml` Python library.[31] The final step is to use the queries to search in the manipulated parse trees, which is done using `lxml`, as well, thanks to its XPath engine.

Finally, structuring MWE queries around a major lemma query allows query results to be cached, providing a more fluent interactive workflow for users. The user does not see anything of this tiered approach, and instead simply sees the results for the selected MWE and type of query, and can quickly switch between them.

GrETEL is open source and its code is available at GitHub.[32] The part of the application that generates queries for MWEs and that performs the tree manipulation is available as a separate Python package, so that it may also be used to create scripts that search treebanks without using GrETEL.[33]

[29] These are in accordance with the alpino_ds DTD.

[30] https://docs.basex.org/wiki/Main_Page. GrETEL uses BaseX version 9.

[31] https://lxml.de/

[32] https://github.com/UUDigitalHumanitieslab/gretel

[33] https://github.com/UUDigitalHumanitieslab/mwe-query

5 Other languages

We presented MWE-Finder for the Dutch language, integrated in a specific treebank query application (GrETEL) for the Dutch language, which uses a specific grammar and parser for the Dutch language (Alpino). However, it is not difficult to make similar systems for other languages. The minimum requirements to make a variant for a different language are a parser for that language, and a query system that can query the kind of structures that the parser yields. A system for a different language could even be integrated in GrETEL, because GrETEL is in itself not bound to any particular language, as shown by the GrETEL variant for Afrikaans (Augustinus et al. 2016a), and Poly-GrETEL (Augustinus et al. 2016b), which enabled simultaneous querying in multiple languages in a parallel treebank.[34]

Moreover, a MWE-Finder for a different language and a different parser has to have a query generation procedure. The procedure described in §4.3 is to a large extent generic, though of course it has some aspects that are specific to the Dutch language or to the specific parser used. In MWE-Finder, the treatment of single word phrases and the treatment of secondary predicates is entirely idiosyncratic to the parser used. Some aspects are entirely specific to Dutch (definite pronouns and sentential complements to an adposition), though surely each language will have its own peculiarities, even if one would use a cross-language framework for grammatical structures such as the Universal Dependencies framework (Nivre et al. 2016).[35] Other aspects will have to be addressed in any grammar/parser but may be implemented in a completely different way in different grammars/-parsers. Displacement and control phenomena (with Alpino using bare indexed phrases), passivisation (with Alpino having displaced objects) are concrete examples. But most aspects are completely generic: the treatment of the grammatical properties (§4.3.2), the modified variants (§4.3.3), subjects, relativisation of MWE-parts, NP PP sequences, adpositional phrases, and left-right order are relevant for any language.

In summary, the implementation of MWE-Finder sets an excellent example for the implementation of similar systems for different languages and parsers.

6 Conclusions

We presented the DUCAME resource and the MWE-Finder as useful research instruments for linguistic and lexicological research into MWEs. MWE-Finder

[34] http://gretel.ccl.kuleuven.be/poly-gretel/index.php.

[35] https://universaldependencies.org/

makes it possible to reliably and quickly search for occurrences of a MWE despite their flexible nature. The search is based on an example in an annotated canonical form. The system searches not only for the MWE, but also generates and executes two more relaxed queries: the results of the *near-miss query* and especially the difference between the results of the *near-miss query* and the *MWE query* are very useful for evaluating the implicit hypothesis on the nature of the MWE as formulated in the annotated canonical form, and for adjusting it if needed. The *major lemma query*, and especially the difference between the results of this query and the other two enable the user to find occurrences of MWEs that one might not have expected at all, and also acts as a fall back option for cases in which Alpino parses the sentence containing a MWE incorrectly, or if MWE-Finder does not generate the correct other queries from the canonical form.

7 Future work

We aim to finalise the work on the dedicated MWE analysis component and to integrate it in the online application.

We also plan to experiment with a different indexing system than BaseX for the major lemma query. This query searches for a set of lemmas irrespective of their grammatical relation, so it is not necessary to use an index system for this query that can deal with very complex XPath expressions. One of the indexing systems we want to experiment with is Solr/Lucene,[36] which has also proven very efficient in OpenSoNaR (de Does et al. 2017) and in Nederlab (Brouwer et al. 2016).

The software behind the system can easily be converted to software to annotate a large corpus for MWEs, and enrich the treebank with metadata on MWE occurrences. We intend to make this software and apply it on a large corpus (e.g., the LASSY-Groot Newspaper corpus; van Noord et al. 2013). We will also write software for converting the metadata on MWE occurrences in the CoNLL-U and Parseme-tsv formats as proposed in the PARSEME consortium.[37] This can then form the basis for the manual verification of these annotations and in particular adding missing annotations, and the resulting data may be relevant for a wide range of natural language processing tools dealing with MWEs.

We furthermore aim to extend the annotation system for the canonical forms with special annotations for collocations and support verb constructions, and to extend MWE-Finder so that it can deal with these new annotations.

[36] https://lucene.apache.org/ and https://solr.apache.org/

[37] https://universaldependencies.org/format.html and https://typo.uni-konstanz.de/parseme/index.php/2-general/184-parseme-shared-task-format-of-the-final-annotation

Finally, there is a small number of MWEs that are currently not dealt with correctly with the canonical forms we currently use. We aim to investigate how we can adapt this.

Acknowledgements

The research described in this chapter was carried out in the Datahub SSH project funded by Utrecht University. We thank the anonymous reviewers for their comments, which led to a significant improvement of the original text.

Acronyms and Abbreviations

BaseX	index system for XML documents (index system)
body	relation of the clause in a wh-question (Alpino grammatical relation)
cat	syntactic category (Alpino attribute)
CONLL-U	Computational Natural Language Learning format version U (text corpus format)
det	determiner (Alpino grammatical relation)
detp	determiner phrase (Alpino syntactic category)
DIM	diminutive (grammatical category)
DUCAME	Dutch Canonicalised Multiword Expressions (lexical resource)
DuELME	Dutch Electronic Lexicon of Multiword Expressions (lexical resource)
E	Dutch *e*-suffix on adjectives (suffix)
GrETEL	Greedy Extraction of Trees for Emprical Linguistics (application)
hd	head (Alpino grammatical relation)
inf	infinitive phrase (Alpino syntactic category)
lid	article (Alpino part of speech code)
lxml	Python module to deal with XML (Python module)
MWE	multiword expression (term)
n	noun (Alpino part of speech code)
np	noun phrase (Alpino syntactic category)
NP	noun phrase (syntactic category)
obj1	direct object (Alpino grammatical relation)
PARSEME	Parsing and Multiword Expressions (project)
PP	adpositional phrase (syntactic category)
ppart	past participle phrase (Alpino syntactic category)
pt	part of speech (Alpino attribute)
rel	grammatical relation (Alpino attribute)

R-pronoun	Dutch pronoun from a particular set, each of which contains an *r* in it (word class)
smain	main clause (Alpino syntactic category)
su	subject (Alpino grammatical relation)
sv1	Verb-initial clause (Alpino syntactic category)
top	top relation (Alpino grammatical relation)
tsv	tab-separated value file (file format)
TwNC	Twente News Corpus (text corpus)
vc	verbal complement (Alpino grammatical relation)
vnw	pronoun (Alpino part of speech code)
VRT	Vlaamse Radio en Televisie 'Flemish Radio and Television' (broadcast organisation in Belgium)
whd	relation of fronted wh-phrase in a question (Alpino grammatical relation)
whq	main wh-question (Alpino syntactic category)
ww	verb (Alpino part of speech code)
XML	eXtensible Mark-up Language (mark-up language)
Xpath	query language for XML-documents (query language)
Xquery	programming language (programming language)

References

Augustinus, Liesbeth, Peter Dirix, Daniel Van Niekerk, Ineke Schuurman, Vincent Vandeghinste, Frank Van Eynde & Gerhard Van Huyssteen. 2016a. AfriBooms: An online treebank for Afrikaans. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the tenth international conference on Language Resources and Evaluation (LREC 2016)*, 677–682. Portorož, Slovenia: European Language Resources Association (ELRA).

Augustinus, Liesbeth, Vincent Vandeghinste & Frank Van Eynde. 2012. Example-based treebank querying. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the eighth international conference on Language Resources and Evaluation (LREC 2012)*, 3161–3167. Istanbul, Turkey: European Language Resources Association (ELRA).

Augustinus, Liesbeth, Vincent Vandeghinste, Ineke Schuurman & Frank Van Eynde. 2017. GrETEL: A tool for example-based treebank mining. In Jan Odijk & Arjan van Hessen (eds.), *CLARIN in the Low Countries*, chap. 22, 269–280. London, UK: Ubiquity. DOI: 10.5334/bbi.22.

Augustinus, Liesbeth, Vincent Vandeghinste & Tom Vanallemeersch. 2016b. Poly-GrETEL: Cross-lingual example-based querying of syntactic constructions. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the tenth international conference on Language Resources and Evaluation (LREC 2016)*, 3549–3554. Portorož, Slovenia: European Language Resources Association (ELRA).

Bouma, Gosse, Gertjan van Noord & Robert Malouf. 2001. Alpino: Wide-coverage computational analysis of Dutch. *Language and Computers* 37(1). 45–59.

Broekhuis, Hans, Norbert Corver & Riet Vos. 2020. The impersonal passive. In *Taalportaal*. https://taalportaal.org/taalportaal/topic/link/syntax__Dutch__vp__V3_alternations__V3_alternations.3.2.1.2.xml.

Brouwer, Matthijs, Hennie Brugman & Marc Kemps-Snijders. 2016. A SOLR / Lucene based multi tier annotation search solution. In *Selected papers from the CLARIN annual conference 2016, 26–28 October, Aix-en-Provence*, 29–37. Linköping, Sweden: Linköping University Electronic Press. https://ep.liu.se/ecp/article.asp?issue=136&article=002&volume=0.

Brugman, Hennie, Martin Reynaert, Nicoline van der Sijs, René van Stipriaan, Erik Tjong Kim Sang & Antal van den Bosch. 2016. Nederlab: Towards a single portal and research environment for diachronic Dutch text corpora. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the tenth international conference on Language Resources and Evaluation (LREC 2016)*, 1277–1281. Portorož, Slovenia: European Language Resources Association (ELRA).

Constant, Mathieu, Gülşen Eryiğit, Johanna Monti, Lonneke van der Plas, Carlos Ramisch, Michael Rosner & Amalia Todirascu. 2017. Multiword expression processing: A survey. *Computational Linguistics* 43(4). 837–892. DOI: 10.1162/COLI_a_00302.

de Does, Jesse, Jan Niestadt & Katrien Depuydt. 2017. Creating research environments with BlackLab. In Jan Odijk & Arjan van Hessen (eds.), *CLARIN in the Low Countries*, 245–257. London, UK: Ubiquity Press. DOI: 10.5334/bbi.20.

Grégoire, Nicole. 2009. *Untangling multiword expressions: A study on the representation and variation of Dutch multiword expressions*. Utrecht: Utrecht University. (Doctoral dissertation).

Grégoire, Nicole. 2010. DuELME: A Dutch electronic lexicon of multiword expressions. *Language Resources and Evaluation* 44(1/2). 23–39. DOI: 10.1007/s10579-009-9094-z.

Grün, Christian. 2010. *Storing and querying large XML instances*. University of Konstanz. (Doctoral dissertation).

Hoekstra, Heleen, Michael Moortgat, Bram Renmans, Machteld Schouppe, Ineke Schuurman & Ton van der Wouden. 2003. *CGN Syntactische Annotatie*. CGN report. Utrecht, the Netherlands: Utrecht University. http://lands.let.kun.nl/cgn/doc_Dutch/topics/version_1.0/annot/syntax/syn_prot.pdf.

Koehn, Philipp. 2005. Europarl: A parallel corpus for statistical machine translation. In *Proceedings of the 10th machine translation summit*, 79–86. Phuket, Thailand.

Nivre, Joakim, Marie-Catherine de Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, Reut Tsarfaty & Daniel Zeman. 2016. Universal dependencies v1: A multilingual treebank collection. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Helene Mazo, Asuncion Moreno, Jan Odijk & Stelios Piperidis (eds.), *Proceedings of the tenth international Conference on Language Resources and Evaluation (LREC 2016)*, 1659–1666. Portorož, Slovenia: European Language Resources Association (ELRA).

Odijk, Jan. 2013a. DUELME: Dutch electronic lexicon of multiword expressions. In Gil Francopoulo (ed.), *LMF: Lexical markup framework*, 133–144. London, UK / Hoboken, US: ISTE / Wiley.

Odijk, Jan. 2013b. Identification and Lexical Representation of Multiword Expressions. In P. Spyns & J. E. J. M. Odijk (eds.), *Essential Speech and Language Technology for Dutch: Results by the STEVIN-programme* (Theory and Applications of Natural Language Processing), 201–217. Berlin/Heidelberg: Springer.

Odijk, Jan. 2022. Eenwoordsconstituenten in GrETEL. In *Liber amicorum Francisci Affinii alias Frank Van Eynde*, 143–150. Leuven, Belgium: KU Leuven.

Odijk, Jan, Martijn van der Klis & Sheean Spoel. 2018. Extensions to the GrETEL Treebank Query Application. In *Proceedings of the 16th international workshop on Treebanks and Linguistic Theories (TLT16)*, 46–55. Prague. http://aclweb.org/anthology/W/W17/W17-7608.pdf.

Odijk, Jan, Gertjan van Noord, Peter Kleiweg & Erik Tjong Kim Sang. 2017. The parse and query (PaQu) application. In Jan Odijk & Arjan van Hessen (eds.), *CLARIN in the Low Countries*, chap. 23, 281–297. London, UK: Ubiquity Press. DOI: 10.5334/bbi.23.

Oostdijk, Nelleke, Martin Reynaert, Véronique Hoste & Ineke Schuurman. 2013. The construction of a 500 million word reference corpus of contemporary written Dutch. In Peter Spyns & Jan Odijk (eds.), *Essential speech and language technology for Dutch: Results by the STEVIN-programme*, 219–247. Berlin: Springer. DOI: 10.1007/978-3-642-30910-6_13.

Ordelman, Roeland J.F., Franciska M.G. de Jong, A. J. van Hessen & G. H. W. Hondorp. 2007. TwNC: a multifaceted Dutch news corpus. *ELRA Newsletter* 12(3-4).

Rosetta, M. T. 1994. *Compositional Translation* (Kluwer International Series in Engineering and Computer Science 273). Dordrecht: Kluwer.

Sag, Ivan A., Timothy Baldwin, Francis Bond, Ann Copestake & Dan Flickinger. 2002. Multiword expressions: A pain in the neck for NLP. In Alexander F. Gelbukh (ed.), *Proceedings of the third international conference on Intelligent Text Processing and Computational Linguistics (CICLing 2002)*, 1–15. Springer.

Stoett, Frederik August. 1923. *Nederlandsche spreekwoorden, spreekwijzen, uitdrukkingen en gezegden.* 4th edn. Zutphen: W.J. Thieme & Cie. https://www.dbnl.org/tekst/stoe002nede01_01/.

van de Camp, Matje, Martin Reynaert & Nelleke Oostdijk. 2017. WhiteLab 2.0: A web interface for corpus exploitation. In Jan Odijk & Arjan van Hessen (eds.), *CLARIN in the Low Countries*, 231–243. London, UK: Ubiquity Press. DOI: 10.5334/bbi.19.

van der Beek, Leonoor, Gosse Bouma & Gertjan van Noord. 2002. Een brede computationele grammatica voor het Nederlands. *Nederlandse Taalkunde* 7. 353–374.

Van Eynde, Frank. 2005. *Part Of Speech Tagging En Lemmatisering van het D-COI Corpus.* LASSY Report. Leuven, Belgium: Centrum voor Computerlinguïstiek, KU Leuven. http://www.let.rug.nl/vannoord/Lassy/POS_manual.pdf.

Van Eynde, Frank, Liesbeth Augustinus & Vincent Vandeghinste. 2016. Number agreement in copular constructions: A treebank-based investigation. *Lingua* 178. 104–126. DOI: 10.1016/j.lingua.2016.02.001.

van Noord, Gertjan. 2006. At last parsing is now operational. In P. Mertens, C. Fairon, A. Dister & P. Watrin (eds.), *Actes de la 13ème conférence sur le Traitement Automatique des Langues Naturelles (TALN 2006)*, 20–42. Leuven, Belgium: ATALA. https://aclanthology.org/2006.jeptalnrecital-invite.2.

van Noord, Gertjan. 2008. Huge parsed corpora in LASSY. In Frank van Eynde, Anette Frank, Koenraad de Smedt & Gertjan van Noord (eds.), *Proceedings of the seventh international workshop on Treebanks and Linguistic Theories (TLT 7)* (LOT Occasional Series 12), 115–126. Groningen: LOT.

van Noord, Gertjan, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang & Vincent Vandeghinste. 2013. Large scale syntactic annotation of written Dutch: Lassy. In Peter Spyns & Jan Odijk (eds.), *Essential speech and language technology for Dutch* (Theory and Applications of Natural Language Processing), 147–164. Berlin: Springer. DOI: 10.1007/978-3-642-30910-6_9.

van Noord, Gertjan, Ineke Schuurman & Gosse Bouma. 2011. *Lassy syntactische annotatie (Revision 19455)*. LASSY Report. Groningen. https://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf.