CHAPTER 23

The Parse and Query (PaQu) Application

Jan Odijk, Gertjan van Noord, Peter Kleiweg and Erik Tjong Kim Sang
Utrecht University – RU Groningen – Meertens Institute, j.odijk@uu.nl

ABSTRACT

In this chapter we describe the web application PaQu (Parse and Query), and carry out a small case study to illustrate its use. PaQu is an application for searching in Dutch treebanks and for analysing the search results. One can search in the LASSY and CGN treebanks, or upload one's own Dutch corpus, which is then parsed and made available for search and analysis. PaQu offers, next to an interface to formulate Xpath queries, a dedicated interface for searching for dependency triples. This makes it easy to search in treebanks for grammatical dependencies, which would otherwise require very complex queries. It offers extensive functionality for analysing the search results. The dedicated search interface makes PaQu a prime example of the kind of applications that CLARIN promotes. The case study provides an analysis of the syntactic selectional differences between two near-synonymous verbs.

23.1 Introduction

In this chapter we describe the web application PaQu[1] (Parse and Query), and carry out a small case study to illustrate its use. PaQu is an application for searching in treebanks (i.e. text corpora in which each sentence has been assigned a syntactic structure) and for analysing the search results. PaQu was developed by the University of Groningen in the CLARIN-NL project and is an extension of the LASSY Dependency Relations Search application originally developed by Tjong Kim Sang et al. (2010). Here is a direct link to the application and its documentation.

[1] This chapter contains many hyperlinks. In electronic versions they are easily visible by special marking (blue colour). In the printed black and white version the hyperlinks may be visible as grey text. The relevant URLs are explicitly listed in the appendix.

How to cite this book chapter:
Odijk, J, van Noord, G, Kleiweg, P and Tjong Kim Sang, E. 2017. The Parse and Query (PaQu) Application. In: Odijk, J and van Hessen, A. (eds.) *CLARIN in the Low Countries*, Pp. 281–297. London: Ubiquity Press. DOI: https://doi.org/10.5334/bbi.23.

PaQu has been used for carrying out research, inter alia for an analysis of the Dutch intensifiers *heel, erg, zeer* (each meaning 'very') by Odijk (2015a) and Odijk (2016), and for an analysis of normative and non-normative variants of Dutch constructions by Odijk (2015b) and Van Noord and Odijk (2016). It is also heavily used in the Taalportaal, a digital grammar for Dutch (Van der Wouden et al., 2016), where many of the links with queries for corpus examples use the PaQu interface (see Bouma et al. (2015), Van der Wouden et al. (2015) and chapter 24).

PaQu offers three types of functionality: corpus management (described in section 23.2), search (described in section 23.3), and analysis (described in section 23.4). Section 23.5 contains an analysis of the predicative complements of the near-synonymous copular verbs *worden* 'become' and *raken* 'get' as a small case study. Section 23.6 summarises our conclusions and sketches future work. The appendix contains a list of all hyperlinks.

23.2 Corpus Management

PaQu enables one to select a treebank to search in from the publicly available Dutch treebanks in PaQu. These include LASSY-Small, the Wikipedia part of LASSY-Large (Van Noord et al., 2013), and the treebank of the Spoken Dutch Corpus (Oostdijk et al., 2002). One can also upload one's own Dutch text corpus.[2] One's own corpora can be kept private or shared with others. Login is required for managing one's corpora, since the user wants to see his/her own corpora but not someone else's (unless they are shared).

PaQu accepts as input plain text (in multiple varieties), which it then parses using Alpino (Van der Beek et al., 2002), or a text corpus already parsed by Alpino in the LASSY XML format. After this, the treebank is available for search and analysis. New input formats are being added (see section 23.6).

PaQu offers full parses of sentences in one's own corpus, but these parses are generated in a fully automatic manner, so they inevitably will contain errors. It is therefore required to evaluate the quality of the automatically generated parses.[3] Odijk (2015a) describes the results of such an evaluation for the words *heel*, *erg* and *zeer* in the CHILDES Van Kampen subcorpus, and he concludes that PaQu is excellent for carrying research on these phenomena, since the accuracy of Alpino for these data is in part very high (accuracy well over 90%), and the part where the accuracy is low is easily identifiable with PaQu. Though these results cannot be automatically generalised to other cases, they show that PaQu can be very useful even in cases where Alpino's accuracy is low, provided that the problematic cases can be automatically identified with PaQu.

Finally, PaQu enables the user to store the results of a query in a new corpus.

23.3 Search

PaQu enables search for utterances in the treebank by selecting for properties in the syntactic structures of these utterances. Before discussing this in more detail in section 23.3.2, we will say some words on the nature of the syntactic structures in the treebanks that PaQu deals with (section 23.3.1). We conclude with a description of the search results in section 23.3.3.

---

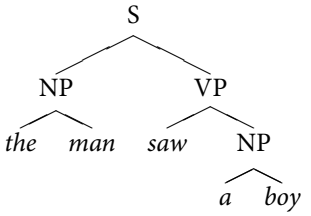
[2] This raises the question of how long these user corpora are retained, as one of the anonymous reviewers noted. On 18 May 2016, PaQu hosted 47 user corpora for 34 different users, and so far there are no problems in hosting these. Ultimately, the retention policy will be determined by the CLARIN Centre that will host PaQu.

[3] See Bloem (2016) for general strategies to evaluate automatically generated parses in the absence of a gold standard.

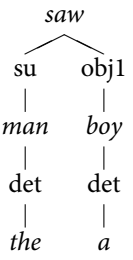
23.3.1 Syntactic Structures in Dutch Treebanks

The character of the syntactic structures in the treebanks in PaQu is in accordance with the *de facto* standard for syntactic structures in treebanks for the Dutch language (Hoekstra et al., 2003; Van Noord et al., 2011), defined in projects for the construction of Dutch treebanks such as CGN (for spoken Dutch; Oostdijk et al., 2002) and LASSY (for written Dutch; Van Noord et al., 2013). They are tree structures of a particular kind.[4] They differ from pure constituent structures (as used e.g. in Chomskyan generative grammar (Chomsky, 1965) and LFG's c-structures (Bresnan, 1982)) in two respects: they explicitly label grammatical relations, and order in the tree is not significant and need not correspond to the surface order.[5] They differ from dependency structures (on the analytical layer) as used in the Prague Dependency Treebank (Hajič and Hajičová, 1997) in explicitly allowing grouping of nodes under non-word nodes. The following simple (English) example illustrates the differences between a constituent structure (1a), a dependency structure (1b) and a CGN/LASSY-structure (2) for the sentence *The man saw a boy*:[6]

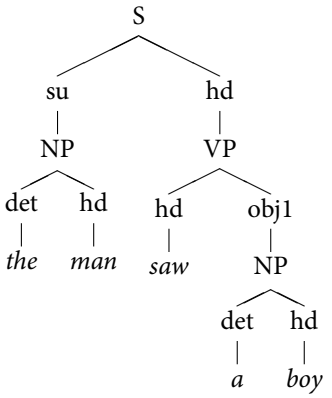
(1) a.



b.



(2)



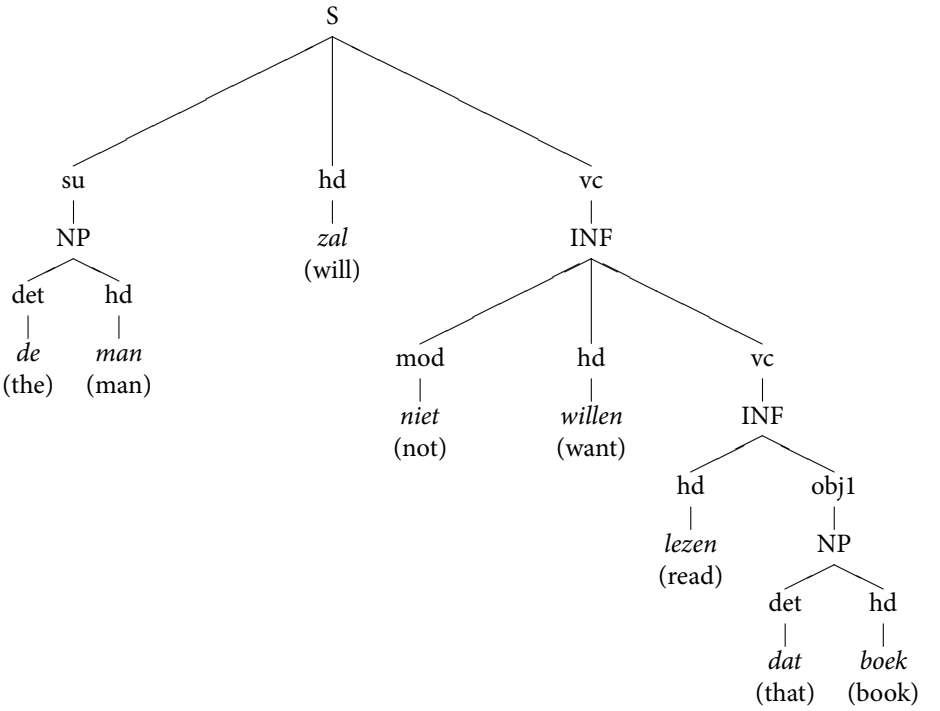
[4] There is no generally accepted name for these tree structures. Van Noord et al. (2011) call them 'dependentiestructuren' (dependency structures), but that is not a very fortunate choice.

[5] The surface order is specified in attributes on nodes for words.

[6] The nonsignificance of order in the CGN/LASSY tree is not visible in these trees.

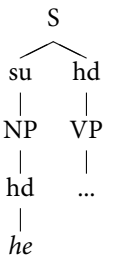
The syntactic structure for the Dutch sentence *De man zal dat boek niet willen lezen* 'The man will not want to read that book' in (3) clearly shows that order in the tree does not necessarily correspond to surface order:[7]

(3)



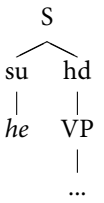
Another characteristic of the syntactic structures that is in accordance with the Dutch *de facto* standard for syntactic structures in treebanks is that a structure cannot contain nodes that dominate a single node. So, e.g., if the subject of a sentence consists of a single pronoun (e.g. a pronoun such as 'he'), then the structure is not like in (4), since that structure contains a node (NP) that dominates a single node (*he*):

(4)



but is instead as in (5):

[7] Thus enabling the representation of discontinuous dependencies or what are called *non-projective structures* in Dependency Grammar.

(5)


Syntactic structures can also contain nodes that have no content except for an index attribute ('empty categories'). These are always co-indexed with some node with content. This mechanism is used to express constructions in which a single word or phrase plays multiple syntactic roles. We will see several examples of this in section 23.3.2.

23.3.2 Queries in PaQu

PaQu offers two interfaces for formulating queries to search in the treebanks. The first one enables one to formulate queries in the Xpath query language. This requires expert knowledge of the query language Xpath. Learning Xpath, or at least the most common elements required in PaQu, is easy, so this actually poses no problems. A more serious problem is that Xpath queries tend to become quite complex very quickly. Here PaQu aids the user by providing a syntax check: is the Xpath query well-formed (white background colour), or not (red background colour), and are valid elements and attributes used in the query or not (yellow background)?. It also provides suggestions for completing the Xpath expression that is being constructed. In addition, extensions of Xpath originally developed for the desktop Alpino corpus tool DACT (Van Noord et al., 2013) can be used: macros, which can be used for abbreviating often occurring complex Xpath expressions, and query pipelines, which are useful for improving performance.

Most problematic for constructing Xpath queries is that the user must know, in every fine detail, what the syntactic structures for a variety of constructions actually look like in the treebank. The GrETEL application ((Augustinus et al., 2012) and chapter 22) avoids this problem through a dedicated user interface that makes example-based querying possible. This imposes restrictions on the types of queries that can be made, but many often used query types can be constructed in this way on the basis of an example without actually having to formulate a query in a formal query language or to know in great detail what the syntactic structures look like.

The second query interface that PaQu offers is, like GrETEL's interface, a highly user-friendly dedicated interface, but it is user-friendly in a completely different way than GrETEL's interface. PaQu enables searching for dependency triples (see Figure 23.1). Such triples are of the form '(dependent word, relation, head word)'. PaQu's dedicated interface enables one to formulate queries in terms of these three elements. One can query for the head word and the dependent word by specifying some of their properties, in particular, their lemma, their word form, and their part of speech. The *relation* element is an attribute that takes atomic labels for grammatical relations such as subject (su), direct object (obj1), modifier (mod), etc. The query can basically be formulated in terms of the seven attributes *lemma, word, postag* (for the dependent word), *hlemma, hword, hpostag* (for the headword) and *rel* for the relation. One has to specify minimally one of these elements in the query. If one does not specify anything at all for any of these elements, it effectively acts as a variable and matches with anything. Some examples may illustrate this:[8]

(6) a. Yield sentences from CGN that contain the lemma *heel* as a modifier: (lemma='heel', rel='mod').

[8] Clicking on the queries executes them.

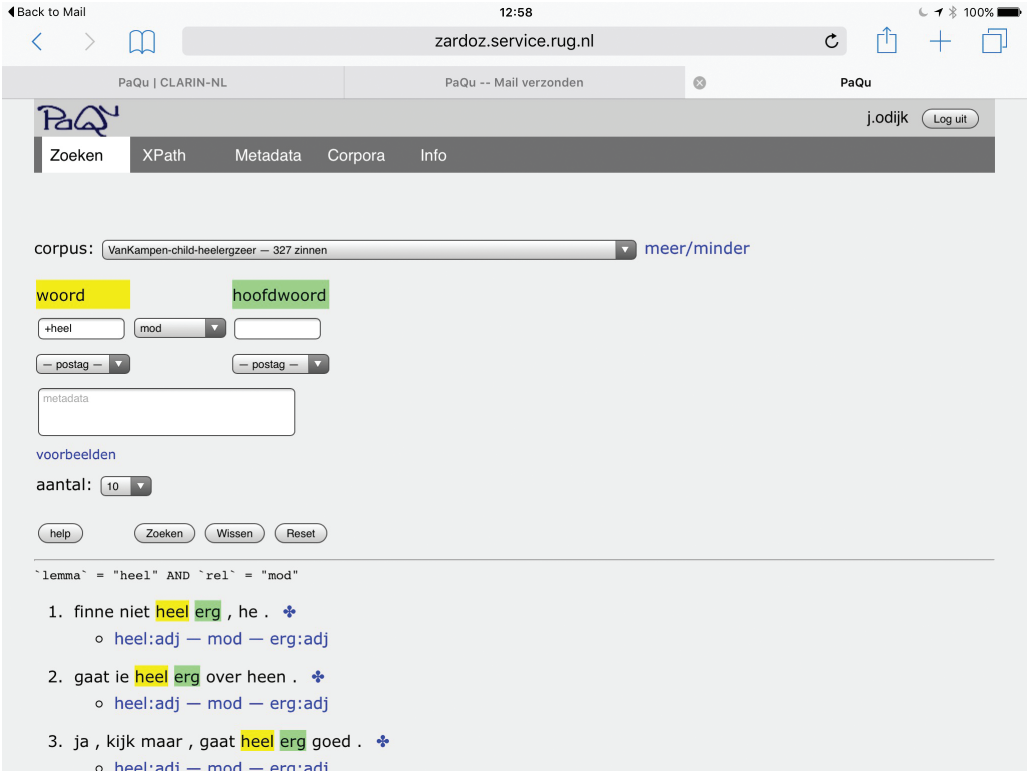


Figure 23.1: PaQu web interface with a query for occurrences of the lemma *heel* as modifier.

b. Yield sentences from CGN that contain the lemma *heel* as a modifier of an adjective: (lemma='heel', rel='mod', hpostag='adj')

c. Yield sentences from LASSY-Small that contain a predicative complement to the verb *raken*: (rel='predc', hlemma='raken')

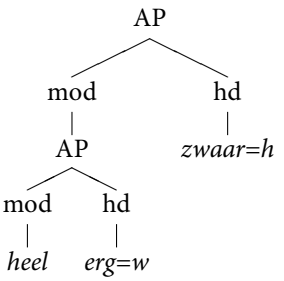
Searching for dependency triples is very often needed in linguistic research, so having a simple and very user-friendly way of formulating such queries is one of PaQu's greatest benefits. Obviously, the simple and user-friendly way in which this can be done also restricts the kind of queries one can pose, but it does so in a completely different way than GrETEL does. In this way PaQu is complementary to GrETEL.

The relevant queries for dependencies triples are not easily expressed in Xpath. Such a query has to take into account not only headed structures but also coordinated structures and co-indexed nodes in the syntactic structure. In addition, the dependent word can be contained in a phrase that is a dependent of the head word. We will elaborate a little bit on this.

A word *w* that modifies some other word *h* can be modified itself. In that case *w* is the head of a constituent, and *h* is actually modified by the constituent dominating *w*. For example, the result of the query (word='erg', rel='mod') in LASSY-Small contains such examples, in particular the resulting sentences 1, 18, 71, 77.[9] The relevant part of the structure of 18 (*heel erg zwaar*, lit. very awfully heavy, 'really very heavy') is given in (7):

[9] The order of the query output is not always guaranteed to be the same, so one might find these examples under other numbers in a different instantiation of the query. The links should work, however.

(7)



In (7) *erg* 'awfully' is the head of the AP (*heel erg*) 'very awfully', which modifies *zwaar* 'heavy': in such cases we want the dependency triple (*erg*, mod, *zwaar*) in the results, which is what PaQu's dedicated interface achieves.

Suppose that one is interested in subjects of the verb *lopen*. This word, as almost any word in natural language, is highly ambiguous: the online Van Dale lists nine senses (and De Boer (2007) even lists eleven), and possible translations into English include 'walk', 'run', 'stream', 'last', 'move', 'work', and others). The ambiguity is often reduced and occasionally even resolved by combining it with its arguments. For investigating combinations of the verb *lopen* with subjects, one can use the query (rel=su, hword='lopen') in LASSY-Small. But a subject of a verb can be a conjunction of multiple words, e.g. *droom en werkelijkheid* (dream and reality) in the sentence *Droom en werkelijkheid lopen door elkaar*[10] (example 10), as well as in many other examples, among them 21, 65 and 90 of the result set. PaQu counts each conjunct and the conjunction itself as a subject of the verb in such cases, which is a reasonable and very useful strategy.

Another pecularity of the syntactic trees is that they can contain nodes that are co-indexed with other nodes (as described above). If a node *a* is co-indexed with node *b*, and *b* is a dependent of some head *h*, then we want to consider *a* to be a dependent of *h* as well. Co-indexed nodes are used for a range of linguistic phenomena. Ellipsis is one: in the query (word='heel', rel='mod') examples 2 and 4 have been analysed as containing ellipsis; in an informal notation, for example 2, the part *heel afwisselende en rijke* is analysed as *heel afwisselende en ~~heel~~ rijke*, in which the striked-out *heel* is an 'empty category' co-indexed with the first occurrence of *heel*.

Coindexation is also used for control, for subject and object raising, and for long distance movements (e.g., preposed relative and interrogative pronouns). In the query (rel=su, hword='lopen') we find subject raising in examples 15 (subject of *was* and subject of *gelopen*), 16 (subject of *blijkt* and *lopen*) and 32 (subject of *leken* and *lopen*); ellipsis and subject control in example 17 (subject of *wil* and *lopen*); object raising in 21 (object of *laat* and *lopen*), subject raising twice in 22 (subject of *hebben*, *kunnen*, and *lopen*); and object raising in 28 (direct object of *zagen* and subject of *lopen*); etc. etc. An example of wh-movement can be found in example 15 (preposing of the relative pronoun *die*) .

Making queries that take into account all these different aspects is very difficult, and these queries are not easily expressed in Xpath (see the DACT Cookbook, section *Antecedents of co-indexed nodes* for an implementation of including indexed nodes in Xpath). PaQu's dedicated search interface, however, does it all automatically without the user having to worry about these cases: when searching for any relation, PaQu not only searches for nodes with that relation, but also searches for nodes that are the head of a phrase with that relation, or that are a conjunct in coordinated phrases; and when it finds a node searched for, it also searches for nodes co-indexed with the found node.

Each executed query in PaQu yields a URL that can be used to replicate the query. The URL encodes the server that PaQu runs on and parameters of the query to be performed. Any web

[10] 'Dream and reality intermingle'.

browser can interpret these URLs (as HTTP GET requests), so that it is easy to replicate the query exactly. We have used many of these URLs in this document in hyperlinks to queries. They have also been used extensively in the Taalportaal (Van der Wouden et al., 2016) to provide the links from descriptions of constructions to examples in text corpora (see Bouma et al. (2015),Van der Wouden et al. (2015) and chapter 24).

23.3.3 Search Results

The output of a PaQu query is a list of utterances that match the query. They are preceded by a specification of the query (for example `‘word‘ = "heel" AND ‘rel‘ = "mod"` for the query (word='heel', rel='mod')).

PaQu shows the results on different web pages, with by default 10 utterances per page, though one can have up to 500 utterances per page. To see more results, one has to browse to the next results page. Representing only a limited number of utterances per page, and initially only one results page, enables users to experiment with a query, to quickly inspect the initial results, and, if needed, to adapt the query, while at the same time saving computing time and avoiding unnecessary waiting time on the user side. If one is interested in all results, one can compute and analyse them in the analysis part.

Each utterance in the results is assigned a number, and the syntactic structure of the utterance can be viewed by clicking on the clover button to the right of the utterance. In the dependency relations search interface the head and the dependent words are marked by colour: green for the head word(s), yellow for the dependent word(s). Each utterance is followed by the dependency triple(s) found in the utterance, and each of these triples is a query itself, automatically generated by PaQu. For example, in the query (word='heel', rel='mod'), result 3 is followed by the triple hele:adj – mod – land:n, which itself is a link containing a query *(word='hele', postag='adj', rel='mod', hword='land', hpostag='n')* that searches for utterances containing the adjective *hele* as a modifier of the noun *land*. Clicking on it executes this query.

23.4 Analysis

The results page of a query shows not only the resulting utterances, but also several options for analysing the search results. The button *Tellingen algemeen* (General Counts) provides counts of the occurrences of each of the seven attributes that define a dependency relation query: *word, lemma, postag, rel, hword, hlemma, hpostag*. The resulting statistics can be downloaded as a tab-separated text file.

One can also compute counts of combinations of these attributes, using the button *Tellingen van Combinaties* (Combination Counts; see Figure 23.2). The result is a table with counts of the examples that meet the criteria, with instantiation of the unspecified attributes by the values occurring in the search results (i.e. they are acting as variables). The counts in the table are hyperlinks with automatically generated queries for exploring specific subcases in more detail. Clicking on the column header sorts the table by the values in this column (the header is then represented in italics).

PaQu already enables analysis of search results in combination with metadata, at least for some corpora, e.g. the Spoken Dutch Corpus. This result was achieved in the CLARIAH project, the successor of CLARIN-NL. This aspect has been heavily used in Van Noord and Odijk (2016), where it is shown that certain non-normative variants of Dutch construction only occur in a specific region, e.g. *hun* 'them' as a subject only occurs in the Netherlands part of the corpus, while *'m* 'him' as a subject only occurs in the Flanders part of the corpus.

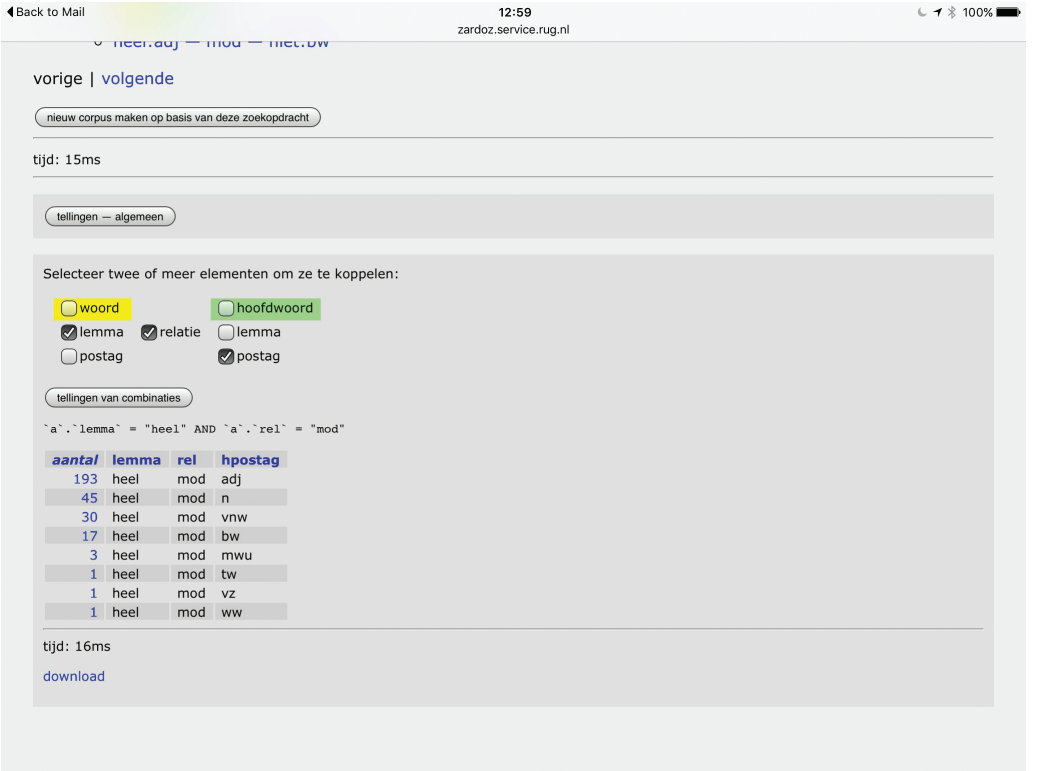


Figure 23.2: PaQu analysis: count of occurrences of the lemma *heel* as modifier by part of speech of the modifiee.

23.5 Case Study

Odijk (2011) sketched a research problem concerning the acquisition of syntactic selectional properties of words. Odijk (2015a) and Odijk (2016) investigate this problem for the words *heel*, *zeer* and *erg* (all meaning 'very'): *heel* (under this reading) only selects adjectival predicates while *erg* and *zeer* select adjectival, verbal and adpositional predicates.

Here we illustrate an initial step in an investigation of this problem for a similar but different case: on the basis of initial data, we assume that the verb *worden* 'become' selects only adjectival and nominal predicates (8), while the almost synonymous[11] verb *raken* 'get' selects only adjectival and adpositional[12] predicates (9):[13]

(8) a. Zij werd zwanger
she became pregnant
'She became pregnant'

[11] The only semantic difference, as far as we can see, is that *raken* implies accidentality of the state or property change, while *worden* is neutral in this respect.

[12] This includes words that are traditionally called locational and directional 'adverbs'.

[13] Of course, these verbs, like almost all natural language words, have multiple uses: *worden* is also used to form passive constructions, and *raken* is also a transitive verb meaning 'hit'. We will ignore these uses here, which is easy since PaQu enables us to separate them from the copular use of these verbs.

b. * Zij werd in verwachting
she became in expectation

c. Zij werd dokter
she became doctor
'She became a doctor'

(9) a. Zij raakte zwanger
she got pregnant
'She got pregnant'

b. Zij raakte in verwachting
she got in expectation
'She got pregnant'

c. * Zij raakte dokter
she got doctor

We search, using PaQu, in the LASSY-Small corpus for the predicative complements of *raken* using the query (rel='predc', hlemma='raken') and analyse the results (see Table 23.1). We do the same for *worden* using the query (rel='predc', hlemma='worden'), and the analysis of the results yields Table 23.2. The counts in the Tables 23.1 and 23.2 are links to queries that search for utterances containing predicative complements with the relevant postag, just as in the analysis table of PaQu. These queries, which are automatically created by PaQu, are characterised in the third column of Tables 23.1 and 23.2.[14]

Count	postag	Query
112	adj	(rel='predc', hlemma='raken', postag='adj')
67	ww	(rel='predc', hlemma='raken', postag='ww')
33	vz	(rel='predc', hlemma='raken', postag='vz')
3	vg	(rel='predc', hlemma='raken', postag='vg')
2	mwu	(rel='predc', hlemma='raken', postag='mwu')
1	n	(rel='predc', hlemma='raken', postag='n')

Table 23.1: Counts of predicative complements to *raken* grouped by postag.

Count	postag	Query
693	n	(rel='predc', hlemma='worden', postag='n')
665	adj	(rel='predc', hlemma='worden', postag='adj')
59	vg	(rel='predc', hlemma='worden', postag='vg')
54	ww	(rel='predc', hlemma='worden', postag='ww')
49	tw	(rel='predc', hlemma='worden', postag='tw')
40	mwu	(rel='predc', hlemma='worden', postag='mwu')
32	vnw	(rel='predc', hlemma='worden', postag='vnw')
13	bw	(rel='predc', hlemma='worden', postag='bw')
8	spec	(rel='predc', hlemma='worden', postag='spec')
4	vz	(rel='predc', hlemma='worden', postag='vz')

Table 23.2: Counts of predicative complements to *worden* grouped by postag.

[14] The relevant part of speech codes used in the LASSY-Small treebank are:adj (adjective), ww (verb), vz (adposition), vg (conjunction), mwu (multiword unit), n (noun), tw (numeral), vnw (pronoun), bw (adverb), and spec (special).

For *raken*, we find, as expected, adjectival predicative complements (*adj*) and adpositional predicative complements (*vz*). PaQu also lists verbal predicative complements (*ww*), but many of these are adjectives that happen to be identical in form to passive participles of verbs: such adjectives are always analysed in the Dutch treebanks as verbs, but in reality they are adjectives, as can be shown with standard tests to distinguish verbal participles from homophonous adjectives. An example is *besmet* 'contaminated' in this example, also represented in (10):

(10) In China zijn varkens besmet geraakt met de vogelpest
in China are pigs contaminated got with the fowl pest
'In China, pigs got contaminated with the fowl pest'

It is not fully clear to me that all examples can be analysed as adjectives, though many researchers claim this (e.g. Broekhuis and Corver (2015:section 6.2.3, I, p. 989)). This requires further research. If not, we must allow verbal participial complements to the verb *raken* as well. Examples with the part of speech tag *vg* concern conjoined structures (in all cases with adjectival conjuncts). The code *mwu* stands for *multiword unit*: the treebank identifies certain word combinations as multiword units without assigning them any part of speech tag other than *mwu*. The two examples are the expressions *in de war* and *door de war*, both meaning 'mixed up' or 'confused'. They are clearly adpositional phrases, even though the treebank does not explicitly state this (probably because the expressions have an idiosyncratic meaning and the word *war* only occurs in these expressions).

Finally, PaQu lists one example with a nominal predicate (*n*), which is not expected. Closer inspection shows that the syntactic structure for this example contains an error: the example contains the transitive verb *raken* and the nominal complement should have been assigned the grammatical relation *obj1*. Even in treebanks that are claimed to have been manually verified some errors will occur. The fact that this error occurs is actually good, because it shows that the grammar behind the treebank does not exclude nominal predicative complements to the verb *raken*. If it did, there would be a danger of circularity: we would then not find any nominal complements because the grammar behind the treebank cannot create such structures. But this example shows that there is no circularity here.

Of course, the presence of such errors raises the question of how reliable the data underlying the analysis given here are. Van Noord et al. (2013) mention an independent validation which reports a sentence accuracy, i.e. sentences without any error in the syntactic analysis divided by all sentences, of 97.8% for LASSY-Small. To assess the accuracy for dependency triples with *raken* or *worden* as hlemma, we manually annotated all 1,039 dependency triples with hlemma=*raken* (yielding an accuracy of over 96.1%) and a sample of 866 out of the 27,697 dependency triples with hlemma=*worden* (yielding an accuracy of over 98.8%). These high accuracy figures clearly indicate that the data that the analysis given here is based on are reliable.

It makes sense to also look at examples where *raken* has a complement marked with the grammatical relation *ld*. The label *ld* is intended for marking locative and directional complements, but it is not obvious that a syntactic distinction with *predc* must or can be made. The distinction between *ld* and *predc* might just be semantic in nature. This distinction is especially difficult to make in many cases because many predicates are expressed by locative phrases in some metaphorical abstract form of space, and many expressions that are literally locative have an idiomatic or metaphorical reading as a state (e.g. *aan lager wal raken*, lit. 'end up at the lee shore', fig. 'come down in the world', as in this example). All *ld*-complements to *raken* are adpositional (*vz*), except for one (this example), in which the complement has postag *bw* (adverb) and is a locative adverb that falls under adpositions.[15] Furthermore, there is one example with an *ld* complement in

[15] The relevant adverb is *waarin*, lit. where-in', for which an analysis as two separate syntactic formatives, *waar* and *in*, that form an adpositional phrase and that happen to be a single word phonologically and orthographically is very plausible.

which the verb *raken* is actually the (passivised) transitive verb (with meaning 'hit'). It is not obvious that the phrase marked as a complement here is indeed a complement.[16]

Finally, we should consider elements with the grammatical relation *svp*, which is for separated verbal particles. Such particles can be adpositional, nominal or adjectival, and single word predicative complements can often behave as such particles. A concrete example is *opraken* 'run out', which is composed of the (intransitive) adposition *op* 'exhausted' and the verb *raken* 'get'. But often the combination of a particle and a verb leads to an unpredictable meaning and unpredictable syntactic selectional properties. There are 19 occurrences of an *svp*-complement to *raken* in LASSY-Small, and in 12 cases it indeed involves *svp*-complements, for the idiosyncratic verbs *aanraken* 'touch', transitive *kwijtraken* 'lose' (which are irrelevant for our analysis), and for *opraken*. In 7 cases, the analysis as an *svp*-complement is dubious. The examples all involve idiosyncratic expressions such as *in de vergetelheid raken* 'become forgotten' and *in zwang raken* 'get fashionable', *slaags raken* 'start fighting' and *(ergens) verzeild raken* 'get (somewhere)'. Even when they are analysed as predicative complements (which is a more plausible analysis and is also adopted in traditional grammars), they conform to our assumptions on the syntactic selectional properties of the copular verb *raken*.

To summarise, these data fully confirm our assumptions about the syntactic selectional properties of the copular verb *raken*, though perhaps participial complements should be added as an option.

For *worden*, we find many nominal (*n*) and adjectival (*adj*) predicative complements, as expected. The examples labelled with *vg* involve nominal and adjectival conjoined structures. The examples labelled with *ww* concern true verbal participles incorrectly labelled as *predc* instead of as *vc* (verbal complement), or adjectives that are identical to participles, e.g. *opgewonden* 'excited' as an adjective, 'wound up' as a participle, in this example, the relevant part of which is in (11):

(11) Je overschat jezelf en wordt opgewonden
one overestimates oneself and becomes excited
'One overrates oneself and becomes excited'

The examples labelled with *tw* concern numerals, both cardinals and ordinals. Many researchers analyse these as nouns (cardinals) and adjectives (ordinals), but if a separate syntactic category *tw* can be justified, we should add it to the syntactic selectional specification of *worden*. The *mwu* examples mostly involve names, titles and citations (all nominal in character). There is one example of an adpositional expression, but here the syntactic structure is incorrect (this example). However, there are also examples with the adpositional expression *van kracht*, lit. 'of power', fig. 'valid, in force', and these are real counterexamples to our original assumptions about the syntactic selectional properties of copular *worden*. We will come back to them below.

The examples labelled with *vnw* concern pronouns, and these are all nominal or adjectival. The examples labelled *bw* concern adverbs: under our grammatical assumptions, all these examples (*zo* 'so, this way', *het eens* 'agreeing', *anders* 'different') are considered adjectives. The examples labelled with *spec* concern unknown (often foreign) words: the complements to *worden* here clearly are all nominal. Finally, there are 4 cases with adpositional predicative complements, which we do not expect. They come in two classes: first, expressions introduced by the word *als* 'as', which is traditionally analysed as a subordinate conjunction, as in (12):

(12) Het zou nooit meer worden als vroeger
it would never more become as in-the-past
'It would never be as before'

[16] It involves a locative PP containing an inalienable body part related to the (surface) subject, just as in the English example *he was hit in the head.*

We must either allow phrases introduced by the conjunction *als* as complements to the verb *worden* or assume that a covert adjective comparable to *zo* 'in the way' is present with the *als*-phrase as its modifier. The second class involves genuine adpositional phrases such as *van belang* 'of importance' and *van het grootste belang* 'of the greatest importance'. These are very similar to the example *van kracht* we saw earlier. We must conclude that *worden* does allow some adpositional predicative complements, though not all (see (8)), and we need to find properties that distinguish these two types of adpositional phrases. The adpositional phrases here are all introduced by the adposition *van*, but that cannot be the distinguishing factor since expressions such as *van streek* 'worried' and *van slag* 'confused' cannot combine with *worden* and must be combined with *raken*. It is also worth noting that the adpositional phrases that occur with *worden* cannot occur as complements to *raken*, which is also not expected.

For the grammatical relation *ld*, only one example is found with *worden*, and it has the wrong grammatical relation: it should be *mod* instead of *ld*.

We conclude, based on our search for relevant examples and the analysis of the search results in PaQu, that our initial assumptions about the selectional properties for *worden* must be adapted. This example thus shows that systematic search of linguistic phenomena in corpora can be used to find evidence to support or falsify hypotheses, and may lead to adaptations of proposed hypotheses. The exact nature of the adaptation that is needed here, however, requires further research, for which PaQu again can be fruitfully used.

23.6 Conclusions and Future Work

We have described the functionality that PaQu offers: corpus management, in particular uploading one's own corpora to have them parsed and make them searchable as a treebank; searching in the treebank; and analysis of the search results. We have shown that queries that are pretty difficult to formulate in a query language such as Xpath are very easy to formulate using PaQu's dedicated search interface. PaQu is a prime example of what CLARIN aims to achieve: it is a web application with a user interface dedicated to researchers (mainly linguists) that supports them in their research. We have illustrated the use of PaQu with one small case study, which could be a first step in a full investigation on the syntactic selectional properties of the near-synonyms *worden* and *raken* and their acquisition. This small case study with PaQu forced us to modify our initial assumptions about the properties of these verbs.

We conclude that the functionality and interface of the PaQu application make it an effective and practical tool for supporting syntactic research of the Dutch language, and this claim is supported by the fact that PaQu has already been used in several research projects and for linking descriptions of grammatical constructions in the Taalportaal to corpus examples.

The development of PaQu was initiated independently of CLARIN with the Lassy Dependency Relations search application, though it was clearly inspired by the goal of CLARIN to make large, richly annotated corpora accessible and easily usable by linguists through dedicated interfaces. CLARIN-NL later ensured the continued existence of the Lassy Dependency Relations application, and requested extensions of its functionality, which resulted in PaQu. PaQu's development is being continued in the CLARIAH-CORE project, and some results of this work are already visible and can already be used, in particular analysis in terms of metadata and support for more input formats than just plain text, among them FoLiA (Van Gompel and Reynaert, 2013 and chapter 6) and TEI.

Acknowledgements

This work crucially uses data and/or tools made available through the CLARIN infrastructure. The work was financed by CLARIN-NL and CLARIAH-CORE, NWO projects in the Netherlands. We thank the anonymous reviewers for valuable comments on an earlier version of this chapter.

References

Augustinus, Liesbeth, Vincent Vandeghinste, and Frank Van Eynde (2012), Example-based treebank querying, *in* Calzolari, Nicoletta, Khalid Choukri, Thierry Declerck, Mehmet Uğur Doğan, Bente Maegaard, Joseph Mariani, Asunción Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC 2012)*, European Language Resources Association (ELRA), Istanbul, Turkey.

Bloem, Jelke (2016), Evaluating automatically annotated treebanks for linguistic research, *in* Bański, Piotr, Marc Kupietz, Harald Lüngen, Andreas Witt, Adrien Barbaresi, Hanno Biber, Evelyn Breiteneder, and Simon Clematide, editors, *Proceedings of the 4th Workshop on Challenges in the Management of Large Corpora (CMLC 4)*, ELRA, Paris, pp. 8–14. http://www.lrec-conf.org/proceedings/lrec2016/workshops/LREC2016Workshop-CMLC_Proceedings.pdf.

Bouma, Gosse, Marjo van Koppen, Frank Landsbergen, Jan Odijk, Ton van der Wouden, and Matje van de Camp (2015), Enriching a descriptive grammar with treebank queries, *Proceedings of the Fourteenth International Workshop on Treebanks and Linguistic Theories (TLT14)*, Vol. 14, pp. 13–25. http://tlt14.ipipan.waw.pl/files/4614/5063/3858/TLT14_proceedings.pdf.

Bresnan, Joan, editor (1982), *The Mental Representation of Grammatical Relations*, MIT Press Series on Cognitive Theory and Mental Representation, The MIT Press, Cambridge, Massachusetts /London, England.

Broekhuis, Hans and Norbert Corver (2015), *Syntax of Dutch: Verbs and Verb Phrases*, Vol. II of *Comprehensive Grammar Resources*, Amsterdam University Press, Amsterdam. http://www.oapen.org/download?type=document&docid=555749.

Chomsky, Noam (1965), *Aspects of the Theory of Syntax*, MIT.

de Boer, Theo, editor (2007), *Groot Woordenboek Hedendaags Nederlands, digitale versie 6.10*, Van Dale, Utrecht.

Hajič, Jan and Eva Hajičová (1997), Syntactic tagging in the Prague Dependency Treebank, *in* Marcinkeviciene, R. and N. Volz, editors, *Proceedings of the Second European Seminar "Language Applications for a Multilingual Europe"*, TELRI, Kaunas, Lithuania, pp. 55–68.

Hoekstra, H., M. Moortgat, B. Renmans, M. Schouppe, I. Schuurman, and T. van der Wouden (2003), CGN syntactische annotatie, *CGN report*, Utrecht University, Utrecht, the Netherlands. http://lands.let.kun.nl/cgn/doc_Dutch/topics/version_1.0/annot/syntax/syn_prot.pdf.

Odijk, Jan (2011), User scenario search, internal CLARIN-NL document, http://www.clarin.nl/node/166. http://www.clarin.nl/sites/default/files/User%20scenario%20Serach%20110413.docx.

Odijk, Jan (2015a), Linguistic research with PaQu, *Computational Linguistics in the Netherlands Journal* **5**, pp. 3–14. http://www.clinjournal.org/sites/clinjournal.org/files/odijk2015.pdf.

Odijk, Jan (2015b), Zoeken naar constructies, Presentation and demo held at the DRONGO Language Festival 2015, Utrecht.

Odijk, Jan (2016), A Use case for Linguistic Research on Dutch with CLARIN, *in* De Smedt, Koenraad, editor, *Selected Papers from the CLARIN Annual Conference 2015, October 14-16, 2015, Wrocław, Poland*, number 123 in *Linköping Electronic Conference Proceedings*, CLARIN, Linköping University Electronic Press, Linköping, Sweden, pp. 45–61. http://www.ep.liu.se/ecp/article.asp?issue=123&article=004.

Oostdijk, N., W. Goedertier, F. Van Eynde, L. Boves, J.P. Martens, M. Moortgat, and H. Baayen (2002), Experiences from the Spoken Dutch Corpus project, *in* González Rodriguez, M. and C. Paz Suárez Araujo, editors, *Proceedings of the third International Conference on Language Resources and Evaluation (LREC-2002)*, ELRA, Las Palmas, pp. 340–347.

Tjong Kim Sang, Erik, Gosse Bouma, and Gertjan van Noord (2010), LASSY for beginners, Presentation at CLIN 2010, Utrecht. http://ifarm.nl/erikt/talks/clin2010.pdf.

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

van der Wouden, Ton, Gosse Bouma, Matje van de Camp, Marjo van Koppen, Frank Landsbergen, and Jan Odijk (2015), Enriching a grammatical database with intelligent links to linguistic resources, *in* De Smedt, Koenraad, editor, *Selected Papers from the CLARIN Annual Conference 2015*, Linköping Electronic Conference Proceedings, CLARIN, Linköping University Electronic Press, Linköping, Sweden, pp. 108–117. http://www.ep.liu.se/ecp/article.asp?issue=123&article=009.

van der Wouden, Ton, Jenny Audring, Hans Bennis, Frits Beukema, Geert Booij, Hans Broekhuis, Norbert Corver, Crit Cremers, Roderik Dernison, Marcel den Dikken, Siebren Dyk, Carlos Gussenhoven, Ger de Haan, Vincent van Heuven, Eric Hoekstra, Jarich Hoekstra, Bart Hoogeveen, Gerbrich de Jong, Evelien Keizer, Anna Kirstein, Björn Köhnlein, Frank Landsbergen, Kathrin Linke, Marc van Oostendorp, Nina Ouddeken, Koen Sebregts, Carole Tiberius, Arjen Versloot, Willem Visser, Riet Vos, Truus de Vries, and Joke Weening (2016), Het Taalportaal, *Nederlandse Taalkunde* **21** (1), pp. 157–168. http://dx.doi.org/10.5117/NEDTAA2016.1.WOUD.

van Gompel, Maarten and Martin Reynaert (2013), FoLiA: A practical XML format for linguistic annotation - a descriptive and comparative study, *Computational Linguistics in the Netherlands Journal* **3**, pp. 63–81. http://www.clinjournal.org/sites/clinjournal.org/files/05-vanGompel-Reynaert-CLIN2013.pdf.

van Noord, Gertjan and Jan Odijk (2016), Goed of fout? Wat gebruikt men feitelijk?, Presentation held at Grote Taaldag 2016, Utrecht.

van Noord, Gertjan, Gosse Bouma, Frank Van Eynde, Daniël de Kok, Jelmer van der Linde, Ineke Schuurman, Erik Tjong Kim Sang, and Vincent Vandeghinste (2013), Large scale syntactic annotation of written Dutch: Lassy, *in* Spyns, Peter and Jan Odijk, editors, *Essential Speech and Language Technology for Dutch*, Theory and Applications of Natural Language Processing, Springer Berlin Heidelberg, pp. 147–164. http://dx.doi.org/10.1007/978-3-642-30910-6_9. http://dx.doi.org/10.1007/978-3-642-30910-6_9.

van Noord, Gertjan, Ineke Schuurman, and Gosse Bouma (2011), Lassy syntactische annotatie (revision 19455), *Lassy report*, RU Groningen, Groningen. https://www.let.rug.nl/vannoord/Lassy/sa-man_lassy.pdf.

Appendix: List of Hyperlinks

(lemma='heel', rel='mod') in CGN http://www.let.rug.nl/alfa/paqu/?db=cgn&word=%2Bheel&rel=mod&hword=&postag=&hpostag=&meta=&sn=10

(lemma='heel', rel='mod', hpostag='adj') in CGN http://www.let.rug.nl/alfa/paqu/?db=cgn&word=%2Bheel&rel=mod&hword=&postag=&hpostag=adj&meta=&sn=10

(rel='predc', hlemma='raken') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&rel=predc&hword=%2Braken&postag=&hpostag=&meta=&sn=10

(rel='predc', hlemma='raken', postag='adj') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=adj&rel=predc&hword=%2braken&hpostag=&meta=

(rel='predc', hlemma='raken', postag='mwu') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=mwu&rel=predc&hword=%2braken&hpostag=&meta=

(rel='predc', hlemma='raken', postag='n') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=n&rel=predc&hword=%2braken&hpostag=&meta=

(rel='predc', hlemma='raken', postag='vg') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=vg&rel=predc&hword=%2braken&hpostag=&meta=

(rel='predc', hlemma='raken', postag='vz') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=vz&rel=predc&hword=%2braken&hpostag=&meta=
(rel='predc', hlemma='raken', postag='ww') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=ww&rel=predc&hword=%2braken&hpostag=&meta=
(rel='predc', hlemma='worden') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&rel=predc&hword=%2Bworden&postag=&hpostag=&meta=&sn=10
(rel='predc', hlemma='worden', postag='adj') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=adj&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='bw') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=bw&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='mwu') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=mwu&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='n') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=n&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='spec') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=spec&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='tw') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=tw&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='vg') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=vg&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='vnw') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=vnw&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='vz') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=vz&rel=predc&hword=%2bworden&hpostag=&meta=
(rel='predc', hlemma='worden', postag='ww') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&postag=ww&rel=predc&hword=%2bworden&hpostag=&meta=
(rel=su, hword='lopen') in LASSY-Small http://www.let.rug.nl/alfa/paqu/?db=lassysmall&word=&rel=su&hword=%2Blopen&postag=&hpostag=&meta=&sn=100
(rel=su, hword='lopen') in LASSY-Small, sent. 10 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=4671&yl=0,1,2&gr=3&ms=5,2,3,6,4
(rel=su, hword='lopen') in LASSY-Small, sent. 15) http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=4815&yl=23&gr=31&ms=49,58
(rel=su, hword='lopen') in LASSY-Small, sent. 16 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=4845&yl=1&gr=15&ms=19,30,4
(rel=su, hword='lopen') in LASSY-Small, sent. 17 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=4868&yl=0&gr=13&ms=16,24
(rel=su, hword='lopen') in LASSY-Small, sent. 21 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=5109&yl=23,24,25&gr=27&ms=41,45,47,42,43
(rel=su, hword='lopen') in LASSY-Small, sent. 21 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=5109&yl=23,24,25&gr=27&ms=43,41,45,47,42
(rel=su, hword='lopen') in LASSY-Small, sent. 22 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=5606&yl=2&gr=8&ms=19,7,12
(rel=su, hword='lopen') in LASSY-Small, sent. 28 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=6506&yl=31&gr=37&ms=57,58
(rel=su, hword='lopen') in LASSY-Small, sent. 32 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=7602&yl=6&gr=11&ms=6,15,20
(rel=su, hword='lopen') in LASSY-Small, sent. 65 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=11901&yl=1,5,7&gr=10&ms=18,2,3,5,11,12,14
(rel=su, hword='lopen') in LASSY-Small, sent. 90 http://www.let.rug.nl/alfa/paqu/tree?db=lassysmall&arch=0&file=15812&yl=3,5,7&gr=9&ms=14,10,15,6,7,11,12