Chapter 1

Introduction

In this introductory chapter, we start with a motivation for the corpus-based approach that we adopt in this thesis (section 1.1). We proceed in section 1.2 with some preliminaries about the corpora and statistics that are used, followed by a brief introduction in the theoretical frameworks assumed in section 1.3. At the end of the chapter, we give an outline of this thesis.

1.1 Corpus Linguistics

The chapters of this thesis each report an independent piece of research. The topics and the approach in each chapter vary widely, from purely theoretical linguistics in chapter 2 to machine learning in chapter 5. There is one thing that ties the chapters together: in each of the chapters, we make use of corpus data. The ways in which the corpus data is used vary just as widely as the the topics of the chapters. While corpora play only a supporting role as a natural source of examples in chapter 2, they form a crucial factor for estimating frequencies in chapter 3, identifying particular constructions in chapter 4 and training the classifiers in chapter 5.

1.1.1 Data collection

Even for traditional, theoretical linguistics research, corpus data form a valuable resource. (Electronic) text corpora may help theoretical linguists to find examples that support an analysis or counterexamples to some previously proposed analysis.

It is common practice to make up examples to illustrate a theoretical claim. This methodology allows the linguist to construct simple, short example sentences that abstract away as much as possible from anything that

is not relevant for the discussion.

For major, frequent constructions it is often very easy to construct such examples. For less probable constructions, however, it is often difficult to come up with a natural sounding example. Corpora may provide such examples for us. Although the sentences are often longer than made-up examples, they are more natural and easier to evaluate for grammaticality. As an extra advantage, corpus examples are usually extracted from large texts, so that the context, which may be important for evaluation and interpretation, is provided as well. Abney (1996) shows how grammaticality judgments may overlook perfectly grammatical though improbable parses, especially when presented out of context: although any subject will judge example (1) ungrammatical without the proper context, it is a fine sentence when uttered in the context of a map where large stretches of land are designated by capital letters and subdivided in pieces the size of one are and designated by lowercase letters. This problem is circumvented by the use of corpus data (in context).

(1) The a are of I.

Another advantage of corpus examples, is that they show the relevant construction in various different linguistic contexts. Browsing through these real world examples may reveal influencing factors that are not yet modeled in the analysis. As such, corpus data may drive further development of linguistic theory. In chapter 2, we almost exclusively use corpus examples to illustrate our analysis.

In some cases, the difficulty to come up with an example of a certain construction has lead to the conclusion that this construction must be impossible, i.e. ungrammatical. Corpus study may prove such claims wrong. Bresnan and Nikitina (2003) show that verb classes which were claimed not to participate in the dative alternation do in fact alternate. The alternative structures proved infrequent, rather than ungrammatical. Meurers (2004) discusses various claims in Germanic linguistics, which can be shown false based on corpus evidence. Counterexamples from corpora can also be found in chapter 2 of this book, where we falsify the claim that clefts have "regular" expletive subjects on the basis of corpus examples with demonstrative subjects, and in chapter 3, where we falsify the claim that only reduced object pronouns can shift (Zwart, 1996) on the basis of a number of counterexamples that we retrieved from corpus data.

The extraction of corpus examples usually proceeds in two steps: first the extraction of candidate examples and then the evaluation of the examples. The extraction of candidate examples is done on the basis of a search query.

Sometimes this query can be formulated in terms of the literal words in a sentences, but more often the query depends on more abstract notions such as part-of-speech or grammatical relation. In these cases, linguistic annotation is necessary. By adding meta-level linguistic information to the words in the corpus, linguists are able to search for abstract patterns, such as dative passives or it-cleft constructions. See Meurers (2004) for illustration of the process of retrieving examples via linguistic descriptions in the form of search queries.

In the evaluation step, the good examples are extracted from the total set of candidates. This set will also contain sentences which are grammatically fine, but do not contain the linguistic structure under investigation. For example, when looking for it-clefts in a corpus, one will come across sentences like example (2), where the relative clause is a modifier of the predicate. Annotation errors may produce more false candidates. On top of that, the corpus is likely to contain a number ungrammatical sentences. The number of typos and grammar errors depends on the type of text, but even in heavily edited text one will come across ungrammaticalities: the examples in (3) are from a national newspaper. One can conclude that there is still an important role for grammaticality judgments in a corpus-based approach: separating the true positives from the false positives.

(2) Het is een ontwikkeling die veel aandacht vraagt.
it is a development that much attention asks
It is a development that asks for much attention.

(3) a. *Volgens de politie van Karlsruhe sloeg het voertuig
according-to the police of Karlsruhe went the vehicle
bij een inhaalmanoeuvre over de kop sloeg.
during a take-over upside down went

b. *Hij vind de teruggang van het aantal juristen in het
He find$_{1st}$ the decrease of the number lawyers in the
parlement [...] zorgwekkend.
parliament [...] disturbing

Linguists have often observed that the grammar of any natural language can generate an unbounded number of expressions. As a consequence, no corpus will ever contain all possible utterances of a language. This means that in case an appropriate search query does not yield any candidate examples, we cannot draw the conclusion that the construction is therefore ungrammatical. In order to draw any conclusion from the absence of a certain pattern in a corpus, one has to carefully evaluate the quality and size of the corpus, the frequencies of the individual construction components and the expected

frequency based on this information.

Summing up, corpora provide natural, real world examples for linguistic analyses which are easy to evaluate and which may reveal phenomena that were previously unaccounted for. As corpora also contain the infrequent or improbable expressions that one rarely comes up with while sitting at a desk grazing one's own intuitions, it forms a good test suite for existing assumptions about grammaticality and ungrammaticality. Finally, corpora may also provide some evidence that a particular pattern is not grammatical, although one must handle this evidence with care. As a final note, corpus data is an important source of linguistic evidence, but that is not to say it is the *only* appropriate type of evidence. See for example Wasow (2002, chapter 6) for a discussion of different types of linguistic evidence and how they can complement each other.

1.1.2 Gradient patterns and probability

Language is often modeled as categorical. Something is either a verb or a noun, either grammatical or ungrammatical, either productive or not. This is a useful simplification, which facilitated much linguistic research and has led to advanced symbolic models of language. Even in this thesis, strict categories are used to formulate generalizations. But it *is* a simplification. The idea that language is non-categorical is now dominating the field of computational linguistics. Jurafsky (2003) reported that 77% of the main conference (ACL) papers in 2000 built on stochastic models of language processing or learning. But also in other disciplines of linguistics we find research showing that various aspects of language are gradient. Ross was the first to show, for example, that syntactic categories are not discrete entities: "[L]et me propose that what is necessary is a relaxation of the claim that sequences of elements either are or are not members of some constituent class, like NPs, V, S etc. Rather, I suggest, we must allow membership to a degree" (Ross, 1973). This gradience is present in all areas of linguistics. Bod et al. (2003a) give an overview of probabilistic linguistic phenomena, ranging from language acquisition to language variation and from phonology to semantics.

Here, we will focus on gradient aspects of syntax and the use of quantitative corpus data for syntactic research. Abney (1996) already showed that many constraints on syntactically well-formed structures are in fact probabilistic in nature. For example, English plurals usually do not function as prenominal modifiers. Nevertheless, Abney lists a handful of examples from edited newspaper text, e.g. *the financial services industry, the bonds market*. In chapters 3, 4 and 5 of this book, we will come across various non-categorical phenomena: neither the ordering of objects in Dutch nor the selection for an

NP or PP recipient, a preposition's selection for a determinerless complement or a noun's countability are categorical.

Facing these gradient patterns, there are several strategies to chose from. One can chose to ignore the stochastic aspect and just allow for the most frequent pattern. As a consequence, all sentences containing one of the many less frequent patterns will be rejected. Alternatively, one can allow for all variants. This will lead to massive overgeneration and ambiguity. A third option is a probabilistic model of language, where different realizations are associated with different probabilities. Note that such a system does not exclude the usage of a symbolic grammar: one may simply attach weights to the rules of the symbolic grammar, or augment a symbolic grammar with a statistical disambiguation model, as in for example the Alpino grammar (Bouma et al., 2001, see also section 1.2). For an overview of ways in which probabilities may be incorporated in a grammar, see Manning (2003).

If one commits to research that allows non-categorical linguistic distinctions, corpora are indispensable. Introspective grammaticality judgments do not tell you which variant is most frequent, nor to what degree extra syntactic weight increases the chance of finding extraposition. In this thesis, we investigate various non-categorical phenomena. In chapter 3, we investigate which factors influence the ordering of the direct and the indirect object in Dutch. These factors are determined by making crucial use of quantitative data, such as difference in average weight in the shifted and the unshifted variant, or the frequency of a particular pronoun in each of the alternants. In chapter 4, we look at prepositions that combine with bare singular count nouns to form a determinerless PP, e.g. *per kind* 'per child', *richting huis* 'towards house'. Some of these prepositions combine almost always with determinerless complements, e.g. *per* in both Dutch and English, but others co-occur both with nouns with or without a determiner and some prepositions hardly ever form a determinerless PP. As a last example of gradual patterns discussed in this chapter, we try to determine the countability class of nouns automatically in chapter 5. Although a noun's countability appears to correlate rather straightforwardly with its potential to combine with certain determiners (*much* vs. *many*), we will see that 1) the notion countability in itself is not categorical and 2) the correlation between countability and various syntactic diagnostics for countability is not categorical. As we will see, it is nevertheless possible to classify nouns in countability classes based on their preferences for certain configurations.

Not all linguists have embraced quantitative data in linguistic research. Chomsky is strongly opposed to any notion of probability. "It must be recognized that the notion 'probability of a sentence' is an entirely useless one, under any known interpretation of this term" (Chomsky, 1969, p.57). I will

not discuss all arguments and counterarguments in this discussion, as Abney (1996) and Manning (2003) convincingly argue for the use of probabilities in syntax and against the denial of probabilistic phenomena in language. I restrict myself to saying that none of the linguistic phenomena mentioned above and described in more detail in this thesis can be properly dealt with unless one accepts the existence of gradient patterns in linguistic theory and the use of quantitative data in accounting for these phenomena. Categorical distinctions simply prove empirically incorrect, while a treatment in terms of free variation fails to account for the influence of linguistic factors on the distribution of the alternants. Probabilistic data reveal a structure in language which is more fine grained than the notions 'grammatical', 'ungrammatical' and 'free variation'. This structure builds on linguistic notions such as pronominality and definiteness. A theory of language may be expected to account for the influence of these factors on linguistic patterns, whether categorical or gradient.

1.1.3 Resources for natural language processing

Corpus data is indispensable in computational linguistics. Models of (various aspects of) language are trained on annotated or raw text corpora. For example consider preprocessors, which themselves make corpora more useful by tokenizing them, splitting them up in sentences, or enriching them with additional linguistic information such as part-of-speech or lemma. Although rule-based approaches to these tasks exist, most successful methods are supervised, crucially making use of corpus data for training.

But corpus data is also used to train models for more high level natural language processing: systems have been built to learn complex syntactic structures or even complete grammars from corpus examples (Bod et al., 2003b; Adriaans et al., 2004). In addition, symbolic systems may be complemented with a probabilistic component in order to model the gradient patterns mentioned earlier. The Alpino parser, for example, is based on a symbolic, handwritten grammar, but relies on corpus data for the training of its disambiguation module.

Corpora are also a natural source for other kinds of linguistic information, which may be extracted automatically. Lexicographers use large corpora to identify new words and automatic or semi-automatic methods have been developed to extract for example verb frames (Briscoe and Carroll, 1997; Kermanides et al., 2004), semantically related words (Hearst, 1992; Lin, 1998; Finkelstein-Landau and Morin, 1999; van der Plas and Bouma, 2005), support verb constructions (Villada Moirón, 2004) or collocational prepositions (Bouma and Villada, 2002) from corpus data. In chapter 4 we will use cor-

pora to extract syntactically marked determinerless PPs automatically and in chapter 5 we use corpus data to classify nouns according to their countability class.

1.1.4 Evaluation

Another application of corpus data is evaluation. Manually annotated corpora may serve as a gold standard for natural language models or processors. Interesting as it is to discuss the beauty or the simplicity of a particular grammar of English, the value of a grammar is mainly determined by its capacity to correctly analyze all sentences of a language. This may be tested by applying the grammar to a set of real world sentences for which the correct analysis is known and count the number of mistakes the grammar makes. If the test set is representative, the performance on the test is a measure of the grammar's performance.

Furthermore, corpus data is used to monitor the effect of new additions to a system. As a model grows larger, it rapidly becomes impossible to oversee all consequences of a minor change. However, running the system over the test set will immediately tell you whether it improved or not. For illustration, see the increase in accuracy of the Alpino parser on the Alpino Treebank over a time span of 1000 days in figure 1.1. As the test set is used over and over again, performance on this set is not a representative measure of the absolute performance of the grammar, as it is likely that the data set will influence the work on the model. It should therefore only be used to obtain information about the performance relative to some other point in time.

In this book, we use corpus data for evaluation in chapter 4. By counting the number of determinerless PPs that we recognize as a particular syntactically marked construction, we measure the recall. In combination with an accuracy figure, this gives us an idea of the performance or quality of the system. Evaluation of the classification methods in chapter 5 is performed on an annotated test set of about a hundred hand-annotated nouns which were randomly extracted from a POS-tagged corpus (in addition to dictionary data and automatically classified data).

1.1.5 Corpus linguistics and this thesis

This book illustrates four applications of corpus data for syntactic research: corpora as a source of linguistic examples, gradient patterns in language, the automatic extraction of syntactically marked constructions from corpora and corpus-based classification. As such it formulates four answers to the question "why use corpora in syntactic research?". In particular we show

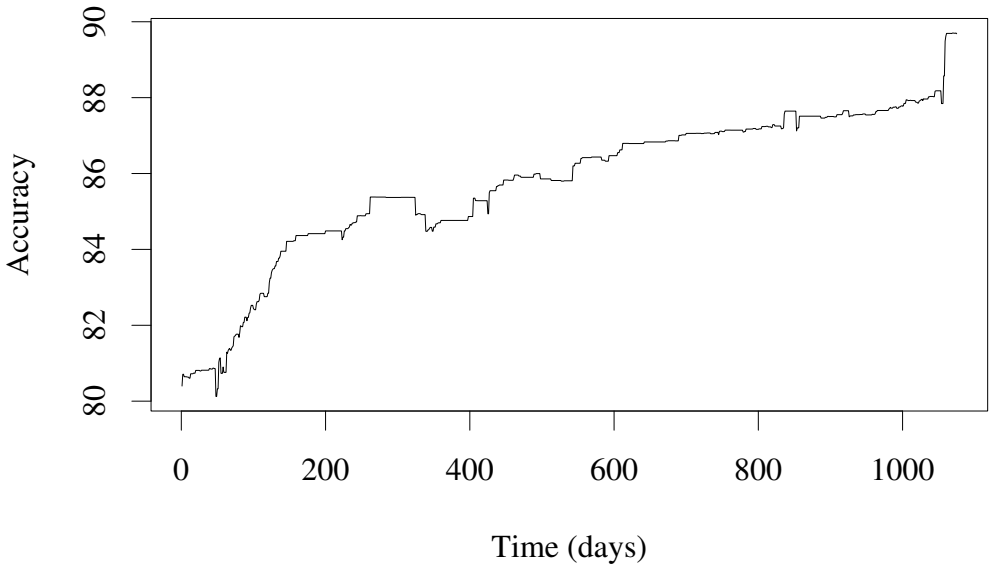


Figure 1.1: Accuracy of the Alpino parser on the Alpino Treebank.

that syntactically annotated corpora, manually built or machine-annotated, offer new opportunities for syntactic research.

The research project that led to this thesis was part of the Pionier program *Algorithms for Linguistic Processing*[1]. This project aims at the development of a broad coverage parser for Dutch. The corpus-based research reported on in this thesis is meant to extend the coverage of the grammar and improve the performance of the parser. The Alpino parser in turn facilitates the development of large, syntactically annotated corpora, which make possible more research into the rules and regularities of Dutch, which will again improve the grammar of the parser. This book thus depends on and contributes to the availability of large, annotated corpora of Dutch.

1.2 Methodological Preliminaries

1.2.1 Corpora

Various corpora are used in this research. First of all, we used unannotated corpus data for finding relevant examples. We used three different national newspaper corpora: the 17M word **Volkskrant97** corpus and the equally large 1994 volumes of *NRC Handelsblad* and *Algemeen Dagblad*, which are both part of the **Twente Nieuws Corpus** (TwNC) [2]. In case these did not provide the required examples, we used the **web** as our last resort.

For most tasks, we relied on annotated corpora. Two good sources of syntactically annotated material were available. First of all, the **Corpus of Spoken Dutch** (CGN), of which about 1M words are syntactically annotated. The annotation consists of dependency structures (Moortgat et al., 2001) and has been manually checked and corrected. The corpus contains spoken Flemish and Dutch of various genres, including (but not limited to) phone conversations, read aloud lectures and classroom discussions. A second manually annotated corpus that we used extensively is the **Alpino Dependency Treebank** (van der Beek et al., 2002a). This is a small size (145K words) corpus, consisting of the cdbl newspaper part of the Eindhoven corpus (den Boogaart, 1975). The sentences have been enriched with dependency structures as output by the Alpino parser (Bouma et al., 2001; van der Beek et al., 2002b, see also below) and familiar from CGN. The automatically generated annotations were all manually checked and corrected if necessary.

As these manually annotated corpora frequently proved too small for certain queries, we additionally made use of automatically annotated corpora.

[1] `http://www.let.rug.nl/~vannoord/alp/`

[2] `http://wwwhome.cs.utwente.nl/~druid/TwNC/TwNC-main.html`

A total of 78M words of newspaper text from TwNC were available (the sections NRC1994, NRC1995, AD1994 and AD1995). Like the Alpino Treebank, the sentences were automatically annotated by the Alpino parser. But unlike the Alpino Treebank, no manual editing was performed. Despite the noise that may be introduced by parse errors, automatically annotated data are a rich source of linguistically relevant information, which one would not be able to extract without the annotation. This thesis illustrates this point amply. However, as the parser may systematically misanalyze a particular construction, one has to use the automatically annotated data with care. Before the data is used for a specific query, we investigate whether the parser makes systematic errors which influence the results. Only if no systematic bias is found, we use the automatically generated data. More information about the Alpino parser in the next section.

In addition to the corpus data, we also made use of **EuroWordNet**[3] (Vossen and Bloksma, 1998) as a resource of linguistic knowledge. EuroWordNet is a manually composed hierarchical network of concepts, populated with words. The nodes are called synsets, as they contain sets of synonyms: words which express the same concept. The hierarchical relations express hyponymy and hypernymy relations between synsets. EuroWordNet furthermore connects the synsets of various languages, facilitating cross-linguistic generalizations.

1.2.2 Tools

Alpino Alpino is a wide-coverage grammar: it is designed to analyze sentences of unrestricted Dutch text. The grammar is based on the OVIS grammar (van Noord et al., 1999), that was used in the Dutch public transportation information system, but both lexicon and grammar have been extensively modified and extended. The lexicon contains about 100,000 entries at this moment (June, 2005) and more than 130 different verbal subcategorization frames are distinguished. Lexical information from the lexical databases Celex (Baayen et al., 1993), Parole and CGN (Groot, 2000) was used to construct the lexicon. Various unknown word heuristics further extend the lexical coverage of the system.

The grammar is rule based. The rules are written in a framework that is based on Head-Driven Phrase Structure Grammar (Pollard and Sag, 1994; Sag and Wasow, 1999). Following Sag (1997) construction specific rules are defined in terms of more general structures and principles. In total, the grammar contains over 330 rules.

[3] `http://www.illc.uva.nl/EuroWordNet/`

The parser usually generates a very large set of analyses for a sentence, upto 161,182 parses for the sentence in (4). A stochastic module selects the most probable parse from this set (Malouf and van Noord, 2004). The disambiguation model is based on a log linear maximum entropy model. In a nutshell, the model counts various features of the parses, e.g. dependency relations, the grammar rules that were used and unknown word heuristics. These features are associated with positive (prefered) or negative (dispreferred) weights. The 'heaviest' parse is then selected as the most probable one. The parser has an accuracy of about 85.5%, measured over the dependency relations.

(4) Aling maakte deel uit van een kopgroep van zeven renners die bijna een ronde voorsprong had op het peloton.
Aling made part out of a breakaway of seven cyclists that almost a lap lead had on the pack
Aling was part of a breakaway of seven cyclists that was almost one lap ahead of the pack.

Dt_search The advantage of syntactically annotated corpora is that one can formulate queries that refer to syntactic rather than lexical units. One can search for 'it'-subjects with plural verb agreement or pronominal indirect objects. The search tool `dt_search` (Bouma and Kloosterman, 2002) facilitates the formulation of these and more complex queries. It is built on top of the more general XML search tool XPath. `Dt_search` queries may refer to POS, dependency relation, word form and string position and. It is thus possible to search for hierarchical relations, such as the verb heading a determinerless PP, or a dative passive (i.e. a noun phrase which is identified both as a subject and as an indirect object).

1.2.3 Statistics

On various occasions, simple statistic measures are used, for instance to quantify the association between two variables. We will briefly discuss the relevant measures.

Entropy Entropy is a measure of uncertainty or unpredictability. A low entropy indicates that there is very little uncertainty about the value of a variable. We apply entropy in the detection of syntactically marked determinerless PPs. For instance, we set a minimum PP-verb entropy, meaning that if we look at the verb that heads a determinerless PP, the entropy must be higher than some threshold value. By setting this minimum, we ensure

that the PP combines with various verbs and effectively exclude fixed PP complements. The formula for the entropy H of a variable X is a weighed average of the probabilities of all possible outcomes, expressed in bits (hence the negative logarithm to the base 2):

$$H(X) = -\sum_{x \in X} p(x) log_2 p(x)$$

Entropy Reduction By adding knowledge to a system, one reduces the uncertainty. The information gain can be quantified by comparing the total entropies of the original system and the final system. This measure is applied to quantify the influence of the verb lexeme and the syntactic category of the direct object on the dative alternations in Dutch.

$$H_{1-2} = |H_1 - H_2|$$

Pearson's Chi Square Test Chi square (χ^2) is a non-parametric test of statistical significance that is applied to contingency tables. It tells you the degree of confidence you can have in rejecting the null hypothesis (i.e. the hypothesis that the distribution is due to chance). In other words: it tells you with some degree of confidence whether or not two variables are independent. The test is applied in chapter 3 to test whether realization of the recipient argument as an NP or a PP is independent of the verb lexeme. χ^2 compares for all cells in the contingency table the observed frequencies (O) with the expected frequencies (E). If this difference is large, we can reject the null hypothesis of independence. The advantage of χ^2 is that it does not assume that probabilities are distributed normally. The disadvantage is that the test is unreliable with small numbers: it should not be used if the total sample size is less than 20 or if the total sample size is between 20 and 40 and the expected value of any cell is 5 or below.

$$\chi^2 = \sum_{i,j} \frac{(O_{ij} - E_{ij})^2}{E_{ij}}$$

Log-Likelihood Ratio The log-likelihood ratio (G^2) is a measure of association, similar to the Pearson's χ^2 test. However, it is less sensible to low-frequency data. Bouma and Villada (2002) showed that log-likelihood outperformed χ^2 on the task of collocational PP extraction. We thus opt for log-likelihood on the related task of extracting syntactically marked determinerless PPs. For two-way contingency tables, the statistic simplifies to the formula

$$G^2 = 2 * \sum_{i,j} O_{ij} log \frac{O_{ij}}{E_{ij}}$$

This ratio is called the *likelihood-ratio chi-squared statistic* (Agresti, 2002).

1.3 Theoretical Framework

Four topics in Dutch syntax are discussed in this thesis: the it-cleft construction, the dative alternation, determinerless PPs and noun countability. We formalize the regularities of these constructions in rules and constraints and we identify the relevant lexical items and features. This linguistic information should be cast in some theoretical framework. The diversity of the topics puts heavy constraints on this framework: it must allow for two constituents mapping to a common syntactic role (in order to account for the Dutch it-clefts), it must allow for gradient patterns and non-categorical distinctions (in order to account for the various linguistic factors influencing the dative alternation) and it should allow for an information rich lexicon (in order to store the determinerless PP and noun countability information).

The basic syntactic framework assumed in this thesis is Lexical Functional Grammar (LFG). This framework separates constituent structure from functional structure, which facilitates the account of constructions where the two levels of syntax do not coincide, as we will see is the case for Dutch it-clefts. It is furthermore a lexicalist theory of syntax. This is crucial for our account of determinerless PPs. As we will see in chapter 4, the lexical specifications of the preposition or the noun determine in which type of determinerless PP the word can participate and what type of modification is allowed. Naturally, the lexicon is also the place to store the results of our noun countability classification efforts in chapter 5.

But classic LFG is a symbolic grammar type which does not allow for gradient patterns. Therefore it does not allow us to express the non-categorical distributional differences that we find in the dative alternation: certain linguistic factors favor realization of the recipient argument as a PP, while other factors prefer the double object construction realization. Optimality Theory (OT) does allow for the modeling of (large numbers of) violable constraints on a particular construction. The stochastic extensions to OT furthermore facilitate the modeling of distributional patterns in linguistic variation, without loosing the linguistic insights. (Stochastic) OT would thus allow us to model our account of the Dutch dative alternation in chapter 3. Fortunately, LFG and OT may very well be combined. Various modes of

combination have been discussed in the literature. These range from variants in which the LFG component is minimal and all the explanatory power is put in the OT component (Bresnan, 2000, 1999, 2002, 2001a; Smolensky and Legendre, 2005), to full-fleshed LFG grammars with OT modules to account for certain preferences (Frank et al., 2001). As we rely on a lexicalist type of grammar, we envisage a model similar to the one described in Frank et al. (2001) and briefly discussed below.

That being said, the results obtained in the following chapters do not rely heavily on any particular theoretical framework. The chapters offer linguistic insight and this information is poured in a framework which allows us to express the relevant notions. But any grammar in any theoretical school will have to account for the agreement facts of Dutch it-clefts and the linguistic factors influencing the dative alternation. And any grammar will need to have access to information about which prepositions and which nouns form determinerless PPs and the countability of nouns. We will come back to this point at the end of section 1.3.4.

In the remainder of this section we will first give a short introduction in LFG, followed by introductions in classic OT, stochastic OT and OT-LFG.

1.3.1 Lexical Functional Grammar

Various good introductions in Lexical Functional Grammar (LFG) have been published in the last couple of years: Bresnan (2001b), Dalrymple (2001), Falk (2001). This introductory section on LFG is not meant to be an alternative to these excellent books; any reader interested in learning LFG, is referred to those works. We included this introductory section in this thesis so that any linguist, regardless his or her theoretical background or prefered school, can read, understand, and evaluate the linguistic analyses in chapter 2 and 4. We therefore introduce the basic LFG terms and concepts below. The topics below are all uncontroversial and generally accepted parts of 'classical' LFG. Novel features or extensions to the theory are discussed in the chapters that follow.

F-structure

LFG separates two levels of syntax: the surface form or constituent structure (c-structure) and the functional structure (f-structure). C-structure represents the overt linear and hierarchical organization of words into phrases, while f-structure represents the more abstract, functional organization of the sentence in the form of grammatical functions such as subject and object.

These two structures are parallel: one is not derived from the other, but both are built in parallel.

The functional information encoded in f-structure takes the form of sets of attribute-value pairs, usually depicted in square brackets. Examples of attributes are PRED for predicate, SUBJ for subject and TENSE for verb tense. A large part of the f-structure attributes encode grammatical relations. These include the argument function SUBJ (subject), OBJ1 (direct object), OBJ2 (indirect object), OBL (oblique), COMPL (verbal complement) and XCOMP (open complement).[4] In addition, there are non-argument functions, such as TOP (topic), FOC (focus) and ADJ (adjunct).

Each attribute has a unique value (**uniqueness condition**). That is, attributes may not have multiple values (but as we will see, it is possible for multiple attributes to have the same value). These values can be atoms, semantic forms, (embedded) f-structures or sets of f-structures. We give an example of each of these possibilities. A sample atomic value is 'past', which is an appropriate value for the attribute TENSE. There is only one attribute which takes a semantic form as its value, and that is the attribute PRED. Semantic forms may be simplex, or complex. A complex semantic form encodes the arguments it selects for. Simplex predicates do not select any arguments. An example of a simplex predicate is 'Bo', an example of a complex predicate is 'sleep<SUBJ>'. As mentioned, f-structures may themselves serve as values. An example of a value which is in turn an f-structure can be found in example (6). The value of the attribute SUBJ is the f-structure in (5).

(5) *Bo*

$$\begin{bmatrix} \text{PRED} & \text{'Bo'} \\ \text{NUM} & \text{sg} \end{bmatrix}$$

(6) *Bo slept*

$$\begin{bmatrix} \text{PRED} & \text{'sleep<SUBJ>'} \\ \text{TENSE} & \text{past} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Bo'} \\ \text{NUM} & \text{sg} \end{bmatrix} \end{bmatrix}$$

Finally, we have values consisting of sets of f-structures. These are the values of the attribute ADJ, or adjunct[5]. As noted by Kaplan and Bresnan (1982),

[4]Our use of OBJ1, OBJ2 and OBL is partly a notational variant on most work in LFG and partly a simplification. In classical LFG, the primary object is labeled OBJ. In addition, OBJ_θ and OBL_θ are used to indicate families of relations, where the θ subscript refers to the semantic role associated with the argument, i.e. source, goal or theme.

[5]There are other attributes which take set values, e.g. conjunctions, but we will not

modifiers, but not arguments, can be multiply specified in a sentence. All specifications are encoded under one attribute in the f-structure as members of a set. For example, in sentence (7), we have two sentential modifiers, *well* and *yesterday*, which form the two members of the set which is the value of ADJ.

(7) *Bo slept well yesterday*

$$\begin{bmatrix} \text{PRED} & \text{'sleep<SUBJ>'} \\ \text{TENSE} & \text{past} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Bo'} \\ \text{NUM} & \text{sg} \end{bmatrix} \\ \text{ADJ} & \{\begin{bmatrix} \text{PRED} & \text{'well'} \end{bmatrix}, \begin{bmatrix} \text{PRED} & \text{'yesterday'} \end{bmatrix}\} \end{bmatrix}$$

Sometimes, two attributes have the same value. If this value is atomic, it is simply repeated, e.g. if both the subject and the object of a sentence are singular. Things are trickier when the value is a semantic form. Since each instance of a semantic form is unique, a duplication of the semantic form should be thought of as having a unique index, indicating that the two are distinct objects. As f-structures always have a predicate, it is not good practice to repeat an f-structure if it is shared between two attributes. Instead, what we use is the notation in example (8), where the line indicates that the structure on one end of the line is shared with the attribute on the other end of the line.

(8) *Bo seemed to sleep*

$$\begin{bmatrix} \text{PRED} & \text{'seem<XCOMP>'} \\ \text{TENSE} & \text{past} \\ \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Bo'} \\ \text{NUM} & \text{sg} \end{bmatrix} \\ \text{XCOMP} & \begin{bmatrix} \text{PRED} & \text{'sleep<SUBJ>'} \\ \text{SUBJ} & \end{bmatrix} \end{bmatrix}$$

All f-structures must meet two welformedness conditions in addition to the uniqueness condition mentioned above: completeness and coherence. **Completeness** tells us that all arguments that are specified in a complex PRED value must be realized.[6] It follows from this requirement that a sentence like:

(9) *Bo seemed

discuss these here.

[6]For a formal definition, see Kaplan and Bresnan (1982)

is ungrammatical, as we just saw that the predicate of *to seem* introduces an argument XCOMP. This argument must be realized in order to meet the completeness condition.

The **coherence** condition furthermore states that argument functions must be governed by the semantic form of the PRED of the f-structure it appears in. That is, we cannot add extra arguments to an f-structure. Thus, example (10) is ungrammatical, as the predicate 'sleep<SUBJ>' only introduces a subject and no object.

(10) *Bo slept the bed

C-structure

The linear and hierarchical organization of words into phrases is modeled in c-structure, usually depicted as a tree. The leaves of the tree are occupied by words. This means that words are the smallest units of a c-structure, and that a c-structure rule cannot refer to any unit smaller than a word. In other words, LFG adheres to the principle of **lexical integrity** as formulated in Lapointe (1980):

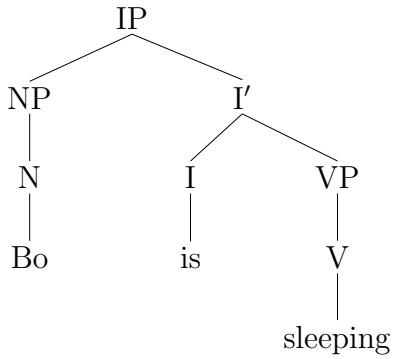
(11) No syntactic rule can refer to elements of morphological structure (Lapointe, 1980, p. 8)

The LFG c-structure assumes four major lexical categories: N(oun), V(erb), A(djective) and Adv(erb). These project the corresponding phrases NP, VP etc. In addition to lexical categories, there are also functional categories such as I and C. In Dutch, I is the category for tensed verbs in main clauses, assuring verb second, and C is used for complementizers. Again, they project the corresponding phrases IP and CP. The difference between lexical and functional categories will be discussed in the next section.

The way in which the categories are organized obeys the general rules of *X-bar theory* (Jackendoff, 1977; Chomsky, 1986). Lexical and functional categories head the phrases they project and combine with complements and specifiers. Two levels of projection are assumed: X′ and XP. As the f-structure welformedness conditions already restrict the space of possible structures, few additional, general restrictions on c-structure exist in LFG. For example, LFG does not assume binary branching: a node may have any number of daughter nodes. Furthermore, any category is optional, even lexical heads. Does this mean that headless constructions are allowed, violating the principle of endocentricity? Yes, although some weaker notion of endocentricity still applies to c-structure (Bresnan, 2001b, p. 133), and the f-structure welformedness conditions ensure that every phrase is headed on

the level of f-structure. Thus, VPs are allowed in Dutch main clauses, even if the only verb is realized in I, and without the need to assume a trace or slash. Although the absence of a strict headedness condition would facilitate the insertion of extra syntactic categories, **economy of expression** tells us that syntactic structure nodes are not used unless required by independent principles such as completeness, coherence or semantic expressivity (Bresnan, 2001b, p. 91), thus preventing the spurious ambiguities that the insertion of void syntactic nodes would give rise to. A sample c-structure is given in (12).

(12) *Bo is sleeping*



Mapping from c-structure to f-structure

C-structure and f-structure are two aspects of one and the same linguistic object. Each node in a c-structure corresponds to an f-structure, as illustrated in figure 1.2. This section describes the way in which this correspondence is established.

The mapping between c-structure and f-structure is defined trough f-structure annotations on c-structure nodes. The terminals of the syntactic tree are lexical items, which are lexically specified with functional constraints. These constraints are passed on to the preterminals and syntactic nodes following general (X-bar based) schemata and c-structure rules with functional annotations. We describe each of the three components (lexical entries, general schemata and c-structure rules) below.

Example (13) shows the lexical entry for the word *Bo*. It specifies the syntactic category of the word to be N. In addition, it constrains the f-structure associated with its mother node, indicated by the ↑, to have an attribute PRED and an attribute NUM with the values 'Bo' and 'sg' respectively.

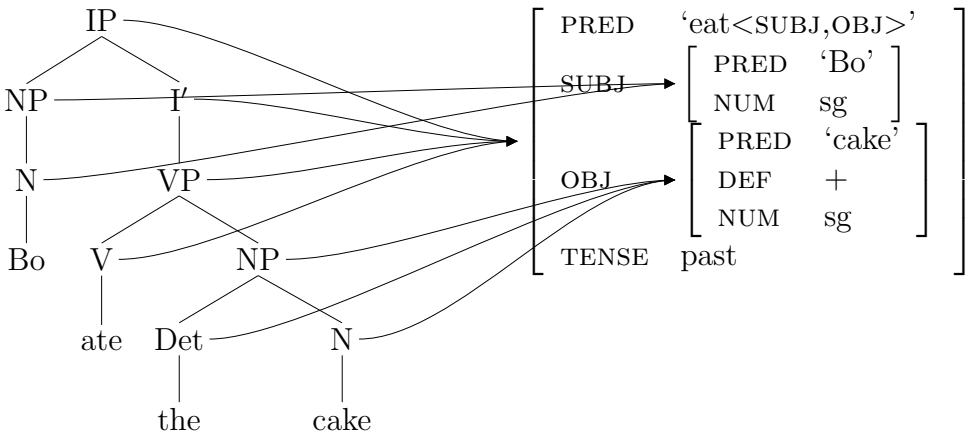


Figure 1.2: c/f-structure correspondence.

(13) *Bo* N (↑PRED) = 'Bo'
(↑NUM) = sg

Lexical entries may also refer to grammatical functions. The lexical entry for the verb *am* in example (14) selects for a subject and an open complement and furthermore requires its subject to be first person singular.

(14) *am* V (↑PRED) = 'be<SUBJ,XCOMP>'
(↑SUBJ PERS) = 1
(↑SUBJ NUM) = sg

The ↑ symbols refer to the f-structures associated with the preterminals. If the preterminal is heading a phrase, it will pass on all functional constraints to the mother node. This follows directly from X-bar theory and the notion of headedness. In LFG this is indicated with the annotation (↑=↓) on the head daughter, saying that all information on the current node (↓) is shared with the mother node (↑). Similar annotation schemata exist for other kinds of nodes. A non-head daughter in a lexical projection is annotated (↑CF)=↓, where CF stands for Complement Function, and CF={OBJ1|OBJ2|OBL}. An example of a non-projecting daughter in a lexical projection is the object NP in figure 1.2. All information on this node will project to a complement function (in this case direct object) of the mother node's f-structure. In functional projections, two daughters project all their information up to the mother: the head and the functional co-head. Examples are the I-node (head) and the VP-node (co-head) in example (12). Non-projecting nodes in functional projections are associated with discourse functions (DF). The annotation is (↑DF)=↓, with (DF={TOP|FOC|SUBJ}). Phrasal nodes may

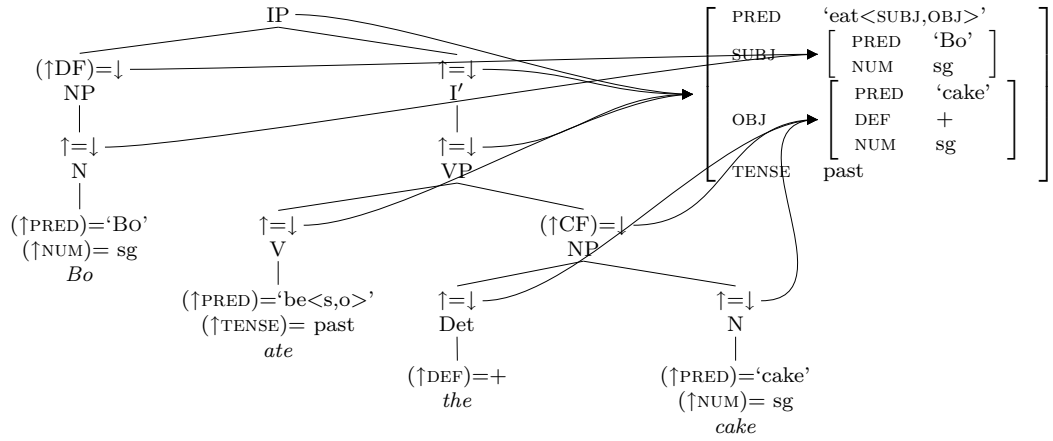


Figure 1.3: c/f-structure correspondence: lexical and predictable annotations.

have non-projecting sisters. These fill the non-argument functions (NAF) TOPIC, FOCUS and ADJ. The annotation on the node: (↑NAF)=↓. So the topic and focus functions may be realized in both lexical and functional projections. Together, these rules give us all predictable annotations. A summary:

(15)
a. A lexical head of a phrase is annotated ↑=↓
b. A functional head is annotated ↑=↓
c. A functional co-head is annotated ↑=↓
d. A non-head daughter in a lexical projection is annotated (↑CF)=↓ (CF={OBJ1|OBJ2|OBL})
e. A non-head daughter in a functional projection is annotated (↑DF)=↓ (DF={TOP|FOC|SUBJ})
f. A non-projecting sister of a phrasal node may be annotated (↑NAF)=↓ (NAF={TOP|FOC|ADJ})

For illustration, we decorated the nodes in the tree from fig. 1.2 with their functional annotations, as shown in fig. 1.3. Note that although the schemata in (15) leave open whether *Bo* is the subject, topic or focus and whether *the cake* is a direct object, an indirect object or an oblique, Coherence and Completeness rule out all other options except the correct one (SUBJ and OBJ1 respectively).

In addition to the predictable annotations, there are unpredictable and language specific annotations, which must be explicitly specified in c-structure rules. For example, in Dutch, a noun indicating some kind of measure may combine with an N′, which indicates the substance. The measure noun acts

as the syntactic head of the construction, which is modified by the substance noun following it. The results is something similar to the N_1 of N_2 construction in English (*a pound of sugar, a bunch of flowers*). To account for this construction, we might write a c-structure rule as in (16).

(16)
$$\begin{array}{lcccc} N' & \Rightarrow & N & & N' \\ & & \uparrow{=}\downarrow & & \downarrow\in(\uparrow \textsc{adj}) \\ & & (\downarrow \textsc{nform})=\text{meas} & & \end{array}$$

Recall that ADJ is a set valued attribute. The annotation $\downarrow\in(\uparrow \textsc{adj})$ defines the f-structure corresponding to the N′-node to be a member of the set of f-structures which is the value of the ADJ. The functional annotation '$(\downarrow \textsc{nform})$=meas' defines the head noun to be of noun type 'meas', i.e. a measure noun. LFG is based on **unification**. This means that parsing (or generating) will succeed unless the noun is otherwise specified for the feature NFORM. If the head noun simply does not have a feature NFORM, it will receive the attribute value pair 'NFORM=meas' as a result of the annotation. We therefore call this type of annotation **defining equations**. If the grammar writer wishes to restrict the application of the rule in (16) to nouns which are predefined as measure nouns, this is possible by using a **constraining equation** instead, indicated by an underscore *c*. The relevant equation would be '$(\downarrow \textsc{nform})=_c$meas'.

Several other options are available for constraining the application of a rule or to add functional information to c-structure nodes. It is possible to formulate a negative constraint, e.g. $\neg(\uparrow \textsc{tense})$ for a non-tensed phrase, or $(\downarrow \textsc{nform})\neq_c$meas for a nominal which is not a measure noun.

Existential constraints simply require that the f-structure corresponding to a node has a particular attribute, without constraining its value. An annotation $(\uparrow \textsc{tense})$ thus requires the dominating node to be tensed.

Optional constraints, printed in parentheses, may or may not be instantiated. They may be used, for instance, to account for clitic doubling: if the clitic is used independently, its PRED value must be 'pro', similar to regular pronouns. However, if it is doubled, it cannot have any predicate, because the NP that it doubles already has a predicate and the two would fail to unify. The clitic is then annotated $((\uparrow \textsc{pred})=$'pro'$)$. Lexical entries with optional constraints can be viewed as shorthand for the combination of two entries: one with the relevant constraint, and the other one without.

Furthermore, it is possible to combine constraints in a conjunction or a disjunction. Example (17) is just a silly way of saying $(\uparrow \textsc{pers})=_c3$, and example (18) is an alternative to $(\uparrow \textsc{pers})\neq_c3$.

(17) $(\uparrow \textsc{pers})\neq_c1 \wedge (\uparrow \textsc{pers})\neq_c2$

(18) $(\uparrow \textsc{pers}) =_c 1 \vee (\uparrow \textsc{pers}) =_c 2$

Finally, it is important to know that it is possible to refer to a large number of paths through an f-structure. The technique that makes this possible is **functional uncertainty**. The technique was introduced by Kaplan et al. (1987). Kaplan and Zaenen (1989) applied it in their treatment of long distance dependencies. We have already seen a simple example of it above: when summarizing the X-bar based annotation schemata in (15), we used abbreviations like CF, DF and NAF for complements functions, discourse functions and non-argument function. Their denotation is a set of grammatical functions, and any member of that set will successfully instantiate the condition. In other words: the exact function was left uncertain. Building on this, it is possible to describe longer paths through an f-structure. Example (19) for example states that the focus of the dominating node fills a grammatical function that is arbitrarily deep embedded in open or verbal complements of this node (describing for example the path between a wh-word and its 'trace'). The star ($\star$) is the Kleene star, indicating *zero or more* of the preceding element. In this case, the preceding element is either an open complement or (|) a verbal complement. GF stands for $\{\textsc{subj}|\textsc{obj}1|\textsc{obj}2|\textsc{obl}|\textsc{comp}|\textsc{xcomp}|\textsc{adj}\}$. Correct instantiations of the complete path would be (COMP OBJ), (COMP COMP ADJ), or (COMP XCOMP XCOMP OBL), for example.

(19) $(\uparrow \textsc{foc}) = (\uparrow \{\textsc{xcomp}|\textsc{comp}\}\star \textsc{gf})$

Not only can we refer to f-structures that are arbitrarily deep embedded, we can also refer to f-structures that enclose the reference point (i.e. in which the reference point is embedded). The technique for this is called **inside-out functional uncertainty**. Again, we explain the technique with an example. In (20), it is stated that the f-structure whose locative oblique is the projection of the annotated node has ergative case.

(20) $((\textsc{obl}_{loc} \uparrow)\textsc{case}) = \textsc{erg}$

Besides c-structure and f-structure, various other levels of syntax have been proposed in the LFG literature, most notably a(rgument)-structure. We will not discuss these here, as they are not relevant to the topics discussed in this book. However, it is clear that mappings between these other structures may take the same form as the f-structure annotations on c-structure nodes that we saw above.

1.3.2 Optimality Theory

From LFG syntax, a new framework has emerged, namely OT-LFG, or Optimality Theoretic LFG. Before we discuss this particular incarnation of OT syntax in the next section, we first describe the basics of 'classic' OT, as it was developed in the field of phonology in the early nineties (Prince and Smolensky, 1993).

In classic OT, the task of generating or interpreting a linguistic object is split up in two subtasks. First a set of candidate realizations or interpretations is generated, and from these output candidates the most optimal is picked in a second step.

The generator component is called GEN. It is a function from the input to the set of output candidates. In phonology, the input consists of an underlying form for generation and a surface form for interpretation. It is assumed that the input is unrestricted. That is, no linguistic explanation may depend on the assumption that some underlying form does not exist in some language. The generator function GEN is furthermore assumed to be universal. Combined with the unrestricted input, this results in a universal set of possible output candidates. This principle is called Richness of the Base (Prince and Smolensky, 1993).

Evaluation of the candidates is based on a ranked set of constraints. The candidate which best meets the constraints, is the optimal (or grammatical) candidate, even if it violates some constraints. The constraints are strictly ranked with respect to each other. If constraint C_1 dominates constraint C_2 ($C_1 \gg C_2$), one violation of C_1 is worse than any number of violations of constraint C_2. Two constraint families are distinguished: markedness constraints and faithfulness constraints. The first requires the output to conform to certain structural patterns. The faithfulness constraints require that the output contains all information from the input and no more. Clearly, the two sets of constraints potentially contradict each other: a markedness constraint may require clauses to have subjects, but if the verb does not specify a subject role, as in the English *rain* and the Dutch *regenen*, this constraint can only be met by introducing a subject which was not in the input, thus violating faithfulness. The set of constraints CON is assumed to be universal. As the input was unrestricted and GEN was also universal, the only source of cross-linguistic variation is the ranking of the constraints, which determines their importance in evaluation.

An OT analysis is usually depicted in a table called tableau, as in fig. 1.4. The tableau only visualizes the evaluation step, as that is the component where all the explanatory power of the model lies. In the upper left cell, we put the input. The relevant constraints are in the upper row, from left to

	[INPUT]	C_1	C_2	C_3	C_4
☞	Candidate 1			*	*
	Candidate 2		*!		**
	Candidate 3		*!		
	Candidate 4	*!	*		*

Figure 1.4: Sample OT tableau.

	[INPUT]	C_1	C_2	C_3	C_4
☞	Candidate 1			*	*
	Candidate 2		*		**
☞	Candidate 3		*		*
	Candidate 4	*!	*		

Figure 1.5: Sample OT tableau with constraint stratum.

right in descending order of importance. In the left column, we list the relevant output candidates. Constraints and candidates which are not relevant to the analysis are left out for clarity and simplicity. Note that one must guarantee that all candidates that are not depicted in the tableau are ruled out somehow. Each cell of the tableau then indicates whether or not a particular candidate violates a constraint. An empty cell indicates that there is no violation, a * indicates a violation and a fatal violation is indicated with a !*. Grammatical output candidates are marked ☞.

Some constraints are equally strong. In this case, they are unranked relative to each other. We call a set of unranked constraints a **stratum**. In a tableau, the constraints in a stratum are not separated by vertical lines. By using strata, it is possible to have two candidates which have the same violation profile. In this case, both candidates are grammatical, and free variation is expected (with both candidates surfacing in about 50% of the cases). Figure 1.5 illustrates optimization with a stratum and two winners.

Constraints may 'gang up' to dominate other constraints. This is called **constraint conjunction**. The conjunction is always higher ranked than all its component constraints. For example, assume we have the following constraint ranking:

(21) $C_1 \gg C_2 \gg C_3$

The conjunction of C_2 and C_3 ($C_2\&C_3$) may outrank the first constraint (22). Constraint conjunction is the only way of modeling cumulativity effects

(multiple violations of lower ranked constraints outweighing one violation of a higher ranked constraint) in classic OT.

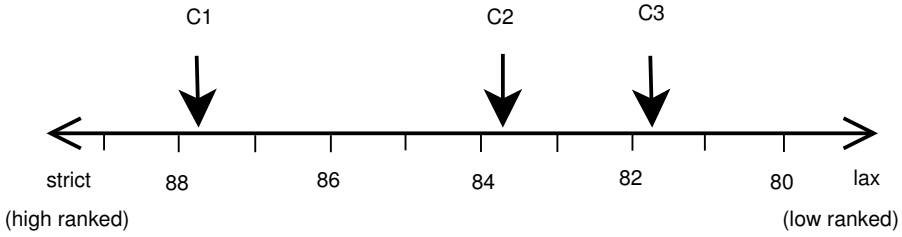
(22) $(C_2\&C_3) \gg C_1 \gg C_2 \gg C_3$

1.3.3 Stochastic OT

Classic OT assumes a strict ranking of constraints (21). Given this strict ranking, C_3 can never dominate C_1 or C_2. Candidates which violate constraints in strata other than the optimal candidate will never surface. Variation is thus only possible if the variants have the same violation profile, i.e. only vary within a stratum, as in table 1.5. In this case we find free variation. However, we will see that other distributions are observed besides the fifty-fifty of free variation. These can be modeled with a stochastic implementation of OT (Boersma and Hayes, 2001).

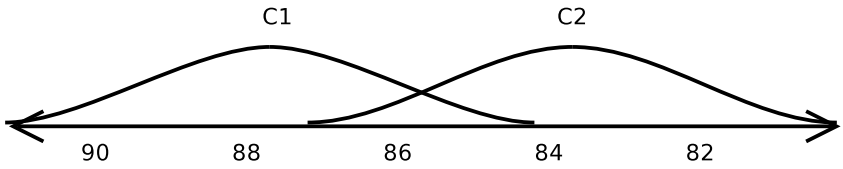
Boersma and Hayes interpret constraint ranking in a different way. First, a linear scale is adopted, as shown in (23). The higher the numerical value of a constraint, the higher the ranking. This allows one to express distances between constraints, e.g. C_1 outranks C_2 more than C_2 outranks C_3.

(23) Categorical ranking on a continuous scale



Furthermore, the candidates are evaluated stochastically. Whenever a candidate set is evaluated, the exact position of a constraint on the scale is determined. This exact numeric value depends on its ranking, but is perturbed by a random variable. This perturbation is a model of noise in the system. The range of possible selection points for a certain constraint is interpreted as a normal probability distribution with the peak at its ranking value. This is illustrated in (24). A small area in the diagram is enclosed by both curves. In this area, it is possible for C_2 to outrank C_1. These alternative rankings may give rise to alternative optimal candidates. The probability of the alternative ranking is thus a inverse function of the distance between two constraints. This probability quickly approaches zero if constraints are further away from each other, modeling categorical distinctions.

(24) Stochastic evaluation of constraint ranking



1.3.4 OT and LFG

Although OT has its roots in phonology, it has been applied to other fields of linguistics, including syntax. The application of OT to the field of syntax raises an important question: what is the input? In phonology, the input was assumed to be the underlying form of a word (for production) or its surface form (for interpretation). But what is the underlying form in syntax? LFG-OT has a straightforward answer to this question: the underlying form is an underspecified f-structure. This f-structure represents the main grammatical information that a given utterance expresses, but abstracts away from morphological or lexical (language specific) information.

GEN is then a function from underspecified f-structures to pairs of c-structures and the corresponding (fully specified) f-structures. The candidates' f-structures are all subsumed by the input f-structure. Faithfulness constraints relate the input f-structure to the output f-structure, markedness constraints and alignment constraint determine the shape of the c-structure.

For illustration, we included a (slightly edited) example from Kuhn (2002) in fig. 1.6. It shows how the input f-structure for a simple question is mapped to an infinite number of candidates, of which only three are depicted in the illustration. The candidates are pairs of c-structures and f-structures. Each pair violates some of the relevant constraints in (25)-(27), which were coined in Bresnan (2000).

(25) Op-Spec: An operator must be the value of a DF [discourse function] in the f-structure.

(26) Ob-Hd: Every projected category has a lexically filled [extended] head.

(27) Stay: Categories dominate their extended heads.

The first candidate violates Op-Spec, as the question operator is not associated with any discourse function. *Will* functions as the extended head of I′ in candidate 3, but cannot function as the extended head of C′ in candidate 2, as all nodes dominating the extended head should also dominate the projection. Thus Candidate 2, but not candidate 3, violates Ob-Hd. Candidate 3 does violate the constraint Stay, as I′ does not dominate its (extended)

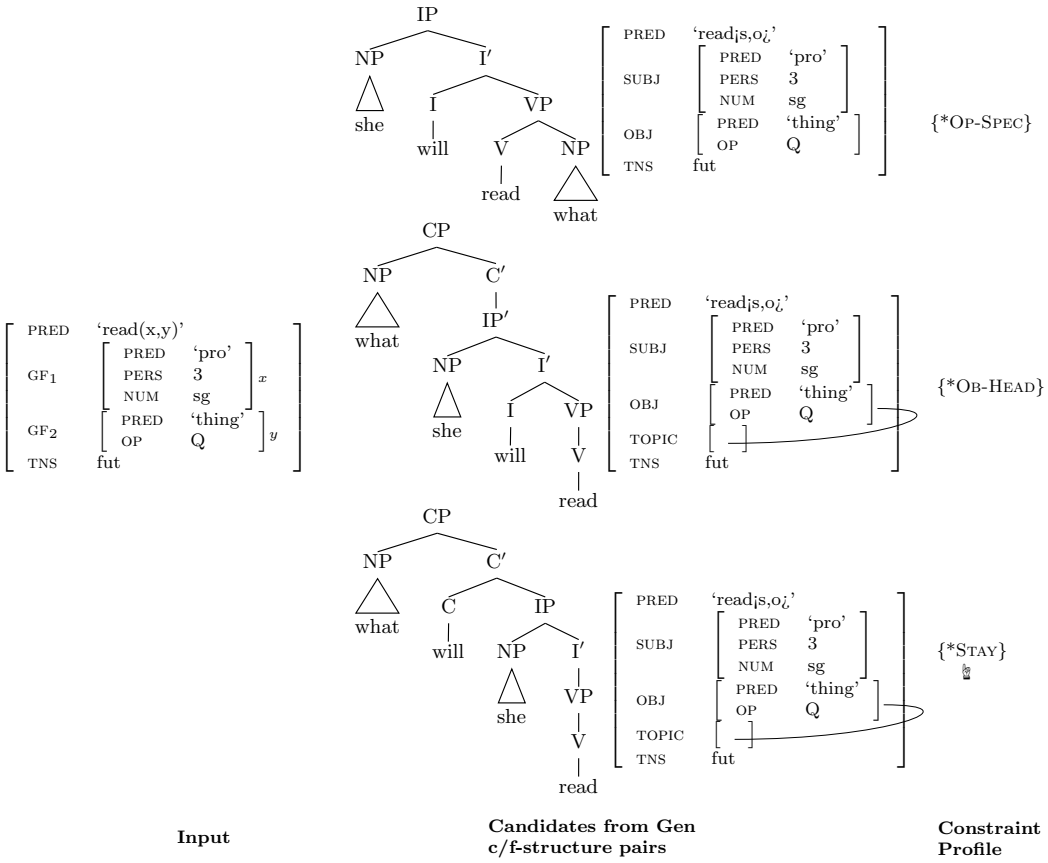


Figure 1.6: General architecture of the OT-LFG framework.

head. Given that the ranking of these universal constraints in English is OP-SPEC≫OB-HD≫STAY, the third candidate is the optimal one.

The OT-LFG framework is relatively new, and various aspects of the framework are still under discussion, including the role of the lexicon and the function of GEN. Bresnan advocates a version of OT-LFG in which the candidates do not contain lexical information Bresnan (2000, 1999, 2002, 2001a). Instead, they are bundles of features, which are decorated with lexical material only after optimization. This ensures that the principle of Richness of the Base is left intact. Van der Beek and Bouma (2004) claim that such an interpretation fails to account for various linguistic phenomena, and instead argue for a function GEN which maps an input *and a language particular lexicon* to a set of output candidates.

In classic OT, the only source of cross-linguistic variation is the ranking of the constraints. Depending on the view on the lexicon, this may or may not be augmented with lexical differences. In both cases, the question arises where that leaves GEN. Radically different answers have been formulated to this question. Kuhn (2003) describes an implementation of an OT-LFG system. The base grammar GEN includes all and only universally inviolable constraints on syntactic structure. In practice, these are the principles of X-bar theory. This leaves intact the principle of a universal GEN (although a language particular base lexicon is used, in line with van der Beek and Bouma (2004), and the candidates are thus language particular). Frank et al. (2001) describe a radically different, but OT-LFG based syntax model. They allow for a full-fleshed language particular LFG grammar. This grammar is complemented with a system for marking a candidate parse as good or bad, based on a ranked set of constraints. Evaluation then proceeds as usual: the candidate with the least important constraint violations wins and is considered grammatical. For an elaborate discussion of the differences and similarities between this variant and 'classic' OT-LFG, we refer to the original paper (Frank et al., 2001).

We have seen various different ways of modeling violable syntactic constraints. In OT and OT-LFG it is a core feature of the framework. In other systems, such as the Alpino parser, violability is restricted to the probabilistic disambiguation module. These are merely different ways of modeling the same concept: not all constraints in natural language are hard constraints. Johnson (2002) for example shows that categorical OT-LFG systems are very similar to probabilistic language models such as Maximum Entropy Models. In this thesis, we model violable constraints as OT constraints. Such a model allows us to illustrate and control the interaction of various constraints and it is linguistically more insightful than a probabilistic black box. However,

the same constraints may well be formulated or implemented in stochastic grammar components.

1.4 Overview

Chapter 2 argues that the Dutch it-cleft construction is in fact *two* distinct constructions. Analyses are developed for both types of it-clefts, accounting for the complex agreement facts without violating the strict subject-verb agreement rules of Dutch. Corpus data is used for examples that illustrate the characteristics of the construction, and for counterexamples to alternative analyses. For example, when we argue that the focused element should be analyzed functionally as part of the relative clause in intransitive clefts, we illustrate this with a corpus example of a cleft in which the focus fulfills an argument function of the embedded verb. Examples of clefts with demonstrative subjects contradict the claim that clefts have expletive subjects.

Chapter 3 investigates the different realizations of the Dutch dative alternation. While in English the syntactic category of the recipient and the order of the two objects alternate together (direct object followed by PP recipient or indirect object followed by direct object), the two may alternate separately in Dutch. We investigate the hypothesis that general alignment principles influence the argument order alternation, and the verb lexeme influences the NP/PP alternation. This hypothesis is proved partly wrong: general alignment principles are shown to influence the order of the arguments and the verb lexeme only influences the NP/PP alternation, in line with our hypothesis. But pronominality, often assumed to influence the order of the arguments, also influences the category of the recipient, and weight does not influence the order of the arguments in the midfield. The influence of linguistic factors on the distribution of the alternants in the dative alternation is quantified by means of simple statistics.

Chapter 4 focuses on a phenomenon that has received little attention in the linguistic literature so far: determinerless PPs. It is shown that nouns which usually require a specifier may occur without one in certain PPs. Various types of determinerless PPs are distinguished and possible accounts are sketched. For all these accounts it is necessary to know which nouns and which prepositions may occur in which type of detless PP and what their modification potential is. We automatically extract this information from automatically annotated corpora.

Chapter 5 is concerned with the automatic classification of nouns as countable and/or uncountable. We first develop a corpus-based approach, applying both monolingual and crosslingual classification. We then experi-

ment with a second classification strategy, which is based on EuroWordNet, a semantic ontology. We conclude that the corpus-based method outperforms the ontology based method.

Finally, in chapter 6 we summarize and conclude. We briefly point to some directions for future research.