Starting a sentence in Dutch
Bouma, G.J.

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date: 2008

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):
Chapter 3

Methods, Techniques & Material

Before proceeding to the corpus study in the next chapter, I will discuss some background issues in this chapter. I will begin by introducing the Corpus Gesproken Nederlands, used for the investigations in this dissertation. I will briefly describe its make-up, the type of annotations, and the amount of available material in Section 3.1. Section 3.2 introduces the syntactic annotation available for the corpus. I will discuss general properties of the annotation and the way in which the annotation may influence what we can learn from the corpus. Moreover, I will briefly consider how much of the information that we need for the investigations we have planned is directly retrievable from the corpus, and how much must be added to it. The discussion of the syntactic annotation also prepares us for Section 3.3. In this section, I will give a definition of the Vorfeld in terms of the syntactic annotation. Topological fields are not annotated directly in the Corpus Gesproken Nederlands, but the amount of syntactic annotation that is present allows us to give a good definition of the Vorfeld.

Finally, the last two sections introduce two important tools. Section 3.4 explains the use of the logic programming language Prolog for the purpose of querying the corpus. The section ends with a Prolog translation of the Corpus Gesproken Nederlands definition of Vorfeld. Section 3.5 provides a general introduction to the use of logistic regression modelling. This statistical technique is used to answer several important questions in the remainder of the dissertation.

3.1 About the Corpus Gesproken Nederlands

The Corpus Gesproken Nederlands (Spoken Dutch Corpus, CGN, 2004), is a corpus of approximately 9mln words, that contains spoken Dutch from adult native speakers in The
Table 3.1: Overview of available syntactically annotated material.

<table>
<thead>
<tr>
<th>Component</th>
<th>Region</th>
<th>nl</th>
<th>fl</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Spontaneous conversation</td>
<td>300 368</td>
<td>146 745</td>
</tr>
<tr>
<td>b</td>
<td>Interviews with Dutch teachers</td>
<td>25 687</td>
<td>34 064</td>
</tr>
<tr>
<td>c</td>
<td>Spontaneous telephone dialogues</td>
<td>69 933</td>
<td>19 886</td>
</tr>
<tr>
<td>d</td>
<td>Spontaneous telephone dialogues</td>
<td>0</td>
<td>6 257</td>
</tr>
<tr>
<td>f</td>
<td>Interviews/discussions/debates (bc)</td>
<td>75 106</td>
<td>25 144</td>
</tr>
<tr>
<td>g</td>
<td>(Political) Discussions/debates/meetings</td>
<td>25 117</td>
<td>9 009</td>
</tr>
<tr>
<td>h</td>
<td>Classroom recordings</td>
<td>25 961</td>
<td>10 103</td>
</tr>
<tr>
<td>i</td>
<td>Live commentaries (bc)</td>
<td>24 986</td>
<td>10 130</td>
</tr>
<tr>
<td>j</td>
<td>News reports/background (bc)</td>
<td>25 065</td>
<td>7 679</td>
</tr>
<tr>
<td>k</td>
<td>News (bc)</td>
<td>25 384</td>
<td>7 305</td>
</tr>
<tr>
<td>l</td>
<td>Commentaries/columns/reviews (bc)</td>
<td>25 071</td>
<td>7 431</td>
</tr>
<tr>
<td>m</td>
<td>Ceremonious speeches/sermons</td>
<td>5 184</td>
<td>1 893</td>
</tr>
<tr>
<td>n</td>
<td>Lectures/Seminars</td>
<td>14 913</td>
<td>8 143</td>
</tr>
<tr>
<td>e</td>
<td>Simulated business negotiations</td>
<td>25 485</td>
<td>0</td>
</tr>
<tr>
<td>o</td>
<td>Read speech</td>
<td>0</td>
<td>44 144</td>
</tr>
<tr>
<td><strong>Total Used</strong></td>
<td></td>
<td>642 775</td>
<td>293 789</td>
</tr>
<tr>
<td><strong>Grand Total</strong></td>
<td></td>
<td>668 260</td>
<td>337 933</td>
</tr>
</tbody>
</table>

*Note:* Counts are in words. Abbreviations nl: Netherlands; fl: Flanders; bc: broadcast material. Segments c and d differ only in recording method.


Netherlands and Flanders. The complete corpus has received orthographic transcription, part-of-speech (POS) tagging, lemmatization and automatic phonetic transcription and alignment. About a third of the data is from Flanders. Meta-data about the background of the speakers and the situational context are also provided.

Parts of the corpus have received additional annotation. In this study, we are interested in lexical, syntactic and information structural properties of Vorfeld placement. However, there is no annotation for information structure. The constituent properties whose relation to the Vorfeld I will be investigating – grammatical function, definiteness/NP form and grammatical complexity – can be read from the combined morphological and syntactic annotation layers. There are about 1mln words of data available with this information.

The corpus is divided in different components that correspond to different sources and/or genres. The amount of material with morphological and syntactic annotation that is available per component is shown in Table 3.1. The table also shows how much data
in each component comes from which region. Two components are excluded from the selection: Component e was excluded because there was only material available from the Netherlands, and component o was excluded because it contains read speech from literary novels – data that we do not expect to be representative of spoken language. The remaining components are by no means homogeneous. They contain speech ranging from spontaneous to prepared, and speech produced in monologues or dialogues. The mixed nature of the data is an advantage because it allows us to better approximate the full range of spoken Dutch. Also, excluding data should be avoided as much as possible, because data sparseness may be a real problem. Note that components c and d are of the same genre but differ in recording method, so there is no reason to throw component d out.

It might be the case that there are subtle differences in Vorfeld placement between regions, registers, and genres. I fully expect that looking for regional variation in Vorfeld occupation in the corpus, and likewise looking for differences between the registers/genres, would prove to be interesting. However, in this dissertation I will not consider these issues, and treat all parts of the corpus just described on a par.

In the annotation, speech is divided into utterances. The selected corpus (components a–d and f–n) has ~125k utterances, averaging 7.6 words per utterance. It should be kept in mind that many of the utterances, especially in the spontaneous speech components, are one or two word utterances, of the sort: ‘uhm’ ‘uh uhm’, etcetera.

The CGN has receive detailed syntactic annotation but there is no annotation for topological fields. An important task in this chapter is therefore to provide a definition of Vorfeld in CGN terms. Section 3.3 gives this definition in prose, and Section 3.4 discusses the implementation of this definition. In the next section, I will begin by introducing the syntactic annotation.

### 3.2 Syntactic annotation in the CGN

The syntactic annotation scheme employed in the CGN is a hybrid between two approaches to syntactic description. First and foremost, the syntactic structures in the CGN are dependency structures. However, CGN trees also contain phrasal nodes, which are not traditionally part of dependency structure, but come from a phrase structure approach to describing syntactic structure. In this section I will describe this hybrid structure, and discuss how the structure influences the kind of information we can extract from the corpus and the way in which we can do so (Subsection 3.2.1). In Subsection 3.2.2, I will briefly consider the special problem of multiple dependencies. In many cases one could argue that a complement is dependent on more than one head. However, annotating multiple dependencies leads to a less constrained data structure which is harder to use in corpus work. Fortunately, we will see that we can ignore multiple dependencies in our definitions and queries.

Section 3.2: Syntactic annotation in the CGN

65
3.2.1 Dependencies and phrases

The CGN syntactic annotation mixes dependency structure and phrase structure. In a dependency structure, syntactic structure is specified in terms of grammatical relations between heads and complements. Grammatical relations are primary in a dependency structure. Traditionally, all nodes in a dependency tree are words. A syntactic structure is then a graph that connects all word in a sentence. In a CGN tree, however, nodes can be phrases, too. A CGN dependency relation is annotated between a mother phrase and a daughter constituent (a phrase or word). A special type of CGN dependency relation is the head relation (HD). Replacing phrasal nodes in a CGN tree with the word that is the head-daughter would result in a more traditional tree. Note that CGN does not require phrases to have heads, nor does it require them to be unique.

The hybrid of dependency structure and phrase structure allows for an effective division of labour: Formal information like the type of a clause or the category of a word can be annotated in the phrase labels, and grammatical functions can be annotated as dependency relations. The mixed scheme was originally put forward for German in Skut et al. (1997). The annotation guidelines for the CGN are given in Hoekstra et al. (2003, in Dutch). Guidelines for morphological analysis in the CGN can be found in Van Eynde (2003, in Dutch). English reports of the morphology- and syntax-annotation efforts can be found in Van Eynde, Zavrel, and Daelemans (2000) and Hoekstra et al. (2001), respectively. The annotation format is also used in the Alpino Treebank (Dutch, with a near identical annotation protocol, Van der Beek et al., 2002), and in the TiGer Treebank (German, Brants et al., 2002). An example CGN syntactic tree is given in (1).

(1)

```
<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>ieder</td>
<td>meisje</td>
<td>dat</td>
<td>daar</td>
<td>komt</td>
<td>loopt</td>
<td>gevaar</td>
<td></td>
</tr>
</tbody>
</table>

Every girl that goes there is in danger.
```

The whole sentence in (1) is marked up as a V2 main clause (SMAIN). Nodes in the tree below this top-level have two labels: the dependency relation between the node and its mother, and the syntactic category (phrase type or part-of-speech) of the node. The sentence in (1) has an NP subject (SU\_NP), the finite verb in second position as its head (HD\_WW,
Section 3.2: Syntactic annotation in the CGN

WW for werkwoord 'verb'), and a noun direct object (\( \text{OBJ}_1 \)). The subject NP has three daughters: a pronominal determiner (\( \text{DET}_{\text{VNW}} \), VNW for voornaamwoord 'pronoun'), a head noun, and a modifying relative clause (\( \text{MOD}_{\text{REL}} \)). The internal structure of the relative clause is not shown. A complete list of dependency relations and syntactic categories is given in Appendix A.

CGN syntactic trees do not encode linear order. The nodes in the dependency tree in (1) are not ordered with respect to each other. This is potentially problematic. Suppose we want to know whether the subject in (1) precedes the finite verb. We would like to be able to decide this by looking at the node that corresponds to the subject \( \text{SU}_{\text{NP}} \) and at the node that corresponds to the finite verb \( \text{HD}_W \). However, since these are not ordered, we cannot say whether the former precedes the latter. In our corpus research, it is crucial that we are able to answer such a question. After all, one of the characterizing properties of Vorfeld material is that it precedes the finite verb that marks the left bracket in the topological fields template (Section 2.1).

As a rather straightforward solution to this, used for instance in the Alpino Treebank (Van der Beek et al., 2002), we can use the linear order of the words that are dominated by two nodes to determine the linear order of the nodes. A node begins where the leftmost word it dominates begins, and ends where the rightmost word it dominates ends. Under this definition of linear order of a node, the subject NP of (1) starts at 0 and ends at 5, and the head verb node begins at 5 and ends at 6. To answer our earlier question: The subject of (1) precedes the finite verb.

The fact that dependency structure does not encode linear order itself is convenient when one studies word order variation: If dependency structure is independent of linear order, word order variation does not change the syntactic structure of a sentence. Consider a word order variant of (1) where the relative clause that modifies the subject is in the postverbal domain. The result is (2).

(2)

\[
\begin{array}{c}
\text{SU}_{\text{NP}} \\
\text{DET}_{\text{VNW}} \quad \text{HD}_N \quad \text{HD}_W \quad \text{OBJ}_1 \quad \text{MOD}_{\text{REL}} \\
0 \quad \text{ieder} \quad 1 \quad \text{meisje} \quad 2 \quad \text{loopt} \quad 3 \quad \text{gevaar} \quad 4 \quad \text{dat} \quad 5 \quad \text{daar} \quad 6 \quad \text{komt} \quad 7 \\
\end{array}
\]

\begin{center}
\text{SMAIN}
\end{center}

\begin{center}
\text{‘Every girl that comes there is in danger.’}
\end{center}

This constancy in syntactic analysis reflects a basic, defining assumption about word order variation, which is that there is a common core (of propositional content, thematic
role assignment, grammatical function assignment, etcetera) between word order variants.

The tree in (2) also shows another salient property of CGN trees: they may contain discontinuous phrases. In a discontinuous phrase, descendants are separated by material that is not descendant of that phrase. In (2), the head of the subject is separated from its sister relative clause by the finite verb and the direct object. Since the trees do not encode linear order directly, this discontinuity cannot be seen in the syntactic structure itself but depends on the linear order of the dominated words. A result of discontinuous phrases is that, given two nodes, we cannot always decide which one comes first. In (2), the subject NP ends at 7 and the finite verb starts at 2, so the subject does not precede the finite verb. However, since the finite verb ends at 3 and the subject begins at 0, the finite verb does not precede the subject either. Without further assumptions, we cannot consider the subject to be either preverbal or postverbal. This issue becomes relevant when we want to know whether the subject in (2) is in the Vorfeld or not. I will therefore return to this issue when I define Vorfeld occupancy in CGN terms in Section 3.3.

Now that we have seen some general properties of the CGN syntactic annotation, let us briefly consider what information is directly available in the annotation and what information we will have to add (automatically) in light of the research intentions stated in Chapter 2. In Section 2.6, I discussed three constituent properties whose relation to Vorfeld occupation will be investigated; grammatical function, definiteness or NP form, and grammatical complexity. I will give the details of the operationalization of each of these properties in terms of the corpus annotation in Chapter 4, when I discuss the results of the corpus investigations.

**Grammatical function**  The first property we are interested in is grammatical function. We need to know the grammatical function of a constituent because we will only look at the behaviour of subjects, indirect objects and direct objects. Moreover, one of the questions to be investigated in the corpus that was raised in Section 2.6 was how grammatical function relates to Vorfeld occupation.

Grammatical function can be read fairly directly from the dependency relations: Subjects are SU-dependents, direct objects OBJ1-dependents and indirect objects OBJ2-dependents. However, there are two caveats. First, a minor point is that the OBJ1-relation is also used for objects of a preposition, and not just direct objects of a verb. Secondly, direct objects and indirect objects in a clause may be embedded under verbs without affecting their (intuitive) markedness as Vorfeld occupants. This is illustrated by (3). The object initial word order in (3a) is no less or more marked than (3b).

(3)  a. **Die boot** zag ik.
     that boot saw I
     ‘I saw that boat.’
As an indication of the syntactic structure I have given the *dependency path* above each word. A dependency path is the list of dependency relations that one crosses if one walks from the word up the tree to the *SMAIN*-node (Van der Beek et al., 2002). The dependency paths show how the initial direct object in (3a) is an immediate *OBJ1* of *SMAIN*, whereas the direct object in (3b) is a dependent of a phrase embedded under two verbs (*VC* for verbal complement). It is tempting to just allow any *OBJ1* descendant of an *SMAIN* at any level to count as a direct object in our investigation, since the embedding does not appear to make a difference for its fronting behaviour. However, this is too unrestricted. A direct object from an embedded clause is marked in the Vorfeld of the matrix clause (4).

The dependency path of the Vorfeld occupant contains a *BODY*-relation, which (in this case) indicates that the object is extracted from an embedded clause. A way around this problem is to define *direct object* as a collection of dependency paths beginning with *OBJ1* that do not differ in their fronting behaviour. The question of which paths can be pooled together is thus in part an empirical question, to be answered in Chapter 4. The same reasoning carries over to indirect objects and *OBJ2*. A positive side effect of defining grammatical relation in terms of dependency paths is that the problem that *OBJ1* is also used for prepositional objects disappears – prepositional objects have different dependency paths.

**Definiteness** The second property of constituents that I will investigate is definiteness. The CGN syntactic annotation does not include an annotation for definiteness, so we will have to add this information ourselves. For our investigations, we have to be able to distinguish at least the following: the three definiteness levels (pronoun, definite full NP and indefinite full NP), and pronominal form (demonstrative, full or reduced). These distinctions can be fairly easily added using lists of distinguishing features. The pronominal forms can be categorized simply by listing all possible forms. The full NPs can be categorized into definite or indefinite mostly by looking at their determiners. More details of the classification can be found Section 4.3.1.

**Grammatical complexity** Finally, I will look at the relation between a constituent’s grammatical complexity and Vorfeld occupation. One measure of grammatical complexity
is syntactic category (NP, PP, CP, etcetera). This can be read almost directly from the syntactic annotation. In the literature on the effects of grammatical complexity, the length of a constituent is also frequently used as a measure of its complexity (see Section 4.4). The length of a constituent or node is the number of words that are dominated by it – retrieving this information from the corpus is unproblematic.

The CGN annotation allows us to gather all the information about constituents that we are after. This information can either be read almost directly from the annotation (grammatical function, grammatical complexity) or we can expect to reliably add this information automatically on the basis of the existing annotation (NP form). That this process can be automated is important because it allows us to use the whole of the CGN without having to worry about the time it takes to manually annotate for the information.

3.2.2 Multiple Dependencies

CGN dependency trees are unlike context free phrase structure trees in that linear order is not encoded in the tree. However, dependency annotation allows for a more serious deviation from the tree as datastructure: Nodes may depend on multiple mothers. Multiple dependencies mean that the datastructure of a rooted tree does not suffice anymore, and that the more general directed acyclic graphs would have to be used to represent the corpus data. Although this in itself is not problematic, searching through trees and inferencing on trees is simpler than working with directed acyclic graphs. Below, I will discuss two situations in which one might want to annotate multiple dependencies. In the first case, the multiple dependencies are (supposedly) lexically derivable from one dependency – these are not annotated in the CGN. In the second case, the multiple dependencies are not lexically derivable – these are annotated by the CGN but do not contain any information that we need to rely on. In conclusion, we can ignore the problem of multiple dependencies and assume that the syntactic annotations are trees in our definitions.

Lexically derivable multiple dependencies In cases of raising, control (equi), AcI, etcetera, one may argue that a constituent is a dependent of more than one node. For instance, in the AcI in (5a), the pronoun hem ‘him’ is an object of horen ‘to hear’ and in some sense also the subject of zingen ‘to sing’. We might annotate this with a dependency OBJ1 from hem to the SMAIN, and a dependency SU from hem to the phrase headed by zingen. In the control-construction in (5b), a relation SU from ik ‘I’ to the SMAIN as well as a relation SU from ik to the phrase headed by horen ‘to hear’ is reasonable.

(5) a. Ik hoor hem zingen.
    I hear him sing
    ‘I can hear him sing.’
b. Ik wil hem horen.
   I want him hear
   ‘I want to hear him.’

The CGN only annotates the highest dependency in the tree. According to the annotation guidelines, the structure of the sentences in (5) is (6).

(6) a.  
   |   |   |   |
   SU HD OBJ1 VC
   VNW WW VNW WW
   Ik hear him sing

   b.  
   |   |   |   |
   SU HD OBJ1 HD
   VNW WW VNW WW
   Ik want him hear

Additional dependencies may be derived from a) the highest dependency, the one that is annotated, and b) information about the matrix verb. For instance, we know about the control verb willen ‘to want’ that its subject is also the subject of the infinitival VP willen takes as an argument.\(^1\)

The question is whether we need the information that would be contained in the extra dependency links that the CGN has chosen not to annotate. For the investigations in this dissertation, we do not need the extra dependencies. For instance, the ungrammaticality of the reduced pronoun direct object ‘m in the Vorfeld in (7) shows that the fact that this direct object is also the subject of the embedded verb horen does not bring it the same privileges as being a subject of the finite verb would bring (see Section 2.3 for a discussion of reduced pronouns and Vorfeld occupation).

(7) *‘m Hoor ik zingen.
    him.RED hear I sing
    ‘I can hear him sing.’

\(^1\)It is not clear whether it is always the case that the matrix verb provides enough information to construct additional dependencies. For English, Pollard and Sag (1994, sec 7.4) point out that in cases like (i), it depends on the lower verb which of the arguments of the matrix verb is shared. In (ia), Kim is attending the party, whereas in (ib), Sandy is allowed to attend the party.

(i) a. Kim promised Sandy to attend the party.
    b. Kim promised Sandy to be allowed to attend the party.

Investigating whether, and if so how often, this occurs in spoken Dutch is not the topic of this dissertation, so I will not consider this issue any further. Here, I would just like to point out that considering these extra dependencies as lexically derivable is a slight simplification.
Apparently, only the dependency relation between 'm and hoor is relevant in deciding whether 'm can front. The other, currently not annotated, dependency relation would only add confusion, albeit easily resolved.

A possible downside of the omission of the additional dependencies is that an argument is not always a dependent of its main verb in the annotation. In Section 2.6, I mentioned that some researchers of German word order have claimed that different verbs prefer different base orders of their arguments. In order to investigate this in a corpus, we would have to know of each constituent what its main verb is. However, if the main verb is not the finite verb, a constituent may not be a dependent of its main verb in the CGN annotation. Adding the additional information of what the main verb is, and which dependency relation a constituent has to that verb, may be possible to do automatically but it adds another non-trivial step to the investigation.

**Non-lexically derivable multiple dependencies** There is a second group of constructions in which multiple dependencies may be called for. In this second group, the additional dependencies can generally not be derived from other parts of the annotation. These cases are annotated by the CGN. An example will follow below.

The CGN annotation formally distinguishes primary dependencies from secondary ones (following Skut et al., 1997). The primary dependencies form a tree. The trees that we have seen until now consisted of only primary dependencies. When there is a need for extra dependencies, these can be added as decorations to this tree (secondary edges). Secondary edges are employed for instance in the annotation of the syntactic structure of questions (8). The secondary edge is drawn as a dotted curve. The phrase labels WHMAIN and SV1 refer to main wh-question and verb-initial clause. The relations WHD and BODY hold between a wh-question and its wh-constituent and a wh-question and the non-wh-part, respectively.

(8) Wie heeft haar aanbeden

Who has adored her.

```
(8) WHMAIN
    / \  
   BODY SV1
     / \  
    SU   VC
     | \  |
    WHD HD OBJ1 HD
    VNW WW VNW WW
    Wie heeft haar aanbeden
    who has her adored
```

‘Who has adored her.’
CGN annotates the wh-word/phrase as the head of a wh-question. At the same time, this word may also be an argument in the clause. This argument relation is encoded with the secondary edge. In (8), the wh-word is a subject.

The corpus work in this thesis is not concerned with questions, so the extra annotation can be safely ignored. Other constructions that are annotated with secondary dependencies are (free) relative clauses and coordinations. Since none of these constructions is directly relevant for the Vorfeld in declarative sentences, we can ignore all annotation that uses the secondary dependencies. When we only look at primary dependencies, the annotations will always be trees. Therefore, we can safely assume in our definitions and queries that all dependency structures are trees. We can now turn to the definition that will be central to all corpus work in this thesis: the definition of Vorfeld in terms of the CGN annotation.

### Section 3.3: Finding the Vorfeld in CGN

The CGN syntactic annotation is detailed enough for us to retrieve information about a constituent’s grammatical function, definiteness and length. What remains to be given, however, is a definition of the Vorfeld in terms of the CGN annotation. In this section I will develop that definition.

The Vorfeld is a region in a main clause. What we are after in this section is a definition of whether a constituent is in this region. We can look at this as a relation between a constituent and the main clause that the constituent is in. This relation is separate from grammatical function, and it is not restricted to certain syntactic categories. We have seen in Chapter 2 that nearly anything could appear in the Vorfeld. As a consequence, characteristics such as syntactic function or syntactic category are not very helpful when we try to decide whether a constituent is in the Vorfeld. On identifying topological fields in a corpus that does not annotate for them, Meurers (2005) writes the following.

> Searching for material in fields with less characteristic membership [than the verbal cluster – gjb], such as the fronted material in the Vorfeld, the freely ordered mixture of elements in the Mittelfeld, or extraposed material in the Nachfeld, is practically impossible in a corpus without topological or structural annotation.

Meurers (2005, p1631)

As we will see below, in spite of the fact that the CGN does not have topological annotation, the structural annotation that is present is enough to define the topological Vorfeld on.

Let us start by briefly repeating what the Vorfeld is from the introduction in Section 2.1. In a main clause in Dutch and German, we can define three linearly ordered regions – Vorfeld, Mittelfeld, and Nachfeld – that are separated by verbal material. In the left
periphery, the finite verb in main clauses marks the right boundary of the Vorfeld and is referred to as the *left bracket*. In addition to the Vorfeld and left bracket, the left periphery of a Dutch utterance may contain extra material that precedes the Vorfeld. This material is in a region known as the *lead*. The relation between the CGN syntactic tree and the topological fields in the left periphery is schematically represented in (9).

Phenomena like contrastive left dislocation and hanging topics (see Section 2.4.2), which involve material in the lead, are annotated as super-sentential or discourse phenomena. Therefore the whole utterance is marked as a *discourse unit* (DU). In this unit, the main clause (SMAIN) is the nucleus (NUCL), and the left dislocated material the satellite (SAT). Material in the lead may be words or phrases of various types (for the CGN), hence ‘X|XP’ as indication of the category for the SAT-dependent. The Vorfeld is however part of the SMAIN, so we know that it must be a descendant of the SMAIN node in the tree. What we do not know is at what level below the SMAIN-node the Vorfeld constituent occurs, what dependency relation the Vorfeld constituent will have, or of which syntactic type the Vorfeld constituent will be. Finally, the left bracket is the finite verb in the main clause: It is the head of the clause (HD daughter) and it is a verb (WW).

We can distill two properties that a constituent in the Vorfeld must meet from this description of the CGN annotation: A Vorfeld constituent is a descendant (of any generation) of an SMAIN node, and a Vorfeld must precede the HD of the SMAIN that it is a descendant of. Let us apply this definition to the example in (10). De finite verb is boldfaced.

(10) [SMAIN Jan des Bouvrie’s achternaam *congrueert* niet in getal].

Jan des Bouvrie’s surname agrees not in number

‘There is something odd about JdB’s last name.’

All of *Jan, des, Bouvrie, achternaam, Jan des Bouvrie’s* and *Jan des Bouvrie’s achternaam* are descendants of the SMAIN that precede the finite verb. However, we are only interested in one of them; the largest constituent *Jan des Bouvrie’s achternaam*. Let us dub this the Vorfeld occupant. We now need a way to separate Vorfeld occupants from other Vorfeld constituents.
It is not correct to say that Vorfeld occupants are Vorfeld constituents that span the entire string between the beginning of the sentence and the finite verb. There are cases that have multiple Vorfeld occupants ('|' marks the boundary between the occupants), such as (11).

(11) Later | met de kinderen *gingen we naar Frankrijk.
    later with the children went we to France
    ‘Later with the children, we went to France.’

I will define Vorfeld occupant as a constituent in the Vorfeld that is not dominated by another constituent in the Vorfeld. This deals with both (10) and (11) in the desired fashion. For the latter, this is shown in the syntactic structure in (12).

(12) Of all the constituents in (12) that precede the head of SMAIN, [ADJ later] and [PP met de kinderen] are not dominated by other constituents also preceding the head. These two constituents are therefore Vorfeld occupants.

There is one final issue we need to address, which relates to discontinuous constituents. Consider example (2), repeated here as (13). The subject in the Vorfeld has an extraposed relative clause.

(13) In the CGN multiple Vorfeld occupancy is possible and our definition thus has to be able to deal with this situation. See Section 2.4.3 for discussion of multiple Vorfeld occupancy.
According to the definition thus far, the subject of (13) is not a Vorfeld occupant. The subject does not precede the head of SMAIN because the rightmost word under the subject node, *komt* `comes`, is not before the head of SMAIN but behind it in sentence final position. This has the unfortunate side effect that *ieder* and *meisje* do count as Vorfeld occupants. After all, they are Vorfeld constituents that are not dominated by another Vorfeld constituent.

There are several constructions that involve discontinuous phrases that start in the Vorfeld and end in the postverbal domain, of which we would like to say that the dependent is the Vorfeld. The problem occurs with stranded or extraposed modifiers (13), coordination, and certain discourse-level dependencies. The problematic constituents have in common that some key descendant like the head noun (in the case of a discontinuous NP), or the first conjunct (in cases of coordination) is properly in the Vorfeld.

We will solve this issue by loosening the precedence requirement. Instead of requiring that the complete phrase precedes the head of SMAIN, we will require that at least one of its key dependents does. Since the head noun is a key dependent of the NP in (13), we allow the NP to be a Vorfeld constituent, and thereby a Vorfeld occupant. The determiner and head noun *ieder* and *meisje* are no longer Vorfeld occupants under this definition. Note that we want to be restrictive in what dependent we allow as *pars pro toto*. For instance, an OBJ1 should not fulfill this role: having an object in the Vorfeld does justify classifying the whole VP as a Vorfeld constituent.

The topological notion of Vorfeld can be successfully defined in CGN terms. The definition can be summarized as in (14).

(14) A constituent V is a Vorfeld occupant when
  a. V is a descendant of an SMAIN clause S
  b. and V or a key direct dependent of V precedes the head of S
  c. and there is no constituent that meets criteria (a) and (b) of which V is a descendant

The selection of a corpus search tool, and the implementation of definition (14) in terms that fit this tool is the subject of the next section.

### 3.4 Implementation

The corpus searches in this dissertation will be conducted using the programming language Prolog. This approach was chosen over using existing, dedicated corpus searching tools.

---

3I used SICStus Prolog v3 (SICS, 2005), although nothing relies on this.
tools suited for the CGN like TiGer Search (TiGer, 2003) and DT_Search (Bouma and Kloosterman, 2002) because of a lack of flexibility in the dedicated tools.

DT_Search, developed in the context of the Alpino treebank (Van der Beek et al., 2002), uses the XPath query language to search through an XML representation of the corpus. Although DT_Search has been successfully used in CGN-related corpus studies before (for instance, Bouma, 2004, Van der Beek, 2005), (linguistically) intuitive formulation of more complicated queries is difficult because XPath does not offer a straightforward way of naming specific nodes for future reference, nor does it allow one to easily construct macros of queries. This means that complex queries cannot be easily composed out of less complex ones. Recently, Bouma and Kloosterman (2007) have proposed to use the more powerful XQuery language to solve these issues. An additional problem is that although DT_Search is good at retrieving fragments from the corpus on the basis of a query, it does not provide many facilities to analyze the data after retrieval. For instance, in Section 3.2.1, I explained that information about definiteness of an NP will have to be inferred because it is not part of the annotation of the corpus. This is the kind of classification that one would like to be able to do in a post-processing stage. Such data manipulation and analysis requires separate tools if one uses DT_Search. The more elaborate TiGer Search does provide the possibility to combine simpler queries into a more complex one, but the post-processing available in the programme itself is still fairly restricted. Using Prolog solves all issues mentioned above, since it is a full-featured programming language rather than a corpus search engine that interprets a query language. One can use variables to refer to the same node at different points in the query and one can define sub-functions (actually: relations) as reusable macros to hide complicated parts of the query. Finally, any manipulation of the data for post-processing can be done right inside Prolog.

In Prolog, there is no principled difference between the data, the queries, the processing and the post-processing. To give a concrete example of why this is useful and efficient from a developer’s point of view, consider the concept of dependency paths, mentioned in Section 3.2.1. Recall that in general a dependency path is the list of dependency relations that we cross travelling from a node in the tree up to a node somewhere above it in the tree. Given our assumption that all data is represented as trees, there is always exactly one such path between two nodes. We can use dependency paths in different ways. For instance, we could look in the corpus for pairs of nodes that are connected by a certain dependency path. In this case, the dependency path is part of the query. Alternatively, we might want to know for a given pair of nodes what the dependency path between them is. Here, the dependency path is used in post-processing. If retrieval of nodes from the corpus and post-processing of retrieved nodes is separated from each other, one has to encode the definition twice, possibly using very different tools or languages. In our Prolog setup, we can define the concept of dependency path once, and use it in different ways at different
stages. Furthermore, although specifying a particular dependency path is probably trivial in a query language that is especially tailored to describe linguistic dependencies, it is more of an effort to collect them in post-processing without a dedicated language.

In comparison to many other general purpose languages, Prolog is well suited for corpus work because it is a language that – ideally, and in practice only to a certain extent – allows one to specify the solution to a problem in terms of logic, rather than describing how to get to that solution. This means that in order to find a certain type of linguistic construction in the corpus, one defines what properties the construction should have, and not how to find it. Note that this is also the purpose of having a query language that is interpreted by another programme, such as DT_Search and Tiger Search have. It allows the linguist to just describe the bits of the tree that they are after. The job of the programme is then to find these bits in a corpus.

In order to use the CGN in Prolog, the corpus had to be converted into a Prolog database: a collection of logical representations of facts and relations. Every node in the CGN receives a unique identifier. Associated with these identifiers is information about the node: Which node is its parent, what relation does it have to this parent node, of what category is the node itself, what part of the string falls under the node, etcetera. This representation format is very similar to one of the native formats the CGN comes in, so the conversion is fairly straightforward. In addition, the morphological annotation layer of CGN was merged with the syntactic layer in order to give easy access to detailed morphological information.

There are two concerns that one might have with the approach taken here: memory and speed. Regarding memory, the whole corpus is loaded into memory at once and this puts limits on the size of the corpus. Tools like DT_Search and TiGer Search do not run into this problem because they are capable of searching through corpora that reside on disc. Because the CGN is a moderately sized corpus, and because we are only interested in a subpart of the corpus, the memory limitations were not a problem. If, in the future, we would like to use Prolog to search the large >10mln or even >100mln word corpora that exist, we will have to find a more creative method of accessing the corpus. Secondly, one might be worried about the speed with which the queries are processed. Due to the way the corpus was represented and the fact that there was no file- or disc-access involved in searching the data, performance compares favourably to DT_Search.

The Prolog-version of the definition of Vorfeld occupant (14) is given in (15). A quick walk-through follows below.4

\[
(15) \text{vf}_\text{occupant}(S,V)\leftarrow \text{vf}_\text{constituent}(S,V) \land \\
\quad \neg (\text{parent}(V,D) \land \\
\quad \quad \text{vf}_\text{constituent}(S,D)).
\]
The (a) and (b) clauses of (14) are implemented in the definition of the relation `vf_constituent` (the constituent is in an `smain` and partly or fully precedes the head of `smain`). Vorfeld constituency is a relation between a node `S` and a node `V`: a descendant `V` of an `smain` `S` is considered a Vorfeld constituent when some ‘key part’ of `V` precedes the `smain`’s head daughter. Key part is defined as a separate relation, which holds between any node and itself, or between a node and its head daughter, or between a node and a daughter that is one of the conjuncts in a coordination, or between a node and some other type of child that I have not specified here (hence the ellipsis). The statement that any node is considered to be a key part of itself is not superfluous since nodes need not have any daughters. Finally, Vorfeld occupancy is a relation `vf_occupant(S,V)` between two nodes, such that `V` is a Vorfeld constituent of `S` and there is no parent of `V` that is also a Vorfeld constituent of `S`.

The other relations that are used in this definition need to be specified as well, but this is a trivial exercise. Some are atomic, that is, they are directly part of the corpus specification. Examples of atomic relations are function, parent and category, and for words `begin` and `end`. Others can simply be defined in terms of these: `child` in terms of `parent`, and `descendant` recursively in terms of `child`, and `begin` and `end` for non-lexical nodes in terms of the dominated lexical material.

Querying the corpus and post-processing the results is done in Prolog. In this section I have motivated this choice and I have shown how the CGN definition of Vorfeld is implemented. Searching the corpus and manipulating corpus data is only half of empirical

---

4The statements are to be read as follows: Capitals indicate variables, and variable scope is from left of the `:-` to the full stop `. `. The relation named on the left hand side is considered to hold between its arguments when all conditions on the right hand side are met. In this snippet of code, `p(X)` is to be read as: ‘`X` is a `p`’, and `p(X,Y)` means ‘`Y` is the/a `p` of `X`’. Relation predicates whose name occurs more than once on the left hand side can be proven to hold in more than one way. ‘`key_part(N,N)`’ means: everything is a key part of itself, unconditionally. Finally, anything that cannot be proven is considered false.
work involved in the corpus investigations. The other half is the statistical interpretation of the retrieved data. One of the statistical methods used will be explained below.

3.5 Statistical methods

Of the statistical methods used to analyze corpus data in Chapters 4 & 6, there is one that deserves a brief introduction: logistic regression. Here and in the rest of the dissertation, I am assuming that the reader has some understanding of basic concepts such as probability, variance, and significance testing; and is familiar with contingency tables, Fisher’s Exact test, $\chi^2$ (test), and Wilcoxon’s test. The description of logistic regression given below is purely meant to introduce its main ingredients and to give an intuitive understanding of the method. A reader familiar with logistic regression will not find anything new here, and the unacquainted but interested reader may find the description to be too superficial. In the latter case, I can refer the reader to the relevant chapters of Agresti (1996) and Rietveld and Van Hout (1993), and references therein, for introduction. Agresti (2002) provides a more rigorous mathematical discussion.

A natural way of displaying and analyzing count data is to use contingency tables. However, contingency tables become impractical and hard to interpret when there are more than two or three dimensions, or when the dimensions have many values. When one of the dimensions is continuous, a contingency table cannot be used at all without transforming the data. In all these cases, it may be advantageous to model the data in order to investigate them (Agresti, 2002). That is, we describe the data with a mathematical model, and make generalizations about the data on the basis of the model parameters. The type of model that we will use here, logistic regression, is a model that is suitable for describing data with a discrete dependent (response) variable and discrete or continuous independent (explanatory) variables. If we have a two-valued dependent variable whose chance of having value 1 (meaning true, yes, success, something being the case, etcetera) given the values of the independent variables is $P$, a logistic regression model looks as in (16).

$$\text{logit}(P) = \ln\left(\frac{P}{1-P}\right) = \alpha + \beta_1 x_1 + \ldots + \beta_n x_n$$  (16)

where $\alpha$ and $\beta$’s are the parameters of the model, $1, \ldots, n$ are indices for the independent/explanatory variables, and $x_i$ is the observed value of independent variable $i$ for the instance that we are trying to predict the probability of.

---

5In linguistics, especially socio-linguistics, logistic regression has been used since the 70s in the form of VARBRUL (Cedergren and Sankoff, 1974).
To give a more concrete example, the dependent variable will be Vorfeld occupancy in Chapter 4. For each constituent in the dataset, we predict the probability of membership of the group of Vorfeld occupants on the basis of its properties – grammatical function, definiteness, and grammatical complexity. Of course, a constituent either is a Vorfeld occupant or not. The probability of a single constituent being a Vorfeld occupant is the proportion of all constituents with similar properties that occupy the Vorfeld.

The explanatory variables in a logistic regression model can be discrete or continuous. Continuous variables are used as they are but since we need numbers to do our calculations, discrete variables are mapped. A discrete variable that can take on \( n \) values is mapped to \( n - 1 \) binary variables, each with numerical values 1 and 0. Each of these new variables represents a value of the original variable. The value of the original variable that is not mapped is referred to as the **base level**. It is represented by all the new binary variables being 0. Examples of mappings for a binary variable ‘Winter’ (true when it is winter, false when it is not) and a ternary variable ‘Springmonth’ (with values ‘march’, ‘april’ or ‘may’) are given in (17).

(17) \[
\begin{align*}
&\text{Winter} \\
&\text{true} \Rightarrow \text{Winter} = 1 \\
&\text{false} \Rightarrow \text{Winter} = 0 \\
\end{align*}
\]

\[
\begin{align*}
&\text{Springmonth} \\
&\text{march} \Rightarrow \text{Springmonth/march} = 1, \text{Springmonth/april} = 0 \\
&\text{april} \Rightarrow \text{Springmonth/march} = 0, \text{Springmonth/april} = 1 \\
&\text{may} \Rightarrow \text{Springmonth/march} = 0, \text{Springmonth/april} = 0 \\
\end{align*}
\]

The binary variable Winter \( \in \{\text{true, false}\} \) is represented by one binary variable Winter \( \in \{1, 0\} \). The ternary variable Springmonth \( \in \{\text{march, april, may}\} \) will be represented by two binary variables Springmonth/march \( \in \{1, 0\} \) and Springmonth/april \( \in \{1, 0\} \). The base level of Springmonth is therefore the value ‘may’.

Building a logistic regression model of the data gives us insight into patterns in the data because we can interpret the \( \beta \)-parameters.\(^6\) The \( \beta \)s are weights that indicate how and how strongly each factor contributes. (Significantly) positive values mean that as the value of the independent variable increases, the predicted probability of a positive value of the dependent variable increases, too. Negative values indicate a negative correlation. The size of the weight tells us something about the strength of the factor, and can be understood in terms of **odds ratios**, which may bear some explanation.

Like probability, **odds** is a way of expressing the chance that something might happen. Whereas the probability of success \( P \) relates the occurrence of a certain event to the total

\[^6\]The \( \alpha \)-parameter is the intercept and is thus related to the overall probability of success at the base level of all factors, or put differently, when all \( x \)s are 0. As such it is not directly relevant.
number of events, the odds \( O \) express how often an event occurs in relation to how often it does not occur. So, if a day being rainy has odds of 3 – or 3 : 1 (“3 to 1”) – this means that for every three days of rain, there is one day of sun. Odds and probability can be defined in terms of each other: \( O = \frac{P}{1-P} \). So, odds of 3 : 1 correspond to a probability of .75 or 75%. Probability ranges from 0 to 1 (0%–100%); odds lie between 0 and infinity.

The logit of a probability \( P \) is defined as the (natural) logarithm of the odds \( O \) associated with \( P \). The logit of \( P \) is the left hand side in (17). The odds ratio is the quotient of the odds of two different events: \( O_1/O_2 \). For instance, if the odds of a rainy day in August are 3 : 1, and the odds of a rainy day in July are 2 : 5, then the odds ratio is \( \frac{3/1}{2/5} = \frac{15}{2} \). That is, the odds of it raining in August are 7.5 times the odds of it raining in July.\(^7\) Because odds are always positive, odds ratios will also be positive. An odds ratio of 1 indicates no difference. Odds ratios relate to understanding the \( \beta \)-parameters in a logistic regression model in the following way: The ratio between the predicted odds for two values of variable \( i \) is \( e^{\beta_i} \).

One of the reasons to not directly predict probability from the model parameters, that is, to not use a simple linear model, is that under the logit transformation, the predicted probabilities always fall between 0 and 1. Without the logit, the predictions might fall outside this range, which is conceptually impossible.

Let us turn to a toy example. Imagine a model that predicts whether a day will be rainy during the two summer holiday months July and August. It uses one independent variable (factor) August, which is true (or 1) when the day falls in the month of August, and false (or 0) when the day falls in July. Say we choose the parameters to be \( \alpha = -0.89 \) and \( \beta_{\text{August}} = 1.95 \). If we fill in the parameters in (16), we end up with (18):

\[
\ln \left( \frac{P(\text{rainy day})}{P(\text{sunny day})} \right) = -0.89 + 1.95\text{August}
\]

This model will for a day in July predict that the odds of a rainy day are \( e^{-0.89+1.95*0} = 0.41 \), which corresponds to a probability of \( \frac{0.41}{1+0.41} = 29\% \). The odds of a rainy day in August are predicted to be \( e^{-0.89+1.95*1} = 2.89 \), a probability of \( \frac{2.89}{2.89+1} = 74\% \). The ratio of the odds of a rainy day in August and the odds of a rainy day in July is 2.89/0.41 = 7, which is \( e^{1.95} \).

Estimating the parameters so that the model describes a given dataset as well as is possible with the information present in the used factors is called model fitting. Model fitting is handled by the statistical software (see below). Note that fitted models do not have to describe the data perfectly. It might be that one is missing relevant factors.

\(^7\)Note, by the way, that the odds ratio can differ radically from the quotient of the probabilities, called relative risk. In our example the relative risk is \( \frac{3/(3+1)}{2/(5+2)} = \frac{3}{7} \). When saying something like: “the chance of \( X \) happening is \( n \) times the chance of \( Y \) happening”, we therefore need to know whether we are talking about chance in terms of odds or probability.
(explanatory variables), that there are errors in the data related to the way of measuring, or it might be that there is inherent variation in the data that cannot be explained.

After fitting the model, we can draw inferences from the model by looking at the model and several statistics. The parameter estimates themselves indicate the size and direction of the effect. They can be converted to odds ratios so that we can interpret them more easily. The estimates come with standard errors, which allows us to create confidence intervals for them and their associated odds ratios. *Wald’s test* allows us to test whether the contribution of an effect is statistically significant, by testing whether a coefficient is significantly not 0.

In Chapters 4 and 6, I will report the results of model fitting in tables like Table 3.2. The tables present parameter estimates, 95% confidence intervals for the odds ratios, and *p*-values from Wald’s test. We can therefore read an estimate of the size of each effect (the OR confidence interval) and the significance of the contribution (*p*-values) directly from the table.

A confidence interval is an indication of how certain we are about the true value of whatever we are trying to estimate from a sample. Wide confidence intervals indicate low certainty, narrow confidence interval high certainty. The width of a confidence interval depends on the sample and on the demanded level of confidence. If we have little data or much variation in the data, the confidence intervals will be wide. If we demand a high level of confidence (for instance 99%), the confidence interval will be wider than if we had demanded a lower confidence level (say, 95%).

Let us return to predicting rainy days during the summer holiday. Assume that the climate doesn’t change and that there thus is such a thing as the true ratio between the odds of rain in August, and the odds of rain in July. We can fit a model predicting the chance of a rainy day in summer using the data from one particular summer. The results of fitting such a model on data from the summer of 2006 are given in Table 3.2. The table shows that on the basis of the available data, we are confident at the 95% level that the odds of a rainy day in August (in general) are between 2.2 and 21.5 times higher than the odds of a rainy day in July. The significance of the difference between August and July (represented by the variable August), can be read from the table in two ways. First, the *p*-value that comes from Wald’s test in the last column is very low. Secondly, the 95% confidence interval for the odds ratio August=true/August=false does not include the value 1. Recall that an odds ratio of 1 indicates no difference. We can therefore be certain that the odds of a rainy day in August are higher than the odds of a rainy day in July. If additional assumptions are met – the sample is representative, we did not ignore other relevant factors, the model predicts a variable that can be meaningfully predicted in this way – the logistic regression model would support the general claim that there are more rainy days in August.

---

8These are two ways of reading the result from the table, they do not represent two different ways of testing whether a factor contributes significantly.
Another way of testing for the significance of parameters is to look at the contribution of a parameter in reducing the deviance (error) of the model. Two nested models\(^9\) can be compared using a likelihood ratio test. If the model with a factor included is significantly better than the model without that factor, the extra factor contributes significantly in predicting the dependent variable. Dropping the factor August from the rain-prediction model leads to a worse fit, which means that August helps to explain variation in the data \((G^2 = 13.131, df = 1, p < 0.001)\). The predictive value of the full model is indicated by the \(c\)-index ('\(c\)' for concordance). The \(c\)-index is an indication of how often the model predicts a higher probability of being true for an actual true outcome than for a false outcome. It is 0.5 when the model has no predictive value, and 1 when the model predicts the data perfectly. Finally, it may also be instructive to compare predicted and observed probabilities.

As mentioned, I will use logistic regression models to investigate what factors influence the probability of a constituent being a Vorfeld occupant. In Chapter 4, I will investigate several factors separately (grammatical function, definiteness and grammatical weight), mostly using contingency tables. However, logistic regression is a perfect tool to investigate the extent to which these factors influence Vorfeld occupation independently and the extent to which the effect of factors can be explained from other factors. Take grammatical function and definiteness as an example. We can expect that the majority of subjects is definite. Suppose we find that both being a subject and being definite increases the chance of appearing in the Vorfeld. We would then want to know whether the subject effect can explain the definiteness effect or vice versa. With a logistic regression model, we can answer this question. If both factors turn out to be significantly positive in the model, then we can conclude that the subjecthood and definiteness effects cannot be explained by each other. A reasonable conclusion would be that being a subject as well as being definite contributes towards appearing in the Vorfeld.

Moreover, we can formulate factors to investigate complicated hypotheses that are hard to test without modeling. In Chapter 6, for instance, I will use logistic regression to investigate whether the relation between definiteness of the subject and definiteness of the object influences Vorfeld occupation of the object.

---

\(^9\)Two models are nested when the factors in the first model are a subset of the factors in the second.

---

**Table 3.2:** Predicting rain in the summer holiday of 2006

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>OR (lo–hi)</th>
<th>(p)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\alpha)</td>
<td>-0.8938</td>
<td>1.9499</td>
<td>2.2</td>
</tr>
<tr>
<td>August</td>
<td></td>
<td>21.5</td>
<td>&lt;.001</td>
</tr>
</tbody>
</table>

Although Prolog is well suited to do corpus work with, it is not an obvious choice for statistical calculations. All calculations were done using R, (R Development Core Team, 2006). The logistic regression models were fitted using maximum likelihood estimation. Model fitting and inspection was done with the Design library (Harrell, 2003). An exception is the last model in Chapter 6. To overcome issues with sparse data and highly correlated factors, a special type of penalized maximum likelihood estimation was used, provided by the logistf library (Heinze and Ploner, 2003). See Harrell, Lee, and Mark (1996) and Baayen (forth.) for tutorials and guidelines for good practice. For the other statistics, functions from the standard packages of R where used.

3.6 Summary

This chapter has introduced the CGN and the techniques used to investigate it. In particular I motivated the use of Prolog as a tool in corpus linguistics, and I introduced the statistical method of logistic regression modelling, that will play an important role in Chapters 4 & 6. I have paid special attention to the definition of the Vorfeld in terms of CGN annotation. A satisfactory definition of Vorfeld occupancy can be given, and moreover, we can easily translate this definition into Prolog. The ability to automatically identify Vorfeld constituents on the basis of the CGN syntactic annotation is a prerequisite for the large scale corpus investigations of the next chapters.

With the expectations about Vorfeld occupation that arose in the discussion in the previous chapter, and the tools and definitions that I have introduced here, we are ready to investigate the Vorfeld in spoken Dutch. In Chapter 4, I will present a corpus investigation that will help us answer the question of how a constituent’s properties influence the chance that it is selected as the Vorfeld occupant. The second question to be answered in this dissertation introduced in Chapter 1, is how the chance of communicative success influences the choice of a Vorfeld occupant. Chapter 5 proposes a theoretical model of the influence of communicative success. The tools discussed in the current chapter will be employed to test predictions of this theoretical model in Chapter 6.