Types and Sorts
*Resource Logic for Feature Checking*

Proefschrift
ter verkrijging van de graad van doctor
aan de Universiteit Utrecht
op gezag van de Rector Magnificus, prof. dr. H.O. Voorma
ingevolge het besluit van het College van Promoties
in het openbaar te verdedigen
op vrijdag 18 juni 1999
des middags te 16.15

door

Dirk Heylen

Geboren op 4 oktober 1963 te Bouwel
Promotor: Prof. Dr. M.J. Moortgat

Published by
Thesus
P.O. Box 53292
2505 AG The Hague
The Netherlands
phone: +31 70 3595110
fax: +31 70 4480177
e-mail: mail@hagpub.com
http: www.thesus.com

Thesus is a subsidiary of Holland Academic Graphics, The Hague.

ISBN 90-5569-103-4
ISSN LOT Series 1389-9473
NUGI 941

Copyright © 1999 by Dirk Heylen. All rights reserved.

Printed in The Netherlands.
# Contents

Abstract 5

I Classification and Composition 7

1 Grammars and Categories 11
   1.1 Phrase Structure Grammars 11
   1.2 Categorial Grammars 15
   1.3 Semantics 20

2 Multi-modal Categorial Grammars 25
   2.1 The Logic of Parts and Wholes 25
   2.2 Structure 32
   2.3 Mixed-Multimodal 36
   2.4 Alternative Proof Presentations 40

3 Generalised Phrase Structure Grammars 45
   3.1 Feature Structure Theories 45
   3.2 Feature Structures and Grammars 53
      3.2.1 Subsumption-based Grammars 54
      3.2.2 Unification-based Grammars and Beyond 56
      3.2.3 Stand-Alone Feature Structure Grammars 60

4 Compare and Combine 69
   4.1 Global Grammar Architecture 70
   4.2 Comparing the Frameworks 72

II Options and Obstacles 87

5 Options for Category Refinement 93
   5.1 Bach's Extended Categorial Grammar 93
   5.2 Categorial Unification Grammar 96
   5.3 Extended Lambek Grammar 101
   5.4 First-order Terms 103
      5.4.1 AB-Version 103
      5.4.2 Lambek version 104
   5.5 Fibered Approaches 105

6 Underspecification in categorial grammars 111
   6.1 Unification-based Categorial Grammars 111
   6.2 Type-Logical Grammars 115
   6.3 Shortcomings 121
III A Resource-Conscious Approach 125

7 A Logic for Feature Checking 129
7.1 Feature checking ........................................ 129
7.1.1 The Framework ....................................... 129
7.1.2 Cross-classification ................................. 132
7.1.3 Discussion ............................................ 133
7.2 Underspecification ....................................... 136
7.3 Distribution ............................................. 138
7.3.1 Distribution Postulates ............................. 138
7.3.2 Distribution and Inclusion ......................... 140
7.4 Multiple Modalities .................................. 142

8 The Use of Feature Checking 145
8.1 Checking and Polarity ................................. 145
8.1.1 Logical Options for Checking .................. 146
8.1.2 Lexical Anchoring of Checking Configurations 147
8.2 Sortal Structure ....................................... 156
8.2.1 Multiple Attributes ............................... 156
8.2.2 Internal structure .................................. 158

IV Underspecification Reconsidered 171

9 Coordination and the Limits of Underspecification 181
9.1 Coordination of unlike types ....................... 181
9.2 On-line underspecified neutral factors ........... 185
9.2.1 Type-Logical analysis ............................ 185
9.2.2 Subsumption versus unification ................ 187
9.2.3 The limits of off-line underspecification ........ 190
9.3 Ambiguity and underspecification ................ 192
9.3.1 Overgeneration ..................................... 192
9.3.2 No underspecification ............................ 194
9.3.3 Off-line underspecification and ambiguity .... 197
9.4 The use of underspecification ...................... 199

Conclusion 203

Samenvatting in het Nederlands 205
Abstract

This thesis is about how formal devices can be used in a grammar to define certain aspects of linguistic description. The general goal is to propose a framework in which the refined logical perspective on the specification of linguistic composition of type-logical grammars is paired with a refined classification schema that includes techniques to organise grammatical information. These techniques should provide the same functionality as (typed) feature structure systems offer to constraint-based grammars, including cross-classification, underspecification and the possibility to formulate general feature distribution principles. We argue that to achieve this goal it is not necessary to integrate feature structures as they are typically found in a constraint-based approach into a type-logical setting. We show that the type-logical framework already provides the necessary tools that can be used for the same tasks.

This thesis consists of four parts. The first part presents an analysis of the functions and components of a grammatical framework. It introduces the central topics and contrasts the type-logical categorial approach with phrase structure approaches of the constraint-based type. It provides an introduction to the frameworks that are being investigated and to the issues involved in the combination. A first general issue concerns an important difference between the frameworks. Whereas the type-logical grammars try to define a dedicated, resource- and structure-conscious logic that captures linguistic properties as characteristics of logical constants, the constraint-based grammars make use of a general purpose logic in which a linguistic theory is formulated. This difference is important when the issue is considered of how the two frameworks could be related (or combined). A second general issue concerns the (empirical) motivation behind extensions to the type-logical framework: what is it that one might want to achieve with a combination. We argue that, from an empirical point of view, it is only the morphosyntactic properties that need a more fine-grained description technique than they receive in traditional type-logical accounts.

In Part II we summarise and evaluate several proposals for combining ideas from the categorial and the constraint-based tradition to refine the classification structures of the former and we discuss the obstacles that are typically encountered in this endeavour. Analysing the different proposals, we compile a list of desiderata for an optimal system. None of the systems discussed meets all of them.

In Part III we define a resource and structure-conscious perspective on cross-classification and feature checking using the multi-modal version of categorial grammar. Underspecification and the possibility to express lexical generalisations form important issues in the discussion. This system combines the benefits of the various systems presented in Part III.

In Part IV we evaluate the merits of the mechanism of underspecification defined in the previous part and combine this with an evaluation
of the use of underspecification in constraint-based grammars. We show the advantages of using the fine-grained logical instruments of type-logical grammars for the analysis of coordination constructions. In this part we also point out that one should carefully distinguish between different cases of underspecification. This leads us to reconsider the relation between the type-logical and the constraint-based grammars. We present the contours of a framework that combines aspects of both approaches to provide an optimised mechanism for expressing linguistic generalisations.

Acknowledgments Thanks are due to Michael Moortgat, not just for acting as promotor but also for the way he did this. I have very much enjoyed working with him, starting about 12 years ago in Leiden. His insights into and opinions on (computational) linguistics and the linguistic community, science in general, colleagues, the Dutch and other such topics often served as a valuable traveller’s guide. I thank the members of the thesis committee Gosse Bouma, Bob Carpenter, Jan van Eijck, Jan Landsbergen, and Anton Nijholt for reading the manuscript and providing many valuable comments. Thanks also to Pim van der Eijk and Ton van der Wouden not only for their willingness to act as panaimf but also for reading the manuscript and for all their support in other ways. For all kinds of advice, for mental support (and for the coffee), I thank the colleagues in Utrecht: Raffaella Bernardi, Herman Hendriks, Heleen Hoekstra, Wolf Huijsen, Esther Kraak, Anne-Marie Mineur, Paola Monachesi, Richard Moot, Louis des Tombe, Willemijn Vermaat and Kees Vermeulen. For technical support Hendri Hondorp and Jules van Weerden. For questions and comments at conferences and workshops: Claudia Casadio, Philippe de Groote, Mark Hepple, Ruth Kempson, Geert-Jan Kruijff, Natasha Kurtonina, François Lamarche, Jim Lambek, Alain Lecomte, Glyn Morrill, Dick Oehrle, Gerald Penn, Guy Perrier, Mario Piazza, Christian Rétoré, and Ed Stabler. Finally, I thank Renée Pohlmann for caring and encouragement, for reading the manuscript, but above all, for making life taste so good in every way.
Part I

Classification and Composition
Introduction to Part I

A grammar is a device that defines a language. The nature of the device, the way it is used to define a language, and the precise notion of what is a language is open to variation. In this first part we present several options for each of these aspects. More particularly, we contrast several types of phrase structure grammar and of categorial grammar and see how such grammar frameworks differ in defining two of their central tasks: (1) classifying expressions into equivalence classes and (2) describing how complex expressions are composed out of smaller expressions.

This introductory first part serves a number of functions. Our main aim is to introduce the main topics and themes of this book and to provide motivation for some of the proposals made in later chapters. An important secondary aim is to provide the necessary background material about the basics of the phrase structure and the categorial framework.

In the first chapter we provide an elementary introduction to basic phrase structure grammars and categorial grammars. The former use atomic symbols, called non-terminals or categories, to refer to classes of expressions. The grammars consist of a lexicon and a set of rules. The lexicon states for each word to which classes it belongs. We say that it assigns categories to words. A rule, like $NP \rightarrow \text{DET} \ N$, states that expressions consisting of the concatenation of some expression in $\text{DET}$ and an expression in $N$ are in the category $NP$. The rule component is thus concerned with the specification of composition (what classes can combine and in what order) and classification (what is the category of the combination). Categorial grammars, on the other hand, have a richer notion of category. Besides atomic categories, they also use complex categories of the form $A/B$ and $B\backslash A$ (where $A$ and $B$ are again categories, possibly complex themselves). A language is defined solely by a lexicon that assigns categories to words. There are no phrase structure rules. Instead, a pair of general combination operations is used: (1) an expression in $A/B$ concatenated with an expression of category $B$ gives an expression in $A$, (2) an expression in $B$ concatenated with an expression in $B\backslash A$ gives an expression in $A$. Notice that these schemata do not mention specific categories, but use variables over arbitrary categories $A, B$. The information about precisely which expressions combine with others is provided by the lexicon. For instance, a determiner, such as the, might be assigned the category $NP/N$, which, by the general schema, means that it combines with expressions in $N$ to form expressions in $NP$. Again, from this elementary summary, it should be clear how classification and composition are defined in this alternative framework. One thing that is important is that the classification of expressions by the categories directly expresses their combination properties.

This first chapter already shows some of the possible variation in the kind of devices that can be used to define a language. Also in this first chapter, we define several notions of what one could take a language to
be. We are interested in languages as sets of signs, i.e. form-meaning pairs. Although our concerns in this book lie mainly with the first component of this pair, we will also briefly indicate how an account of the second can be provided by the two grammar frameworks considered.

From the basic categorial grammars we move to more refined versions in the second chapter. We introduce type-logical versions of categorial grammars in which the combinatorial schemata are formulated as logical rules with / and \ defined as logical operators (connectives). Categories should thus be thought of as logical formulas. We show how the logical rules for the connectives can be fine-tuned to express the specific characteristics of linguistic composition. The analogy between the combination schema above (from \(A/B\) and \(B\) construct \(A\)) and the modus ponens rule (from \(B \Rightarrow A\) and \(B\) infer \(A\)) is obvious. Similarly, there are type-logical versions of hypothetical reasoning or conditionisation. The rules of inference have to be tuned to the resource and structure-sensitive nature of linguistic composition: the order, the number and the structure of premises are all important. We are dealing with substructural or resource-conscious logics in this linguistic setting. We show how multiple logics can be defined which differ in their structural characterisation and how they can be combined. This provides a device that can characterise a landscape of structural options with respect to linguistic composition.

In the kinds of phrase structure grammar that form the topic of Chapter 3, the categories are no longer atomic symbols but have internal structure as well. In this case the aim is to define a notion of category that is able to refer to multiple properties of expressions. Feature structures are used to model categories which will enable the classification of expressions along a number of dimensions. Also in this chapter, we will see that feature structures need not be restricted to the representation of categories but can be used to encode other information, like phrase structure itself.

We not only show how these structures can take on the role just mentioned of cross-classification, but also how they are used to structure linguistic information in various ways. More specifically, we discuss the use of multiple inheritance and related techniques for feature structure grammars to capture certain linguistic generalisations. We will look more in particular at Head-Driven Phrase structure Grammar (HPSG).

In Chapter 4, we contrast the two grammatical frameworks. In the discussion of the type-logical grammars we focus on how they offer a refined logic of composition, whereas in the discussion on the feature-based grammars we emphasise the introduction of more delicate ways to capture a logic of classification. In Chapter 4 we turn to a discussion of the complementary and contrasting aspects of these approaches.
Grammars and Categories

On a general level, we are concerned with the use of formal devices used in grammatical frameworks for the description of natural languages. In this chapter we introduce some basic frameworks to point out the major tasks of grammars and the devices that are used to carry them out.

In order to be able to phrase more precisely the problems that will be addressed, we start by rephrasing some elementary definitions of terms like language and grammar from formal language theory thereby isolating the basic concepts that we will be concerned with in this and following chapters. For similar definitions and further explanation about technical terms and notation we refer to a basic textbook like Partee et al. (1990). In fact, we assume familiarity with the concepts so that reading this presentation should merely involve getting attuned to some notational conventions and slightly non-standard formulations that make the comparison between different systems easier.

In formal language theory a language is defined as a set of strings, i.e. a set of finite sequences of vocabulary items. A grammar for a language then is a formal device that defines which strings belong to that language. We will start with such an elementary definition of language and look at two types of grammar: a context-free grammar (Section 1.1) and a simple categorial grammar (Section 1.2). The concept of what constitutes a language is further refined by taking into consideration the notion of category, imposing a classification on the set of strings (expressions), and a more refined notion of structure, reflecting the construction of complex expressions from their constituents.

The comparison of the two grammar frameworks will concentrate on the way they classify expressions and on the way they define complex structures. However, it should be noted that as a model for natural languages this definition of language falls short in that it ignores that expressions are not just syntactic structures but that they carry meaning. In a more adequate model, linguistic objects are form-meaning pairs (signs). We will briefly indicate how the two approaches to grammatical description can be extended to take this aspect of linguistic description into account (Section 1.3).

1.1 Phrase Structure Grammars

One particular kind of formal system that is used to define a language is commonly known as a context-free phrase structure grammar.
Definition 1 (Context Free Grammar) Given two finite, disjoint sets of symbols \( V \) and \( C \), a context free phrase structure grammar is a pair of finite sets \( \langle \text{Lex}, \text{Rules} \rangle \), where \( \text{Lex} \subseteq V \times C \) and \( \text{Rules} \subseteq C \times C^* \).

We will refer to the set \( V \) as the vocabulary. Elements in the set will be called words. They are also commonly called terminal symbols. An element of \( C \) is called a category. They are the non-terminal symbols. By \( C^* \) we mean a sequence of categories. We use the common notation for rules and write \( c \rightarrow c_1...c_n \) for a rule \( \langle c, c_1...c_n \rangle \). Contrary to usual definitions of context-free grammars we do not introduce a distinguished category as the initial (or start) symbol.

We will refer to words and sequences of words as expressions. We can define expressions using the Backus-Naur notation to be the least set \( E \) such that:

\[
E ::= V \mid (E \circ E)
\]

We have used \( \circ \) as an explicit sequence or concatenation operator. We will often simply leave it out. The concatenation operation is usually defined to be associative as expressions are taken to be strings. We will mostly leave out the brackets. This recursive definition should be read as follows: the set \( E \) consists of \( V \) and pairs of elements from \( E \) that are combined by \( \circ \).

A language can be defined as a set of expressions. We will often take a language to be a set of pairs \( \langle E, C \rangle \), where \( E \) is an expression and \( C \) a category. In the definition of language below we use variables \( s_i \) to denote sequences (= strings) of words. Variables \( c, c_i \) stand for categories.

Definition 2 (A CFG-Language) The language, \( L_G \subseteq E \times C \) as defined by the context-free grammar \( G = \langle \text{Lex}, \text{Rules} \rangle \) is the smallest set such that:

(i) \( \text{Lex} \subseteq L_G \)

(ii) if \( \langle c, c_1...c_n \rangle \in \text{Rules} \) and \( \langle s_1, c_1 \rangle \in L_G, ..., \langle s_n, c_n \rangle \in L_G \) then \( \langle s_1 \circ ... \circ s_n, c \rangle \in L_G \)

This defines how a context-free grammar is used to specify a language. The relation between grammar and language is often defined in terms of a rewriting system (Chomsky (1957)). In such a system each step consists in rewriting a string of elements from \( V \cup C \) to another string of elements from this set. At each step a category \( c \) in the string is replaced, either by another string of categories \( c_1...c_n \), provided \( c \rightarrow c_1...c_n \in \text{Rules} \), or by a word \( w \), provided \( \langle w, c \rangle \in \text{Lex} \). For each sequence of rewriting steps that starts with a single category \( c \) and arrives at an expression \( s \), (a string consisting of elements from \( V \) only), we say that \( \langle s, c \rangle \in L_G \). Notice that we do not assume a single distinguished category as our start symbol in this case either.

We will now define a few other notions to provide the terminology to refer to various components in this relation.
**Definition 3 (Interpretation of a category)** The interpretation (denotation) of a category \( c \) in a grammar \( G \), written as \( \nu(c) \) (or \( \nu_G(c) \)), is the set of expressions \( s \) such that \( \langle s, c \rangle \in L_G \).

Note that categories are called *category labels* by Gazdar et al. (1985). They reserve the term *category* for its interpretation.

**Definition 4 (String Set)** The string set \( S_G \subseteq V^* \) defined by a grammar \( G \) is defined as the set of strings \( s \) such that there is a \( c \in C \) with \( \langle s, c \rangle \in L_G \).

Another important concept which we want to define is that of phrase structure. We first define a more general notion, that of *tree*. We consider trees that are labelled with categories (internal nodes) and words (leaves).

**Definition 5 (Trees)** The set of trees, \( T \), is defined recursively to be the smallest set such that:

1. \( V \times C \subseteq T \)
2. if \( t_1, \ldots, t_n \in \mathcal{T} (n \geq 0) \) and \( c \in C \) then \( \langle t_1 \ldots t_n, c \rangle \in \mathcal{T} \)

We will refer to trees that are defined by a grammar as *phrase structure trees*.

**Definition 6 (CFG-Phrase structures)** The set of phrase structure trees \( T_G \subseteq \mathcal{T} \) defined by the grammar \( G = \langle \text{Lex}, \text{Rules} \rangle \) is the smallest set such that:

1. \( \text{Lex} \subseteq T_G \)
2. if \( \langle c_1, c_2, \ldots, c_n \rangle \in \text{Rules} \) and \( t_1 = \langle t_{11} \ldots t_{1m}, c_1 \rangle \in T_G \), ..., \( t_n = \langle t'_{n1} \ldots t'_{nm}, c_n \rangle \in T_G \) then \( \langle t_1 \ldots t_n, c \rangle \in T_G \)

An important difference between strings and phrase structures is that whereas string concatenation is assumed to be associative, trees are bracketed structures. The latter thus preserve aspects of the compositional (constituent) structure or derivation which is lost in the string representations.

**Example 1 (Grammars and Languages)** We consider a grammar for a small fragment of English: \( G = \langle \text{Lex}, \text{Rules} \rangle \), with vocabulary \( V \) and categories \( C \).

- \( V = \{ a, the, cook, pizza, baked \} \),
- \( C = \{ \text{DET}, \text{N}, \text{NP}, \text{S}, \text{V}, \text{VP} \} \),
- \( \text{Lex} = \{ \langle a, \text{DET} \rangle, \langle \text{the}, \text{DET} \rangle, \langle \text{cook}, \text{N} \rangle, \langle \text{pizza}, \text{N} \rangle, \langle \text{baked}, \text{V} \rangle \} \)
- \( \text{Rules} = \{ \text{S} \rightarrow \text{NP} \text{VP}, \text{NP} \rightarrow \text{DET} \text{N}, \text{VP} \rightarrow \text{V} \text{NP} \} \)

Among the elements of the language \( L_G \) are \( \langle \text{the}, \text{DET} \rangle \) because this is in the lexicon, and \( \langle \text{the cook baked a pizza}, \text{S} \rangle \) which is in the language by the repeated application of the second condition in Definition (2) of language.
In a similar way we can show that \( T_G \) contains the following tree which we also depict in the usual graphical way.

\[
\{\langle \text{the} \rangle, \langle \text{cook}, \rangle, \langle \text{baked} \rangle, \langle \text{a} \rangle, \langle \text{pizza} \rangle, \langle \text{NP} \rangle, \langle \text{VP} \rangle, \langle \text{S} \rangle\}
\]

As the following table shows, we can also derive that the sentence is in the language by the rewriting procedure mentioned above.

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td>VP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td>V</td>
<td>NP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DET</td>
<td>N</td>
<td>V</td>
<td>NP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DET</td>
<td>N</td>
<td>V</td>
<td>DET</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>N</td>
<td>V</td>
<td>DET</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>cook</td>
<td>V</td>
<td>DET</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>cook</td>
<td>baked</td>
<td>DET</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>cook</td>
<td>baked</td>
<td>a</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>the</td>
<td>cook</td>
<td>baked</td>
<td>a</td>
<td>pizza</td>
<td></td>
</tr>
</tbody>
</table>
The left column represents the sequence of categories and words that is arrived at by replacing one of the categories (c) (identical to the left-hand side of the rule in the second column) on the line above by the right-hand side of the rule or by a word that is assigned the category C by the lexicon.

**Conclusion**  Looking at this simple recapitulation of some definitions of grammar and language, we can isolate some of the functions a grammar is required to serve. In the definitions above we have presented grammars to define several sets of objects: (i) a set of expressions (the string set), (ii) a set of pairs of expressions and categories (language), (iii) a set of phrase structures. More abstractly we can say that a grammar is a device that is concerned with two aspects.

(1) Defining the membership of elements to some (sub)set: *classification* or *categorisation*.

(2) Specifying the *compositional* structure of complex elements.

The grammars that we introduced categorise expressions in two steps. The lexicon component *Lex* of a grammar deals with the categorisation of the atomic expressions (the words), whereas the *Rules* determine the category of complex expressions. The rules also take care of the definition of compositional structure. They define which parts can combine to form larger structures. Categories play an important role in defining the compositional structure because they group together the set of expressions that behave similarly with respect to compositional structure. In other words, the rules make reference to the categories, not to the individual expressions.

Besides classification, another important function of a grammar is to define structure. Analysing this aspect, we can say that the context-free grammar fixes the order of (sub)expressions (precedence) and a part-whole structure (dominance). Phrase structure trees make this compositional structure and the categorisation of wholes and parts explicit. We will now turn to another type of grammar that performs the same tasks in a different way.

### 1.2 Categorial Grammars

We present an alternative way to define classification and composition by simple categorial grammars. In this type of grammar the lexical component takes over the role of the rule component in fixing the compositional structure of a language. We will see how, in order to effect this, the notion of category is modified.

Among the primary tasks of a grammar that we mentioned, is a characterisation of the classes that expressions fall into and a specification of how expressions are constructed out of others. These two aspects are intimately
connected in the categorial approach to grammatical description. Whereas in a phrase structure grammar the categories are simply atomic symbols, the categories in a categorial grammar have internal structure reflecting the combination possibilities of the expressions they denote.

**Definition 7 (Categories/Types)** The set of categories is defined inductively relative to a finite set of (atomic) basic categories \( B \) and a set of operators called *connectives*. In the simple categorial grammars of this section we use the connectives: / and \. The set of categories \( C \) is defined as follows.

\[
C ::= B \mid (C/C) \mid (C\backslash C)
\]

Categorial grammars can differ in the set of basic types that are assumed and in the precise set of connectives. We will often omit the outermost brackets around complex categories. In order to distinguish categories in a categorial grammar from categories in a phrase structure grammar, we will sometimes refer to the former as *types*. In categories \( A/B \) and \( B\backslash A \) we refer to \( A \) as the *range* and \( B \) as the *domain* subcategory.

We assume the same definition of language, expression and lexicon as in Section 1.1. Just like categories of the phrase structure grammar, the types are used to classify expressions. In categorial grammars, they provide information about the compositional properties of the expressions directly. The structure of complex types provides information which in a phrase structure is expressed in a rule. An expression \( e_2 \) in \( C_1\backslash C_3 \) can combine with an expression \( e_1 \) in \( C_1 \) to form an expression \( e_1 \circ e_2 \) of type \( C_3 \). Similarly, an expression \( e_1 \) in \( C_3/C_2 \) can combine with an expression \( e_2 \) in \( C_2 \) to form an expression \( e_1 \circ e_2 \) of type \( C_3 \).

We will now provide some definitions for a basic type of categorial grammar which we will refer to as AB (for Adjukiewicz (1935) and Bar-Hillel (1964)).

**Definition 8 (AB-grammar)** Given a set of types \( C \) (as defined in Definition 7) and a set \( V \) (of words) an AB-grammar, \( G \), consists of an assignment of types to words \( (\text{Lex}) \): \( G = \text{Lex} \subseteq V \times C \).

**Definition 9 (AB-language)** The AB language \( L_G \subseteq E \times C \) is defined by the AB-grammar \( G \) (= \( \text{Lex} \)) to be the smallest set such that conditions (i-iii) hold.

(i) \( G \subseteq L_G \)

(ii) if \( \langle s_1, C_3/C_2 \rangle \) and \( \langle s_2, C_2 \rangle \) are in \( L_G \) then so is \( \langle s_1 \circ s_2, C_3 \rangle \),

(iii) if \( \langle s_2, C_1\backslash C_3 \rangle \) and \( \langle s_1, C_1 \rangle \) are in \( L_G \) then so is \( \langle s_1 \circ s_2, C_3 \rangle \).

In these combinations, the first category \( C_3/C_2 \) or \( C_1\backslash C_3 \) is called the *functor* and the second is called the *argument*. Rule (ii) represents rightward application and rule (iii) leftward application. The definition of the string set is identical to Definition 4.
Definition 10 (AB-String set) The set of expressions or string set $S_G \subseteq E$ defined by a grammar $G$ is defined as the set of strings $s$ such that $(s, c) \in L_G$ for some $c \in C$.

Definition 11 (Interpretation of a category) The interpretation of a category $c$ in a grammar $G$, written as $v(c)$ (or $v_G(c)$) is the set of expressions $s$ such that $(s, c) \in L_G$. For the complex categories this means the following.

\[
\begin{align*}
v(c_1/c_2) &= \{ e \mid \forall e_2 \in C_2[e \circ e_2 \in v(c_1)] \} \\
v(c_2\setminus c_1) &= \{ e \mid \forall e_2 \in C_2[e_2 \circ e \in v(c_1)] \}
\end{align*}
\]

Whereas the categories in the phrase structure grammar as such tell us nothing about their distributional properties but leave this aspect to the rule-component, the categorial types encode information about their combination potential. The relation between the information expressed by complex categories and the information expressed in phrase structure rules can be made explicit as follows. For each type $c_3/c_2$ we need a corresponding phrase structure rule: $c_3 \rightarrow C c_2$, where $C$ is some category symbol corresponding to the type $c_3/c_2$ and similarly for types $c_1\setminus c_3$. In a categorial grammar we can do without these rewrite rules. We only need something like a general rule schema that corresponds to the semantics of the types we gave before. So instead of all these rules we could have rule schemata like the following, corresponding to clauses (ii) and (iii) of Definition 9 above. For all types $X, Y$:

\[
\begin{align*}
X & \rightarrow X/Y Y \\
X & \rightarrow Y Y\setminus X
\end{align*}
\]

The proof of the weak equivalence between this type of categorial grammar and context-free grammars can be found in Bar-Hillel et al. (1960). For more discussion see Buszkowski (1997).

The schemata are language or grammar independent. This means that the only language specific statements a grammar contains are the assignments of types to lexical expressions: $G = Lex(\subseteq V \times C)$.

AB-grammars can also be used to define phrase structure trees. We first define a general notion of AB-tree.

Definition 12 (AB-Trees) The set of AB-trees, $T$, is defined recursively as the least set such that:

(i) $V \times C \subseteq T$.

(ii) if $t_1$ and $t_2 \in T$ and $c \in C$ then $(\langle t_1, t_2 \rangle, c) \in T$. 

These AB-trees are similar to the phrase structure trees defined before. They differ from these in two ways. The nodes in the tree are decorated with categorial types and branching structures are always binary (condition ii). We will refer to the second component of a tree \( t \) as the \textit{category of} \( t \). We can now define the (AB-) phrase structure trees defined by an AB-grammar.

\textbf{Definition 13 (AB-Phrase structures)} The set of AB-phrase structure trees \( T_G \) defined by the grammar \( G \) is defined as the smallest set such that:

(i) \( G \subseteq T_G \)
(ii) if \( t_1 \) and \( t_2 \) \( \in T_G \) and the category of \( t_1 \) is \( c_3/c_2 \) and the category of \( t_2 \) is \( c_2 \), then \( \langle t_1, t_2 \rangle, c_3 \rangle \in T_G \)
(iii) if \( t_1 \) and \( t_2 \) \( \in T_G \) and the category of \( t_1 \) is \( c_1 \) and the category of \( t_2 \) is \( c_1 \setminus c_3 \), then \( \langle t_1, t_2 \rangle, c_3 \rangle \in T_G \)

\textbf{Example 2 (AB-fragment)} We consider a small fragment of English defined by the following grammar \( G \), with vocabulary \( V \) and basic categories \( B \).

\( V = \{a, \text{the}, \text{baked}, \text{cook}, \text{pizza}\} \)
\( B = \{N, NP, S\} \)
\( G = \text{Lex} = \{a, \text{NP}/N, \langle \text{the}, \text{NP}/N \rangle, \langle \text{baked}, \text{NP}/S/NP \rangle, \langle \text{cook}, N \rangle, \langle \text{pizza}, N \rangle\} \)

Among the elements of the language \( L_G \) is \( \langle \text{NP}/N, \text{the} \rangle \) because this is in \( G \). It is easy to show that \( \langle \text{the cook baked a pizza}, s \rangle \) is in \( L_G \).

1. \( \langle a \circ \text{pizza}, NP \rangle \in L_G \) because
   \( \langle a, \text{NP}/N \rangle \in L_G \) (Lex)
   \( \langle \text{pizza}, N \rangle \in L_G \) (Lex)
   clause (ii) of Definition (9)

2. \( \langle \text{baked} \circ a \circ \text{pizza}, NP \setminus S \rangle \in L_G \) because
   \( \langle \text{baked}, (NP \setminus S)/NP \rangle \in L_G \) (Lex)
   \( \langle a \circ \text{pizza}, NP \rangle \in L_G \) (1)
   clause (ii) of Definition (9)

3. \( \langle \text{the} \circ \text{cook}, NP \rangle \in L_G \) because
   \( \langle \text{the}, \text{NP}/N \rangle \in L_G \) (Lex)
   \( \langle \text{cook}, N \rangle \in L_G \) (Lex)
   clause (ii) of Definition (9)

4. \( \langle \text{the} \circ \text{cook} \circ \text{baked} \circ a \circ \text{pizza}, s \rangle \in L_G \) because
   \( \langle \text{the} \circ \text{cook}, NP \rangle \in L_G \) (3)
   \( \langle \text{baked} \circ a \circ \text{pizza}, NP \setminus S \rangle \in L_G \) (2)
   clause (iii) of Definition (9)

An important difference between this derivation and the phrase structure one is that in the categorial case there are no references to individual rules (because there are none) but only to the general combination clauses. The combination restrictions originate from the type-assignments to the
lexical items. This reflects the lexicalised character of categorial grammars, where the language specific information is provided in the lexicon.

The AB-phrase structure corresponding to \(<\text{the cook baked a pizza}, S>\) can be represented as a tree as well.

Ignoring the leaves of the tree, this structure is similar to a Prawitz style natural deduction derivation for propositional logic (Prawitz (1965)). The inference step as exemplified in the following subtree is licensed by clause (ii) of Definition 9.

From a proof-theoretical perspective, we can thus formulate clause (ii) from the definition as a general rule of inference:

This rule of inference parallels the following modus ponens step from propositional logic:
A basic type corresponds to a propositional variable and a complex type \( c_3/c_2 \) corresponds to an implicational formula: \( c_2 \supset c_3 \). In the propositional case, the order of the premises of the inference rule are not important. In the categorial case, however, we want to introduce order-sensitive inference rules, corresponding to the order in which expressions combine. This means that the propositional implication connective, \( \supset \), has 2 counterparts in the categorial logic, / and \( \setminus \), and two rules of inference. The other rule of inference — the reformulation of clause (iii) from Definition 9 — looks as follows.

\[
\begin{array}{c c c}
C_1 & C_1 \setminus C_3 \\
\hline
C_3
\end{array}
\]

The logical rules and the corresponding derivations can be presented in several ways. In the following chapters we will mostly use the so-called Gentzen-style natural deduction formulation. The variant that we will be using shows clearly how expressions are built up step by step. The lexical steps are notated like this: \( \text{the} \vdash NP/N \) or \( \text{cook} \vdash N \). The rules of combination are similar as in the Prawitz-style when we look at the categories that are to the right of the turnstile. The left-hand side of the conclusion combines the left-hand sides of the premises in accordance with Definition 9.

\[
\begin{array}{c c c}
\text{the} \vdash c_3/c_2 & \text{cook} \vdash c_2 \\
\hline
\text{the} \circ \text{cook} \vdash c_3
\end{array}
\]

**Summary** In the preceding paragraphs we have shown how the basic tasks of a grammar such as the definition of membership of expressions to a language, their classification into categories and a specification of their compositional structure, is accounted for in simple categorial grammars.

We have pointed out the major difference between the categorial and the phrase structure grammar. The internal structure of the categories expresses the combination potential of expressions. In a phrase structure grammar this information is expressed by the rules. The mechanisms responsible for combination in a categorial setting are (1) a language independent application schema, which can be rephrased as a direction sensitive rule of inference (modus ponens) and (2) the type-assignments to the lexical elements.

### 1.3 Semantics

**Signs** The discussion so far has paid attention only to the syntactic aspects of linguistic description. Our definitions of 'language' have been kept
very simple, referring only to expressions as strings of words or other elementary notions like phrase structure trees. A more accurate model of natural language should, however, also take into account that expressions are used to convey information. In this study we will not have to say much about semantics. However, we will present here the outlines of the techniques that can be used to capture meaning in the frameworks that we presented above. For more details we refer to Hendriks (1993) and Carpenter (1999).

The expressions, as we have used them, are meant to represent the prosodic form of natural language objects. Semantic forms can be represented by lambda terms from some suitable variant of the $\lambda$-calculus. We will take semantic forms to be formulas from the simply typed $\lambda$-calculus.

**Definition 14 (Language)** We define a language as a set of signs, i.e. triples $(E, C, M)$, where $E$ is an expression, $C$ a category and $M$ is a semantic form.

For our purposes, we can assume an elementary definition of typed $\lambda$-terms. The definitions are provided only to introduce notation. We first define the types.

**Definition 15 ((\(\lambda\))-Type)** Given a set of symbols $B_\lambda$, the basic types, we define the set of ($\lambda$)-Types, $\text{Typ}$, as follows:

$$\text{Typ} ::= B_\lambda \mid \text{Typ} \to \text{Typ}$$

As variables over types we will use Greek letters like $\sigma$ and $\tau$, possibly subscripted by a natural number. A type of the form $\sigma \to \tau$ is called a functional type.

**Definition 16 ($\lambda$-Term)** For each $\lambda$-type $\tau$ we assume a set of constants, $\text{Con}_\tau$, and a denumerably infinite set of variables, $\text{Var}_\tau$. The set of $\lambda$-Terms, $T_\lambda$ can be described as follows (note that we leave out the $\lambda$ subscript), where the terms must be typed appropriately as indicated.

$$T^\sigma ::= \text{Con}^\sigma \mid \text{Var}^\sigma \mid (T^\sigma \to T^\sigma)$$

$$T^{\sigma \to \tau} ::= (\lambda \text{Var}^\sigma. T^\tau)$$

A term of the form $((\lambda x^\sigma. t_1^\sigma \to t_2^\tau) t_2^\tau)$ can be converted by $\beta$-conversion to another term $t_1[x \mapsto t_2]$. The notation indicates the term $t_1$ in which every free occurrence of $x$ is replaced by the term $t_2$.

These lambda-terms are semantic forms that are interpreted on a frame (the semantic domains of the basic types). For instance, consider $B_\lambda = \{e, t\}$ (entities and truth values), and $A$ a domain of entities, then we can take the domains of interpretation for typed terms to be as follows ($D_\tau$ stands for the set of possible denotations of terms of type $\tau$).
A model is an ordered pair consisting of a frame and an interpretation of the constants (a function having the set of constants as its domain and as range the set of possible denotations corresponding to the type of that constant). Together with assignments of values to variables this takes care of the interpretation of the terms.

We will now update the grammars presented in the previous sections with semantic information.

**Phrase Structure Grammar** In the case of phrase structure grammars, changing the grammar to account for the semantic component means changing the lexicon (Lex) and the rules (Rules). Instead of pairs of expressions and categories, we now have triples, which we write as follows: $E < C > T$. For the example we gave before, the change in the lexicon has the following result. The vocabulary CON, VAR, TYP that is used, can easily be recovered from the example.

```
Lex:
  a  <DET>  a\text{\textsuperscript{n-\textit{np}}}
  the <DET> the\text{\textsuperscript{n-\textit{np}}}
  cook <N>  cook\text{\textsuperscript{n}}
  pizza <N> pizza\text{\textsuperscript{n}}
  baked <V> baked\text{\textsuperscript{np-\textit{(np-s)}}}
```

We use abbreviated terms. The term baked\text{\textsuperscript{np-\textit{(np-s)}}} will be abbreviated to b in the derivations. We also have to change the format of the rules. Note that the computation of the semantic term of complex constructions is specified for each rule separately (or rule-to-rule, see Bach (1976)).

```
Rules:
  s_1 \circ s_2 < S > (t_2 t_1) \rightarrow s_1 < NP > t_1
  s_2 < VP > t_2
  s_1 \circ s_2 < NP > (t_1 t_2) \rightarrow s_1 < DET > t_1
  s_2 < N > t_2
  s_1 \circ s_2 < VP > (t_1 t_2) \rightarrow s_1 < V > t_1
  s_2 < NP > t_2
```

The following derivation shows that the string *the cook baked a pizza* of category s with semantics ((b (a p)) (t c)) is in the language defined by the grammar above. The leaves of the derivation are formed by lexical elements, each further step is licensed by a rule. The branches of the tree correspond to the right-hand side of the rule, the root corresponds to the left-hand side of the rule. We have abbreviated the semantic terms even
further and performed $\beta$-reduction immediately when applying a rule. We have also abbreviated the prosodic forms for *cook* and *pizza*.

\[
\begin{array}{c}
\text{the} \triangleleft \text{DET} \triangleright t \\
\text{c} \triangleleft \text{N} \triangleright c \\
\text{baked} \triangleleft \text{V} \triangleright b \\
\text{a} \triangleleft \text{DET} \triangleright a \\
\text{p} \triangleleft \text{N} \triangleright p
\end{array}
\]

\[
\begin{array}{c}
\text{the} \triangleright \text{c} \in \text{NP} \triangleright (t \text{c}) \\
\text{baked} \triangleright \text{a} \in \text{VP} \triangleright (b \text{a} \text{np})
\end{array}
\]

\[
\text{the} \ triangleright \text{c} \ triangleright \text{baked} \ triangleright \text{a} \ triangleright \text{p} \triangleright \text{S} \triangleright ((b \text{a} \text{np}) (t \text{c}))
\]

In this presentation we have left out the types of the lambda terms. If we had filled them in it would become clear that all the terms, including the terms constructed in the course of the derivation, are well-typed. The grammar writer has to take care that application of the rules leads to well-typed lambda terms. This involves a number of precautions. Consider the first rule.

\[
s_1 \circ s_2 \triangleleft \text{S} \triangleright (t_2 t_1) \rightarrow s_1 \triangleright \text{NP} \triangleright t_1 \\
\text{NP} \triangleright t_2
\]

First the type of $t_1$, $\tau$, must match the type of $t_2$, $\tau \rightarrow \sigma$. Secondly, across rules, the type of terms associated with some category must be the same wherever that category occurs. In our example, we have the following match.

\[
\begin{align*}
type(\text{DET}) &= n \rightarrow np \\
type(\text{N}) &= n \\
type(\text{V}) &= np \rightarrow (np \rightarrow s) \\
type(\text{VP}) &= np \rightarrow s \\
type(\text{NP}) &= np \\
type(\text{S}) &= s
\end{align*}
\]

**Categorial Grammar** To extend the categorial grammars with semantic terms, we only need to change the lexicon.

\[
\text{Lex}:
\begin{align*}
a &\triangleleft \text{NP/N} \triangleright a^{n \rightarrow np} \\
\text{the} &\triangleleft \text{NP/N} \triangleright \text{the}^{n \rightarrow np} \\
\text{cook} &\triangleleft \text{N} \triangleright \text{pizza}^n \\
\text{baked} &\triangleleft (\text{NP/S})/\text{NP} \triangleright \text{baked}^{np \rightarrow (np \rightarrow s)}
\end{align*}
\]

In a categorial grammar there are no phrase structure rules. Instead of a rule-to-rule translation as in the phrase structure grammars, we calculate the semantics of a combination by an extended version of the general application combination schema (modus ponens). This method can be said to be type-driven (Klein and Sag (1985)). In a Prawitz-style presentation this can be formulated as follows.
Looking at the way categories and types are lined up in these combination schemata, we can define the mapping between the complex categories and the types recursively.

\[
\begin{align*}
\text{type}(A/B) &= \text{type}(B) \rightarrow \text{type}(A) \\
\text{type}(B\setminus A) &= \text{type}(B) \rightarrow \text{type}(A)
\end{align*}
\]

Such a mapping may be assumed to be universal. Specific categorial grammars are parameterised for the basic categories that are assumed and the types that correspond to these. For instance:

\[
\begin{align*}
\text{type}(s) &= s \\
\text{type}(\text{NP}) &= \text{np} \\
\text{type}(\text{N}) &= n
\end{align*}
\]

The reader can see that the lexicon presented above agrees with this typing schema.

\[
\begin{align*}
a < \text{NP}/\text{N} > a^{n \rightarrow \text{np}} & \quad \text{pizza} < \text{N} > p^n \\
 a \text{ pizza} < \text{NP} > (a \ p)^{\text{np}}
\end{align*}
\]

This simple illustration shows that in the categorial grammars there is a close fit between the syntactic categories and the semantic types. The syntactic rule of combination corresponds to application in the semantic dimension (the syntactic rule is also often called \textit{application}).

**Summary**

Aspects of the syntax of natural languages can be accounted for by modeling linguistic objects as expressions classified into categories, or as phrase structure trees decorated with categorial information. Semantic representations can be built up in parallel.

Grammars define the membership of linguistic structures to some language, classify them into categories and specify their compositional structure.

Different formal devices can be used as grammar formalisms. In this chapter we have introduced two basic frameworks: context-free phrase structure grammars and applicative categorial grammars. In the following chapters we will discuss extensions to both of these. We first discuss extensions to the latter, focusing on techniques to refine the description of the compositional structure of languages. Next, we turn to refinements of phrase structure grammars, designed to deal with more fine-grained classification distinctions.
2

Multi-modal Categorial Grammars

In this and the next chapter we explore more refined ways of describing languages by extended versions of the grammatical frameworks we discussed earlier. We first discuss extensions of categorial grammars that focus on refining the concept of grammatical composition. In the next chapter, we will introduce more complex structures for categories in phrase structure grammars to refine the classification mechanisms of linguistic expressions.

We will extend the simple AB-grammars from the previous chapter in several ways. First we continue the logical (deductive) perspective on categorial grammars by introducing type-logical grammars in which the rule of modus ponens or application is complemented by a rule of condition- alisation (hypothetical reasoning). Next, we will generalise this logic to connectives of other arities as well. Besides the binary connectives \(/,\ \backslash\) we introduce unary connectives \(\Diamond, \Box\). The inventory of connectives is further refined by allowing different structural options for the logical connectives which leads to the use of multiple connectives of the same arity. For instance, with respect to the binary connectives for which the base logic defines the operation of selection and composition, the structural options (defined by structural rules) distinguish between different versions of the composition operation like an associative versus a non-associative one. By varying the structural rules, it is thus possible to model different notions of composition. Finally, we show how different options can be mixed together into a single, multimodal system and how the interaction between different modes of composition can be defined.

Our presentation draws heavily on Došen (1992), Kurtonina (1995), and especially Moortgat (1997).

2.1 The Logic of Parts and Wholes

In a categorial grammar, categories provide information about the combination options of expressions. The rules of combination tell us in general terms how expressions can be composed. They reflect what we want the complex categories to mean. If some expression is in the set of expressions \(A/B\) it means that it can combine with any expression in \(B\) and the combination will be in \(A\).
In the previous chapter we interpreted combination in terms of string concatenation. Let us now consider a more general notion of composition as a relation between parts (components) and wholes (composites). In the case where we assume a composite made up out of two components we have a ternary relation, which we can represent as \( R(w, p_1, p_2) \), where \( w \) is the whole, and \( p_1, p_2 \) are the parts. We can also add category names for the wholes, by introducing a binary category forming operator \( \bullet \), such that \( w \) is of type \( A \bullet B \) if \( p_1 \) is of type \( A \) and \( p_2 \) is of type \( B \).

### Models

In the general setting that we are concerned with in this section, we turn to a relational interpretation of categories and provide a Kripke-style semantics in which the connectives are treated as modal operators.

We introduce a set of nodes \( W \) ("worlds") which we call linguistic resources. For the logical connective, \( \bullet \), we introduce a ternary relation on \( W \), \( R^3 \). The set \( W \) and the accessibility relation define a frame \( (W, R^3) \). To define the interpretation of the basic categories we add a valuation function \( v \) that assigns subsets of \( W \) to the basic types and that satisfies the following conditions for the complex types.

\[
\begin{align*}
v(A \bullet B) &= \{ w \mid \exists x, \exists y[R^3wxy & x \in v(A) \& y \in v(B)] \} \\
v(C/B) &= \{ x \mid \forall y \forall w[(R^3wxy \& y \in v(b)) \Rightarrow w \in v(C)] \} \\
v(A\setminus C) &= \{ y \mid \forall x \forall w[(R^3wxy \& x \in v(A)) \Rightarrow w \in v(C)] \}
\end{align*}
\]

For the AB grammars we used the following clauses to make explicit how expressions combine.

(ii) if \( \langle s_1, A/B \rangle \) and \( \langle s_2, B \rangle \) are in \( L_G \) then so is \( \langle s_1 \circ s_2, A \rangle \),

(iii) if \( \langle s_2, B \setminus A \rangle \) and \( \langle s_1, B \rangle \) are in \( L_G \) then so is \( \langle s_1 \circ s_2, A \rangle \)

With these rules we can reason from the parts to the whole. We deduce the category of the whole by inspecting the parts. However, these rules are not enough to define a complete logic for the interpretation above. We should also be able to reason the other way round.

(ii') if \( \langle s_1 \circ s_2, A \rangle \in L_G \) and \( \langle s_2, B \rangle \in L_G \) then \( \langle s_1, A/B \rangle \in L_G \).

(iii') if \( \langle s_1 \circ s_2, A \rangle \in L_G \) and \( \langle s_1, B \rangle \in L_G \) then \( \langle s_2, B \setminus A \rangle \in L_G \).

In this case we deduce the type of one part from the type of the whole and the type of the other part. Whereas clauses (ii) and (iii) are essentially modus ponens steps from a logical perspective, the new rules are conditionalisation steps.

### Residuation

The duality between composition expressed by \( \bullet \) and selection expressed by the connectives /, \setminus, or the duality between construction and deconstruction is an instance of the residuation schema. In the axiomatic presentation, this duality is immediately apparent.

\[
A \rightarrow C/B \quad \text{iff} \quad A \bullet B \rightarrow C \quad \text{iff} \quad B \rightarrow A \setminus C
\]
Natural Deduction  In terms of formal rules of inference we can formulate the logic of combination in a Gentzen-style natural deduction format. We also include the rules for •, which is the logical connective that corresponds to the structure building operation ◦. A judgment like Γ ⊢ C is called a (natural deduction) sequent. The left-hand side, Γ, will be referred to as the antecedent and the right-hand side, C, as the succedent.

In a sequent Γ ⊢ C the Γ-part consists of a term constructed from categories, by means of the term constructor ◦. This is taken to be associative in the so-called associative Lambek calculus. This term represents the structured set of assumptions. The interpretation of the structured terms that form the antecedent parallels the interpretation of •.

\[ v(\Delta_1 \circ \Delta_2) = \{ w | \exists x, \exists y[R^3 wxy \ & x \in v(\Delta_1) \ & y \in v(\Delta_2)\} \]

In the following definition we write Γ[\Delta] for a term Γ containing a distinguished occurrence of the subterm \( \Delta \). For each relevant inference rule, the distinguished occurrences in premise and conclusion are supposed to occupy the same position in Γ.

**Definition 17 (Lambek Calculus Gentzen Style ND)** The following rules define the Lambek calculus in the Gentzen Style Natural Deduction presentation.

\[
\begin{align*}
A \vdash A & \\
\frac{\Delta \vdash (A \bullet B) \Gamma[(A \circ B)] \vdash C}{\Gamma[\Delta] \vdash C} & \quad E\bullet \\
\frac{\Gamma \vdash A/B \ \Delta \vdash B}{(\Gamma \circ \Delta) \vdash A} & \quad E/(ii) \\
\frac{\Gamma \vdash B \ \Delta \vdash B\setminus A}{(\Gamma \circ \Delta) \vdash A} & \quad E/(iii) \\
\frac{(\Gamma \circ \Delta) \vdash (A \bullet B)}{(\Gamma \circ \Delta) \vdash A/B} & \quad I/\bullet \\
\frac{\Gamma \vdash A \ \Delta \vdash B}{(\Gamma \circ \Delta) \vdash (A \bullet B)} & \quad I/(ii') \\
\frac{(\Gamma \circ \Delta) \vdash (B \circ \Gamma) \vdash A}{(\Gamma \circ \Delta) \vdash B \setminus A} & \quad I/(iii')
\end{align*}
\]

The inference rules in the left column are called elimination rules because they eliminate a logical connective in going from the premises to the conclusion. The rules in the right column are called introduction rules because they introduce a logical connective in the conclusion.

The relation between the clauses above and the rules of inference comes out clearly when we align them as follows.

(i) if \( \langle s_1, A/B \rangle \) and \( \langle s_2, B \rangle \) are in \( L_G \) then so is \( \langle s_1 \circ s_2, A \rangle \),
(ii) if \( \Gamma \vdash A/B \) and \( \Delta \vdash B \) then \( \Gamma \circ \Delta \vdash A \).
(iii) if \( \langle s_1 \circ s_2, A \rangle \in L_G \) and \( \langle s_2, B \rangle \in L_G \) then \( \langle s_1, A/B \rangle \in L_G \).
(iii') if \( \Gamma \circ B \vdash A \) then \( \Gamma \vdash A/B \).
The expressions $s_1, s_2, s_3 \circ s_2$ constructed from words, are replaced by terms, consisting of categories. We can say that an expression $s$ of category $C$ is in the language if we can derive $\Gamma \vdash C$, where $\Gamma$ is like $s$ but all the words have been replaced by categories. Of course, this substitution has to accord with the lexical type assignments.

The axiom says that from $A$ one can derive $A$. In terms of the interpretation of categories and terms we can read $\vdash$ as $\subseteq$. So, for the axiom we have $v(A) \subseteq v(A)$, which is trivially true. We can show that with the calculus and interpretation as above a sequent $\Gamma \vdash C$ is a theorem of the calculus iff $v(\Gamma) \subseteq v(C)$ (for all frames and valuations).

**Lambek Calculus** This version of categorial grammar is known as the Lambek calculus (Lambek (1958), Lambek (1961)). What distinguishes this calculus from the AB grammars is that we have two rules for each connective $/$, $\bullet$, $\setminus$: introduction rules that derive a complex category and elimination rules that show us what can be done with a complex category. In other words, rules that tell us how parts can be assembled into wholes and rules that tell us how combinations can be broken up into parts.

**Example 3 (Gentzen Style Natural Deduction Derivation)** The following derivation is a simple instantiation of rule $(ii)$.

\[
\frac{\text{NP/N} \vdash \text{NP/N} \quad \text{N} \vdash \text{N}}{(\text{NP/N} \circ \text{N}) \vdash \text{NP}} \quad E/\]

It is easy to see that this derivation shows that $((a \circ \text{pizza}), \text{NP}) \in L_G$, provided that $\text{NP/N}$ is a lexically assigned category for $a$ and $\text{N}$ a category for $\text{pizza}$. To make this relation between expression and category explicit, we can systematically replace the categories in the antecedent by the lexical items. This is the shorthand format we will be using in most linguistic examples. It is also the format which we introduced at the end of the presentation of AB-grammars in the previous chapter.

\[
\frac{a \vdash \text{NP/N} \quad \text{pizza} \vdash \text{N}}{(a \circ \text{pizza}) \vdash \text{NP}} \quad E/\]

Let us now consider the use of the introduction rule. Intuitively, it says that if we know that some compositional structure $\Gamma \circ B$ is of type $A$ then we can conclude that the structure $\Gamma$ (i.e. the original structure without $B$) is of the type $A/B$ (i.e. the type of constructions that combine with $B$ elements to form $A$ expressions).

**Example 4 (Argument Lowering)** The introduction rule can be used to derive argument lowering: a lexical type assignment for a verb like need, $\text{IV/((S/NP)/S)}$, can be lowered to $\text{IV/NP}$. We use $u$ and $t$ to represent the hypothetical terms in the antecedent.
MULTI-MODAL CATEGORIAL GRAMMARS

Semantics  In the previous chapter, the meaning dimension of natural language expressions was modelled by means of $\lambda$-expressions. In the categorial calculus, the semantics of words was provided by the lexicon and the meaning of phrasal expressions was calculated on the basis of the rules of inference. We repeat the procedure for the elimination rules in Prawitz notation ($E < C > M$).

\[
\begin{align*}
\frac{t \vdash S/NP \quad u \vdash NP}{t \circ u \vdash S} & \quad E/ \\
\frac{\text{need} \vdash IV/((S/NP) \setminus S)}{u \vdash (S/NP) \setminus S} & \quad I/ \\
\end{align*}
\]

The semantic effect of this combination rule is that the function associated with the functor category is applied to the semantic term associated with the argument category. The application rule corresponds to the elimination rule in the Gentzen style natural deduction format. The rule of introduction in this calculus is coupled with in the meaning recipe. We only provide the rules for $/$.

\[
\begin{align*}
\frac{s_1 \triangleleft A/B \triangleright t_1 \quad s_2 \triangleleft B \triangleright t_2 \quad s_1 \circ s_2 \triangleleft A \triangleright (t_1 t_2)}{s_1 \circ s_2 \triangleleft A \triangleright (t_1 t_2)} & \quad E/ \\
\end{align*}
\]

This shows that the way a semantic representation is determined in this more extended categorial calculus is similar to the method used before. The interpretation of complex structures is derived in the spirit of the so-called Curry-Howard correspondence. For more discussion and extensions to other connectives we refer to Morrill (1994), Carpenter (1999) and Moortgat (1997).

Unary Connectives  The logic of residuation is the general starting point for further extensions. We first consider extensions to connectives of a different arity. We define a unary product connective that parallels the binary one by leaving out the second ($B$) argument of $\cdot$ and $\circ$ from the rules of Definition 17.

\[
\begin{align*}
\Delta \vdash A \cdot & \quad \Gamma[A_0] \vdash C \quad \Gamma \vdash A \\
\Gamma[\Delta] \vdash C & \quad (\Gamma_0 \vdash A) \cdot \\
\end{align*}
\]

We can do the same for the slash connectives.

\[
\begin{align*}
\Gamma \vdash A/ & \quad (\Gamma_0) \vdash A \\
(\Gamma_0) \vdash A & \quad \Gamma \vdash A/ \\
\end{align*}
\]
We write the unary equivalent of the product operator $\bullet$ as $\Diamond$ and the unary slash operator as $\Box$, use prefix notation and replace the structural operator $\cdot \circ \cdot$ by brackets: $\langle \cdot \rangle$, to get a more readable notation:

$$
\begin{align*}
\Delta \vdash \Diamond A & \quad \Gamma \langle A \rangle \vdash C \\
\Gamma \langle \Delta \rangle \vdash C & \\
\Gamma \vdash \Box A & \\
\langle \Gamma \rangle \vdash A & \\
\Gamma \vdash \Diamond \Box A & \\
\langle \Gamma \rangle \vdash A
\end{align*}
$$

In axiomatic form, we get the following.

$$
\Diamond A \rightarrow C \iff A \rightarrow \Box C
$$

We can introduce connectives of other arities as well. In this book, however, we will only use the connectives $\div, \bullet, \backslash, \Diamond, \Box$. We provide examples of their linguistic use in Section 2.3. They will also figure prominently in Part II.

**Models** For the model-theoretic interpretation, we add binary accessibility relations to the frame: $\langle W, R^3, R^2 \rangle$ and add clauses for the unary connectives.

$$
\begin{align*}
v(\Diamond B) &= \{ w \mid \exists x [ R^2 w x \land x \in v(\langle B \rangle)] \} \\
v(\Box B) &= \{ x \mid \forall w [ R^2 w x \Rightarrow x \in v(\langle B \rangle)] \}
\end{align*}
$$

**Language** After this summary of the proof-theoretic and model-theoretic properties of the generalised Lambek system, it remains to be shown how the system is used for the purpose of linguistic description, i.e. how it is used as a grammar formalism. In fact, there is not much new to be said here. A grammar is still merely an assignment of types to words: $G = \text{Lex} \subseteq V \times C$, but the set of categories is now extended and the terms in the antecedent become more complex as well.

$$
\begin{align*}
C ::= B \mid C \bullet C \mid C/C \mid C\backslash C \mid \Diamond C \mid \Box C \\
T ::= C \mid (T \circ T) \mid \langle T \rangle
\end{align*}
$$

We require interpretation clauses for terms which are straightforward extensions of the interpretation function $v$ for categories. The denotation of a term is similar to the denotation of a category. There are three cases to consider. A term is either a category (so the interpretation is the same) or constructed out of subterms $(T \circ T)$, or bracketed as $\langle T \rangle$. We assume that $\circ$ is non-associative. The interpretation of these complex terms is the same as the interpretation of complex categories: $(C \bullet C)$, or $\Diamond C$, as the following definition makes clear.
**Definition 18 (Interpretation of Terms)** The interpretation of complex terms is as follows.

\[
v(T_1 \circ T_2) = \{ w | \exists x, \exists y[R^3 wxy & x \in v(T_1) & y \in v(T_2)] \}
\]

\[
v(\langle T \rangle) = \{ w | \exists x[R^2 wx & x \in v(T)] \}
\]

A language, \( L \subseteq E \times C \) is a set of pairs of expressions (non-empty strings \( E = V^+ \)) and categories. We define a language \( L_{NL} \) by a lexicon \( Lex \) as in Moortgat (1997, p. 104). For a general model \( M = \langle W, R^3, R^2, v \rangle \) to qualify as appropriate for a lexicon \( Lex \), we assume \( V \subseteq W \), and we require the valuation \( v \) to be compatible with lexical type assignments, in the sense that, for all \( x \in V, \langle x, A \rangle \in Lex \) implies \( x \in v(A) \).

**Definition 19 (Language - NL)** Given a model \( M \) that qualifies for a lexicon \( Lex \), we will say that the grammar assigns type \( B \) to a non-empty string of lexical resources \( x_1 \cdots x_n \in V^+ \) or \( \langle x_1 \cdots x_n, B \rangle \in L_G \) provided there are lexical type specifications \( \langle x_i, A_i \rangle \in Lex \) such that we can deduce \( B \) from some \( o(A_1, ..., A_n) \) in the general type logic. By \( o(A_1, ..., A_n) \) we mean the possible products of the formulas \( A_1, ..., A_n \) in that order.

From a linguistic perspective, we are not just interested in the language as a set of pairs of expressions and categories, but also in the information carried by the derivation(s) associated with these pairs. "Linguistic composition in the form dimension is captured in the term structure over antecedent assumptions." (Moortgat (1997, p. 119)). "Semantic interpretation can be read off directly from the proof which establishes the well-formedness (derivability) of an expression." (Moortgat (1997, p. 115)).

**Example** We already provided an example of how the calculus is used to define the grammaticality of the simple noun phrase \((a \circ pizza)\). Following the definition, to show that \( \langle (a \circ pizza), NP \rangle \in L_G \), we have to derive \( \Gamma \vdash NP \). Provided that \( NP/N \) and \( N \) are among the lexically assigned types to \( a \) and \( pizza \), respectively, we can take \( \Gamma \) to be \( (NP/N \circ N) \).

\[
\frac{NP/N \vdash NP/N \quad N \vdash N}{(NP/N \circ N) \vdash NP} E/
\]

In the alternative presentation, we use expressions as antecedent terms. The type assignments that are assumed for these derivations can be read off from the axiom leaves.

\[
\frac{a \vdash NP/N \quad pizza \vdash N}{(a \circ pizza) \vdash NP} E/
\]
2.2 Structure

In the previous section we have introduced the pure residuated version of a categorial grammar with operators of various arities. The modus ponens rule from AB-grammars was complemented by a rule of conditionalisation. We will now look at variations of the calculus in which this logic of residuation is paired with a package of structural rules thus defining a landscape of structural options with respect to the composition relation (associative versus non-associative, commutative versus non-commutative etc.). Structural rules for the unary connectives will be presented in the next section.

In this context it is important to realise that the assumptions that make up the antecedent are configured into a structured term. We will point out what this structure of the antecedent involves precisely and how it can be manipulated to define useful variants of the calculus. In the following section we discuss the combination of several options in a mixed system and we introduce additional machinery to handle the interaction between the different systems.

In the previous chapter, when discussing the parallels between application in an AB-calculus and the rule of modus ponens in propositional logic, we mentioned an important difference between the two logics. In categorial systems the logic is sensitive to the order of the premises which relates to the order of the words and phrases. We therefore have two distinct connectives / and \ corresponding to .

This already points out that a grammar logic must be sensitive to aspects of linguistic structure. It must be resource and structure-sensitive. Not only the order of assumptions is important, but also other structural aspects: the number (multiplicity) of occurrences of assumptions (categories in the antecedent) matters, and the way they are grouped together (combined by brackets).

**Multiplicity** Two sequents that differ only in the number of occurrences of the formulas in the antecedent are different sequents and may thus differ in derivability. The sequents \( A/B \rightarrow B \) and \( (A/B)/B \rightarrow A \), for instance, are not derivable because the first has one formula too many and the second is one formula short.

The following attempts to derive these sequents can be used as an indication of what goes wrong. The square brackets in the first attempt are here used to collapse two derivations into one. Both the one with and without this extra premise will fail.

\[
\frac{A/B \rightarrow A \quad B \rightarrow B}{A/B \rightarrow B} \quad E/ \\
\frac{A/B \rightarrow A}{A/B \rightarrow B} \quad [B \rightarrow B] \quad \therefore \\
\frac{A/B \rightarrow A}{A/B \rightarrow B} \quad \therefore
\]


In propositional logic, the number of times an assumption occurs does not matter because besides the elimination and introduction rules for the connectives, there are structural rules that can manipulate the assumptions in a term. These may include rules that allow one to duplicate copies of a formula (contraction) or to throw away copies (weakening), for instance.

Because not all the structural rules are generally available in the categorial logic, we are dealing with an instance of a so-called substructural or resource-conscious logic (see Došen and Schröder-Heister (1993), Troelstra (1992)). The reason that we do not want such rules unrestrictedly is because the categories (formulas) represent linguistic material which we do not want to be copied and deleted at random but only in a controlled way, if at all.

**Order** Another aspect of linguistic objects is that words and constituents cannot permute freely. The calculus as we have presented it incorporates this restriction. For instance, we can derive \( \frac{A/B \circ B \vdash A}{\vdash A} \) but not \( \frac{B \circ A/B \vdash A}{\vdash A} \).

If we add the structural rule of permutation to the calculus we get a version of a categorial grammar that is known as the Lambek/Van Benthem calculus, also known under the abbreviation LP: Lambek with Permutation (van Benthem (1986)).

\[
\frac{\Gamma[\Delta_2 \circ \Delta_1] \vdash C}{\Gamma[\Delta_1 \circ \Delta_2] \vdash C}^{\text{PERM}}
\]

In axiomatic form, this structural rule looks as follows.

\[ A \bullet B \rightarrow B \bullet A \]

Whereas the axioms are defined on categories, the natural deduction format uses terms and introduces contexts \( \Gamma \). Using this rule we can derive \( B \circ A/B \vdash A \).

\[
\frac{A/B \vdash A/B \quad B \vdash B}{\frac{\quad A/B \circ B \vdash A}{\frac{B \circ A/B \vdash A}^{\text{PERM}}}}^{\text{PERM}}
\]

**Associativity** We pointed out that the order and number of occurrences of categories in the antecedent is important. For instance, the structural rule of permutation is not present in the \( \text{NL} \) system. In terms of the property of the structural operation \( \circ \) this can be rephrased by saying that in the Lambek calculus it is non-commutative. The brackets that we have put
around terms are, in the absence of structural rules and further conventions, assumed to be non-associative as well. The brackets provide extra information about the construction of the term, defining constituent structure. If we want the \( \circ \) operation to be associative then we have to take care that we can change a structure \((C_1 \circ (C_2 \circ C_3))\) into a structure \(((C_1 \circ C_2) \circ C_3)\) and vice versa. This can be effected by introducing the following structural rule.

\[
\frac{\Gamma[((\Delta_1 \circ \Delta_2) \circ \Delta_3)] \vdash C}{\Gamma[(\Delta_1 \circ (\Delta_2 \circ \Delta_3))] \vdash C} \quad \text{ASSOC}
\]

The double line is used as a notational convention to show that the rule works both ways. The notation captures the following two structural rules.

\[
\frac{\Gamma[(\Delta_1 \circ (\Delta_2 \circ \Delta_3))] \vdash C}{\Gamma[((\Delta_1 \circ \Delta_2) \circ \Delta_3)] \vdash C} \quad \text{ASSOC}_1 \quad \frac{\Gamma[((\Delta_1 \circ \Delta_2) \circ \Delta_3)] \vdash C}{\Gamma[(\Delta_1 \circ (\Delta_2 \circ \Delta_3))] \vdash C} \quad \text{ASSOC}_2
\]

In axiomatic form the rule \text{ASSOC} can be presented as follows.

\[
A \bullet (B \bullet C) \leftrightarrow (A \bullet B) \bullet C
\]

The difference between an associative and a non-associative system can be illustrated in several ways. The sequent \((A/B \circ B/C) \Rightarrow A/C\), for instance, is a theorem in the associative system but not in the non-associative one.

\[
\frac{\Gamma[(\Delta_1 \circ (\Delta_2 \circ \Delta_3))] \vdash C}{\Gamma[((\Delta_1 \circ \Delta_2) \circ \Delta_3)] \vdash C} \quad \frac{\Gamma[((\Delta_1 \circ \Delta_2) \circ \Delta_3)] \vdash C}{\Gamma[(\Delta_1 \circ (\Delta_2 \circ \Delta_3))] \vdash C} \quad \text{ASSOC}
\]

The difference between the associative and the non-associative system can also be illustrated by the possible types for a simple transitive verb. If we choose the assignment \((NP\text{s})/NP\) then we can derive both (1) and (2) in the associative calculus, but only the first in the non-associative calculus.

\[
(1) \quad NP \circ ((NP\text{s})/NP \circ NP) \vdash S
\]

\[
(2) \quad (NP \circ (NP\text{s})/NP) \circ NP \vdash S
\]

The following derivations show at what point we need the rule of associativity.

\[
\frac{\text{baked} \vdash (NP\text{s})/NP \quad \text{pizzas} \vdash NP}{\text{Toni} \vdash NP \quad \frac{\text{baked} \circ \text{pizzas} \vdash NP\text{s}}{E/}}\quad E/\quad \text{E/}
\]

\[
(1) \quad \frac{\text{Toni} \circ (\text{baked} \circ \text{pizzas}) \vdash S}{E/}
\]

\[
\frac{\text{baked} \vdash (NP\text{s})/NP \quad \text{pizzas} \vdash NP}{\text{Toni} \vdash NP \quad \frac{\text{baked} \circ \text{pizzas} \vdash NP\text{s}}{E/}}\quad E/\quad \text{ASSOC}
\]

\[
(2) \quad \frac{\text{Toni} \circ (\text{baked} \circ \text{pizzas}) \vdash S}{\text{ASSOC}}
\]

\[
(2) \quad \frac{((\text{Toni} \circ \text{baked}) \circ \text{pizzas}) \vdash S}{E/}
\]

\[
(2) \quad \frac{((\text{Toni} \circ \text{baked}) \circ \text{pizzas}) \vdash S}{\text{ASSOC}}
\]
In fact, the associative calculus is 'structurally complete' in the sense that when some sequent is derivable then all sequents that differ from it only in the way the antecedent is bracketed are derivable as well (see Buszkowsi (1988) for the formal proof and Moortgat (1988) for the linguistic consequences).

**Associativity and Introduction** We provide another linguistic example of the associativity rule, this time in combination with an introduction rule.

**Example 5 (Use of Introduction rule)** We illustrate the use of the introduction rule with a derivation for a relative clause. We have already seen derivations for simple sentences like *he baked the pizza*. Now consider constructions like the pizza that he baked. If we have lexical assignments such that we can derive \( he \circ baked \circ the \circ pizza \vdash S \), then using the introduction rule we can prove that \( he \circ baked \) is in \( S/NP \) (an expression that needs a noun phrase to form a sentence). Of course, when proving that \( he \circ baked \) is in \( S/NP \), there is no object \( NP \). We proceed by hypothesising one which we have to withdraw later. We use the label \( t \) for the hypothesised noun phrase. We mark the sequent with brackets to show that this hypothesis has been withdrawn.

\[
\frac{he \vdash NP \quad \text{baked} \vdash (NP\backslash S)/NP \quad \frac{t \vdash NP}{E/}}{he \circ (baked \circ t) \vdash NP\backslash S \quad \frac{E\backslash}{ASSOC}} \quad \text{he} \circ (baked \circ t) \vdash S \quad \text{he} \circ baked \vdash S/NP \quad \frac{I/}{(he \circ baked) \vdash S/NP}
\]

Now this phrase is ready to combine with a relative pronoun to form a relative (noun modifier) clause, by modus ponens.

\[
\frac{that \vdash (N\backslash N)/(S/NP) \quad (he \circ baked) \vdash S/NP}{(that \circ (he \circ baked)) \vdash N\backslash N \quad E/}
\]

**Multiple Systems** We started this section on the Lambek calculus by defining a calculus based on the pure logic of residuation. We have now shown how this calculus captures the number (multiplicity), order and structure of the linguistic resources. Adding structural rules to the calculus provides us with interesting alternatives. In the following paragraphs we will see how the different alternatives can be combined into a mixed system and how postulates can be defined that specify the interaction between them.
2.3 Mixed-Multimodal

Instead of choosing between an associative or non-associative calculus it is also possible to mix them together into one system. For this we need a device to make a distinction between the connectives, distinguishing the associative \( /_a, \cdot, \ \backslash_a \) from the non-associative \( /_n, \cdot, \ \backslash_n \) ones. The subscripts, or resource management modes, distinguish between different connectives. The bracketings of the terms \((\cdot \circ \cdot)\) must be made distinct as well by indexing the structural connective \( \circ \). For each triple of connectives we assume the base logic expressing the residuation relations. For the associative case we add the appropriate structural rule. Because the calculi differ only in the structural rules, we can present the logical rules more compactly as follows, where \( i \) is a variable ranging over resource management modes.

\[
A \vdash A
\]

\[
\frac{\Delta \vdash (A \cdot B) \quad \Gamma[(A \circ_i B) \vdash C]}{\Gamma[\Delta] \vdash C} \quad \frac{E_{i}}{\Gamma \vdash A \quad \Delta \vdash_i B \quad \Gamma \vdash_i A \quad \Delta \vdash_i B \quad I_{i}}
\]

\[
\frac{\Gamma \vdash A /_i B \quad \Delta \vdash B}{(\Gamma \circ_i \Delta) \vdash A} \quad \frac{(\Gamma \circ_i B) \vdash A}{\Gamma \vdash A /_i B} \quad \frac{I /_i}{(\Gamma \circ_i \Delta) \vdash A}
\]

\[
\frac{\Gamma \vdash B \quad \Delta \vdash A /_i B}{(\Gamma \circ_i \Delta) \vdash A} \quad \frac{(B /_i \Gamma) \vdash A}{\Gamma \vdash B /_i A} \quad \frac{I /_i}{(\Gamma \circ_i \Delta) \vdash A}
\]

\[
\frac{\Gamma[(\Delta_1 \circ_a \Delta_2) \circ_a \Delta_3] \vdash C}{\Gamma[(\Delta_1 \circ_a (\Delta_2 \circ_a \Delta_3))] \vdash C} \quad \text{ASSOC}
\]

\[
\frac{\Delta \vdash \circ_i A \quad \Gamma[(\Gamma)_i \vdash C]}{\Gamma[\Delta] \vdash C} \quad \frac{\Gamma \vdash A \quad \Gamma \vdash \circ_i A}{(\Gamma)_i \vdash \circ_i A} \quad \frac{I \circ_i}{(\Gamma)_i \vdash A}
\]

\[
\frac{\Gamma \vdash \square_i A}{(\Gamma)_i \vdash A} \quad \frac{\Gamma \vdash \square_i A}{(\Gamma)_i \vdash A} \quad \frac{I \square_i}{(\Gamma)_i \vdash A}
\]

Besides the modes \( a \) and \( n \) we can introduce others, for instance a commutative mode \( c \) or a mode that is both associative and commutative \( ac \), etc. The unary connectives can similarly be indexed by resource management modes.

**Interaction** Besides having structural rules in which one kind of structural connective is manipulated, we can also define postulates in which the interaction between structural modes is expressed. We first present a series of postulates in which unary structural configurations interact with binary modes.
The distribution postulates $\kappa$ (strong distribution) and $\kappa_1, \kappa_2$ (weak distribution) allow us to replace the structural brackets $i$ around a complex construction by structural brackets $i$ around the parts or around one part. Inclusion postulates allow us to change some structural configuration $j$ into another $i$.

$$\frac{\Gamma[(\Delta_1)_i \circ x (\Delta_2)_i] \vdash C}{\Gamma[(\Delta_1 \circ x \Delta_2)_i] \vdash C} \kappa$$

$$\frac{\Gamma[(\Delta_1)_i \circ x \Delta_2] \vdash C}{\Gamma[(\Delta_1 \circ x \Delta_2)_i] \vdash C} \kappa_1$$

$$\frac{\Gamma[(\Delta_1)_i \circ x (\Delta_2)_i] \vdash C}{\Gamma[(\Delta_1 \circ x \Delta_2)_i] \vdash C} \kappa_2$$

Although we will use this format for postulates in derivations, we will often define them in the axiomatic form which is easier to read.

$$\frac{\Gamma[(\Delta)_i] \vdash C}{\Gamma[(\Delta)_j] \vdash C} \iota$$

Note that the left-hand side of the axiomatic rule corresponds to the left-hand side of the bottom sequent in a natural deduction presentation (and the right-hand side to the left-hand side of the top sequent).

The unary structure-building operators provide extra structure in the antecedent term represented by brackets. The logical rules for $\Diamond$ and $\Box$ can be used to introduce and remove different types of brackets. The behaviour of the operators is further refined by collections of structural postulates. This extra structure can be used for different purposes. In Part III we will let the unary modalities and their corresponding structures refine the classification structure with morphosyntactic information and use postulates like the one above to distribute this information in trees.

Unary modalities are often used to define different domains of locality (see Moortgat (1997) for examples and further references). Originally, the unary modalities were introduced in substructural logics like linear logic to recover the options normally provided by structural rules like permutation in a controlled way. The following postulate illustrates this use.

$$\text{PERM} \quad \Diamond_p A \bullet_c B \leftrightarrow B \bullet_c \Diamond_p A$$

In this case, we introduce a restricted form of permutation. It is restricted to the mode $c$ and allowed only if one of the subcategories is marked with $\Diamond_p$.

We will call this generalisation of the Lambek calculus with connectives of different arities and connectives with different structural properties (but
all defined by a base logic of residuation) the generalised Lambek calculus or the mixed multimodal Lambek calculus.

We will often write \( \langle i \rangle \) and \([i]\), for \( \Diamond_i \) and \( \Box_i \), respectively. Note that the brackets \( \langle \cdot \rangle \) will be used both around resource management modes and in terms. The context should make it clear which use is intended.

**Models and Language** The interpretation of the multimodal types is a simple generalisation of the relational semantics we provided for the simpler system. As before, we define a Kripke-style semantics in which the connectives are treated as modal operators. This means that we introduce a set of nodes \( W \) (worlds) which we call *linguistic resources*. For each logical connective, we introduce a relation on \( W \) of the arity of the connective plus one: \( (R^2_i, R^3_i) \). The set \( W \) and the families of relations (accessibility relations in modal terms) define a frame \( \langle W; \{R^2_i\}_{i \in I}, \{R^3_i\}_{i \in I} \rangle \) (where \( I \) obviously refers to the set of resource management modes). To define the interpretation of the basic categories we add a valuation function \( v \) as before but now relativised to the resource management modes.

Extra conditions on the accessibility relations that are proof theoretically defined in terms of the structural rules, including the interaction and inclusion postulates, are semantically defined as constraints on the frame, defining further properties of the accessibility relation, i.e. the linguistic structure. Corresponding to the various structure manipulating rules above we have the following frame conditions.

\[
\begin{align*}
\text{ASSOC} & \quad \exists t(R_a u x t \& R_a t y z) \iff \exists t'(R_a u t' z \& R_a t' x y) \\
\text{K} & \quad (R_y w z \& R_x z u t) \Rightarrow \exists u'[R_y u' u \& R_y t' t \& R_x z u' t'] \\
\text{K}_1 & \quad (R_y w z \& R_x z u t) \Rightarrow \exists u'[R_y u' u \& R_x z u' t] \\
\text{K}_2 & \quad (R_y w z \& R_x z u t) \Rightarrow \exists t'[R_y t' u \& R_x z u' t] \\
\text{P} & \quad (R_x w z \& R_y w' w) \iff (R_x z w' \& R_y w' w) \\
\text{I} & \quad (R_x w z) \Rightarrow (R_y w z)
\end{align*}
\]

In order to let such a mixed, multimodal grammar logic define a language we have to make some obvious changes to the definitions we provided above, indexing logical and structural connectives by a variable ranging over resource management modes. These changes are trivial. They concern definitions of categories, terms, etcetera.

A language, \( L \subseteq E \times C \), is a set of pairs of expressions and categories. A language \( L_C \) is defined relative to a lexicon \( \text{Lex} \) and a concrete instantiation of a generalised Lambek system. By a concrete instantiation we mean that the set of basic types, the set of resource management modes and the various structural rules are all fixed. Also in these multimodal systems we are not just interested in the set \( L_C \), but rather in the derivations. From a linguistic perspective, the form dimension is captured in the term structure of the antecedent assumptions and the meaning is read off from the proof.
To illustrate the use of structural rules we present a derivation of an example taken from the analysis of Dutch verb raising by Moortgat and Oehrle (1996). We will not discuss the details of this type of construction here. It is important to note that it involves cross-serial dependencies that are not context-free. We want to be able to derive a verb phrase category for Marie wil plagen (Mary wants to tease), where Marie is the object of plagen, as in Jan weet dat Piet Marie wil plagen (Jan knows that Piet Marie wants to tease = Jan knows that Piet wants to tease Marie).

Example 6 (Language Fragment – GL) We assume the following concrete generalised Lambek system, presenting the postulates in axiomatic form. Mode 1 refers to phrasal composition and 0 is used for verb clusters.

Signature
\[ B = \{NP, VP, INF\} \]
\[ I = \{0, 1\} \]

Postulates
\[ I \quad \Diamond_1 A \rightarrow \Diamond_0 A \]
\[ K_2 \quad \Diamond_1 (A \bullet_1 B) \rightarrow A \bullet_1 \Diamond_1 B \]
\[ K \quad \Diamond_0 (A \bullet_0 B) \rightarrow \Diamond_0 A \bullet_0 \Diamond_0 B \]
\[ MC \quad A \bullet_1 (B \bullet_0 C) \rightarrow B \bullet_0 (A \bullet_1 C) \]

Lexicon
\[ wil \quad \Box_0 (VP/0\ INF) \]
\[ Marie \quad NP \]
\[ plagen \quad \Box_0 (NP/1\ INF) \]

We start (at the top) by assembling the words wil, Marie, and plagen in the structural configuration and in the order that is required by the selection operators /, \. Next, we apply structural reasoning to derive the proper verb-raising order. The main rule is that of mixed commutativity (MC). This takes care that the verbs get clustered.
2.4 Alternative Proof Presentations

In the presentation of the grammar logic above, we have made use of different styles of presenting a derivation. In the sequel we will use the style that is most convenient for the problems at hand. Besides the Prawitz-style trees and the (Gentzen-style) natural deduction format we will also make use of the Gentzen sequent calculus.

**Gentzen Sequent** As in the Gentzen-style natural deduction presentation, we use sequents of the form \( \Gamma \Rightarrow C \), where \( \Gamma \) is a term (antecedent) and \( C \) a category (succedent). We use \( \Rightarrow \) instead of \( \vdash \) to mark that we are using the Gentzen sequent presentation.

To show that some expression \( E \) is of category \( C \), we first simply replace the words in \( E \) by a category (one assigned to the word in the lexicon), and then we organise the result in a structural configuration to obtain a term \( T \). If we can derive \( T \Rightarrow C \) by the rules below, then we have shown that \( E \in v(C) \).

\[
\begin{align*}
A \Rightarrow A \quad & \text{ax} \\
\Gamma \vdash A \circ_i B \Rightarrow C & \quad \text{L}_i \\
\Gamma \vdash A \bullet_i B \Rightarrow C & \quad \text{L}_i \\
\Delta \Rightarrow B \quad & \quad \Gamma \vdash A \circ_i B \Rightarrow C \quad \text{L}_i \\
\Gamma \vdash \Delta \circ_i B \Rightarrow C & \quad \text{L}_i \\
\Gamma \vdash (\langle A \rangle_i) \Rightarrow B & \quad \text{L}_i \\
\Gamma \vdash \langle \Delta \rangle_i \Rightarrow \Diamond_i A & \quad \text{L}_i \\
\Gamma \vdash A & \quad \text{R}_i \\
\langle \Gamma \rangle_i \Rightarrow \Diamond_i A & \quad \text{R}_i \\
\Gamma \vdash B & \quad \text{R}_i \\
\end{align*}
\]

We will sometimes use the Gentzen sequent presentation because it shows the duality between the connectives very well. It is also useful for the definition of backward chaining proof search. The rules can therefore best be read from conclusion to premise. Instead of elimination and introduction rules, the Gentzen sequent presentation uses left and right rules. The left rules manipulate connectives in the antecedent (to the left of \( \Rightarrow \)) and the right rules manipulate connectives in the succedent (to the right of \( \Rightarrow \)). The right rules correspond to the introduction rules of the natural deduction presentation.
The left rules for $\bullet$ and $\Diamond$ are less complicated than the elimination rules in the natural deduction version: the logical connective is replaced by its structural counterpart. The left rules for $/\, \text{and} \, \backslash$ define application. Consider the $L_i$ rule which says that an antecedent $\Gamma$ that contains the substructure $A_iB \circ_\Delta \Delta$ within it is of type $C$ provided that $\Delta$ is of type $B$ and the sequent $\Gamma$ with $A_iB \circ_\Delta \Delta$ replaced by $A$ is of type $C$. This expresses the behaviour of $A_iB$: form a combination of type $A$ with a structure $\Delta$ of type $B$. Also important in the derivations below are the rules for $\Box$. The $R_i$ rule says that in order to show that some structure $\Gamma$ is in $\Box_iA$ we have to show that the 'marked' version $<\Gamma>_i$ is in $A$. This corresponds to the axiomatic form $\Diamond \Box A \to A$. The $L_i$ rule nicely shows how structural brackets check (and remove) the $\Box$-operators in the antecedent.

To present certain linguistic examples, we will also use a syntactically sugared version of this calculus in which we replace categories in the antecedent by the lexical forms. As an illustration, we present the derivation of *that he baked* in this notation. We will often omit the line closing off the axiom leaves for the sake of readability. In this example we have used associativity implicitly.

\[
\begin{align*}
\text{NP} & \Rightarrow \text{NP} & S & \Rightarrow S & L/ \\
\text{NP} \circ \text{NP} \backslash S & \Rightarrow S & (\text{lex}) & \text{NP} \Rightarrow \text{NP} & L/ \\
\text{he} \circ \text{NP} \backslash S & \Rightarrow S & (\text{lex}) & \frac{\text{he} \circ \text{baked} \circ \text{NP} \Rightarrow S}{R/} \\
\frac{\text{he} \circ \text{baked} \Rightarrow S/\text{NP}}{L/} \\
\text{he} \circ \text{baked} & \Rightarrow \text{NP} \backslash \text{NP} \\
(\text{NP}_\text{\textbackslash NP}) \circ \text{he} \circ \text{baked} & \Rightarrow \text{NP}_\text{\textbackslash NP} & (\text{lex})
\end{align*}
\]

**Prawitz** We have introduced the elimination or application rule in the Prawitz formulation above, but did not present the introduction rules yet. We provide both of them here. The introduction rules show clearly the procedure involved in hypothetical reasoning. In the first step a category $A$ is derived using a hypothetical assumption $B$ which is withdrawn in the introduction step when $A/B$ is derived. The brackets around the hypothetical assumption mark that it has been withdrawn.

\[
\begin{align*}
\begin{array}{ccc}
\vdash & \vdash & \vdash \\
A/B & B & A/\text{e} \\
\backslash & & \backslash \\
\text{e} & & \text{e}
\end{array}
& \quad
\begin{array}{ccc}
\vdash & \vdash & \vdash \\
B & B \backslash A & A \\
/ & / & \backslash \\
\text{e} & \text{e} & \text{e}
\end{array}
\end{align*}
\]

\[
\begin{align*}
\frac{}{A/B} & \text{e} \\
\frac{}{B} & \text{e} \\
\frac{}{A/B} & \text{e}
\end{align*}
\]

\[
\begin{align*}
\frac{}{A/B} & \text{e} \\
\frac{}{B/A} & \text{e} \\
\frac{}{A/B} & \text{e}
\end{align*}
\]

\[
\begin{align*}
\frac{}{A/B} & \text{e} \\
\frac{}{B/A} & \text{e} \\
\frac{}{A/B} & \text{e}
\end{align*}
\]

\[
\begin{align*}
\frac{}{A/B} & \text{e} \\
\frac{}{B/A} & \text{e} \\
\frac{}{A/B} & \text{e}
\end{align*}
\]
In this presentation the derivation for *that he baked* proceeds as follows.

```
  baked  \[(NP\backslash S)/NP\]  \[NP]^1
  he  \[NP\backslash S\]  \[NP\backslash S\]^1
  that  \[(N\backslash N)/(S/NP)\]  \[S\]  \[S/NP\]  \[\]  \[i]^1
```

**Summary**

In this chapter we have presented a logical, deductive perspective on grammars. We have defined a dedicated grammar logic as a specific variant of a substructural, resource-sensitive logic. The sensitivity involves the number of times a formula can be used in a derivation and the structural configuration of the assumptions, which reflects important properties of linguistic expressions.

The notion of linguistic *composition* and of its dual *selection* is logically characterised by the properties of the logical constant $\bullet$ and its left and right residuals $/\, \backslash \, \$. These connectives and their (complete) logic define the basic rules of composition and selection in the grammar logic.

In the first chapter we discussed applicative categorial grammars that admit only the modus ponens type of reasoning. Moortgat (1997) writes: “The insight that Modus Ponens and Hypothetical Reasoning are two inseparable aspects of the interpretation of the ‘logical constants’ $/\, \backslash \,$ is the key contribution of Lambek’s work in the late Fifties”. From a linguistic point of view, we have provided an illustration of how hypothetical reasoning relates the treatment of local and non-local dependencies (the latter exemplified by the relative clauses) and how it complements the rule of functional application with a rule of $\lambda$-abstraction in the semantic dimension.

By varying the structural characteristics of the logical constants it is possible to define a landscape of compositional configurations. In this way the base logic, expressing the residuation relation between composition and selection, is refined. Aspects of structure are taken into account by distinguishing, for instance, between commutative or non-commutative versions and associative versus non-associative ones. This provides the basis for a modular construction of the grammar. The logical core is formed by the logic of residuation defining selection and composition, whereas structural rule packages provide more fine-grained modulation of the structure.

In the multimodal systems, different structural options live next to each another. In a mixed multimodal system as we have presented it, unary connectives $\Diamond$ and $\Box$ (related by the same base logic of residuation) can be used to control the interaction between different modes of combination and thereby trigger further structural manipulation. We have illustrated
this with an analysis of the Dutch verb-raising constructions that are not context-free.

In the third part of this book we will consider a different application for this type of communication between the unary and binary connectives. There, the logical rules for unary connectives are used to characterise a feature checking procedure for morphosyntactic information that parallels the logical rules defining the selection and composition procedure by the binary ones. In this case, the unary connectives will make use of the binary structures to move around in, percolating and distributing morphosyntactic information through phrase structure. This will enable us to complement the fine-grained potential of the binary connectives to characterise selection and composition by more refined mechanisms to classify expressions using the unary modalities. Before we come to that, however, we will look at more elaborate techniques to classify expressions in particular types of phrase structure grammars.
3

Generalised Phrase Structure Grammars

In the simple phrase structure grammars introduced in Chapter 1 we used categories as atomic symbols. However, in many contemporary linguistic theories it is customary to model categories by structured objects called feature bundles or feature structures. In theories like Head-Driven Phrase Structure Grammar (HPSG) such structures are also used to model lexical entries, phrase structure trees, semantic content and even the rules and principles of grammar (Pollard and Sag (1987)).

In this chapter we want to indicate what motivates the use of feature structures in grammatical description. We illustrate some of the ways in which feature structures are used and point out the specific benefits they offer to the descriptive grammarian. We first show how they provide a more fine-grained description language to classify expressions along multiple dimensions. Also, the partial informativeness ordering defined on feature structures is used as a taxonomic classification schema on the universe of expressions (sub- and superclasses) and makes it possible to simplify the grammar by generalising over certain properties (abstraction). The decomposition of information also provides a handle on presenting the grammar as a collection of principles that fix the behaviour of some individual property or of a collection of properties (factorisation).

3.1 Feature Structure Theories

The use of feature structures to model categories has a long tradition in linguistics. A formal theory of these structures was presented in Gazdar et al. (1985) and Gazdar et al. (1988), synthesising earlier work by several of the authors involved in these publications and others. The formal rigour and the precise description of non-trivial fragments was attractive not only to linguists but also to logicians, mathematicians, computer scientists and computational linguists. During the last two decades both the formal aspects of feature structures and their linguistic applications have been investigated extensively.

Here we sketch the apparatus that is used in constraint-based grammar formalisms only briefly. For a more in-depth discussion we refer to the literature. The classic introduction is Shieber (1986). For surveys and the discussion of technical issues we refer to Blackburn (1994), Carpenter (1992b), Keller (1993), King (1989) and Rounds (1997). These monographs and dissertations provide insights into different types of feature structure systems,
and into different approaches to formalising feature structures and feature description languages. Many of them also provide a survey of historical developments. Our brief presentation is only meant to recapitulate the major aspects of feature structures that are important for linguistic description.

**Feature structures** Feature structures are often defined as rooted, directed, acyclic graphs with labels on the edges (features) and on the nodes (sorts), with some extra requirements like the functionality constraint which states that from each node in a feature structure there can at most be one outgoing arc for each label. We discuss some further aspects using the modal approach, (Blackburn (1993), Blackburn et al. (1993), Blackburn (1994)), in which feature structures are presented as Kripke models.

**Definition 20 (Feature Structure)** Let \( \mathcal{L}, \mathcal{A} \) and \( \mathcal{B} \) be non-empty enumerable sets called attributes (or features), sorts (or types) and nominal symbols respectively. \( \langle N, \{R_i\}_{i \in \mathcal{L}}, \{Q_\alpha\}_{\alpha \in \mathcal{A} \cup \mathcal{B}} \rangle \), is a feature structure of signature \( (\mathcal{L}, \mathcal{A}) \), where

- \( N \) is a non-empty set called the set of nodes;
- for all \( l \in \mathcal{L}, R_i \) is a binary relation on \( N \) that is a partial function (transition);
- for all \( \alpha \in \mathcal{A}, Q_\alpha \) is a subset of \( N \),
- for all \( \alpha \in \mathcal{B}, Q_\alpha \) is a singleton subset of \( N \).

It is often assumed that each node can be decorated with a single sort only (we relax this requirement below). Note that nominals refer to unique points in the model.

Further restrictions can be imposed on this structure to define the specific kind of feature structure that is assumed in most of the grammatical theories using feature structures that we will discuss. The binary relation \( R_i \) (defining transitions between nodes) is defined to be a partial function. This accounts for the functionality constraint we mentioned above. Suggestions for axiomatising the notion of feature structure can be found in Blackburn (1993) who discusses several of the restrictions together with formula schemata that express the appropriateness conditions on the accessibility relation. For instance, the condition that they should be partially functional is expressed by the schema: \( (f) \phi \rightarrow [f]\phi \). However, to state such a constraint we need a language to talk about feature structures.

One option is the language \( L^N \) discussed in Blackburn (1994). Syntactically the language \( L^N \) (of signature \( (\mathcal{L}, \mathcal{A}, \mathcal{B}) \)) is a (multi-)modal propositional language containing an \( \mathcal{L} \) indexed collection of distinct modalities; an \( \mathcal{A} \) indexed collection of propositional symbols (S) and a set of symbols called the nominals \( \mathcal{N} \) which are indexed by \( \mathcal{B} \). These are used to refer to individual nodes. They will be represented by symbols \( i, j, k \ldots \). The set of well-formed formulas \( \mathcal{F} \) of \( L \) is defined as follows.
\[ \mathcal{F} ::= S \mid \mathcal{N} \mid \langle \mathcal{L} \rangle \mathcal{F} \mid \mathcal{F} \land \mathcal{F} \]

In this simple language conjunction is the only boolean operator. In line with Blackburn (1994) we could add negation and disjunction, but we will not need these. In our examples, we will also make use of other feature description languages.

Semantically, the modal formulas can be interpreted in the Kripke models as usual. The sorts represent subsets of the domain and modal operators correspond to accessibility relations. So, the Kripke structures alias feature structures serve as models for this language. The important interpretation clauses for \( L^N \) are as follows (\( p_\alpha \) is used as a variable over sorts and nominals).

\[
\begin{align*}
v(p_\alpha) &= \mathcal{Q}_\alpha \\
v(\langle l \rangle \phi) &= \{ y \mid \exists x[R_t(x, y) \& x \in v(\phi)] \} \\
v((\phi \land \psi)) &= v(\phi) \land v(\psi)
\end{align*}
\]

Similar definitions of a modal feature language can be found in Dörre et al. (1996) and van Eijck (1998). Quite a few other feature description languages have been proposed in the literature which is much too vast to do justice here. For these we refer to the literature mentioned above.

**Examples**  Given the features \( \mathcal{L} = \{ \text{SYN, BAR, NUM} \} \) and sorts \( \mathcal{A} = \{ v, 2, pl \} \), we can form the formula: \( (\text{cat} \land \langle \text{SYN} \rangle v \land \langle \text{BAR} \rangle 2 \land \langle \text{NUM} \rangle pl) \). This formula is a description of a feature structure. We can depict its interpretation by a graph where the nodes correspond to the elements of the domain, the edges between the nodes correspond to the accessibility relations (in this case labelled by a feature, i.e. the name of the relation) and the labels on the nodes to the sorts (values).

```
SYN
\downarrow v
\quad cat
BAR
2
\quad NUM
\downarrow pl
```

Such a graph can also be depicted by an attribute-value matrix like the following.

```
\begin{bmatrix}
cat & v \\
SYN & v \\
BAR & 2 \\
NUM & pl
\end{bmatrix}
```
The term *feature structure* is used to refer to objects in the Kripke model, the graphs, i.e. to the interpretation of the modal formulas. A formula from the modal language describing a feature structure is called a *feature description*. A logical language for describing feature structures will be called a *feature description language*.

As we have just illustrated, the attribute-value matrices are used to depict feature structures (some examples of this use are Pollard and Sag (1987), Carpenter (1992b), Johnson (1988)). However, they can also be used as formulas from some "attribute-value description language" (Blackburn (1994), Pollard and Sag (1994)). For the sake of readability, we will often use the attribute-value matrices as the description language as well. In the absence of negation and disjunction, the attribute-value matrix interpreted as a feature structure is the most general satisfier of the (same) attribute-value matrix interpreted as a description.

Not all formulas from the feature description language have a model that meets the requirements we mentioned earlier. For instance the formula \((\text{SYN})v \land (\text{SYN})2\) has no feature structure model if we assume that only one sort can decorate a node. The reason for this is that the functionality requirement forces us to consider only models in which the transitions on SYN lead to the same node and the formula requires us to assign two sorts to this node (v and 2).

Feature structures can get more complex than the one above. First of all the value of a feature can be a node that has further transitions. Secondly it is possible that the values for a certain feature are ‘re-entrant’ or ‘structure-sharing’. In the graph representation, this means that two edges in a graph will end up at the same node.

This graph is described by the following formula that uses a nominal to refer to the (re-entrant) node.

\[
(\text{phrase} \land \\
\langle \text{CAT}\rangle \langle \text{HEAD} \rangle i \land \\
\langle \text{DTRS}\rangle \langle \text{HDTR} \rangle \langle \text{CAT}\rangle \langle \text{HEAD} \rangle (i \land \text{noun}))
\]
In the attribute-value matrix notation the re-entrant structure is represented with so-called tags (boxed numbers are often used).

\[
\begin{array}{c}
\text{phrase} \\
\text{CAT} \\
\text{DTRS}
\end{array}
\begin{array}{c}
\text{HEAD} \\
\text{HDTR} \\
\text{CAT}
\end{array}
\begin{array}{c}
1 \\
\text{HEAD} \\
\text{noun}
\end{array}
\]

**Signature** We have seen above that the set of feature structures for a given application is defined relative to a signature defining the kind of transition relations \( \mathcal{L} \) and sorts \( \mathcal{A} \). For specific applications further constraints can be defined. In so-called typed feature structure systems, the sorts are organised into a sort-hierarchy. The set of well-formed structures can be further restricted by providing constraints on the domain and range of the accessibility relations (transition functions). We will briefly consider these restrictions on the ontology.

**Sort-hierarchy** In most current versions of feature-structure systems, the sorts are assumed to be partially ordered \( \preceq \) in the signature. If \( \alpha_1 \preceq \alpha_2 \), where \( \alpha_1 \) is a supersort of \( \alpha_2 \) (or alternatively put \( \alpha_2 \) is the subsort of \( \alpha_1 \)), this means semantically that \( v(\alpha_2) \subseteq v(\alpha_1) \). In this case, the restriction on nodes in a feature structure that they only have one sort does not hold. If a node is of some sort \( \alpha_x \) it is also of the supersort \( \alpha_y; \alpha_y \preceq \alpha_x \). For instance, if \( \text{cat} \preceq \nu \) then \( (\text{SYN}\nu \land (\text{SYN}\text{cat}) \) has a model, where the node that is reached by following the SYN edge has two sorts.

The sort-hierarchy provides a taxonomic classification of the elements in the domain.

**Appropriateness** Not every combination of a feature and a value makes sense. For instance, in a linguistic setting, where feature structures are used to represent categories, nom and acc are sorts that are appropriate for the attribute \textit{case} but not for \textit{person}.

\[
\begin{array}{c}
\text{Appropriate} \\
\text{nom-cat} \\
\text{CASE} \\
\text{nom}
\end{array}
\begin{array}{c}
\text{Not appropriate} \\
\text{nom-cat} \\
\text{PERSON} \\
\text{nom}
\end{array}
\]

Secondly, some features do not make sense for structures of a particular sort. \textit{Case} is an attribute that is suited for nouns but not verbs and \textit{tense} (with possible values like \textit{present} and \textit{past}) is suited for verbs (represented by feature structures of sort \textit{v-cat} for instance) but not nouns so a feature structure that is specified for both these attributes does not make sense. Other features like \textit{number} and \textit{person} can be appropriate for both sorts.
 CHAPTER 3

**Appropriate**

<table>
<thead>
<tr>
<th>nom-cat</th>
<th>acc</th>
<th>third</th>
<th>singular</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERSON</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUMBER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Appropriate**

<table>
<thead>
<tr>
<th>v-cat</th>
<th>past</th>
<th>third</th>
<th>singular</th>
</tr>
</thead>
<tbody>
<tr>
<td>TENSE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERSON</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUMBER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Not Appropriate**

<table>
<thead>
<tr>
<th>nom-cat</th>
<th>acc</th>
<th>third</th>
<th>singular</th>
</tr>
</thead>
<tbody>
<tr>
<td>TENSE</td>
<td>past</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERSON</td>
<td></td>
<td>third</td>
<td></td>
</tr>
<tr>
<td>NUMBER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Not Appropriate**

<table>
<thead>
<tr>
<th>v-cat</th>
<th>acc</th>
<th>third</th>
<th>singular</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PERSON</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUMBER</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Therefore, for a given application, the signature will not merely fix the sets \( L \) and \( A \) but will also indicate constraints on the domain and the range of the relations indexed by \( L \), relative to the sorts of the elements. We will say that when a feature or attribute \( l \) is appropriate for a sort \( s \), then elements of the sort \( s \) are included in the domain of the function \( R_l \). If we say that a sort is an appropriate value for a feature \( l \) then elements of the sort \( s \) are in the range of \( R_l \). Variations on this type of appropriateness conditions can be put in the signature as well.

**Totally Well-Typed and Sort Resolved**  In Pollard and Sag (1994), feature structures that model linguistic expressions are required to be totally well-typed (see also Carpenter (1992b)) and sort-resolved. Note that type and sort in this context are taken as synonyms. Such feature structures must obey the appropriateness restrictions and each node must have a 'resolved' sort (one that has no proper subsorts). Furthermore, if a feature is defined for a sort, then it must be present in the feature structure.

In this way, the signature can be used in a given application to check certain well-formedness aspects of structures, or to complete certain partial structures by type or sort inference procedures. For instance, if a structure has feature \( l \) then it must be of type \( s \) to be well-formed, if a structure is of sort \( s \) then it must (either) have feature \( l \) (or \( l' \)) to be totally well-typed, etc.

**Constraints**  For a specific application, the specification of the set of well-formed feature structures can be further refined by providing a set of sort constraints. In Carpenter (1992b) they take on the following form.

\[ \sigma \Rightarrow \Phi \]

Here \( \sigma \) is a sort and \( \Phi \) a feature description formula. The constraint is expressed in the form of an implication because "the intuitive idea is that a feature structure of type \( \sigma \) must satisfy the constraint \( \Phi \)" (Carpenter (1992b, p. 227)). In the examples below we will adopt the notation of Sag (1997) and use attribute-value matrices as the feature description language in these constraints.
So besides classes of feature structures that are well-formed, totally well-typed, sort-resolved, or combinations of these requirements we can also talk about the feature structures that satisfy the system of sort-constraints.

**Inheritance**  When a feature is appropriate for a sort it is also appropriate for all its subsorts. In other words, subsorts inherit the appropriateness conditions of their supersorts. Subsorts can differ from their supersorts by having extra features defined for them that are not defined for their supersorts. When a sort is an appropriate value for a feature then all its subsorts are appropriate for it as well.

Inheritance also applies to sort-constraints. In this case, each subsort inherits the constraints of all of its supersorts. So, if we have a constraint $\sigma \Rightarrow \Phi$ and $\sigma \preceq \sigma'$ then a feature structure of sort $\sigma'$ must also satisfy the constraint $\Phi$.

The notion of inheritance and the organisation of the appropriateness and constraint information along a sortal hierarchy plays an important role in the architecture of grammar frameworks that use sorted feature structures.

**Example**  Consider a set of features $\mathcal{L}$ and sorts $\mathcal{A}$ as follows. $\mathcal{L} = \{\text{CASE, PERSON, NUMBER, SYNCAT, TENSE}\}$ and $\mathcal{A} = \{\text{nominal, verbal, nom, acc, first, second, third, singular, plural, np, n, v, vp, present, past}\}$. The following is a well-formed feature structure according to this signature.

$$
\begin{array}{c}
\text{nominal} \\
\text{CASE} \quad \text{acc} \\
\text{PERSON} \quad \text{third} \\
\text{NUMBER} \quad \text{singular} \\
\text{SYNCAT} \quad \text{np}
\end{array}
$$

For a given application, we will define the set of sorted feature structures relative to a signature that does not only state which features and sorts can be used but also for each sort what the appropriate features are and what the appropriate values are for each of these features.

**Example 7 (Signature)** $\mathcal{L} = \{\text{CASE, PERSON, NUMBER, SYNCAT, TENSE}\}$ and $\mathcal{A} = \{\text{nominal, verbal, nom, acc, first, second, third, singular, plural, np, n, v, vp, present, past}\}$

- **nominal** has appropriate features: CASE, PERSON, NUMBER, SYNCAT, where CASE can have values nom and acc, PERSON can have values first, second, third, NUMBER can have values singular and plural, SYNCAT can have values np, n.
• verbal has appropriate features: PERSON, NUMBER, SYNcatid and tense, where PERSON can have values first, second, third, NUMBER can have values singular and plural, SYNcatid can have values vp, v and TENSE can have values present, past.

• None of the other sorts have appropriate features.

Given a signature Σ we take the set $\mathcal{FS}_\Sigma$ to be the set of sorted feature structures for which the appropriateness constraints formulated in the signature Σ hold. To illustrate this we provide some instances of sorted feature structures that are not in $\mathcal{FS}_\Sigma$ with Σ as in the example above.

\[
\begin{bmatrix}
\text{verbal} \\
\text{CASE nom}
\end{bmatrix}
\begin{bmatrix}
\text{verbal} \\
\text{TENSE first}
\end{bmatrix}
\begin{bmatrix}
\text{verbal} \\
\text{SYNcatid np}
\end{bmatrix}
\]

The first structure is not in $\mathcal{FS}_\Sigma$ because the feature CASE is not appropriate for verbal. The second structure is not in $\mathcal{FS}_\Sigma$ because first is not a sort that is appropriate for the feature TENSE in feature structures of sort verbal. Similarly, the third structure is not in $\mathcal{FS}_\Sigma$ because np is not appropriate for the feature SYNcatid in feature structures of sort verbal (although it is appropriate for the feature SYNcatid in structures of the sort nominal).

Subsumption  Note that sorted feature structure are not required to contain all the features that are appropriate for its sort. All of the following structures are in $\mathcal{FS}_\Sigma$.

\[
\begin{bmatrix}
\text{verbal} \\
\text{TENSE past}
\end{bmatrix}
\begin{bmatrix}
\text{verbal} \\
\text{TENSE past} \\
\text{NUMBER third}
\end{bmatrix}
\]

Typical for this collection is that the second is an extension of the first and the third is an extension of the second and the first. The first is said to subsume (⊆) the second and the third one and the second subsumes the third. We will also say that each structure subsumes itself. This subsumption ordering is a partial ordering. It orders the feature structures on an informativeness scale. A feature structure A that subsumes a feature structure B is less informative than B (or as informative as).

For technical definitions we refer to the literature mentioned above. We can make more precise what it means to be less informative by looking at the different information carriers in a feature structure. We say that a feature structure A subsumes a feature structure B if (i) B is of the same sort as A or of a more specific sort (subsort), (ii) B has (at least) all the features in A, (iii) for all these (common) features the values of those in A subsume those in B and (iv) all re-entrancies in A also hold in B.
Unification

Unification refers to an operation on feature structures. Informally, the unification of two feature structures is the least informative feature structure that contains all the information of the two structures. Note that unification can fail if the two structures contain incompatible information.

Now that we have presented the essentials of feature structures and how they are used to classify some domain of interpretation, we will turn to their application in grammatical description.

### 3.2 Feature Structures and Grammars

In the following sections, we illustrate some of the functions of feature structures in grammatical description. One of these involves the classical use as lexical or syntactic category. The work by Kay (1985), Kay (1984) on functional unification grammar was important in introducing the idea that feature structures are 'general purpose', in that not only categories, but also phrase structure and other aspects of linguistic description could be so encoded. Currently HPSG is the prime example of what has become known as a 'stand-alone formalism'.

We illustrate some of these aspects below by presenting some small grammar fragments. The first example is taken from Borsley (1991) using simple feature structures as categories. Next we present a rule in a style that is familiar from unification-based grammars. The third example is a kind of slimmed down version of HPSG, using similar techniques to specify a stand-alone grammar.

With these illustrations we want to show which benefits feature structures can provide to the descriptive grammarian. First of all, feature structures provide a way to refer to multiple properties of elements in a domain. This means that the classification possibilities are refined to account for cross-classification. The grammar writer can conveniently classify linguistic structures along a number of different dimensions each one indicated by an attribute.

Secondly, the structure also provides a partial ordering which provides names for the hierarchical classification of elements in a domain. This will make it possible to express generalisations succinctly (or at least, to simplify the formulation of the grammar).

Finally, the decomposition of the description along several dimensions makes it possible to present a grammar as a combination of partial descriptions that each describe a specific aspect of the linguistic structures. In other words, the use of feature structures and their descriptions makes it possible to factorise grammatical information into different components.

We will now present three examples in which feature structures and their descriptions are used in grammars pointing out the benefits of feature structures that we just mentioned.
3.2.1 Subsumption-based Grammars

In this section we present a simple type of phrase structure grammar in which non-terminal (category) symbols are replaced by feature structures in both rewriting rules and lexical entries. We define a subsumption-based system adapted from Borsley (1991) that shows how feature structures allow for cross-classification and how the subsumption ordering can be used to express certain linguistic generalisations. It uses only very simple feature structures: unsorted, no embedding (features can only have atomic values) and no structure sharing (re-entrancy).

**Subsumption-based Grammars**  We start with a simple change to the definition of context free grammars that we presented in Chapter 1.

**Definition 21 (CFG with Feature Specifications)** Given a set of symbols $V$ and a signature $\Sigma$, let $C$ be the set of feature specifications $FS$. A context-free grammar with feature specifications as categories is a pair $(Lex, Rules)$, where $Lex \subseteq V \times C$ and $Rules \subseteq C \times C^*$. The set of $Rules$ is finite.

This parallels Definition 1 of Context Free Grammar almost exactly. The only difference involves the definition of category.

**Definition 22 (Language)** A language, $L_G \subseteq V^* \times C$ is defined by the context free grammar with sorted simple feature structures as categories $G = \langle Lex, Rules \rangle$ as the least set, such that the following conditions hold:

(i) $Lex \subseteq L_G$

(ii) if $(c, c_1 ... c_n) \in Rules$ and $(s_1, c'_1) \in L_G, ..., (s_n, c'_n) \in L_G$ then $(s_1 ... s_n, c)$ \in L_G provided $c_1 \sqsubseteq c'_1$ or $c'_1 \sqsubseteq c_1$ ... $c_n \sqsubseteq c'_n$ or $c'_n \sqsubseteq c_n$.

The only difference from Definition 2 involves the proviso that the categories of the constituents do not have to be identical to the categories in the rules but that only a subsumption relation has to hold between them in one direction or the other. In Borsley (1991), one direction $c'_i \sqsubseteq c_i$ is reserved for lexical elements and the other for non-lexical ones.

**Example 8 (Simple Sorted Language)** We provide a simple grammar for phrases like the sentence *A girl sees the girls*. We simplify the notation for feature structure matrices, leaving out the attribute names and connecting the values by $\wedge$.

- **Signature:**
  
  $L = \{\text{SYN, BAR, NUM}\}$

  $A = \{\text{cat, d, n, v, 0, 1, 2, pl, sg}\}$

  *cat* has appropriate features: SYM with possible values $d, n, v$; BAR with possible values $0, 1, 2$; and NUM with possible values $sg$ and $pl$. 

• Lex = {
  \{ (o,ldn o n sg), (the,ldn no)1, (girl,ln O Ast)), (girls,ln O Apl)), (sees, [v O A st]), (see, [v O onpl]) \}
• Rules = {
  [v \wedge 2 \wedge sg] \rightarrow [n \wedge 2 \wedge sg][v \wedge 1 \wedge sg],
  [v \wedge 2 \wedge pl] \rightarrow [n \wedge 2 \wedge pl][v \wedge 1 \wedge pl],
  [v \wedge 1 \wedge sg] \rightarrow [v \wedge 0 \wedge sg][n \wedge 2],
  [v \wedge 1 \wedge pl] \rightarrow [v \wedge 0 \wedge pl][n \wedge 2],
  [n \wedge 2 \wedge sg] \rightarrow [d \wedge 0 \wedge sg][n \wedge 1 \wedge sg],
  [n \wedge 2 \wedge pl] \rightarrow [d \wedge 0 \wedge pl][n \wedge 1 \wedge pl],
  [n \wedge 1 \wedge sg] \rightarrow [n \wedge 0 \wedge sg],
  [n \wedge 1 \wedge pl] \rightarrow [n \wedge 0 \wedge pl]
}

The following steps show that sees the girls is of the category [v \wedge 1 \wedge sg].

1) \langle girls, [n \wedge 1 \wedge pl] \rangle \in L_G because
   [n \wedge 1 \wedge pl] \rightarrow [n \wedge 0 \wedge pl] \in Rules,
   \langle girls, [n \wedge 0 \wedge pl] \rangle \in L_G (Lex), and
   [n \wedge 0 \wedge pl] \subseteq [n \wedge 0 \wedge pl]

2) \langle the \circ girls, [n \wedge 2 \wedge pl] \rangle \in L_G because
   [n \wedge 2 \wedge pl] \rightarrow [d \wedge 0 \wedge pl][n \wedge 1 \wedge pl] \in Rules,
   \langle the, [d \wedge 0] \rangle \in L_G (Lex),
   [d \wedge 0] \subseteq [d \wedge 0 \wedge pl],
   \langle girls, [n \wedge 1 \wedge pl] \rangle \in L_G (1), and
   [n \wedge 1 \wedge pl] \subseteq [n \wedge 1 \wedge pl]

3) \langle sees \circ the \circ girls, [v \wedge 1 \wedge sg] \rangle \in L_G because
   [v \wedge 1 \wedge sg] \rightarrow [v \wedge 0 \wedge sg][n \wedge 2] \in Rules,
   \langle sees, [v \wedge 0 \wedge sg] \rangle \in L_G (Lex),
   [v \wedge 0 \wedge sg] \subseteq [v \wedge 0 \wedge sg],
   \langle the \circ girls, [n \wedge 2 \wedge pl] \rangle \in L_G (2), and
   [n \wedge 2] \subseteq [n \wedge 2 \wedge pl]

Partial Information and Generalisation The use of underspecified structures in the lexicon and the rules allows one to simplify the grammar. If we were not allowed to do this we would have to duplicate the entry for the (for sg and pl) and the rules combining a verb with an object. In this case we assign the a category that is underspecified with respect to number to signal that it can have either value. The same holds for the rules. In this particular case, the object noun phrase can be either singular or plural. By using an underspecified category, we say that the rule applies to the more general class of noun phrase expressions, including the singular and plural subclasses.
3.2.2 Unification-based Grammars and Beyond

Subsumption-based grammars do not exploit all the possibilities offered by the feature structures for their use in grammar formalisms. Unification-based grammars, as they are usually defined, go a step further. We will not paraphrase their definition here as our purpose at this point is merely to show by a simple example how they can capture certain linguistic generalisations.

The example grammar above contains some typical regularities. For instance, there are two rules rewriting \([v \wedge 2]\) categories: one for the singular and one for the plural. They require that the values for the number feature in the mother and the daughters be the same. In unification-based phrase structure grammars, this pattern is captured by collapsing the two rules into one, using a notation like the following.

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

This rule abstracts over the differences between the two rules and shows what they have in common. The rule also expresses a pattern of feature distribution in sisters and daughters. The use of the same tag in different structures is meant to indicate that the values co-vary. This means that when the rule is applied, all the tags \([1]\) must be instantiated in the same way. If we interpret the matrices as descriptions, we can view the tags as a kind of variables. In such a rule they are then assumed to be in the scope of a universal quantifier. Note that we use \(\alpha\) instead of \(1\).

\[
\forall \alpha(
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM} \ \alpha
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM} \ \alpha
\end{bmatrix}
)
\]

For the formal details that ensure that this type of rule behaves in the intended way, we refer to Kay (1979), Pereira and Shieber (1984), and Carpenter (1992b, p. 185ff). Here we will only illustrate how such a rule can be applied in linguistic description. We use one of the formulations that is given in Shieber (1986, p. 21ff), using what he calls the bottom-up (destructive) unification strategy.

Observe that the same pattern occurs in the rules for \([n \wedge 2]\) as in the rules for \([v \wedge 2]\). They can also be collapsed into one rule.

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\rightarrow
\begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\]
Such a rule can be seen as a description or a notational variant of a feature structure like the following.

\[
\begin{align*}
0 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{n} \\
2 \\
1
\end{bmatrix} \\
1 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{d} \\
0 \\
1
\end{bmatrix} \\
2 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{n} \\
1 \\
1
\end{bmatrix}
\end{align*}
\]

As before, we consider pairs of expressions and categories. In this case, the latter are feature structures. Suppose that the following pairs are in the language.

\[
\langle \text{the}, \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR}
\end{bmatrix}
\begin{bmatrix}
\text{d} \\
0
\end{bmatrix} \rangle
\]

\[
\langle \text{girls}, \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{n} \\
1 \\
pl
\end{bmatrix} \rangle
\]

Now we can apply the rule to derive the category for \textit{the o girls} by ‘unifying in’ the structures for the determiner and for the noun at the values for the features 1 and 2 in the rule. This results in the following structure, which can be seen as a representation of the phrase structure tree for \textit{the o girls}.

\[
\begin{align*}
0 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{n} \\
2 \\
1
\end{bmatrix} \\
1 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{d} \\
0 \\
1
\end{bmatrix} \\
2 & \quad \begin{bmatrix}
\text{cat} \\
\text{SYN} \\
\text{BAR} \\
\text{NUM}
\end{bmatrix}
\begin{bmatrix}
\text{n} \\
1 \\
pl
\end{bmatrix}
\end{align*}
\]
CHAPTER 3

The category of the combination, *the girls* can be taken to be the value of the feature $0$.

\[
\begin{bmatrix}
\text{cat} \\
\text{SYN} & n \\
\text{BAR} & 2 \\
\text{NUM} & \text{pl}
\end{bmatrix}
\]

Because of the re-entrancies in the rule, we know that the value for \text{NUM} must be plural. In this bottom-up interpretation, the value of the noun is passed on, so to speak, to the noun phrase. The re-entrancies are not just used for passing-on values but can also be used to check agreement of values. This would have happened in the example when the value for \text{NUM} was specified on the determiner as well.

**More generalisations** In the same spirit we can continue the search for other patterns in the rules, i.e. for regularities across rules. The general goal could now become stating general constraints that rules have to obey, either per feature or per constellation of features and by these constraints characterising the set of possible phrase structures for a language. Such a move away from concrete phrase structure rules to principles or constraints on possible rules or possible phrase structures occurred in generative (transformational) grammars with the introduction of, for instance, X-bar theory. Similarly, in Generalised Phrase Structure Grammars (GPSG, (Gazdar et al. 1985)) or HPSG (Pollard and Sag 1994) feature distribution principles and rule schemata replace the specific phrase structure rules. In the next section we provide a small HPSG-style grammar. We will now point out the kind of generalisations that will be captured.

One of the patterns that emerges from the rules in Example 8 concerns the features \text{SYN} and \text{BAR}. Looking closely at the example we see that all rules are instantiations of two schemata:

\[
\begin{align*}
\text{SYN} \alpha & \rightarrow \text{BAR} 2 \\
\text{BAR} & \rightarrow \text{SYN} \beta
\end{align*}
\]

\[
\begin{align*}
\text{SYN} \alpha & \rightarrow \text{BAR} 2 \\
\text{BAR} & \rightarrow \text{SYN} \beta
\end{align*}
\]

Here the round brackets around the feature structure denote optionality. These schemata reflect, of course, two of the generalisations expressed in X-bar theory (adjunct structures are not present in our simple grammar):

\[
\begin{align*}
\text{XP} & \rightarrow \text{YP} & \text{X'} & \rightarrow \text{YP} \\
\text{X'} & \rightarrow \text{X} & \alpha^2 & \rightarrow \beta^2 & \alpha^1 \\
\text{XP} & \rightarrow \text{YP} & \alpha^1 & \rightarrow \alpha^0 & \beta^2
\end{align*}
\]

The decomposition of the information contained in a category allows us to see regularities within and across rules. Instead of providing separate
rules for each construction, it is now possible to characterise the set of rules indirectly by stating the general formats and principles they have to obey. We illustrate this approach in the next section. This type of generalisation is in many respects similar to the one reached in the rules of combination of say an applicative categorial grammar where the schemata generalise over all specific category instantiations.

\[
\begin{align*}
X & \rightarrow Y \quad Y\backslash X \\
X & \rightarrow X/Y \quad Y
\end{align*}
\]

It is possible to go a step further still. The decomposition of features also allows one to state regularities per feature. This means that we can either replace concrete phrase structure rules by general principles ('feature distribution principles') or, seen from another perspective, formulate general constraints on the set of possible phrase structure rules. The information contained in the set of rules can thus be factorised into different principles. To make this more precise, we could represent the information in the trees, abstracting away from the precise categories into several principles, which we will call the category feature principle, the agreement principle and the bar-level principle. At this point we will formulate them informally to point out how such principles generalise across rules.

- **Category feature principle:** Each rule must specify a daughter that has the same category as the mother (if the bar-level of the mother is two, this is the right daughter, if the bar-level of the mother is one, this is the first daughter). This daughter will be called the head.

- **Agreement principle:** Head-daughters carry the same values for the agreement features (for instance NUM) as their mothers. They are said to agree with them. In case the bar-level of the head is 2, then the non-head daughter will agree with the mother as well.

- **Bar-level principle:** The bar-level of the mother is the bar-level of the head daughter + 1. The bar-level of non-head daughters is always 2.

**Summary** Instead of atomic symbols we have used feature structures to model categories which makes it possible to combine information about several properties of expressions in a structured notation. We say that feature structures allow for cross-classification. Feature structures are also ordered by a subsumption relation which makes underspecification possible in the lexicon and in the set of rules.

Underspecification reduces the number of rules in a grammar or captures regularities across rules. The use of the possibilities offered by the hierarchical classification of the domain allows one to abstract away from properties where they are not needed.

We have also illustrated the use of variables in rules. These allow underspecification of rules (collapsing different rules into one) together with the
statement of co-variation of instantiation. This provides a way to express generalisations pertaining to specific features or feature configurations. The grammatical information is thus factorised into different principles.

### 3.2.3 Stand-Alone Feature Structure Grammars

The move from concrete phrase structure rules to general principles was taken further in feature structure grammars in GPSG and HPSG. We will now sketch the latter approach in which feature structures have a much more prominent role than in the phrase structure grammars presented above. We illustrate the decomposition of phrase structure information into principles and also point out the parallel decomposition of lexical information. The presentation will also make more precise the move away from concrete phrase structure grammar rules to a system where the rule component has almost completely been dissolved in favour of general principles.

So far we have considered the use of feature structures as categories replacing the atomic category symbols in phrase structure grammars. Theories like HPSG make use of more complicated feature structures not only for the representation of categorial information but also for other purposes like the representation of constituent structure or semantics.

We will present a slimmed-down version of an HPSG-like grammar instead of a complete HPSG grammar. This should suffice as an outline of the general approach to specify a grammar the HPSG way. In the next chapter we will contrast this approach with the deductive approach of the type- logical grammars which we discussed in the previous chapter.

In general, an HPSG grammar consists of two components. The first is a signature, defining the attributes and the sorts and the appropriateness relations holding between them. The signature plays an important part in characterising the domain of interpretation. The second component of grammatical specification is the grammar proper, which in this case is simply a formula from the feature description language. A language in this set-up is defined as a particular set of feature structures. A grammar, $G$, defines a language $L_G$ as the set of feature structures in terms of the satisfaction relation: $L_G = \{FS \mid FS \models G\}$. Again, we will show how feature structures allow for cross-classification, generalisation and factorisation.

**Signature** The feature structures that represent linguistic expressions in HPSG are those of sort sign that conform to the constraints as formulated in the grammar formula.

In the HPSG signature, there are two kinds of signs: *words* and *phrases*. Feature structures of sort *sign* have features PHON and SYNSEM. Feature structures of sort *phrase* have the extra feature DTRS. PHON values represent the phonology of the expression modelled by the feature structure. SYNSEM values encode the syntactic (category) and semantic information.
about the expression. The attribute DTRS provides information about the constituent structure of the expressions. The attribute value matrices of the sort sign will thus have the following form:

\[
\begin{bmatrix}
\text{word} & \text{phonology} \\
\text{PHON} & \text{synsem} \\
\text{SYNSEM} & \text{DTRS}
\end{bmatrix}
\begin{bmatrix}
\text{phrase} & \text{phonology} \\
\text{PHON} & \text{synsem} \\
\text{SYNSEM} & \text{DTRS}
\end{bmatrix}
\begin{bmatrix}
\text{local} & \text{category} \\
\text{HEAD} & \text{subject} \\
\text{OBJEJECT} & \text{content} \\
\text{CONT} & \text{context} \\
\text{NONLOC} & \text{nonlocal} \\
\text{constituent-structure}
\end{bmatrix}
\]

In more detail, feature structures of type PHRASE look like this, in the simplified version of HPSG that we present here.

The value of the SYNSEM feature is itself a complex feature structure with attributes LOCAL and NONLOCAL. The structures that are the value of LOCAL are again complex structures with attributes CATEGORY for syntactic, CONTENT for semantic and CONTEXT for pragmatic information (roughly speaking). The structures that form the value of NONLOCAL represent information pertaining to dislocated constituents involved in long-distance dependencies.

CATEGORY represents more or less the information of a type in categorial grammar. In Pollard and Sag (1987) it bears the attributes HEAD and SUBCAt, where HEAD represents the part of speech of the expression together with morphosyntactic information (the precise kind of information depending on the part of speech) and SUBCAT represents information about the valency of the expression. This information is represented by a list of feature structures of sort synsem and encodes the restrictions on the number and type of complements that combine with the expression.

In our simplified version of HPSG we will replace the idea of a list of complements by providing separate features for each type of complement. In our simplification we can make do with SUBJECT and OBJECT. In the literature on HPSG one can find various versions of the treatment of complementation that differ in the precise formulation. Our version here is a combination of aspects of several of these. It resembles the different
VALENCE features in more recent versions of HPSG where the selection of subject, specifier and complement daughters is distinguished by features like \textit{subj}, \textit{spr} and \textit{comps} (see for instance Sag (1997) and Chapter 9 of Pollard and Sag (1994)). We ignore the feature \textit{arglist} here.

By now it is obvious that in HPSG, feature structures are not simply used to model categories but also other aspects of grammar. Constituent structure is represented by the value for the feature \textit{dtrs}, which is only appropriate for structures of the sort \textit{phrase}. This structure can be said to represent a phrase structure tree. Different kinds of trees are distinguished depending on the kind of relation that holds between the daughters. In most cases there is one daughter that is the \textit{head} of the constituent structure. The attribute for this daughter is called \textit{hdtr} and the value is of sort \textit{sign}. Sisters of the head daughter can be distinguished by the kind of dependency that holds between them and the head. In HPSG, 1987 style, the complement daughters (\textit{cdtrs}) form a list of signs. In our presentation we will distinguish between subject daughters (\textit{sdtr}) and object daughters (\textit{odtr}). Other types of daughters are adjunct daughters and filler daughters. The precise inventory does not matter here.

\textbf{Example} The following three simplified structures model the expressions \textit{he}, \textit{walks} and \textit{he walks}.

\begin{itemize}
  \item \textbf{PHON} \textit{he} \textbf{CAT} \textbf{HEAD} \textbf{noun} \textbf{nom}
  \item \textbf{PHON} \textit{walks} \textbf{CAT} \textbf{HEAD} \textbf{subject} \textbf{verb} \textbf{vform} \textbf{fin} \textbf{noun} \textbf{nom}
  \item \textbf{PHON} \textit{3 \textit{he} \textit{walks}} \textbf{CAT} \textbf{HEAD} \textbf{1 \textit{verb} \textit{fin}} \textbf{PHON} \textbf{4 \textit{walks}} \textbf{CAT} \textbf{1 \textit{verb} \textit{fin}} \textbf{PHON} \textbf{2 \textit{he}} \textbf{CAT} \textbf{2 \textit{noun} \textit{nom}}
\end{itemize}
In a more conventional tree form we can represent the feature structure for *he walks* as follows, where the feature structures for HDTR and SDTR represent the constituent daughters and the feature structure for CAT decorates the mother node.

```
[ HEAD 1 [ verb VFORM fin ] ]
```

```
[ HEAD 2 [ noun CASE nom ] ]
```

```
[ SUBJECT HEAD [ verb VFORM fin ] ]
```

```
[ CAT 2 ]
```

```
[ CAT 1 ]
```

```
[ HDTR CAT [ HEAD 1 ] ]
```

**Grammar and Language** In HPSPG-style formalisms, a language is modelled by the set of (totally well-typed and sort-resolved) feature structures that satisfy a grammar (a formula from the feature description language). For the simple language consisting of the three feature structures above we can define the grammar to consist of subformulas for the lexical expressions and principles governing the combination of constituent expressions into phrases.

The grammar formula has the form \( G = Lex_1 \lor Lex_2 \lor (Prin_1 \land Prin_{2a} \land Prin_{2b}) \). The principles are defined as type-constraints. In this sense they are similar to the definitions in Sag (1997). We will use his notation for type-constraints with attribute-value matrices as description language.
A note on the simplifications: we have left out quite a few features in the example grammar (like \textit{CONTENT} and several morphosyntactic features) and thereby reduced the structure in size:

\[
\begin{bmatrix}
\text{SYNSEM} & \text{LOCAL} & \text{CAT} & \cdots & \cdots\n\end{bmatrix} \sim \begin{bmatrix}
\text{CAT} & \cdots \n\end{bmatrix}
\]

This gives rise to a certain number of oversimplifications. For instance, the second principle states that the \textit{CAT} information of the \textit{SUBJECT} feature of the head daughter must be re-entrant with the \textit{CAT} information of the subject daughter (SDTR). In HPSG the values that are shared are \textit{synsem} objects, including the feature \textit{CAT}, but also \textit{CONTENT} and \textit{CONTEXT}. Also note that in head-object combinations we want the information on the object value of the verb to be re-entrant with the head value of the object daughter. The subject value of the verb must percolate to the verb phrase.

**Principles for Factorisation** HPSG uses feature structures to model every aspect of linguistic analysis as we have just seen. Feature structures are general information structures that need not be used merely for syntactic categories but they can encode phrase structure and semantic information as well. The information that is contained in the rules in a context-free grammar is now distributed over a number of principles that are formulas constraining the possible grammatical feature structures.

The first principle is called the \textit{head feature principle}. It captures a part of the X-bar theory, or projection principle. It says that in headed phrases the head features of the head daughter are projected onto the mother by requiring them to be re-entrant. This captures part of the agreement principle that we mentioned before.

The second and third principle, which can be called the \textit{subcategorisation principles}, resemble the application schema of AB-categorial grammars. The \textit{SUBJECT} feature on the verb \textit{walks} corresponds to the argument position of a functor category. The second principle requires that in head-subject-phrases this information on the head-daughter is re-entrant with the infor-
mation on the daughter that is the subject. The third principle works the same way.

**Semantics** The meaning of expressions is represented by a logical language coded in a feature-structure notation. In HPSG, situation semantics is taken as the basis for the semantic analysis of signs. For the encoding of the semantic values it is necessary to define the appropriate sorts and features in the signature. Lexical items are provided with a feature structure representation of their lexical semantic values. The semantic representation of phrases is computed from the lexical semantics of the parts by principles. The major semantic principle in HPSG states that the semantic value of a phrase can be identified with the semantic value of the head daughter.

**Inheritance** The grammar above presents principles which illustrate the decomposition of the presentation of phrase structural information. A principle can be seen as a module that deals with a particular aspect of phrase structure (particular features in particular configurations). It provides partial information and generalises over different rules.

Our principles have been formulated as type constraints. The first principle holds for all phrases that have a HDrR (those of type headed-phrase). The second principle holds for those headed phrases that have a subject daughter head-subject-phrase. We assume a phrasal hierarchy such that head-subject-phrase is a subsort of headed-phrase. This means that, by inheritance, the first principles also holds for the more specific head-subject phrase. This becomes clear in the example, where the feature structure for he walks is constrained by both principles. The example thus shows two important aspects of the organisation of grammatical information.

- Information is distributed hierarchically into taxonomies.
- Information is factorised into principles according to the feature or group of features involved (and the type of structure).

A similar decomposition and factorisation can be effected for lexical information. For instance we can introduce types like \( v \) and \( n \) to the type system and introduce constraints to define the feature structures that correspond to such types.

\[
\begin{align*}
  v & \Rightarrow [ CAT [ HEAD \text{ verb} ] ] \\
  n & \Rightarrow [ CAT [ HEAD \text{ noun} ] ] \\
  vf & \Rightarrow [ CAT [ SUBJECT [ v [ HEAD fin [ VFORM \text{ noun} ] [ CASE \text{ nom} ] ] ] ] ] \\
\end{align*}
\]

These types capture information that is common to a lot of lexical entries. All verbs can be said to be of type \( v \). The information contained in
individual entries can now be reduced drastically. For instance, the verb *walk* from the example above can now be specified in the lexicon as:

\[
Lex_2 \left[ \begin{array}{c}
    vf \\
    PHON \quad walks
\end{array} \right]
\]

This technique of structuring the information has two benefits. First, it functions as an abbreviation of the information contained in individual entries. Secondly, it allows the formulation of general patterns common to a whole class of structures.

In this example, the lexical entry is assigned the type *vf*. It is said to inherit the information that comes along with this type. The type *vf* in its turn inherits information from *v*. In many implementations of lexicons for feature-structure grammars, inheritance involves not just one type but multiple types, hence multiple inheritance. We can illustrate this by assuming a further type 3sg which is associated with the following constraint.

\[
3sg \Rightarrow \left[ \begin{array}{c}
    v \\
    CAT \quad SUBJECT \\
    CAT \quad \begin{array}{c}
        \text{PERSON} \\
        \text{NUMBER} \\
        3 \quad sg
    \end{array}
\end{array} \right]
\]

The lexical entry for *walks* provides the information about the verb through multiple inheritance from *vf* and 3sg, given that *vf* ≤ *vf3sg* and 3sg ≤ *vf3sg*.

\[
Lex_2 \left[ \begin{array}{c}
    vf3sg \\
    PHON \quad walks
\end{array} \right]
\]

Such techniques for structuring lexical and phrasal information through multiple inheritance have been studied extensively in the literature. For more details we refer to Shieber (1986) for simple inheritance mechanisms using so-called *macros*, and to Gazdar and Daelemans (1992a), Gazdar and Daelemans (1992b) and Briscoe et al. (1993) for a collection of papers all dealing with this topic including discussion of nonmonotonicity and defaults in inheritance hierarchies (what to do when inheritance from different types leads to conflicting information). A related mechanism to specify linguistic information into hierarchies using non-monotonic inference rules is provided by the DATR system (Evans and Gazdar (1989)).

**Summary**

In this chapter we have introduced the notion of feature structure and discussed several options for using it in grammatical description. We saw how these general purpose information structures can be used to represent categories or other linguistic structures. The system of typed feature structures and the associated logic can be used as a general-purpose grammar formalism in which different types of grammar can be encoded.
Feature structures provide a more fine-grained classification of objects by allowing groupings of objects along several dimensions, they also offer a way to classify objects hierarchically or taxonomically, making it possible to refer to more general or more specific classes of objects. By allowing reference to different aspects of signs it also provides a way to organise the grammatical information in different principles. Both the hierarchical classification and the factorisation potential can be used to capture linguistic generalisations in the lexicon and in phrasal constructions.
In the previous chapters we have presented two approaches to the specification of (natural) language grammars, emphasising the way in which the grammars classify expressions and define the complex constructions of a language.

We have shown in Chapter 2 how generalised versions of the Lambek-style categorial grammars provide a dedicated, fine-grained instrument to define and control different ways in which linguistic objects can be combined. The general logic of residuation for the binary connectives \( \backslash, \cdot, / \) is used as a base logic for composition. Different modes are distinguished by different versions of these connectives whose behaviour is fixed by a set of structural rules. This leads to a fine-grained classification (a landscape as it has been called) of compositional options.

Feature-structure grammars, on the other hand, provide mechanisms to fine-tune the classification potential of grammars in several ways: expressions can be classified along multiple dimensions, information is structured hierarchically in cross-classifying inheritance structures and factored into different principles. This makes it possible to express linguistic generalisations succinctly.

This discussion leads up to the obvious question whether it is possible to define a grammatical framework in which the logical approach of the extended categorial grammar is combined with mechanisms that have the same benefits as feature structures. Several of such frameworks have been proposed in the literature and they will be reviewed in Part II. Our own version is presented in Part III. Before we turn to these proposals, however, we want to discuss some more general aspects of this type of combination.

There are a number of reasons why a naive combination that comprises the complete type-logical approach together with the complete constraint-based approach is unmotivated, unfeasible, and unwanted. In the next section, we will point out a first reason that stems from a technical difference in the approaches that leads to problems for combinations. These problems will be illustrated by several systems proposed in the literature in Part II.

A second type of reason why a simple combination is perhaps not the optimal framework for linguistic description is worked out in Section 4.2 of this chapter. In the previous chapters we have seen how both the type-logical grammars and the constraint-based grammars define a language. For many aspects of linguistic description each approach has its own specific way of dealing with them. Now the question arises what we need feature structures for in type-logical grammars if we already have other techniques.
to describe the same phenomena. In Section 4.2 we will try to get an estimate of the existing overlap between the coverage of the empirical issues and the techniques used for this. More particularly, we want to find the residual phenomena for which refining the categories seems warranted.

The third type of argument against a simple combination builds on the previous idea. Once we have isolated the domain for which constraint-based extensions are useful, the question arises whether this is the only technique available to get the same results or whether other techniques can be used instead. For instance, one could look for other options that fit better in the resource and structure-sensitive perspective of the type-logical grammars. In Part III this argument will be further developed and resource-sensitive alternatives will be presented.

4.1 Global Grammar Architecture

Let us first compare the global set-up of the categorial grammar and the HPSG grammar. If we consider the modal perspective on feature structures than both the categorial grammars and the feature structure grammars operate with formulas from a multi-modal language. However, there are important differences between the precise syntax of the language and the use of the formulas on several levels. Obviously, the same device is used for different functions.

**Logic: theory and framework** An important difference between the type-logical and the constraint-based framework which we pointed out before is that in the former the logic of the constants is used to define part of the linguistic theory, whereas in a constraint-based framework the logic is used to define the framework in which the theory is formulated. The typical binary modal operators /, \ are only found in categorial grammars. They are the primitive constants of the grammar logic defining linguistic selection and composition. In HPSG these aspects are described by feature structures that appear as values for such attributes as SUBCAT (or other selection features like SUBJECT and OBJECT as we defined in the fragment of the previous chapter) and DTRS. Notice that there is nothing logical about the fact that precisely these features are used to define selection and composition. As the symbols SUBCAT and DTRS belong to the non-logical part of the vocabulary of the modal language, there is nothing in the logical rules for (SUBCAT) that defines selection. Indeed, the logical rules for (SUBCAT) are the same as the rules for any other attribute, like (PHON) or (VFORM). Its meaning as a selection feature is stipulated as part of the 'theory' that is laid down in the formula specifying the subcategorisation principle. The logical properties of (·) are not used for defining a language but for the definition of the framework. Of major interest, therefore, is the work by Moshier (1997) to capture foundational concepts (such as the head feature principle) as in-
dependent principles in a language of category-theoretic constraints and to define a featureless HPSG.

When we look at implementations of constraint-based formalisms, then the major benefit of having a general purpose system is that “it allows the linguist to rapidly prototype new linguistic theories and test their behaviour computationally” (Carpenter (1992b), see also Shieber (1987)). As Morrill (1994, p. 180) points out: “From a type-logical perspective the merits and limitations of unification can be understood, but unification does not provide an instructive ‘basis’ for grammar.”

**Logic: application and subcategorisation** Consider the following HPSG-style lexicon which allows us to describe the verb phrase *kisses Mary* given the principles specified in the previous chapter.

\[
\begin{array}{c}
\text{word} \\
\text{CAT} & \text{Mary} \\
\text{HEAD} & \text{noun} \\
\end{array}
\]

\[
\begin{array}{c}
\text{word} \\
\text{CAT} & \text{kisses} \\
\text{HEAD} & \text{VERB} \\
\text{VFORM} & \text{fin} \\
\text{NOUN} & \text{nom} \\
\text{CASE} & \text{acc} \\
\end{array}
\]

\[
\begin{array}{c}
\text{PHON} & \text{kisses Mary} \\
\text{CAT} & \text{HEAD} \\
\text{SUBJECT} & \text{1} \\
\text{DTR} & \text{HEAD} \\
\text{SUBJECT} & \text{2} \\
\text{ODTR} & \text{HEAD} \\
\text{Mary} & \text{3} \\
\end{array}
\]

What is interesting about this example is that the information on *Mary* in the verb phrase combines information that stems from the lexical entry for *Mary* (*noun*, *sg*) and from the lexical entry for *walks* (*acc*). The combination is effected through the subcategorisation principle that requires the structures to be re-entrant. Both the noun phrase and the selection feature
(OBJECT) on the verb are not completely specified. The subformula describing Mary is underspecified for CASE and the object selection feature is underspecified for the attribute NUMBER. The constraint on head-object-phrases requires the descriptions of Mary and the OBJECT feature to combine in the representation of the phrase.

Now, let us compare this with a similar verb-object combination in a categorial analysis. The combination of the verb and the object is effected using application in a categorial grammar. The verb is categorised \((\text{NP}\backslash\text{S})/\text{NP}\), which we will abbreviate to \(\text{VP}/\text{NP}\), and the noun phrase simply \(\text{NP}\). In the following derivation we use subscripts to identify the different occurrences of the noun phrases. The combination of the object noun phrase with the verb is shown in the following derivation. We choose the Gentzen sequent notation here because it shows how the basic categories in a derivation match up.

\[
\begin{align*}
\text{VP}_1 & \to \text{VP}_4 & \text{NP}_3 & \to \text{NP}_2 \\
\text{VP}_1/\text{NP}_2 \circ \text{NP}_3 & \to \text{VP}_4
\end{align*}
\]

Semantically, the sequent operator \(\Rightarrow\) corresponds to inclusion, \(\text{NP}_3 \Rightarrow \text{NP}_2\) means that \(v(\text{NP}_3) \subseteq v(\text{NP}_2)\). Now if we translate this back into feature structure language, by taking atomic categories to be sorts, for instance, then this means that \(\text{NP}_2 \sqsubseteq \text{NP}_3\).

In the type-logical case, underspecification is said to be asymmetric in the sense that the argument type must be more specific than the domain of the functor. In the unification-based grammars underspecification is said to be symmetric (see Bayer and Johnson (1995) and Dörre and Manandhar (1997)).

The asymmetry in the type-logical case arises from the polarity of the (sub)types. The example shows that antecedent types in the axioms must be more specific than the succedent types. This makes the use of underspecification dependent on the polarity of the subtypes (domain or range position) in functor types. In the following chapter in which extensions to the categorial machinery are discussed that were inspired by the constraint-based HPSG approach, the difference between symmetrical versus asymmetrical underspecification (or between the subsumption-based approach to binding versus the unification-based approach as Dörre and Manandhar (1997) call it) will show up again in a number of proposals. We should already point out that this difference is not merely a technical issue. The asymmetry and polarity sensitivity properties have been used to argue in favour of type-logical approaches to agreement (Bayer and Johnson (1995)). We will discuss these arguments extensively in Part IV.

### 4.2 Comparing the Frameworks

In general terms, our aim is to complement the fine-grained theory of grammatical composition of type-logical grammars with a fine-grained hierar-
chical mechanism for classification. The latter should allow the descriptive grammarian to express linguistic generalisations by abstraction and factorisation. In many approaches in the literature, it has been taken for granted that to achieve this it is necessary to incorporate feature structures somehow in the categorial framework. However, as we will see in the third part, it is also possible to reach the effect by using devices that are already present within the type-logical framework.

In this section, we want to get a more precise idea of what empirical aspects of linguistic description could motivate refinements to the category system. More particularly, we want to know which attributes or linguistic properties are not yet accounted for in the type-logical grammars as presented in Chapter 2 and for which an analysis using feature-structure like extensions would constitute an improvement.

In order to get a rough idea of the kind of phenomena for which we may want to consider the introduction of instruments for refined classification, we look at the major features in an HPSG signature to see what they are used for and compare this with the mechanisms in categorial grammars. We choose HPSG as it is the prime example of a feature-based theory. Because we only want a rough idea, we only discuss the main aspects of the signature of a basic version of HPSG.

In the previous chapters, we mentioned composition and selection as two important dual notions that are to be defined by a grammar. We have presented these from both a type-logical and a constraint-based perspective. We start the comparison of both frameworks with these aspects.

**DAUGHTERS** The feature structures of sort sign that model expressions in HPSG, also encode information about the constituent structure of complex expressions. The attributes that represent the information about daughters in a tree: DTRS (daughters), HDTR (head-daughter), SDTR (subject-daughter), and ODTR (object-daughter). These are the attributes that we used in our slimmed down version. In the HPSG literature, the attributes CDTRS (complement), ADTR (adjunct), etc. are used. Information about the mother node is encoded in the SYNSEM value of the sign, which in our examples was reduced to CAT.

In the type-logical descriptions, this information must be recovered from the derivations. In the Gentzen style natural deduction presentation, a sequent consists of a structured term on the left-hand side and a category on the right-hand side of the turnstile. The information in the sequent can be represented as a tree and the corresponding HSPG style matrix as follows.
Note that nodes are decorated by the combination mode and the category of the mother. The information about the internal node of the tree decorated by \( j \) is not present in the sequent. These internal category nodes become visible when the proof unfolds. This is an effect of the deductive (derivational) perspective on grammatical description. So, to look for the information corresponding to phrasal signs we should not merely look at the sequent but at the derivation as well. However, neither the sequent nor the derivation is the exact mirror-object of a phrase structure tree. The sequent as such contains less information, whereas the derivation is richer. The former contains information about the structure of the components and the type of the whole, but the information about the type of the parts has to be read off from the derivation. From a complete derivation it is possible to read off a complete phrase structure tree, but it is generally richer in that it also contains information about the way the structure is derived. In this sense, the derivation corresponds to a derivation tree rather than a phrase structure tree.

The schematic HPSG matrix that corresponds to this tree is the following.
When we represent the information contained in the structure above in tree-format, we can map it on the categorial tree.

What is peculiar about this tree is that the \([1]\) decoration corresponds to several types: \(s\), \(\text{NP}\_s\) and \((\text{NP}\_s)/\_j\text{NP}\). This discrepancy between the categorial tree and the HPSG case is due to the fact that we have ignored the subcat decorations on the nodes for the HPSG tree. The type-logical categories contain information that relates to both the \text{HEAD} feature and the \text{SUBCAT} feature, as we will see next.

\text{SUBCAT} In categorial trees, the functor types that decorate some of the nodes encode the selectional properties of the lexical and phrasal expressions. In HPSG, the dual of composition, selection, is expressed by a number of features like \text{SUBCAT}, \text{MODIFIER}, \text{SPEC}, or more particularly \text{SUBJECT} and \text{OBJECT} in our prototype grammar.

In categorial grammars these will all be accounted for by means of the selection connectives /\_i, \_i, where the modes on the connectives can be made sensitive to the type of combination /\_h-o-ph, \_h-s-ph. Using this correspondence we can refine the simple phrasal representations as follows.
The correspondence between the selection features and the binary connectives can be carried further to the way they are used in the derivation of phrasal structures. As Pollard and Sag (1987) point out: “Readers familiar with categorial grammar should note that the Subcategorization Principle is analogous to the cancellation of categories, with heads and complements corresponding to functor categories and argument categories respectively.” (p. 72)

\[
\Delta_1 \vdash B \quad \Delta_2 \vdash B_{\text{h-s-ph}}^{h-s-ph} A \\
\Delta_1 \circ_{\text{h-s-ph}} \Delta_2 \vdash A
\]

\[
\text{hd-subj-ph} \quad \Rightarrow \left[ \begin{array}{c}
\text{SUBJ} \\
\text{HD-DTR} \\
\text{NON-HD-DTRS}
\end{array} \right] \left[ \begin{array}{c}
\{\text{SUBJ} \} \\
\{\text{SYNSEM} \} \\
\end{array} \right]
\]

We have taken over the constraint on the head-subject-phrase almost literally from Sag (1997). We only ignored the empty SPR requirement on
the head daughter. The resemblance with the constraints we put forward earlier as principles in our HPSG fragments should be obvious. An important difference between the type-logical connective and the selection features in HPSG is that we do not have anything that corresponds to a rule of introduction (hypothetical reasoning) for the latter. Several analyses in the HPSG literature allow a more flexible manipulation of the selection features that is reminiscent of certain structural rules of categorial grammar. For instance, argument composition, which involves the merging of the selection requirements of a verb with the selection requirements of its arguments, is sometimes claimed to be a lexicalised version of function composition in categorial grammar (see Hinrichs et al. (1998) and references therein).

We already pointed out a difference between the unification-based perspective on selectional requirements versus the subsumption check in the type-logical approach. There are also differences that relate to choices in the specific linguistic analysis rather than the formal framework. For instance, in the case of determiner-noun combinations, the head of the noun phrases is the noun in most HPSG analyses with mutual selection between specifier and head. It is commonly assumed in categorial grammars that the determiner is the head (and functor) in such combinations. These are not differences related to the framework but to the specific type of analysis. It should also be noted that in the multi-modal categorial grammars, headedness need not just be defined in terms of functor or argument status (Moortgat and Morrill (1992)).

Although the correspondence between the selection features and the selection connectives is not complete, we can still conclude from this parallelism between the descriptive linguistic function of the various attributes just mentioned and between the corresponding logical (and structural) operators that we do not have to extend the categorial formalism with devices to account for features like DTRS, SUBJECT, OBJECT, SUBCAT. These aspects of linguistic analysis are already accounted for in the basic type-logical set-up which we started out from. We can summarise the correspondence as follows.

\[
\begin{align*}
\text{DTRS} & \leadsto \bullet_i, \\
\text{SUBCAT} & \leadsto E_i, E_i
\end{align*}
\]

**NONLOCAL.** The non-local feature in HPSG is used to treat long-distance dependencies. It would take us too far to consider all the cases here and compare their treatment in HPSG with their treatment in current categorial grammars. Here we make do with pointing out the parallels between the use of the non-local features and hypothetical reasoning in categorial grammars in relative clause constructions.

As Pollard and Sag (1994) (following Gazdar (1981)) explain, a typical analysis of an unbounded dependency consists of a bottom, a middle and a top. At the bottom, a dependency is introduced, in the middle section
this is passed up the constituent structure and at the top it is discharged. Schematically, this structure can be represented by the following tree.

```
        [ ]
       /\  
      /   
     (a) [ FILLER1 ] (c) [ SLASH1 ]
        /      
       /  \    /  \ 
      /    /  \  
     ...  [ SLASH1 ]
        /      
       /  \    /  \ 
      /    /  \  
     ...  (b) [ SLASH1 ]
```

The non-local feature SLASH functions as a kind of register in which the information about some dependency is percolated up a tree. The precise way in which this is introduced differs in various accounts of HPSG, so we abstract from this. Some principle of slash inheritance takes care that the information about the SLASH values on daughters is inherited by the mother. Finally, the SLASH dependency is said to be discharged. In the example this happens when the slashed constituent combines with a filler daughter that satisfies the selectional constraints expressed by the slash values. So SLASH is another kind of selection feature (like SUBCAT) that takes care of unbounded dependencies. This becomes obvious when one looks at the constraint on head-filler phrases as defined in Sag (1997), which parallels the constraints on other types of phrases in which selection features play a role, like the head-subject phrase we presented above.

```
hd-fill-ph  \rightarrow  [ SLASH \{1\} \cup \{2\} ]
            [ HD-DTR  ]
            [ NON-HD-DTR ]
```

One could say that in a categorial analysis the slash introduction corresponds to the introduction of a hypothetical assumption. What is stipulated as the behaviour of the SLASH feature in the constraint above, is a logical consequence in the type-logical analysis with hypothetical reasoning triggered by the higher-order type assignment to the relativiser. Discharging the assumption corresponds to the last step before the "top". In the HPSG analysis the selectional requirements found at the bottom (b) of unbounded dependencies are transferred to the position (c). In the Prawitz natural deduction notation, which looks as follows, we have marked the same positions in the introduction rule.
The procedure is in some ways similar, but certainly not identical. Particularly, if one looks at detailed analyses of particular phenomena then the differences become apparent. We present a simplified lexicon for the words in \textit{that he baked} in a HPSG style analysis (based on Sag (1997)) and contrast this with a categorial analysis.

If we compare these lexical assignments in light of the correspondences between selection features and selection connectives above, there are some interesting differences to note. Whereas in the HPSG analysis the typical properties of the construction are accounted for by the special instance of the verbal category, it is the relative pronoun in the categorial analysis that is responsible for the construction specific aspects.

It is important to point out that, while in the HPSG we need extra features, principles and lexical entries to analyse non-local dependencies, the type-logical approach treats local and non-local dependencies using the same logical operators but exploits the duality expressed by the logic of residuation between selection and composition.

Again, as with the local selection and composition features, the main purpose of this section is not to define a complete mapping between the two theories but to point out that the same descriptive task is performed by different mechanisms: instead of \textsc{nonlocal} features the categorial machinery makes use of its logical basis and the potential this offers for hypothetical reasoning.

\begin{align*}
     \text{CAT} & \quad \text{HEAD} \quad \text{NP} \\
\end{align*}

\textsc{content, qstore, phonology} In the presentation of the categorial grammar above we have limited the discussion to issues of syntactic construction and we left out aspects of semantic and prosodic interpretation. In
the kind of categorial grammar that we take as the basic system of reference, these aspects are defined by semantic and prosodic algebras where the objects are typed by the categories. Proof-theoretically, the prosodic and semantic objects are represented by terms decorating the formulas. This idea is often implemented in the form of a labelled deductive system (Gabbay (1996)). The lexicon associates words not just with a category but also with phonological and semantic representations. The representations for complex expressions are defined by operations on the labels in parallel with the proof-theoretical derivation steps. In the style of deduction that we are using, a variant of the Gentzen style natural deduction format, the phonological label corresponds to the expression term on the left-hand side of the turnstile. As regards the treatment of word order, we should mention the type-logical analysis of word-order domains given in Versmissen (1996) and its relation to the linearisation versions of HPSG that make use of ideas based on Reape (1993).

In the past few years, there has been a growing interest within computational semantics research to deal with semantic ambiguity by means of underspecified representations of meaning (van Deemter and Peters (1996)). As we have seen above, feature structures are an adaptable format and have the benefit of being designed for underspecification. Within the HPSG community, the development of Minimal Recursion Semantics (Copestake et al. (1997)) can be taken as an example of this type of development. In many cases the use of underspecification is motivated by processing issues. In the context of this section, the question raised by these developments is whether this necessitates the import of a feature structure-based minimal recursion-like semantics in the type-logical framework?

To answer this question, a number of things have to be taken into account. First of all, the tight connection between the syntactic and the semantic algebras in categorial grammars is considered an important advantage. However, staying within these constraints, the options concerning alternative representation languages are open. As Moortgat (1997) notes: "The proof terms associated with categorial derivations relate structural composition in a systematic way to the composition of meaning. The derivational semantics is fully neutral with respect to the particular 'theory of natural language semantics' one wants to plug in: an attractive design property of the type-logical architecture when it comes to portability. An illustration can be found in Muskens (1996), who proposes a type-logical emulation of Discourse Representation Theory driven by a categorial proof engine." (p. 123).

Of course, underspecified semantic languages need not be encoded as feature structures. In fact, the feature structure representations can often be considered as a notational variant of some logic defined in other terms. In other words, if it is considered worthwhile to use underspecified languages for the semantic analysis in type-logical grammars, there is nothing in the general set-up that seems to prohibit this, but neither is one forced to adopt
a feature structure type notation.

Also relevant in this respect is the work by Muskens and Krahmer (to appear) in which an approach to underspecified semantics is integrated with tree-adjoining grammar which can be adapted for the proof net version of type-logical grammar. Using the sign notation we introduced in Chapter 2, we can summarise the correspondence as follows.

\[
\begin{align*}
\text{CONT} & \leadsto (E) \triangleleft C \triangleright M \\
\text{PHON} & \leadsto E \triangleleft C \triangleright (M)
\end{align*}
\]

Other Properties  Although this exercise of comparing the use of HPSG features with the way these phenomena are treated in categorial grammars can only be suggestive, it does give us an idea about where feature structure-like constructs may pay off in extensions to categorial grammars. Consider the feature structure below, which is a summary of the types of features that appear in HPSG, with an indication of how the property that corresponds to each feature is handled in the categorial tradition.

\[
\begin{array}{c}
sign \\
\text{PHON} \\
\text{SYNSEM} \\
\text{DTRS}
\end{array}
\quad
\begin{array}{c}
\leadsto E \triangleleft \\
\text{LOC} \\
\text{CAT} \\
\text{HEAD} \\
\text{SUBCAT} \\
\text{CONT} \\
\text{CONTXT} \\
\text{NONLOC} \\
\leadsto I, I^
\end{array}
\quad
\begin{array}{c}
\text{PRD} \\
\text{MOD} \\
\text{SPEC} \\
\text{CASE} \\
\text{VFORM} \\
\text{AUX} \\
\text{INV} \\
\text{PFORM}
\end{array}
\quad
\begin{array}{c}
\leadsto \overline{m}, \underline{m} \\
\leadsto /s, \underline{s} \\
\end{array}
\]

If we strip off all the features that are already covered in categorial grammar by the basic apparatus that we are assuming, then we are left with the following feature structure.

\[
\begin{array}{c}
\text{SYNSEM} \\
\text{LOC} \\
\text{CAT} \\
\text{HEAD}
\end{array}
\quad
\begin{array}{c}
\text{PRD} \\
\text{CASE} \\
\text{VFORM} \\
\text{AUX} \\
\text{INV} \\
\text{PFORM}
\end{array}
\]
We can characterise the information that is left as the syntactic category (or part of speech) together with morphosyntactic properties. As we will see in the next chapter, the type of information that is often represented in the extensions found in the literature concerns precisely the morphosyntactic properties of expressions.

The syntactic categories are represented by basic types in a categorial grammar. A simple solution, proposed before, is to augment the inventory of basic categories to cover the morphosyntactic properties of expressions as well. However, this simple extension has the same drawbacks as extending the inventory of atomic categories in a simple phrase structure grammar (in fact the effect is even worse as we will see in the next chapter). If we want to allow cross-classification, underspecification, etc. we must find a more complex solution.

From a more technical point of view, it is obvious that we do not need all the intricacies of the feature-structures as we have considered them before, when we just consider this morphosyntactic information. The attributes that we are left with are not recursive. The attributes SYNSEM, LOCAL, CATEGORY, HEAD do not carry any function and can be left out. Furthermore, if the sets of possible (atomic) values for each of the remaining attributes (PRED, CASE,...) are mutually disjoint, then we do not need attributes either to mark the dimension along which elements are classified. This can be recovered from the value. In many proposals that extend the categorial system with mechanisms that enable cross-classification, such simplifications have been carried out. We will look at some in the next part. In Part III, we will propose to treat these features using the operators $\diamond$ and $\Box$.

$$\text{PRED, CASE, VFORM, AUX, FIN, PFORM} \sim \diamond, \Box$$

Summary

In this chapter we have contrasted the kinds of analysis that are typical in a type-logical framework with those of the constraint-based framework. We discussed some aspects of a combination of the logical, deductive characterisation of grammatical composition with the general logic of information used to cross-classify linguistic structures.

When investigating such combinations it is useful to look at those aspects of linguistic analysis that are covered by both frameworks, possibly by means of different mechanisms. We sketched such a comparison by looking at the signature of common HPSG grammars and indicating the corresponding treatment of these aspects in typical categorial grammars. We pointed out differences on both a formal and a descriptive level. Part of the reason for this exercise was to find those aspects of linguistic description which can motivate refinements to a categorial grammar that are comparable to that in constraint-based theories. We ended up with a only
a small number of properties like part-of-speech and morphosyntactic attributes, which are encoded by basic categories in categorial frameworks. This raises the question whether we actually need feature structures for these refinements.
Summary of the first part

In this first part we have provided an introduction to the topics and the kinds of questions that will concern us in the following chapters.

We started with an analysis of the elementary tasks that have to be carried out by a formal grammar to characterise a language. We mentioned the definition of membership of expressions in a language, their classification into equivalence classes (categories), and a characterisation of their compositional structure and of their semantic interpretation.

Next, we pointed out that there are several ways to implement these tasks by different frameworks. We outlined two approaches, the common phrase structure grammar and the categorial approach. Also within these two frameworks there is room for variation, as we have demonstrated. The formal details that have been given serve as an introduction to the systems and variants that will be discussed in the following chapters.

In the multi-modal, type-logical, categorial grammars, the characterisation of the compositional structure proceeds by deduction in a substructural logic that fits the resource-sensitive nature of the linguistic material. The basic logic is that of residuation. This is generalised to connectives of various arities, where binary and unary connectives will figure most prominently in the sequel. By introducing packages of structural rules for different connectives, a landscape of structural options unfolds that enables us to characterise different modes of composition and to control their interaction.

We continued the discussion by a presentation of feature structure augmentations to the phrase structure grammars. We saw how feature structures can be used to represent information about categories and other linguistic constructs, focusing on the benefits with respect to the classification of expressions and the formulation of possible grammars. We isolated the following benefits (which are interconnected):

- Cross-classification: expressions can be classified along multiple dimensions.

- Hierarchical classification: the information about expressions is organised hierarchically. This allows underspecified categories to appear in lexical assignments and rules, abstracting away from certain properties, which leads to simplification.

- Variables or similar devices are used to express identity of information in different parts of the structure. This allows further reduction of the number of rules in cases of co-variation.

These aspects of feature structures allow the grammar writer to state generalisations by abstraction and factorisation over groups of features and of expressions.
• Decomposition (1): the information contained in phrase structure rules can be decomposed into different principles each specifying a part of the distribution of some feature or group of features in phrase structure.

• Decomposition (2): the information contained in the lexicon can be decomposed into different clusters or macros, each expressing information common to a whole class of words. Organised into a hierarchy of types, the specific information for each word is provided by multiple inheritance. The same procedure applies to phrasal information.

We raised the question how categorial grammars could benefit from extensions with feature structures. The following chapters will reflect further on this question, but here we started thinking about the kind of effects that we want to achieve.

In linguistic theories like HPSG, feature structures and their descriptions form the basis of the formalisation of grammars. We take the position that we only want to extend the categorial machinery as it is currently established in the categorial research tradition for those aspects that are not yet well covered by other techniques. Our comparison of the use of feature structures in HPSG grammars with the practice of categorial grammars showed that for most aspects of linguistic description, categorial grammars use other techniques, not implemented by feature structures. We isolated a small collection of properties (part-of-speech, morphosyntax) for which standard categorial grammars have no refined techniques and for which a feature structure-like decomposition might be beneficial. The comparison between the categorial type of analysis and the HPSG type of analysis shows that the information that is usually specified by the basic categories is most likely to benefit from further decomposition.

This raises the question whether we need the feature structure machinery to handle these aspects of linguistic description or whether we can use other devices to take care of these aspects. In Part III we will present our own proposal in this matter.
Part II

Options and Obstacles
Introduction to Part II

In the previous chapters we have contrasted the type-logical approach to grammatical description with the constraint-based approach. The former defines a dedicated logic for composition and selection and thereby focuses primarily on classifying expressions in terms of their combination potential. The latter, on the other hand, uses feature structures to describe and classify expressions in several dimensions and provides techniques to organise the linguistic information into principles. This enables the descriptive grammarian to capture linguistic generalisations. Over the past few decades the two approaches have influenced each other mutually. In this part, we will look at several extensions to the categorial framework that have more refined classification devices, often inspired by the constraint-based tradition. Besides a brief presentation of the various systems (in Chapter 5), we also want to introduce the issues involved in such combinations and the parameters along which they vary (Chapter 6).

We saw in Chapters 2 and 3 how the familiar categorial types $A/B, A\bullet B$, and $B\backslash A$ are used to categorise expressions with respect to their combination properties. Expressions of type $A/B$ and $B\backslash A$ combine with expressions of type $B$ to form expressions of type $A$. Expressions of type $A \bullet B$ are composed out of two expressions of type $A$ and type $B$. In their common linguistic usage such categories define the syntactic 'constituent structure' of a language, as an alternative to the definition of such structures by phrase structure grammars (Moortgat (1988)).

It is common to assume a restricted set of basic categories, so that the information expressed by the types is restricted to part of speech and subcategorisation information. Morphosyntactic distinctions like case, number, person, etc. that involve dependencies such as government and agreement are often ignored. They concern the morphological form of words rather than the pure syntactic construction. The treatment of these morphosyntactic aspects has been a recurrent theme in the literature on categorial grammar in the past two decades, though never a central concern. One of the reasons why extensions to the calculus for morphosyntax have not received more attention is probably because of the general consensus that the basic categorial machinery can, in principle, handle the phenomenon as well by using more basic types in the grammar and imposing a further subtyping structure upon them (Lambek (1961), Lambek (1997), Dörre and Manandhar (1997)). The motivation to look for extensions parallels the need in phrase structure grammars, where atomic nonterminal categories are replaced by feature structures to allow for cross-classification, partial information, etc. (Borsley (1991, Chapter 4), Bouma (1993, p. 95-96)). To illustrate this, we consider some of the morphosyntax involved in a small set of Subject - Verb Phrase combinations in English. This example is a categorial variant of the motivation behind the introduction of feature structures and their information ordering presented in Chapter 3.
In a categorial grammar with atomic basic categories, the difference between present tense forms like *walk* and *walks* requires at least three basic types for noun phrases to account correctly for their distribution.

NP₁ third person singular nominative noun phrases
  *he*, *she*, *the boy* ...

NP₂ other nominative noun phrases
  *I*, *they*, *the boys*

NP₃ noun phrases that are not nominative
  *him*, *her*, *the boy*, *me*, *them*, *the boys*

With these distinctions we can type the verbs as NP₁\$ for *walks* and NP₂\$ for *walk*. It is already apparent that some expressions must be assigned at least two types; *the boy* is both NP₁ and NP₃ and *the boys* is both NP₂ and NP₃. If we look at the traditional type for *the*, NP/N, in this context, we see that it has to be assigned the following types: NP₁/N₁, NP₂/N₂, NP₃/N₁ and NP₃/N₂, where N₁ and N₂ index singular and plural nouns respectively. A first way to reduce the number of assignments consists of three steps.

(i) We introduce two new basic types NP₄ (*the boy*) and NP₅ (*the boys*), representing singular and plural noun phrases that can be both nominative and accusative (so they contain the full noun phrases and not the case marked pronouns). This also leads to a simplification for the types for *the*: NP₄/N₁ and NP₅/N₂.

(ii) We introduce a partial ordering relation on the basic types: NP₄ ≤ NP₁, NP₅ ≤ NP₂, NP₃ ≤ NP₄, NP₃ ≤ NP₅.

(iii) We change the rules of combination so that a functor A/B combines with an argument C if C ≤ B.

In grammars using feature structures a similar kind of reduction of assignments takes place, as we have seen in the first part. There the ordering of information was defined by the subsumption ordering (⊆) and by changing the rule of application in terms of compatibility (or merging/unification) of the information in C and B.

\[
\begin{align*}
\text{NP₁} : & \begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{AGR} & \text{3-sg} \\
\text{CASE} & \text{nom} \\
\end{bmatrix} \\
& \{\text{the boy, he}\}
\end{align*}
\]

\[
\begin{align*}
\text{NP₂} : & \begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{AGR} & \text{non-3-sg} \\
\text{CASE} & \text{non-nom} \\
\end{bmatrix} \\
& \{\text{the boys, they}\}
\end{align*}
\]

\[
\begin{align*}
\text{NP₃} : & \begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{CASE} & \text{non-nom} \\
\end{bmatrix} \\
& \{\text{the boy, the boys, him, them}\}
\end{align*}
\]
INTRODUCTION

The sort \textit{both} is special. Suppose \textit{case} is the supertype of all possible \textit{case} values. Then we now have the following subsumption ordering: \textit{case} \(\subseteq\) \textit{nom} \(\subseteq\) \textit{both} and \textit{case} \(\subseteq\) \textit{non-nom} \(\subseteq\) \textit{both}. This means that the feature structure corresponding to \(NP_4\) does not subsume the feature structure corresponding to \(NP_1\) but rather the other way round. This can easily be seen from the examples: \{\textit{the boy}\} \(\subseteq\) \{\textit{the boy, he}\} and not vice versa.

Feature structure grammars provide a second way to reduce the number of assignments. If we replace the basic types by their feature structure equivalents in the types we assigned to \textit{the}, we get the following assignments.

\[
\begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{AGR} & 3\text{-sg} \\
\text{CASE} & \text{both} \\
\end{bmatrix}
\quad \text{and} \quad
\begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{AGR} & \text{non-3-sg} \\
\text{CASE} & \text{both} \\
\end{bmatrix}
\]

The determiner combines with both types of \textit{N} and produces all types of \(NP\), but there is an important dependency between the agreement values of the domain and the values of the range of the determiner: the determiner can pass on the values from the noun to the noun phrase and vice versa. This distribution fact is accounted for in unification-based grammars by forcing these values to be identical. In this case, the re-entrancy also reduces the number of assignments to just one. Parallel to a suggestion for phrase structure grammars which we made in Chapter 3, we can imagine categorial types like this. This notation could be used as an abbreviation for the two assignments above.

\[
\forall \alpha \left( \begin{bmatrix}
\text{CAT} & \text{NP} \\
\text{AGR} & \alpha \\
\end{bmatrix} / \begin{bmatrix}
\text{CAT} & \text{N} \\
\text{AGR} & \alpha \\
\end{bmatrix} \right)
\]

These examples illustrate two ways to simplify the grammar (or the lexical information in the case of categorial grammars).

(i) The information ordering on the basic types allows us to reduce the number of type assignments through underspecification.

(ii) Re-entrancies allow us to state distribution principles and also reduce further the number of assignments needed. They thereby allow a more concise presentation of the grammar.

(iii) The decomposition of information also allows generalisations concerning different features to be stated separately.
The second reduction depends on the fact that we have decomposed the information into two separate features (\textsc{cat} and \textsc{agr}) so that we can say something specifically about the \textsc{agr} feature (factorisation) and on the fact that we can express identity of values through re-entrancies.

The methodological preference for economy in the specification of a grammar plays an important role in the proposals discussed below. It involves reducing the number of type-assignments to individual 'ambiguous' lexical elements and generalisations concerning a whole class of items by techniques like type-constraints and inheritance.

We will present a selection of different extensions to categorial grammars that are designed to deal with morphosyntactic structure. This will enable us to point out the various options that are available technically and the ways that the formal machinery is put to use in actual linguistic description. It also allows us to put our proposals to use unary modal operators as bearers of morphosyntactic information, to be presented in Part III, in a wider perspective; comparing them to other categorial logics, illustrating different formal techniques and also the motivation behind the extensions.

The second part is structured as follows. In Chapter 5 we briefly present and illustrate the various proposals and in Chapter 6 we present more general issues. The major issues presented in Chapter 6 concern the differences in the way underspecification is treated in the constraint-based systems and in the type-logical systems.
In this chapter we provide concise outlines of some categorial grammar extensions that have refined classification structures to deal with cross-classification. We will also discuss how these extensions treat aspects like underspecification, feature distribution, factorisation of grammatical information, etc. We will focus here on the technical aspects of the different proposals. This serves as a background for the discussion in Chapter 6 about their merits in dealing with the issues just mentioned. There we will discuss a number of issues that arise in using such formalisms in grammatical description. The effects of the extensions on the possibilities and limits of choosing different options will be discussed there as well.

In this chapter the presentation involves the following aspects for each of the systems.

- **Cross-classification** How is the definition of *category* modified to take into account the decomposition of information?
- **Abstraction** How is the information ordering defined and how is it used?
- **Factorisation and Feature distribution** Do the systems incorporate some technique like re-entrancy to express identity?
- **Calculus or rules of combination** What categorial system (AB, Lambek or other variant) is used?

We discuss the following extensions. (i) A reconstructed version of an *extended categorial grammar* as introduced by Bach in a number of papers: Bach (1983a), Bach (1983b), Bach (1984). (ii) A version of a *categorial unification grammar* based on Bouma (1993). (iii) Proposals in which basic types are replaced by first-order terms as in Carpenter (1992a) and Morrill (1994). (iv) The *extended Lambek grammar* as it is presented in work by Bayer and Johnson (Bayer and Johnson (1995), Johnson and Bayer (1995), Bayer (1996)). (v) Other proposals related to that by Bayer and Johnson and also based on layering a feature logic on a categorial logic, like those by Dörre et al. (1996), Dörre and Manandhar (1997) and Francez (1997).

### 5.1 Bach's Extended Categorial Grammar

Among the first papers to investigate morphosyntactic issues in the context of categorial grammars were those by Bach (Bach (1983a), Bach (1983b),
Bach (1984)). In these papers, Bach extends the notation for categories to take into account the specification of morphosyntactic features, which are defined informally by Bach (1983b, p. 71) as those “that we must keep track of somehow in phrase-grammar and that make a difference in word-grammar”. The definition of the extended notation varies somewhat from paper to paper. Our reconstruction is based on the version in Bach (1983a).

**Categories** As far as the language of categories is concerned, the only change to the traditional system, in which basic categories are atomic symbols, resides in changing the latter into more complex structures, which we will call extended basic categories $T$.

$$
T ::= B \mathcal{F}\mathcal{S} \\
\mathcal{F} ::= T | \mathcal{F}\mathcal{F} | \mathcal{F}\backslash\mathcal{F}
$$

The extended basic categories are pairs consisting of a traditional basic category (atomic symbols $B$) and a restricted kind of feature structure ($\mathcal{F}\mathcal{S}$). The feature structures are restricted in two ways. First, the value of each feature can only be an atomic value. This means that there is no embedding of one structure into another. Secondly, there are no devices to indicate re-entrancies. In other words, the restricted feature structures are simply bundles of feature-value pairs (both simple atomic symbols). The usual functionality requirement holds that there cannot be two pairs in a bundle with the same feature but with a different value.

For particular grammars it is assumed (again as usual) that some kind of signature is defined that constrains the possible values for each feature, that restricts the co-occurrence of features in structures and that defines what kind of feature structure information can be combined with specific basic categories.

**Rules of combination** Bach uses the applicative AB-system. The definition of a language defined by the grammar is thus completely parallel to that given in Definition 9. The only difference concerns the definition of category. We use $XB$ to identify this system.

**Definition 23 (Language)** Given a lexicon $\text{Lex}_{XB}$, the language $\mathcal{L}_{XB}$ over this lexicon is specified as the smallest set such that: (i) $\text{Lex}_{XB} \subseteq \mathcal{L}_{XB}$ and (ii) $(e_1 \circ e_2, A) \in \mathcal{L}_{XB}$ if either (ii-a) $(e_1, A/B)$ and $(e_2, B)$ are in the language, or (ii-b) $(e_1, B)$ and $(e_2, B\backslash A)$ are in the language.

**Ordering** For the purpose of underspecification and the hierarchical classification of expressions, it is possible to make use of the subsumption ordering on feature structures. Bach uses this ordering to reduce the number of lexical assignments by abbreviation (see the example below). The application schema requires the domain of the functor to be *identical* to the argument.
Re-entrancy  Bach does not use re-entrancy in his feature structures. In Part IV we will discuss the technique that Bach uses to factor grammatical information into different principles.

Example  We specify a simple grammar for the sentence he o walks. In the signature part we specify the features, the sorts and the basic categories we will use. Note that we use a different typeface for each kind of atomic symbol. We will write down (restricted) feature structures in the familiar attribute value matrix notation as before and simply juxtapose the basic category and the feature structure in writing down extended basic categories.

Lexicon:  

\[
\begin{array}{l}
he : \text{NP} \\
\text{CASE nom} \\
\text{NUMBER sg} \\
\text{PERSON 3rd} \\
\hline
walks : \text{NP} \\
\text{CASE nom} \\
\text{NUMBER sg} \\
\text{PERSON 3rd} \\
\hline
\end{array}
\]

For this simple applicative system we will use Gentzen style natural deductions.

\[
\begin{array}{l}
he \vdash \text{NP} \\
\text{CASE nom} \\
\text{NUMBER sg} \\
\text{PERSON 3rd} \\
\hline
walks \vdash \text{NP} \\
\text{CASE nom} \\
\text{NUMBER sg} \\
\text{PERSON 3rd} \\
\hline
\end{array}
\]

\[
\begin{array}{l}
he \circ \text{walks} \vdash \text{S} \\
ii - b
\end{array}
\]

A proper name like John could be assigned an abbreviated category like:

\[
\text{NP} \\
\text{NUMBER sg} \\
\text{PERSON 3rd}
\]

If we assume that the only values for CASE are nom and acc then this abbreviates 2 full entries:

\[
\begin{array}{l}
\text{NP} \\
\text{CASE nom} \\
\text{NUMBER sg} \\
\text{PERSON 3rd} \\
\hline
\text{NP} \\
\text{CASE acc} \\
\text{NUMBER sg} \\
\text{PERSON 3rd}
\end{array}
\]

In Bach’s approach it is not possible for the underspecified (abbreviated) entry to combine with the verb walks, because in the applicative schema a functor A/B combines only with categories B: identity is required. We will have more to say about this in the last part where we will also discuss his suggestions to factorise the information in this type of grammar.
5.2 Categorial Unification Grammar

In Chapter 3 we defined stand-alone feature structure grammars that use feature structures to represent not only categories but complete linguistic expressions. Categorial versions of such an approach have been worked out in Bouma (1993), Uszkoreit (1986), Zeevat et al. (1987). We will now discuss the essentials of these grammatical frameworks. More particularly, we will reconstruct a version of categorial unification grammar, \textsc{cug}, based on Bouma (1993).

**Categories** In \textsc{cug}, categories are defined as typed feature structures. A feature structure of the sort category is either a basic or a complex category (of sort basic and complex respectively). The basic categories have two features, SYN and MOR. The complex categories have four features, ARG, DIR, VAL and MOR. The values for the features ARG and VAL are categories themselves, representing the domain and the range of a functor category, respectively. The feature DIR is used to specify the connective: / or \. For more details concerning other features, we refer to the examples.

The following feature structures visualise these definitions of categories. Notice that in such categories the slot for morphological information not only appears on basic categories but also on complex categories. This means that in principle a complex category can bear morphosyntactic information that is different from the specification on either of its constituent categories (VAL, ARG).

\[
\begin{array}{c}
\text{basic} \\
\text{SYN} \\
\text{MOR}
\end{array} \quad \begin{array}{c}
\text{complex} \\
\text{VAL} \\
\text{DIR} \\
\text{ARG} \\
\text{MOR}
\end{array}
\]

**Rules of Combination** The \textsc{cug} grammar is a constraint-based system like \textsc{hpsg}. A language is again defined as a set of feature structures. The set of well-formed, i.e. grammatical, feature structures is specified as those that satisfy the constraints of the grammar, which is stated as a formula in a logical language. In the case of categorial unification grammar, the grammar consists of a lexicon and constraints defining the rules of combination.

The shape of the feature structures representing complex expressions (phrase structure trees) has to be fixed in the signature. Phrase structure trees also come in two forms: branching structures (phrases) and leaves (words). Feature structures of sort word have the features PHON, of sort phonology, and CAT, of sort category. Feature structures representing local binary trees are of sort phrase and have the features CAT, FDTR, ADTR (besides PHON) with CAT encoding the category of the mother, FDTR the functor...
OPTIONS FOR CATEGORY REFINEMENT

daughter and \texttt{ADTR} the argument daughter (the latter two are both of sort \textit{tree} themselves). This means that feature structures representing words and phrases typically look as follows.

\[
\begin{array}{c|c}
\text{word} & \text{phonology} \\
\text{PHON} & \\
\text{CAT} & \text{category} \\
\end{array}
\]

\[
\begin{array}{c|c}
\text{phrase} & \text{phonology} \\
\text{PHON} & \\
\text{CAT} & \text{category} \\
\text{FDTR} & \text{tree} \\
\text{ADTR} & \text{tree} \\
\end{array}
\]

The lexical expressions are modelled by feature structures of sort \textit{word}. The lexicon is a description of this set. A language consists of the feature structures in the lexicon and phrases that respect the constraints defined by the application schema. We can formulate the restrictions as type (sort) constraints, but for the sake of readability we first present the constraints as the elementary feature structures (depicted by attribute-value matrices) that are licensed by these constraints. We present the type-constraints, or at least similar ones, below. Note that the distinction between left and right looking functors $\backslash /$ is reflected in the order of combination of the phonological values.
The following presentation of the right application rule as a rule of inference shows how the application constraints correspond to the components familiar from the derivational representation of an AB-rule.

**Ordering** For the purpose of underspecification it is possible to make use of the subsumption ordering defined on the feature structures. These also allow the grammar writer to organise the information into inheritance hierarchies as in the HPSG grammars.

There is an important difference between the interpretation of the application schema in this CUG framework and the Bach system. Whereas in Bach's system *identity* is required between all the occurrences of A and B categories in the derivation, the categorial unification grammars only need the various structures that will match with the positions with labels like 1 to be unifiable.

**Examples** We provide a simple lexicon and show the phrase structure tree that corresponds to he o walks. We leave out sorts like word, bascat, etc. which can easily be reconstructed.
We illustrate the use of inheritance hierarchies to organise the information with two examples. The first one applies the technique to the phrasal domain by abstracting generalisations from the rule system. The second example deals with the structure of the lexical domain.

The two constraints defining left- and right-application respectively have many elements in common. The only differences between them are the connective involved in the functor and the order in which the phonological values are combined. To account for this overlap, or to state the generalisation over both instances, we define two subtypes of phrase, called left-headed-phrase and right-headed-phrase. The following constraints are associated with each of these types.
As for the lexical hierarchy, it is also possible to abstract general patterns from the specific lexical descriptions and formulate them as type-constraints (in Bouma (1993) and Shieber (1986) these are specified as macros). In specific lexical descriptions this information need then no longer be specified as it is inherited from the supertype. For instance, we can define the sort modifier as a subtype of word and associate it with the following constraint.

\[
\text{modifier} \Rightarrow \begin{bmatrix}
\text{CAT} \\
\text{VAL} \\
\text{ARG}
\end{bmatrix}
\]

The descriptions of each individual modifier (for instance of the German adjective guten) can now be simplified.

\[
\begin{bmatrix}
\text{word} \\
\text{guten} \\
\text{CAT} \\
\text{VAL} \\
\text{ARG}
\end{bmatrix}
\]

reduces to

\[
\begin{bmatrix}
\text{modifier} \\
\text{guten} \\
\text{CAT} \\
\text{VAL} \\
\end{bmatrix}
\]

**Related proposals** The categorial unification grammar we have presented above, is just one example from a family of proposals to encode a categorial grammar in a constraint-based formalism: Shieber (1986), Gazdar et al. (1988), Moens et al. (1990), Zeevat et al. (1987), Uszkoreit (1986),

### 5.3 Extended Lambek Grammar

In a series of papers on coordination Bayer and Johnson use a categorial grammar that differs from Lambek systems only in that the basic categories are replaced by formulas from a simple fragment of propositional logic (Bayer and Johnson (1995), Bayer (1996), Johnson and Bayer (1995)). We will sometimes call this version of the Lambek calculus the extended Lambek calculus and the categories extended Lambek categories. Most of the time however we will refer to it as the Bayer and Johnson grammar.

**Categories** Given a set $S$ of sorts, the set of terms $T$ and categories $C$ is defined by the following grammar.

\[
T ::= S \mid (T \land T) \mid (T \lor T) \\
C ::= T \mid (C/C) \mid (C\setminus C)
\]

It is important to note that the connectives $\land$ and $\lor$ only appear in the basic categories ($T$).

**Calculus** The rules of inference are given in Gentzen sequent style. They are the standard rules except for the axiom. $PL$ stands for propositional logic.

\[
\frac{T_1 \quad T_2}{T_1 \Rightarrow T_2} \quad ax
\]

The only rules of $PL$ that will be relevant here are the rules for $\land$ (and of course the axiom rule for $PL$). We will use the rules of the additive connections here.

\[
\frac{T}{T \Rightarrow T} \quad ax_{PL} \\
\frac{\Gamma[T_1] \quad C}{\Gamma[T_1 \land T_2] \quad \Gamma[T_2]} \quad L\land_1 \\
\frac{\Gamma[T_2] \quad C}{\Gamma[T_1 \land T_2] \quad \Gamma[T_1]} \quad L\land_2 \\
\frac{\Gamma \quad T_1 \quad \Gamma \quad T_2}{\Gamma \quad T_1 \land T_2} \quad R\land
\]
Booleans/Additives The reader should be aware that the addition of additive or boolean connectives to the Lambek calculus poses interesting technical challenges with respect to recognising power, completeness and resource management. For discussion on these matters we refer the reader to Kanazawa (1992), Restall (1994) and Kurtonina (1995).

In the Bayer and Johnson framework, the introduction of extra connectives is restricted to the level of basic categories to avoid these technical problems. In this way, the additive connectives do not interact to a great extent with the type-logical connectives \(/, \bullet, \backslash\). For a discussion on the formal properties of this type of system see Dörre and Manandhar (1997).

Ordering Comparing the propositional language with a feature structure language, we see that the propositional rules function as a kind of subsumption check. We can adopt the feature structure vocabulary, \(T_1 \Rightarrow T_2\) if \(T_2\) subsumes \(T_1\), or equivalently, \(T_1\) is more informative than \(T_2\).

In this hybrid system, two logical languages are combined. A derivation in this system proceeds as usual in the Lambek calculus, up to the point in a branch where all the Lambek connectives have been removed and only terms are left. At that point, the propositional rules of inference take over. Whereas the normal axiom of the Lambek calculus reads \(A \Rightarrow A\) (where \(A\) is a basic category), we can now have two different terms, \(T_1 \Rightarrow T_2\). The relation of logical derivability that holds between them reflects inclusion of the sets of expressions that interpret these terms.

The ordering can be used to match domains of functors and their arguments that are not identical but where one can derive the other.

\[
\begin{align*}
fail & \\
A \Rightarrow A & B \Rightarrow B \land C & A \Rightarrow A & B \land C \Rightarrow B \\
A/(B \land C) \circ B & \Rightarrow A & A/B \circ (B \land C) & \Rightarrow A
\end{align*}
\]

Re-entrancy There are no device like re-entrancy or structure-sharing in this system. The consequences for grammatical description will be discussed in the next chapter.

Example We define an extended Lambek Grammar for the expression \((he \circ walks)\).

Lexicon: \\{(he, (NP \land nom \land sg \land 3rd)), (walks, (NP \land nom \land sg \land 3rd)\backslash s)\}\n
Derivation for \((he \; walks, s)\):

\[
\begin{array}{c}
NP \land nom \land sg \land 3rd \Rightarrow NP \land nom \land sg \land 3rd \\
NP \land nom \land sg \land 3rd \Rightarrow NP \land nom \land sg \land 3rd \\
(NP \land nom \land sg \land 3rd) \circ (NP \land nom \land sg \land 3rd) \Rightarrow S \\
\end{array}
\]

\[
\begin{array}{c}
\text{ax}_{\text{PL}}
\frac{S \Rightarrow S}{S \Rightarrow S} \\
\text{E}\backslash
\end{array}
\]
Related proposals  Boolean or additive connectives have been used to reduce lexical ambiguity not only by Bayer and Johnson but also by Lambek (1961), Morrill (1990), Hendriks (1995) and Carpenter (1999).

5.4 First-order Terms

Several authors have proposed versions of categorial grammars that have first order terms as basic categories. In Carpenter (1992a) this move is presented for an AB-style grammar. For a Lambek-style grammar a good reference is Morrill (1994). We will present both these proposals here.

5.4.1 AB-Version

Categories  Basic categories are constructed out of predicate symbols, individual variables and individual constants. Each predicate symbol is of some fixed arity $n \geq 0$. The arguments can be filled by variables or constants. Constants and variables are assumed to be sorted. The signature fixes for each predicate symbol which constants can fill each of the argument positions.

We take the following illustration from Carpenter (1992a). The signature specifies the sorts (person, number, case, verb-form), the variables ($P_i$, $N_i$, $C_i$, $V_i$, for $i$ a natural number), the constants (1, 2, 3, sing, plu, subj, obj, bse, fin, perf, pred, infl) and the predicates (NP, N, S).

- Variables $P_i$ range over individuals of sort person, $N_i$ over individuals of sort number, $C_i$ ... case and $V_i$ ... verb-form.
- 1, 2, 3 are of sort person; sing, plu, of sort number; subj, obj of sort case; bse, fin, perf, pred, infl of sort verb-form.
- The predicate NP has arity 3, the first argument position is of sort person, the second of sort number and the third of sort case. N has arity 1 of sort number. S has arity 1 of sort verb-form.

The following are well-formed basic categories according to this signature: NP($P_1,N_1$obj), N(plu), S(V_3).

The set of first-order AB categories is the smallest set such that: (i) First-order AB basic categories are first-order AB categories. (ii) If A and B are first-order categories than so are A/B and A\B.

Given this definition of categories there are various ways to define a categorial grammar of the Adjukiewicz/Bar-Hillel style. Our definition here is a straightforward adaptation from Definition 23. We refer to the lexicon and the language as $FAB$: first order Adjukiewicz/Bar-Hillel. Obviously, the lexicon $Lex_{FAB}$ is defined as a set of pairs of words and categories (this time first-order AB categories) as usual.
**Definition 24 (Language)** Given a lexicon $\text{Lex}_{\text{FAB}}$, the language $\mathcal{L}_{\text{FAB}}$ over this lexicon is specified as the smallest set such that: (i) $\text{Lex}_{\text{FAB}} \subseteq \mathcal{L}_{\text{FAB}}$ and (ii) $(e_1 \circ e_2, A) \in \mathcal{L}_{\text{FAB}}$ if either (ii-a) $(e_1, A/B)$ and $(e_2, B)$ are in the language, or (ii-b) $(e_1, B)$ and $(e_2, B\setminus A)$ are in the language.

**Ordering and Re-entrancy** The usual rules of variable substitution and renaming apply. The variables are implicitly assumed to be universally quantified with scope over the whole category. With the use of these variables it is possible to define an information ordering on the categories. We provide some examples. A category $N(N_1)$ generalises over the categories $N(sg)$ and $N(pl)$. The category $N(N_1)/N(N_2)$ can be instantiated to the more specific categories $N(N_1)/N(sg)$, $N(N_1)/N(pl)$ and the categories $N(sg)/N(N_2)$, $N(pl)/N(N_2)$. The category $N(N_1)/N(sg)$ in its turn is more general than $N(sg)/N(sg)$ and $N(pl)/N(sg)$ (and so on for the other categories). Of course, the category $N(N_1)/N(N_1)$ can only be instantiated further to either $N(sg)/N(sg)$ or $N(pl)/N(pl)$.

Variable sharing can thus perform similar functions as re-entrancies in feature structures.

**Example** Assuming the signature as defined above, we can use the following lexicon to describe the sentence *he walks*.

$$
\{(\text{he}, \text{NP}(3,sg,subj)), (\text{walks}, \text{NP}(3,sg,subj)\setminus s(fin))\}
$$

The proof that shows that $(\text{he} \circ \text{walks}, s(fin))$ is in the language is a simple adaptation of the proof in the Bach system.

### 5.4.2 Lambek version

We will be short about the technical aspects of the first-order version, assuming familiarity with first-order logics. More important in the discussion are the examples of its use. These are provided in the next chapter.

**Categories** The first-order version of the Lambek calculus as defined in Morrill (1994) differs from the AB grammar in the choice of inference rules (Lambek instead of AB) and in the syntax of categories. Morrill introduces explicit quantification. If $C$ is a category and $v$ a variable then $\forall v C$ and $\exists v C$ are also categories.

**Ordering** An ordering on the categories and corresponding expressions is defined by the use of variables and quantifiers. In order to establish the ordering one should turn to the interpretation of the categories. The
semantics of categories with variables and quantifiers can be defined by introducing assignments $G$ that assign constants to the variables. The assignments are, of course, assumed to respect the sortal requirements imposed by the signature.

$$v^G(\forall vA) = \{ x | \forall c[x \in v^G[v:=c](A)] \}$$

$$v^G(\exists vA) = \{ x | \exists c[x \in v^G[v:=c](A)] \}$$

Consider a simple example, with a basic category built from the unary predicate $NP$, where the argument positions can be filled by $sg$ and $pl$. The set of expressions in $\forall x[NP(x)]$ is included in the set of expressions denoted by $NP(sg)$. If an expression is in $\forall x[NP(x)]$ it must be in $NP(sg)$ and $NP(pl)$.

**Calculus** Extending the language with these quantifiers also requires, from a proof theoretic point of view, adding left and right rules for these logical operators. If we strip off the semantic labels we can copy the following rules of inference from Morrill (1994, p. 173/182). We provide them here for reference only.

$$\frac{\Gamma[A[v \leftarrow t]] \Rightarrow C}{\Gamma[\forall vA] \Rightarrow C} \quad \forall L$$

$$\frac{\Gamma \Rightarrow A}{\Gamma \Rightarrow \forall vA} \quad \forall R^*$$

$$\frac{\Gamma[A] \Rightarrow B}{\Gamma[\exists vA] \Rightarrow B} \quad \exists L^*$$

$$\frac{\Gamma \Rightarrow A[v \leftarrow t]}{\Gamma \Rightarrow \exists vA} \quad \exists R$$

Here * refers to the side-condition that $v$ does not occur free in $\Gamma$ and $\leftarrow$ indicates substitution.

In the first-order AB grammar as defined above and taken from Carpenter (1992a, p. 236) implicit wide scope universal quantification is assumed.

By using $\forall$ as well as $\exists$ and by integrating the rules in the Lambek calculus, this type-logical framework offers a more fine-grained, polarity-sensitive instrument for linguistic description. We will provide some examples of this approach in the next chapter.

### 5.5 Fibered Approaches

The Bayer and Johnson approach which we just sketched can be characterised as an instance of a fibred (or layered) logic in the sense of Gabbay (1996). In the same vein, other combinations of a categorial grammar logic with a feature-based grammar logic have been worked out in Dörre *et al.* (1996), Dörre and Manandhar (1997) and Francez (1997). The logical system which we will present next is referred to as $L(FC)$ in Dörre *et al.* (1996).
Categories  An indexed syntactic type of $L(\mathcal{FC})$ has the form:

- $b(x)$ (indexed basic type, with $b$ a basic type as usual and $x$ a variable)
- $C/B$ ($C$ and $B$ types)
- $B\setminus C$ ($C$ and $B$ types)

The basic types are said to be indexed, because the variables are used to index the basic types: $b(x)$ and $b(y)$ are of different types.

Calculus  Besides this change in syntax, the system makes use of a slightly different kind of (natural deduction) sequent or declarative unit as Dörre et al. (1996) call it.

Definition 25 (Declarative Unit in $L(\mathcal{FC})$) A declarative unit has the form $\Delta \vdash C :: \Phi$, where $\Delta$ is a sequence of types, $C$ is a type and $\Phi$ is a feature constraint. The first part before :: is called the type skeleton and the constraint $\Phi$ is called the global constraint.

The feature constraint is a formula taken from the language of the feature constraint logic $\mathcal{FC}$ characterised in Dörre et al. (1996) as "function-free first-order logic with equality over the binary predicate symbols $\mathcal{F}$ (features), the unary predicate symbols $\mathcal{P}$ (sorts), and an infinite set of variables $\mathcal{V}$". This language is used to talk about feature structures. "[A]tomic formulas are of the forms $xfy$, $p(x)$, and $x = y$. We take the connectives $\neg$ and $\land$, as well as the quantifier $\exists$ to be primitive and all others to be defined in the usual way." (We will use the prefix notation for binary predicates and write $f(x,y)$).

The formulas of this feature description language are interpreted on feature structures, similar to the structures we defined in the first chapter. The interpretation clauses are fairly standard so we will not provide them here, as we are not interested so much in the formal details as in the application of the system to grammars.

The general idea of the system is to link a type-logical categorial part and a constraint-logic feature structure part together. The link between these parts is established syntactically, first by indexing the basic categories with variables of the constraint-logic and secondly by annotating a natural deduction sequent with a formula from the constraint-logic that is meant to constrain the variables. Semantically, the interpretation of the two logics is effected through fibering relations. The type skeleton is interpreted in a model for the categorial logic as usual (a string model in Dörre et al. (1996)) and the global constraint is interpreted in a feature structure model. The fibering relations define a mapping from the interpretation of basic types with some variable ($x$) in the string semantics to the interpretation of the variables in the feature structure model.
**Ordering** In this calculus one can make use of the ordering on the feature structure constraints to define underspecified structures. The matching proceeds partly by a check on the basic categories and partly by combining the constraints associated with the indices.

**Example** We define a layered Lambek Grammar which generates the expression \((he \circ walks)\).

- **Signature:** The set \(P\) contains the sorts (unary predicates) \(nom, sg, 3\); the set \(F\) (binary predicate symbols or features) contains \(NUM, \ PER\) and \(\text{CASE}\); basic category names are \(NP\) and \(S\), indexed by variables from the constraint language.
- **We assume that the lexicon consists of pairs of words and assignments. Assignments are pairs themselves consisting of a category and a formula from the constraint logic.**

**Lexicon:**

\[
\begin{align*}
(he, \ & NP(x_1)) \quad :: (NUM(x_1, x_2) \land sg(x_2)) \land \\
& \quad \land PER(x_1, x_3) \land 3(x_3) \land \\
& \quad \land CASE(x_1, x_4) \land nom(x_4)), \\
(walks, \ & NP(y_1) \land S(z)) \quad :: (NUM(y_1, y_2) \land sg(y_2)) \land \\
& \quad \land PER(y_1, y_3) \land 3(y_3) \land \\
& \quad \land CASE(y_1, y_4) \land nom(y_4))
\end{align*}
\]

- **Derivation for \((he \ walks, S(z))\):** We assume that lexical insertion proceeds by combining the category parts of the lexical assignment in the type skeleton and conjoining the constraint formulas into the global constraint, with possible renaming of variables or variable substitution.

\[
\begin{align*}
\Phi & \quad \quad \Phi \\
NP(x_1) \vdash NP(x_1) :: \Phi & \quad \quad NP(x_1) \land S(z) \vdash NP(x_1) \land S(z) :: \Phi \\
NP(x_1) \circ NP(x_1) \land S(z) \vdash S(z) :: \Phi
\end{align*}
\]

where \(\Phi = (NUM(x_1, x_2) \land sg(x_2)) \land \quad PER(x_1, x_3) \land 3(x_3) \land \quad CASE(x_1, x_4) \land nom(x_4))

It is interesting to look at the axiom rule and the rule of elimination as it is given in Dörre et al. (1996). These are defined as:

\[
\begin{align*}
A \vdash A :: \Phi \\
\Delta_1 \vdash B :: \Phi & \quad \Delta_2 \vdash B \setminus A :: \Phi \\
\Delta_1 \circ \Delta_2 \vdash A :: \Phi
\end{align*}
\]
This means that the constraint $\Phi$ is truly global. It is passed on at each inference step and will appear on all the axioms. As is pointed out in Leiß (1994) this system combines a validity part and a satisfaction part without genuine communication between the two. The rules of inference and their interpretation require us to check whether the theorem is true in all models, the (open) constraint assumption restricts the models to those in which the constraint holds.

**Other Layered Systems** Other systems besides the one discussed above have been proposed by Calcagno (1996), Heylen (1996a), Dörre and Manandhar (1997), and Francez (1997). In all of these proposals there is an attempt to resolve in one way or the other the conflicts that we pointed out in the previous chapter between the feature-structure systems (unification) and the Lambek systems (subsumption). The aim is to have a logical system with both elimination and introduction rules for the logical connectives in which as much of the unification-based approach to underspecification is preserved as possible. We will discuss several of the problems that arise in this connection in the next chapter.

**Summary**

This presentation of some extended categorial grammars was intended to provide an overview of the various options available. It does not claim exhaustiveness. It shows how frameworks can differ along a number of options with respect to the kind of category system that is used. These are:

- The technique or formal language or logic that is used for the extension of the category.
- The choice whether only basic categories are changed, or whether all categories are affected by the refinements.
- The possibility of stating identities (like re-entrancies) between different parts of a category (domain and range).

**Categories** Each of the proposals above redefines the notion of category to enable some degree of cross-classification. Feature structure-like extensions are found in the proposals by Bach and Bouma (CUG). The former replaces basic categories by pairs of a category and a feature structure for the morphosyntactic information. The CUG system, on the other hand, is an instance of a stand-alone system, using feature structures as representations for categories, phrase structure and other dimensions of linguistic description. In the hybrid approach of Dörre et al. (1996) feature structures are connected to basic categories by means of fibering relations.

The Bayer and Johnson system replaces basic categories by propositional formulas, but an alternative which replaces them by simple feature structures (or rather formulas from some feature description language) is of
course readily conceivable. In the first-order systems they are replaced by first-order terms.

A typical difference between the categorial unification grammar and all the Lambek-based systems is that the former allows morphosyntactic decorations on both basic and functor categories, but the latter only allows refinements on the basic categories.

What is shared by the CUG, first-order systems and the layered approach from Dörrre et al. (1996) is that they all provide a mechanism to require identity between values in different subtypes of a category. The mechanisms used are re-entrancies and variable sharing.

**Ordering** All systems discussed provide a technique to order the information provided by categories and thereby to underspecify information. In most systems this ordering plays an important role in matching the domain of a functor with the argument that it takes in a combination. In the Lambek systems combinations $A/B \circ C \vdash D$ are possible when $C \vdash B$. This means that $C$ and $B$ do not have to be identical, but derivability must hold. Extensions to this notion of derivability typically correspond to the information ordering provided by the extensions. The only Lambek system that we presented which is more permissive in this respect is the layered approach from Dörrre et al. (1996). The underspecified information coded in the constraints is not subject to the derivability requirement but to a satisfaction requirement or compatibility (unifiability) constraint.

The CUG approach, on the other hand, does not require derivability. It is defined as a constraint-based, stand-alone unification system. The grammar framework as proposed by Bach requires identical (extended) categories for matching. Underspecification in his system is restricted to abbreviations in the lexicon and does not figure in derivations.

We will discuss the consequences of the different types of matching extensively in the next chapter.

**Factorisation** The issue of factorisation of linguistic information into principles is discussed explicitly only by Bach and by Bouma. In this chapter, we have only discussed proposals by the latter. The proposals by Bach will be discussed in Part III. The extensions to the Lambek-based systems are concerned mostly with reducing the number of category assignments to individual lexical items and less with extracting generalisations from the grammar or organising the lexical information into a type hierarchy.

In general, one could say that these systems are restricted with respect to the rules of combination, or they are restricted with respect to the refinements (basic categories only), or both. In the next chapter we investigate in more detail the effects of these design choices on the way the systems can be used for linguistic analysis, focusing on the differences between the constraint-based AB system and the type-logical frameworks. This will lead to a list of requirements that an optimal combination should meet.
6

Underspecification in categorial grammars

In the previous chapter, we presented a number of proposals that extend categorial frameworks with techniques that make it possible to decompose linguistic information. Among these, the categorial unification grammars combine a categorial perspective on linguistic composition with a constraint-based perspective on classification. In this chapter we will contrast this constraint-based framework with the extended versions of the type-logical Lambek frameworks with respect to such issues as underspecification, abstraction, and generalisation.

Important differences between the systems are the presence or absence of devices like variable sharing or re-entrancy and the logical regime that governs underspecification. The constraint-based theories, only have wide scope universal quantification, whereas in the Lambek systems underspecification is polarity-sensitive.

In Section 6.1 we will present the typical use that is made of categorial unification grammars, showing how the benefits of a unification-based approach work out for categorial grammars. In Section 6.2, we turn to the Lambek-style grammars and show how the deductive perspective on linguistic description brings with it a different approach to these issues.

6.1 Unification-based Categorial Grammars

In Chapter 3, we showed how feature structures can be used in the grammatical description of natural languages. We discussed how they allow expressions to be classified with respect to several dimensions, how the notion of partial information and unification can be used to simplify grammars, accounting for certain generalisations, how grammatical information can be organised in modules, and how re-entrancies can be used to define feature distribution principles.

In the third chapter we discussed feature structure grammars in the context of phrase structure frameworks, and now we will see how the same issues turn up in a categorial context. In the case of categorial unification grammar, the partial information resides in the lexicon and in the application schema, these being the only components used in defining a language. In the following paragraphs we provide a small fragment that illustrates at which points the use of feature structures has effects and how underspecification of morphosyntactic information works in this type of grammar.
**Underspecified Lexical Entries** We look at the entries for the German adjective *guten* and the noun *Frau* (Bouma (1993, p. 111ff)). In German, adjectives and determiners agree in number, case and gender with the head noun whereas adjectives agree in declension class with the determiner. Nouns are marked for number and case. Many forms in the paradigm have an identical form and could therefore be left underspecified in the lexical assignments as the following example shows. For the sake of convenience, we use attribute-value matrices as the description language.

In this case, the adjective has no specifications for number and gender. As these attributes can have two and three different values respectively, this description abbreviates six elements of the paradigm. The entry for *Frau* does not specify a value for *case* because it is not marked for this attribute. This type of underspecification is of a similar type as the one introduced in Chapter 3 in the context of phrase structure grammars. There we used the following assignment for the definitive determiner, with underspecification for number.

\[
\langle \text{the}, [d \land 0] \rangle
\]

This is a notational variant for:

\[
\langle \text{the}, \begin{bmatrix} \text{cat} \\
\text{SYN} \\
\text{BAR} \end{bmatrix} \rangle
\]

The precise status of this type of underspecification is an issue which we will discuss in some detail in Part IV. The general question is whether we take the universe of expressions to be inhabited by a single expression *the*
with the properties as above, or whether we interpret the description as a notational abbreviation for two entries, corresponding to two expressions in the domain (one singular, the other plural), abstracting the common information from the different assignments. Note that the assumption here is that for the attributes that are not specified, any of the appropriate values for that attribute can be filled in.

**Functor-Argument Matching** The combination of the adjective and the noun is constrained by the type-constraint on phrases defining the unification-variant of function application. This means that for this combination the information as specified on the FDTR is re-entrant with the entry for *guten* and the information on the ADTR will be re-entrant with the entry *Frau*. The re-entrancy marked as [2] in the application schema between the ARG value of the functor and the category of the argument daughter requires this. In other words, the properties of *Frau* must be unifiable with the properties required by the functor. This means that with respect to underspecification we have a symmetric situation in which both the adjective and the noun can be underspecified for certain features.

### Co-variation and Feature Distribution

In the entry for *guten* it is specified that the domain and range part of the functor category are re-entrant. The distribution of features is defined by the patterns of re-entrancy in the lexical categories and in the application constraint. In the entry for *guten* it is specified that VAL and ARG are re-entrant. If the adjective is filled in as FDTR in the structure above this means that [1] and [2] become re-entrant. This means that the MOR value of the ARG part of the DTR becomes re-entrant with [3] as well, as will the MOR value of the ADTR.

It is important to point out the effect of the interaction between the re-entrancies in the modifier and the use of underspecification. Although the modifier can be underspecified for certain features, the re-entrancies force identity of domain and range. We will refer to this as co-variation. This means that when the adjective combines with a noun, the adjective
also passes on the values of the noun to the mother via the various reentrancies.

**Partiality and Grammatical Organisation** The entries for *guten* and *Frau* show one particular use of partial information structures. The entries as we presented them abstract away from those properties that are common to different elements in the paradigm.

We can decompose the entry for *guten* into partial descriptions like the following.

```
[\begin{array}{c}
\text{word} \\
\text{PHON}
\end{array} \begin{array}{c}
\text{guten} \\
\text{VAL}
\end{array} \begin{array}{c}
\text{SYN} \\
\text{N}
\end{array} \begin{array}{c}
\text{DIR} \\
\text{CASE}
\end{array} \begin{array}{c}
\text{gen} \\
\text{wk}
\end{array} \begin{array}{c}
\text{MOR} \\
\text{DECL}
\end{array} \begin{array}{c}
\text{wk} \\
0
\end{array}]
```

The latter description abstracts away a property that the adjective has in common with a large class of lexical items: the modifiers. This can be formulated as a type-constraint.

```
\text{modifier} \Rightarrow \begin{array}{c}
\text{CAT} \\
\text{ARG}
\end{array} \begin{array}{c}
\text{VAL} \\
1
\end{array} \begin{array}{c}
\text{1}
\end{array}
```

This simplifies the description of the lexicon. Information common to a whole class of entries is given once as a constraint on some type. Lexical elements inherit the information by being marked as being of this type. We already indicated in the previous chapter that the same strategy can be applied to the application schemata.

We have now discussed some aspects of the workings of a categorial unification grammar. In the following section we will look at the extended versions of type-logical grammars to see how these issues turn up in a setting in which the rule of application (elimination or modus ponens) is complemented by a rule of abstraction (introduction or hypothetical reasoning) and, more importantly, in which the constraint-based view on grammatical description is replaced by a deductive perspective.
6.2 Type-Logical Grammars

The deductive set-up of the type-logical grammars leads to some important differences with the constraint-based treatment of underspecification and partial information. In this section we will discuss these issues using the same simple example of the German adjective-noun combination as above and taking the Bayer and Johnson approach as the prototypical type-logical extension.

Because the introduction of many extended formalisms was motivated by a desire to capture typical aspects of the constraint-based analysis, some of the differences with respect to underspecification (asymmetry and co-variation) have sometimes been considered as problems (Dörre et al. (1996)) for the type-logical approach. This has motivated the introduction of hybrid systems in which the descriptive tasks of a grammar are divided into a type-logical derivation part and a unification-based part. We will briefly illustrate how these systems overcome these putative shortcomings. However, we will also argue that an important part of these 'problems' can be solved by using techniques within the type-logical framework and that the other part in a sense is not problematic at all, but rather an advantage of the type-logical treatment.

**Functor-Argument Matching**  In the proof-theoretical perspective on type-logical grammars, the basic judgments are of the form $\Gamma \vdash C$, where $\Gamma$ is a structured configuration of categories corresponding to a structured configuration of expressions and $C$ is some category. It expresses the basic categorisation relation: structure $\Gamma$ is of type $C$. Semantically, the unit is interpreted as saying that all the expressions in $\upsilon(\Gamma)$ are included in $\upsilon(C)$ (for all frames and valuations).

In the basic type-logical system as discussed in the first part, the axiom is of the form $A \vdash A$. In extended systems, like the Bayer-Johnson framework, the axioms for basic categories are replaced by another layer of derivability. Because the basic categories are propositional formulas, the axiom for the type-system is replaced by a notion of propositional derivability $A \vdash_{PL} B$. From a semantic perspective this corresponds again to inclusion.

The propositional formulas are not simply used to decompose the information. Their logic also makes it possible for an argument and a functor to combine because the argument propositionally derives the domain sub-category (and not because the domain of the functor is identical to the argument). In other words, the argument can be more informative than is required by the domain of the functor. This should be compared with the standard type-logical system in which the argument and the domain must be identical in the case of basic categories. This means that with respect to underspecification the situation is asymmetrical: in combinations $A/B \circ C$, $B$ must be less specific than, or as specific as $C$. In the CUG system both the
The domain of the functor and the argument can be underspecified.

The inclusion relations between the various categories and subcategories that are allowed can best be seen in Gentzen sequent derivations. \( A \Rightarrow B \) corresponds semantically to \( v(A) \subseteq v(B) \). In the simple examples we will have to look at the following patterns:

\[
\begin{align*}
A \Rightarrow D & \quad C \Rightarrow B \\
A/B \circ C & \Rightarrow D \\
C \Rightarrow B & \quad A \Rightarrow D \\
C \circ B \setminus A & \Rightarrow D
\end{align*}
\]

The asymmetry is due to the polarities of the (sub)categores: whether they occur in the antecedent or succedent of a sequent or in the domain or range part of a functor.

**Example 9 (Informativeness)** We demonstrate the use of propositional logic in the extended Lambek calculus with a simple example in which we combine a third person singular object noun phrase (say the boy) with a transitive verb of which the domain is underspecified for person/number. Note that we will no longer separate \( \vdash \text{pl} \) from \( \vdash \), and simply omit the subscript when using propositional rules.

\[
\begin{align*}
\text{NP} & \Rightarrow \text{NP} \\
\text{VP} & \Rightarrow \text{VP} \\
\text{NP} \land 3\text{-sg} & \Rightarrow \text{NP} \\
\text{VP}/\text{NP} & \circ (\text{NP} \land 3\text{-sg}) \Rightarrow \text{VP}
\end{align*}
\]

We will now turn to two important differences between the Bayer and Johnson system and the categorial unification system. The first relates directly to the type-logical perspective on linguistic description and the asymmetrical checking relation between the domain of the functor and the argument. The second concerns the effect of the absence of re-entrancy and thereby communication between domain and range subcategories.

**Polarity-Sensitive Underspecification** We will represent the information that *Frau* is a singular, third person, feminine noun, by the formula \((N \land \text{sg} \land 3\text{rd} \land \text{fem})\). If we do the same for *guten* we get \((N \land \text{gen} \land \text{wk})/(N \land \text{gen} \land \text{wk})\).

We have seen that in the CUG analysis, as a result of the re-entrancies in the lexical entries and the application schema for the CUG grammar, the category of the adjective-noun combination bears the morphological information which is the combination of the information arising from both daughters. This can be represented as: \((N \land \text{sg} \land 3\text{rd} \land \text{fem} \land \text{gen} \land \text{wk})\). With the type assignments for the adjective and noun we cannot derive this category for the combination as the following attempt shows. (Note that we abbreviate each category and sort to its initial letter.)

\[
\begin{align*}
\text{fail} & \\
(N \land g \land w) & \Rightarrow (N \land s \land 3 \land f \land g \land w) \\
(N \land g \land w)/(N \land g \land w) & \circ (N \land s \land 3 \land f) \Rightarrow (N \land s \land 3 \land f \land g \land w)
\end{align*}
\]
It is easy to see that the left premise is not derivable by looking at the semantics. We interpret sorts as sets of expressions and conjunctions as intersections, \( \Rightarrow \) as \( \subseteq \). Obviously \( (N \cap n \cap w) \nsubseteq (N \cap s \cap 3 \cap f \cap g \cap w) \). Note that we have abused notation by using the same notation for the categories and their interpretations. In intuitive terms the first premise fails because the antecedent is more informative than the succedent, whereas derivability holds if the situation is reversed. The second premise fails similarly, although in this case the situation is slightly different: the antecedent contains information not contained in the succedent and vice versa. Essentially what this illustrates is that application (or elimination) is defined here in terms of a subsumption relation, which is asymmetric, whereas in CUG, application is defined in terms of the symmetric unification relation.

Part of the problem has to do with the asymmetry of the derivability relation or, alternatively put, the difference in positive and negative occurrences of formulas. In unification grammars, all quantification (underspecification) is universal, wide-scope. Another problem with the analysis has to do with the absence of devices like re-entrancy (see the paragraph on Feature Percolation below).

In Example 9 we could combine a third person singular object noun phrase with a transitive verb in which the domain was underspecified for number and person. However, a problem arises if we want the agreement source to be underspecified as well. A simple example involves an intransitive verb which must combine with nominative noun phrases, hence \( (NP \cap nom) \backslash S \), but where the noun phrase (the boy, for instance) is left underspecified for CASE and simply typed NP. In contrast to the CUG type analysis, this combination is not derivable in the Bayer and Johnson framework.

\[
\begin{align*}
\text{fail} \\
NP \Rightarrow NP \land nom & \quad S \Rightarrow S \\
NP \circ (NP \land nom) \backslash S & \Rightarrow S
\end{align*}
\]

There is a solution to this problem by assigning the type \( (NP \land nom \land acc) \) to an unmarked noun phrase like the boy. In effect, this says that such an expression is both nominative and accusative. Although this may look strange when one is used to the restriction on feature structures that each feature can only be specified by one value, this treatment follows from the interpretation of categorisation in a Lambek style grammar. We could say that this is a case of overspecification. Note that we will often use the term underspecification as a generic term to refer to both underspecification and overspecification.

\[
\begin{align*}
(NP \land nom) & \Rightarrow NP \land nom \\
(NP \land nom \land acc) & \Rightarrow NP \land nom \quad S \Rightarrow S \\
(NP \land nom \land acc) \circ (NP \land nom) \backslash S & \Rightarrow S
\end{align*}
\]

In the first-order variant of the Lambek calculus, underspecification is also sensitive to the polarity of the categories. Whereas \( \forall x (NP(x)) \Rightarrow \)
NP(sg) is derivable, NP(sg) \Rightarrow \forall x(NP(x)) is not. The solution here is to make use of existential quantification in addition to universal quantification. This can be illustrated by the following simple examples. The first concerns a subject-verb combination, where the subject is overspecified for case (a proper noun, for instance) and the verb requires a nominative subject. The second example concerns the combination of a verb with its object where the verb does not care about the number of the object but the object is specified as singular.

\[
\frac{NP(nom) \Rightarrow NP(nom)}{
\forall C(NP(C)) \Rightarrow NP(nom) \quad S \Rightarrow S
}
\]

\[
\frac{NP(sg) \Rightarrow NP(sg)}{
VP \Rightarrow VP \quad NP(sg) \Rightarrow \exists N(NP(N))
}
\]

\[
VP/NP(sg) \Rightarrow \exists N(NP(N)) \Rightarrow VP
\]

At first sight, the distinction between underspecification and overspecification or existential versus universal quantification might be felt to be a kind of unwanted complication as compared to the simplicity of the constraint-based treatment. However, the duplication resulting from the polarity sensitivity should rather be seen as a refinement that offers extra possibilities. Some of the advantages of the subsumption based treatment of underspecification are presented in Bayer and Johnson (1995) with respect to certain coordination constructions. These examples will be discussed extensively in Part IV.

**Feature Percolation** Now consider the situation in which the assignment for guten is as before, but the assignment for Frau is the fully specified \((N \land s \land 3 \land f \land g \land w)\). In this case the agreement source is fully specified but the adjective is not. Now we can derive:

\[
(N \land g \land w)/(N \land g \land w) \circ (N \land s \land 3 \land f \land g \land w) \Rightarrow (N \land g \land w)
\]

\[
(N \land g \land w) \Rightarrow (N \land g \land w) \quad (N \land s \land 3 \land f \land g \land w) \Rightarrow (N \land g \land w)
\]

\[
(N \land g \land w)/(N \land g \land w) \circ (N \land s \land 3 \land f \land g \land w) \Rightarrow (N \land g \land w)
\]

However, we meet the problem that the succedent (representing the information of the mother category in the phrase structure tree) is also underspecified for all the features that do not appear on the adjective. What we would like is that the information on the noun also percolates to the mother (represented by the succedent type), just as in the CUG analysis, but this sequent is not derivable.

\[
(N \land g \land w)/(N \land g \land w) \circ (N \land s \land 3 \land f \land g \land w) \Rightarrow (N \land s \land 3 \land f \land g \land w)
\]
The asymmetry is only partially responsible for the failure of the Bayer and Johnson system to underspecify both the adjective and the noun as in the CUG system. Another problem lies in the fact, mentioned before, that the domain and the range values do not communicate through variables or re-entrancies. Even if we take into account the asymmetry between positive and negative occurrences (using the appropriate underspecified and overspecified forms) we are not able to provide adequate type assignments as can easily be seen by modifying the example. For Frau, underspecified for CASE and DECL in the CUG grammar, we will have to overspecify the assignment for these features; it has all cases and belongs to all declension classes. guten is underspecified for NUM, PER, and GEN.

Frau \( N \land s \land 3 \land f \land g \land n \ldots \land w \ldots \land strong \ldots \)

guten \( (N \land g \land w \land s \land p \land 1 \land 2 \land 3 \land \ldots )/(N \land g \land w) \)

The problem with these category assignments is that they make it possible to derive the combination but the result is typed \( (N \land g \land w \land s \land p \land 1 \land 2 \land 3 \land \ldots ) \) which, as an overspecified category, does not take into account that the combination is a singular, third person, feminine nominal expression. In other words, we can now also derive that guten Frau is a first person, plural, masculine noun phrase.

Let us now consider the version of the Lambek calculus that replaces basic categories with first-order terms. The obvious categories for guten and Frau in such a framework defined above (in Section 5.4) are, respectively:

\[
\forall N, P, G(N(N(P, G, g, w)/N(N, P, G, g, w))) \quad \text{and} \quad \\
\forall C, D(N(s, 3, fem, C, D)).
\]

We assume sorts like Number, Person, Gender, Case and Declension class, with \( N \) and \( s \) of sort Number, \( P \) and \( 3 \) of sort Person, etc. With these assignments we can derive that the combination must be of type \( N(s, 3, fem, g, w) \) as follows, where we abbreviate this type to \( N() \).

\[
\begin{align*}
N() & \Rightarrow N() \\
N()/N() & \Rightarrow N() \\
N(s, 3, fem, g, w)/N(s, 3, fem, g, w) & \Rightarrow N() \\
\forall D(N(s, 3, fem, g, w)) & \Rightarrow N() \\
\forall G(N(s, 3, G, g, w)/N(s, 3, G, g, w)) & \Rightarrow N() \\
\forall CD(N(s, 3, fem, C, D)) & \Rightarrow N() \\
\forall PG(N(P, G, g, w)/N(P, G, g, w)) & \Rightarrow N() \\
\forall CD(N(s, 3, fem, C, D)) & \Rightarrow N() \\
\forall NP(N(P, G, g, w)/N(P, G, g, w)) & \Rightarrow N()
\end{align*}
\]
Because the quantifier in front of the modifier category has scope over both the domain and the range of the functor, the instantiations of the variables that are shared between these two are guaranteed to be identical. For application (elimination of the slash) to work, the variables have to be instantiated to the values provided by the noun. Together these constraints take care that the features specified on the noun and the modifier are all passed on to the mother, represented by the succedent.

**Hybrid Systems** In Section 5.5 we presented one of the layered approaches of Dörre et al. (1996). Other systems presented in that paper resemble the system of Bayer and Johnson. Instead of replacing the basic categories by propositional formulas, they are replaced by some other feature description language. However, these systems suffer from the same problem as the Bayer and Johnson language with respect to underspecification. Simply put, an expression of category $A/B$ combines with an expression of category $C$ (by 'application') only if $C \Rightarrow B$ which, for the feature structure involved, means that $B$ must subsume $C$. Dörre et al. (1996) remark: "Now, this proof-theoretical behaviour leads to an unfortunate complication, when we want to make use of underspecification as it is done in unification-based grammars." (p. 408). The language $L(FC)$ with global constraints is introduced to solve this problem by separating categorial information from morphosyntactic (and other) information and by subjecting the two types of information to different regimes.

The global constraints also allow underspecification for both the adjective and the noun in the German *guten Frau* example with a correct analysis.

**Example 10 ($L(FC)$: *guten Frau*)** We define an $L(FC)$ grammar for the expression *guten Frau*.

- **Signature:** The set $P$ contains the sorts (unary predicates) $fem$, $sg$, $3$, $wk$, $gen$; the set $F$ (binary predicate symbols or features) contains $\text{GENDER}$, $\text{NUM}$, $\text{PER}$, $\text{DECL}$ and $\text{CASE}$; basic category names include $N$ indexed by variables from the constraint language.

- **Lexicon:**

  \[
  \begin{align*}
  \text{guten}, & \quad N(x_1)/N(x_1) \\
  & :: (\text{CASE}(x_1,x_2) \land \text{gen}(x_2) \land \\
  & \quad \text{DECL}(x_1,x_3) \land \text{wk}(x_3)) \\
  \text{Frau}, & \quad N(y_1) \\
  & :: (\text{NUM}(y_1,y_2) \land \text{sg}(y_2) \land \\
  & \quad \text{PER}(y_1,y_3) \land \text{3}(y_3) \land \\
  & \quad \text{GENDER}(y_1,y_4) \land \text{fem}(y_4)))
  \end{align*}
  \]

- **Derivation for** $(guten \, Frau, N(x_1))$: 

The feature structure associated with the phrase *guten Frau* combines the information associated with the adjective and the noun. The variables that are shared between the domain and range category of the functor and the identities that are enforced in the derivation by the axioms take care of that.

### 6.3 Shortcomings

In the previous sections we have discussed the potential of several extended categorial frameworks to deal with cross-classification, partial information (underspecification) and modularisation of the linguistic information.

It appears from this discussion that there is an important contrast with respect to these issues between the AB-style categorial grammars and the Lambek-style grammars. The Categorial Unification Grammar is a stand-alone feature system like HPSG and inherits from this type of framework all the possibilities for classification which we discussed in the first part. Unfortunately, it does not provide the same fine-grained logical perspective on grammatical composition as the type-logical grammars do. It fails to provide introduction rules and thus a complete logic for the logical constants of grammatical reasoning is missing.

According to Dörre et al. (1996), the proof-theoretic behaviour of the Lambek style systems, on the other hand, leads to some complications when one wants underspecification as in unification-based grammars. The polarity sensitivity that arises from treating the categorial connectives as logical constants and from defining grammaticality in terms of derivability means that the matching of functor and argument leads to an asymmetric situation with respect to underspecification. In order to maximise the possibility of stating partial information in this asymmetric situation it is possible to make the partial information also polarity-sensitive and define both overspecified and underspecified information as in the Bayer and Johnson system.

The problematic cases that remain are those of co-variation in which the problem of underspecification is combined with that of distribution of information in categorial structures. In the absence of re-entrancy markers or alternatives like variable sharing, it does not seem possible to pro-
vide a good account for these cases. The first-order versions do not have this problem. In the hybrid frameworks as proposed by Dörre et al. (1996) this is taken care of by combining a validity-based categorial part with a unification-based part.

There are a number of objections that one could raise against this type of hybrid approach. The first is of a general methodological nature. The question arises whether it is really necessary to move to a mixed system to incorporate the benefits of a unification-based system or whether one could provide the same benefits within a resource-conscious, type-logical setting. This argument against the approach relates to an objection raised by Leiß (1994): “its semantics is presented as a mixture of satisfaction and validity and is fairly unfamiliar. I am not convinced that this complication is necessary, and hope that further efforts result in simpler solutions.”

The second objection has to do with the shortcomings of the unification-based approach with respect to underspecification in the context of certain co-ordination constructions. For these cases polarity sensitivity is required, also in the morphosyntactic dimensions. So, the differences between the type-logical grammars with asymmetrical underspecification and the unification grammars with symmetrical underspecification may not be a bad thing. As Bayer and Johnson (1995) point out, the subsumption-based approach provides certain benefits with respect to the analysis of co-ordination of unlike types. This argument is presented in the last part of this book. It also argues against the unification-based frameworks that only have modus ponens rules and no logical characterisation of the grammatical connectives.

The third objection is one that can be raised against all of the type-logical extensions presented above (including the first-order approach), namely that morphosyntactic decorations can only appear on basic categories. Bouma (1993) provides a number of reasons why morphosyntactic information should be associated with complex categories as well. It is obvious, for instance, that many morphosyntactic properties are properties of expressions with complex categories (like verbs, determiners and prepositions) rather than of the expressions they combine with or of the expressions they are a part of.

We can improve on this situation if we can define a framework with the following properties.

- Connectives are treated as grammatical constants with a complete logic (both elimination and introduction rules).
- Underspecification is polarity-sensitive, respecting the logic of the connectives.
- The framework provides a way to deal with co-variation.
- Morphosyntactic decorations can appear on both basic and complex types.
The resource-conscious perspective extends to the treatment of morphosyntactic properties.

We will define such a framework in Part III.

Summary

In this chapter we have discussed some general issues regarding the use of partial information in grammatical description, using the extended categorial frameworks that we presented in the previous chapter. We started our presentation with a unification-based applicative system in order to show the possibilities of using partial structures in a categorial context and then we contrasted this approach with their type-logical treatment.

An important contrast between the AB systems and the type-logical systems involves the notion of polarity and the opposition between a unification-based treatment of underspecification and a validity or subsumption-based treatment. This leads to an asymmetric underspecification configuration in the type-logical case that can be solved by introducing overspecification as well as underspecification, depending on the polarity of the (sub)categories. A problem that remains is that of co-variation. In the cases where re-entrancy or variable sharing is used, the issue of underspecification is connected with that of the specification of feature distribution.

One of the challenges that will be faced in the next part is to provide an account of these co-variation cases. We will show how the multi-modal, type-logical framework can be used to provide an analysis of these and other cases.
Summary of the second part

We have reviewed several extensions to the basic categorial frameworks that we presented in the first part. The extensions refine the category system to allow cross-classification for the description of morphosyntactic information. The techniques used to achieve these extensions vary from replacing basic categories by propositional formulas, first-order terms or feature structures, to redefining a categorial grammar as a stand-alone feature structure system.

Besides cross-classification, the extensions also provide a way to reduce the number of assignments to lexical expressions through underspecification. Moreover, they provide different ways to express the distribution of morphosyntactic information in the compositional structure.

Categorial unification grammars behave in many ways differently from the type-logical versions with respect to these issues of classification and distribution. These differences relate to the fact that CUGs only use the rule of application and not a complete logic for composition and selection (with both elimination and introduction rules). We have discussed the consequences for underspecification and the matching of functor-argument structures above.

The AB-systems suffer from incompleteness from a logical point of view. The type of underspecification that they employ is not polarity-sensitive which leads to certain descriptive problems for co-ordination constructions. The Lambek-style systems that we presented above on the other hand, are restricted in the sense that they can specify morphosyntactic information only on basic categories. In cases like the Bayer and Johnson system, they do not allow the expression of generalisation on type-assignments in which co-variation is concerned.

In the next part we will investigate the possibilities offered by the multimodal version of the type-logical grammar presented in Chapter 2. We will see that it already provides the tools needed for cross-classification, underspecification, feature distribution etcetera, so that extensions with other logics (propositional, first-order, layered systems) are not necessary.
Part III

A Resource-Conscious Approach
Introduction to Part III

In the previous part we have presented various ways to extend categorial grammars with formal devices to describe the morphosyntactic properties of linguistic expressions. We also pointed out how these devices can or why they cannot deal with underspecified information and the distribution of morphosyntactic information in phrase structure.

Our aim in this third part is to use the mechanisms available in the type-logical framework presented in Chapter 2 to refine the classification potential. What we want is a classification schema that has the same expressive possibilities as the constraint-based systems outlined in Chapter 3, but avoids the obstacles discussed in the previous chapter. We thus want to refine the notion of category in order to cross-classify expressions along multiple dimensions. More specifically we want categories to provide morphosyntactic information as well as syntactic information. Expressions should be classified hierarchically to reduce the number of lexical assignments through underspecification. Furthermore, we need a mechanism that accounts for co-variation. Simply put, this means that categories can be underspecified in different parts and the extensions require the same values to be filled in for these parts. We have already illustrated this in the introduction to Part II.

We propose to use the unary modal operators $\Diamond$ and $\Box$, introduced in Chapter 2, to carry morphosyntactic information. The resource modes indexing the operators will represent specific values for the morphosyntactic properties. The morphosyntactic markings on expressions are checked in the course of the derivation by the logical rules of the modal operators.

Inclusion postulates are introduced to structure the information hierarchically, thereby allowing underspecified (and overspecified) categories. Distribution postulates, i.e. interaction postulates between the unary morphosyntactic modalities and the binary composition operations, govern the way information is distributed in phrase structure. This framework complies with the goals we established at the end of the previous part.

In Chapter 7 we concentrate on the technical possibilities of the modal approach to define the basic feature-checking mechanism and show how structural postulates define inclusion relations and distribution principles. More details are provided in Chapter 8 with a number of analyses for small linguistic fragments.
7

A Logic for Feature Checking

In this chapter we show how in a mixed multimodal categorial framework unary modalities can be used to represent morphosyntactic properties of expressions. The residuation logic for the unary connectives $\odot_i$ and $\square_i$ is used to define a feature checking procedure that is resource and polarity-sensitive (Section 7.1). Each mode $i$ represents some morphosyntactic property. The logical rules are complemented by a collection of structural postulates. Underspecification is dealt with by assuming general modes that are related to specific instances by inclusion postulates (Section 7.2). The distribution of individual features and feature complexes is regulated by means of distribution postulates (Section 7.3).

These investigations extend some proposals that have been made by authors including Versmissen (1996), Johnson (1999), and Kraak (1998).

7.1 Feature checking

We start with a brief recapitulation of the multimodal framework. Next, we show how unary modalities can be used to encode morphosyntactic properties and how their logic defines a feature checking operation that parallels the binary selection and composition operation.

Unary operators have been used as bracketing operators, marking domains of locality or as structural modalities to define modally controlled structural rules for the compositional connectives (see Morrill (1994)). A simple example of this type of control is provided by a permutation modality $\langle p \rangle$, which replaces a global, structural rule by a modally controlled version.

$$\odot_p A \otimes_c B \rightarrow B \otimes_c \odot_p A$$

Moortgat (1997) offers extensive discussion of the ways in which the unary modalities can control resource management by imposing or relaxing certain structural options of the composition operators. We will use them here for a specific domain of application: the classification of expressions with respect to their morphosyntactic properties, and the distribution of these properties along the compositional axis.

7.1.1 The Framework

A categorial grammar defines a language by assigning categories to the atomic expressions of the language (the words). The categories are formulas from a logical language. It is the task of the grammar to define
which complex expressions are grammatical and in which category they are to be classified. In the type-logical approach this is taken care of by a proof-procedure. To show that an expression $E$ is of some category $C$, we replace the words by their categories (as specified in the lexicon) and assemble these formulas into a term from which we try to derive $C$ by the logic governing the formulas. We have presented the formal language and the rules of inference already in Chapter 2. Here we will repeat the major aspects, focusing on the presentation of the unary connectives.

The formula language, $\mathcal{F}$, is defined as follows, where $\mathcal{B}$ is a set of atomic symbols, called the basic categories. The indices $i$, are taken from a set of symbols $\mathcal{T}$.

$$\mathcal{F} ::= \mathcal{B} | (\mathcal{F} \cdot_i \mathcal{F}) | (\mathcal{F} /_i \mathcal{F}) | (\mathcal{F} \backslash_i \mathcal{F}) | \diamond_i \mathcal{F} | \square_i \mathcal{F}$$

A formula or category represents a set of expressions, or more generally, a set of 'linguistic resources'. The structure of the formula tells us something about the properties of the expressions they denote. For instance, we want to say that a resource in $A \cdot_i B$ is related (in mode $i$) to a resource in $A$ and a resource in $B$. This 'parts-whole' relation between the three expressions is interpreted as composition in the grammar logic. For a resource in $\diamond_i A$ we could say that it is an $i$-marked version of a resource in $A$. We will distinguish between different types of composition and different types of markings, hence the indices.

For the interpretation we have assumed a set of linguistic resources $W$, a family of ternary and binary relations (both indexed by $\mathcal{T}$) used to interpret the binary ($/_i$, $\cdot_i$, $\backslash_i$) and unary ($\square_i$, $\diamond_i$) connectives, respectively. If the ternary relation is interpreted as composition ($R(wxy)$ means that $w$ consists of an $x$ and a $y$) then $A \cdot_i B$ denotes, as we already indicated, the 'composed' resource $w$ (whole). $A/B$ denotes the resources related to the second argument of the relation $(x)$: those that select $B$ resources $(y)$ to form $A$ resources $(w)$. The unary connectives show the same duality, which will be exploited in our account of morphosyntactic features.

We have also defined a notion of derivability between formulas $A \rightarrow B$ so that $v(A) \subseteq v(B)$ (for all frames and valuations), which respects this interpretation. The so-called pure residuation logic below, also known as the non-associative Lambek calculus NL, captures these requirements.

\[
\begin{align*}
\text{(REFL)} & \quad A \rightarrow A \\
\text{(TRANS)} & \quad \text{if } A \rightarrow B \text{ and } B \rightarrow C, \text{ then } A \rightarrow C, \\
\text{(RES)} & \quad A \rightarrow C /_i B \text{ iff } A \cdot_i B \rightarrow C \text{ iff } B \rightarrow A \backslash_i C \\
& \quad A \rightarrow \square_i B \text{ iff } \diamond_i A \rightarrow B \\
\end{align*}
\]

It is important to realise that we have a whole family of binary and unary connectives (indexed by $\mathcal{T}$) but that all of the binary and unary connectives are governed by the same logical rules which relate $\cdot_i$ to $/_i$, $\\backslash_i$, and $\diamond_i$ to $\square_i$ for each mode $i$. The distinction between $\cdot_i$ and $\cdot_j$ or $R_i$...
and $R_j$ in the model is not one of 'logic' but of 'structure'. $R^3_j$ can be an associative relation and $R^2_j$ non-associative, $R^3_i$ might be commutative but $R^2_i$ not. It is also possible that two modes of composition differ only in the way they interact with certain unary operators. These characteristics are defined as frame conditions on the relations in the model theory. In the proof theory, they are defined by so-called structural rules. In the Gentzen sequent presentation, the rules appear as follows.

$$\Gamma[\Delta_2 \circ_c \Delta_1] \Rightarrow C \quad \Gamma[(\Delta_1 \circ_a (\Delta_2 \circ_a \Delta_3))] \Rightarrow C$$

This corresponds to the following axiomatic form.

**PERM**

$A \bullet_c B \rightarrow B \bullet_c A$

**ASSOC**

$(A \bullet B) \bullet_a C \leftrightarrow A \bullet_a (B \bullet_a C)$

Besides structural postulates that fix the characteristics of single modes of composition and marking, we can also specify certain rules for the interaction of different modes.

In the following rule for **PERM**, we see a composition relation where the components can permute, provided the first component is marked $p$.

$$\Gamma[\Delta_2 \circ_c (\Delta_1)p] \Rightarrow C \quad \Gamma[(\Delta_1)p \circ_c \Delta_2] \Rightarrow C$$

**PERM**

$\Diamond_p A \bullet_c B \rightarrow B \bullet_c \Diamond_p A$

In the following rule, we allow an $x$-checking to be changed into a $y$-checking.

$$\Gamma[(\Delta)_y] \Rightarrow C \quad \Gamma[(\Delta)_x] \Rightarrow C$$

This enables us to derive a sequent like $\Box_y A \Rightarrow \Box_x A$. We call this an inclusion postulate, because from a semantic perspective it means that $\nu(\Box_y A) \subseteq \nu(\Box_x A)$ (or $R^2_y \subseteq R^2_x$).

$$\frac{A \Rightarrow A}{\Box_y A \Rightarrow A} \quad A \Rightarrow \Box_y A \quad \frac{\Box_y A \Rightarrow A}{\Box_x A} \quad R^2 \Box_y A \Rightarrow \Box_x A$$

Because the inclusion postulates will be used quite a lot in Gentzen sequent derivations below, we will present them in the Gentzen format: $\Box_y A \Rightarrow \Box_x A$. Most of the time we will not unfold the derivation further than this.

In the next rule, we allow a $z$-marking on an $n$-composite to be replaced by a $z$-marking on the first component of the composite.
\[
\begin{align*}
\Gamma[(\Delta_1 z \circ_n \Delta_2)] & \Rightarrow C \\
\Gamma[(\Delta_1 \circ_n \Delta_2)_z] & \Rightarrow C \\
\kappa_1 \diamond_z (A \bullet_n B) & \rightarrow \diamond_z A \bullet B
\end{align*}
\]

Now that we have presented the formal machinery again we will be more precise about how it is used to define grammars. At this point we only look at the use of the logical operators and their potential to effect cross-classification. The use of postulates will be discussed in the following sections which will deal with underspecification and distribution of morphosyntactic information in phrase structure.

### 7.1.2 Cross-classification

The duality between \( \diamond \) and \( \Box \) is similar to the duality between \( \bullet \) and \( /, \backslash \). We have seen an interpretation of the residuation duals in very general models. In the case of the binary connectives we interpret these more specifically as composition and selection operations (wholes and parts). A similar interpretation holds for the unary structure building operators. However, because there is only one part, one could think of this operation in terms of marking instead of composition. We use the duality between them to define the morphosyntactic properties of a language in terms of checking. An expression typed as \( \Box_n A \) is similar to \( B \backslash_n A \) and can be read as “requires mode i marking”, parallel to “requires composition with argument”. By the duality between \( \diamond \) and \( \Box \) we can define a cancellation schema that we interpret as feature checking. Just as \( B \bullet_i B \backslash_i A \rightarrow A \) is valid so is \( \diamond_i \Box_i A \rightarrow A \). The \( \diamond \) cancels or checks the \( \Box \). Metaphorically speaking we can also say that the \( \Box \) functions as a lock that can be opened by a corresponding key \( \diamond \).

To illustrate the use of unary operators to mark morphosyntactic distinctions we consider the assignments of the example given in the introduction to Part II again.

\[
\begin{align*}
\text{NP}_1 & \rightarrow \Box_1 \text{NP} = 3\text{-sg-nom-np} \quad \text{he, she, the boy...} \\
\text{NP}_2 & \rightarrow \Box_2 \text{NP} = 123\text{-sgpl-nom-np} \quad I, \text{they, the boys} \\
\text{NP}_3 & \rightarrow \Box_3 \text{NP} = 123\text{-sgpl-acc-np} \quad \text{him, her, the boy, me, them, the boys}
\end{align*}
\]

Given these translations of the categories for the noun phrases, the categories for the verbs walk and walks are as follows.

\[
\begin{align*}
\text{NP}_1 \backslash S & \rightarrow \Box_1 \text{NP} \backslash S = 3\text{-sg-subject-v} \quad \text{walks} \\
\text{NP}_2 \backslash S & \rightarrow \Box_2 \text{NP} \backslash S = \text{non-3-sg-subject-v} \quad \text{walk}
\end{align*}
\]

Because we use only a single mode of composition, we can omit the index on the binary connectives (both the logical and the structural ones). To show that he \( \bowtie \) walks is a sentence, we derive the sequent \( \Box_1 \text{NP} \circ \Box_1 \text{NP} \backslash S \Rightarrow S \) as follows.
The Gentzen sequent presentation clearly shows how feature checking by the unary modalities proceeds. In the bottom sequent there are two \( \text{NP} \) formulas marked by \( \Box_1 \), but they are of opposite polarity. The derivation shows how the logical rules for \( \Box \) are used here to implement feature checking. Different strategies are discussed in the next chapter.

### 7.1.3 Discussion

Our definition of grammar provides a deductive perspective on the characterisation of grammatical structures. We have classified expressions using logical formulas as categories and we have defined the typing relation of complex expressions in terms of logical deduction using a resource-conscious grammar logic. The connectives \( /, \cdot, \setminus \) define composition and selection to cover the basic structure building component of the grammar. The classification potential of the formulas is further refined by decorating categories with morphosyntactic properties which are marked by the operators \( \Diamond \) and \( \Box \), whose logic defines a checking procedure. In this checking procedure, the feature information behaves in a resource-sensitive way. In this sense it is like the treatment of features in the minimalist program (Chomsky (1995), Stabler (1997)) and unlike the treatment in unification-based grammars. We provide a simple illustration here.

**Checking Procedure** The checking theory of morphosyntactic properties bears some resemblance to certain ideas in the minimalist program with respect to the cancellation of features and the modal control on phrase structure operations. We can make this more precise by taking a closer look at some reconstructions of minimalism by Stabler (for instance Stabler (1997) and Stabler (1999)). In this framework a grammar consists of a lexicon and a pair of *generating functions: merge* and *move*. The lexicon is a set of feature terms represented as strings of features. Restrictions may be put on the order and the number of occurrences of the features that make up this string (either as universal or language specific constraints). Different types of features can be distinguished.

- **Selected categories** \( c, t, d, n, v, p, \ldots \)
- **Selector features** \( =c, =t, =d, =n, =v, =p, \ldots \)
- **Licensors** \( +\text{wh}, +\text{case}, +\text{focus}, \ldots \)
- **Licensees** \( -\text{wh}, -\text{case}, -\text{focus}, \ldots \)
- **Phonetic features** \( \text{every}, \text{some}, \text{student}, \ldots \)
- **Semantic features** \( \text{every}', \text{some}', \text{student}', \ldots \)
The syntactic and morphological features come in different polarities: c versus =c and +wh versus -wh. This parallels the polarities of the type-logical grammars.

Expressions are either lexical items (strings of features) or binary trees: \( \Delta_1 < \Delta_2 \) or \( \Delta_2 > \Delta_1 \), where \( \Delta_1 \) and \( \Delta_2 \) are either binary trees themselves or lexical items. If we talk about the features of a tree, we mean the features of its head, which is the 'smaller' element of the pair. For the purpose of illustration here, we will ignore the distinction between < and > and we will use the \( \circ \) operator to represent structure.

The operation merge is defined on two expressions.

\[
=x\Delta_1 \quad x\Delta_2 \quad \text{merge:} \quad (\Delta_1 \circ \Delta_2)
\]

This should be read as follows: two expressions (lexical or phrasal) can merge into a tree, provided the first feature of one is \( =x \) and the first feature of the other is \( x \). In Stabler (1999) the lexical/phrasal status of the selecting expressions determines whether the selected expression is put to the right, in complement position, or to the left, in specifier position. In the resulting tree these features are cancelled (erased). A simplified version of move can be presented as follows.

\[
(+x\Delta_1 \circ (\Delta_2 \circ (-x\Delta_3 \circ \Delta_4))) \quad \text{move} \quad ((\Delta_3 \circ \Delta_4) \circ (\Delta_1 \circ \Delta_2))
\]

Note again, that in this case, the movement is triggered by a pair of opposing features. The parallel with the modal operators of different polarities that are used to cancel each other out points to the analogous idea of resource-conscious feature checking.

We provide a simple example. The lexicon consists of the items \( d-x \ she \) and \( =d+x \ sinks \). We can apply a merge step followed by a move step, \( \Delta_2 \) and \( \Delta_4 \) are empty.

\[
=dx sinks \quad d-x \ it \quad \text{merge:} \quad (+x sinks) \circ (-x \ it)
\]

\[
(+x sinks) \circ (-x \ it) \quad \text{move:} \quad it \circ sinks
\]

The idea that feature checking defines a kind of agreement-matching is thus not unique to the approach we have sketched here. The work by Johnson (1999) deals with a resource-conscious perspective on grammatical description in LFG. The important point to note about these approaches is that they use logical operators, exploiting their logical properties (residuation, polarity, etc.) for linguistic description. For alternative and more refined reconstructions of the minimalist program in a type-logical context see Cornell (1997), Lecomte (1998), Heylen (1998), Vermaat (1999). We present a type-logical reconstruction of this example in Chapter 8.
Feature Structures  As we mentioned at the beginning of this chapter, we want to extend the categorial grammar to account for cross-classification and generalisations (abstraction and factorisation) in grammatical description. These are the kinds of benefits that are normally achieved by the introduction of feature structures. We have now introduced more structure to the categories in a type-logical grammar to account for multiple properties (decomposition). However, it is important to see that what we have ended up with are not feature structures in the technical sense as we defined them in Chapter 3.

First of all, it is important not to confuse the models and the logic of type-logical and constraint-based approaches despite superficial similarities. In a modal perspective on feature structures, the structures are also taken to be Kripke structures and a mixed, multi-modal logic is used to talk about the structures (see our discussion in Chapter 3 and Blackburn (1994) for example). Although the two approaches may use the same logical tools (multimodal logic), they use it for a different purpose. For instance, the elements that populate the models are clearly not the same. In typical feature structure models there are many kinds of objects, most of which are not linguistic resources. Only elements of sort sign could count as such in a theory like HPSG, whereas the others are reified properties like nom or boolean. In our type-logical set-up such properties are not expressed as nodes in the model, but rather as relations between linguistic resources. Also, the modal language is used to define the grammar formalism in the constraint-based theories, whereas in type-logical grammars, the logic as such captures part of linguistic theory. For instance, the logic governing the binary connectives is specially attuned to capture aspects of linguistic composition and selection.

It is also important to see that the grammars work differently. In the resource-sensitive categorial case there is a need to check and cancel morphosyntactic features. For this we need a balance between boxes and diamonds or negative and positive positions. In the constraint-based theories there is no need to check features. In particular cases we may require a compatibility check (or unification) but features are not cancelled. This has repercussions on how we treat underspecification as we will see in the following sections.

Another important difference involves the sensitivity of underspecification to the polarities of types and subtypes in a type-logical derivation. In the previous chapters we have seen how this technical difference poses specific problems for combinations of constraint-based approaches with type-logical ones as has been pointed out by, for instance, Dörre et al. (1996) and Dörre and Manandhar (1997) and Francez (1997). In the following sections we will turn to these issues in the context of our modal treatment of morphosyntax.
7.2 Underspecification

In the introduction to Part II we discussed how an information ordering on the categories, corresponding to a hierarchical classification of the expressions, can be used to simplify a grammar by reducing the number of assignments to lexical expressions, capturing certain generalisations. Now that we have defined a more refined notion of classification in the previous section, we want to discuss an ordering for the system we have just presented. We will see how inclusion postulates can be used to define an information ordering on the morphosyntactic properties.

The type-logical calculus defines an ordering relation on categories in terms of logical derivability. If \( A \rightarrow B \) this means that \( v(A) \subseteq v(B) \) as we saw above. However, we can also define an order on the non-logical vocabulary. In our case this consists of the basic categories \( \mathcal{A} \) and the indices \( T \). We can view these orderings as part of the signature that comes with the specification of the grammatical vocabulary used in a specific grammar. For the basic categories we can add non-logical axioms to define a sortal hierarchy (Lambek (1961), Dörre and Manandhar (1997)).

\[
\frac{}{A \Rightarrow B} A \leq B
\]

For the indices we use inclusion postulates, which we will present by sequents as follows. This notation makes the semantics of the relation immediately clear: \( X \Rightarrow Y \) iff \( v(X) \subseteq v(Y) \).

\[
\Box_y A \Rightarrow \Box_x A.
\]

In the simple example from the introduction to Part II we presented the advantages of having an information ordering defined on the basic categories. Such an ordering provides a way to refer to classes of expressions and also to subclasses and superclasses (\( \text{NP}_1 \leq \text{NP}_3 \) means that \( v(\text{NP}_1) \subseteq v(\text{NP}_3) \)). This ordering can be used to reduce the number of assignments.

In using the ordering relation defined by the inclusion postulates, we have to take into account the position where a diamond or box occurs in a category, i.e. its polarity. In order to be able to derive \( \Box_i A \circ \Box_j B \Rightarrow B \), it must be the case that \( \Box_i A \Rightarrow \Box_j A \) and not the other way round. This has important consequences for our view on underspecification. This is best explained by a very basic example. We take a tiny fragment of Dutch. Note that we write \([i]\) for \( \Box_i \) and provide both transitive and intransitive assignments for the verbs.

\[
\begin{align*}
zij & \quad \text{she/they} & \quad [\text{Num}]^{\text{NP}} \\
zungt & \quad \text{sings} & \quad [\text{sg}]^{\text{NP}\backslash S/\text{num}}^{\text{NP}} & \quad [\text{sg}]^{\text{NP}\backslash S} \\
zungen & \quad \text{sing} & \quad [\text{pl}]^{\text{NP}\backslash S/\text{num}}^{\text{NP}} & \quad [\text{pl}]^{\text{NP}\backslash S} \\
liedjes & \quad \text{songs} & \quad [\text{pl}]^{\text{NP}}
\end{align*}
\]
The type assignment to *zij* captures both a singular and plural assignment. For the transitive verbs, the assignments also abbreviate two options. The *num* decoration on the object noun phrase tells us that the verb does not care whether its object is singular or plural. The fact that we have to make a distinction between \([Num]\) and \([num]\) is due to the polarity of the position in which the occur. In terms of feature checking we want \([Num]\)NP to be able to be checked by either a singular or a plural verb. We want \([num]\) to check both singular and plural noun phrases. We first present the essential steps in a derivation for *zij zingt* (she sings). The combination of the subject \([Num]\)NP with the verb \((\text{sg} \text{NP})\)S, leads to the premise: \([Num]\)NP \(\Rightarrow [\text{sg}]\text{NP}\):

\[
\frac{\epsilon}{[\text{Num}]\text{NP} \Rightarrow [\text{sg}]\text{NP} \quad S \Rightarrow S} \quad L/}
\]

For this derivation to work we thus need the following inclusion postulate.

\([\text{Num}]\text{A} \Rightarrow [\text{sg}]\text{A}\)

As we already said, we want \((\text{sg} \text{NP}\)S)/\([\text{num}]\text{NP}\) to check both singular and plural objects. Because we are only interested in the checking of the object, we abbreviate \([\text{sg}]\text{NP}\)S/\([\text{num}]\text{NP}\) to \(\text{IV}/\text{num}\)NP. The first step of the derivation for *zingt liedjes* (sings songs) then goes as follows.

\[
\frac{\epsilon}{[\text{pl}]\text{NP} \Rightarrow [\text{num}]\text{NP} \quad \text{IV} \Rightarrow \text{IV} \quad L/}
\]

For this derivation to work we need the following inclusion postulate.

\([\text{pl}]\text{C} \Rightarrow [\text{num}]\text{C}\)

It is immediately apparent from these examples that the inclusion postulates are polarity-sensitive: they work in one direction only. Again we are dealing with subsumption-type checks rather than unification-type checks. And as with the other type-logical grammars, this asymmetry is related to the semantics. In the first case, the argument (subject) is underspecified, or rather overspecified. *Num* noun phrases can be checked by both *sg* and *pl*. Because both *Zij zingt* (she sings) and *Zij zingen* (they sing) is grammatical we want the inclusion postulates to run as follows.

\([\text{Num}]\text{C} \Rightarrow [\text{sg}]\text{C}\)

\([\text{Num}]\text{C} \Rightarrow [\text{pl}]\text{C}\)

Semantically this means:

\[
v([\text{Num}]\text{C}) \subseteq v([\text{sg}]\text{C})
\]

\[
v([\text{Num}]\text{C}) \subseteq v([\text{pl}]\text{C})
\]

\[
v([\text{Num}]\text{C}) \subseteq v([\text{sg}]\text{C}) \cap v([\text{pl}]\text{C})
\]
For the underspecified verb, which does not care about the number of its object, the inclusion relations are reversed. Here the 'checker'-mode is underspecified.

\[
\begin{align*}
|sg|C & \Rightarrow |num|C \\
|pl|C & \Rightarrow |num|C
\end{align*}
\]

Semantically this means that

\[
\begin{align*}
v(|sg|C) & \subseteq v(|num|C) \\
v(|pl|C) & \subseteq v(|num|C) \\
v(|sg|C) \cup v(|pl|C) & \subseteq v(|num|C)
\end{align*}
\]

We can depict this ordering relation in a Hasse diagram (see Partee et al. (1990)), where the smaller elements (subsets) are below the bigger elements (supersets).

![Hasse diagram]

Also in further examples, we will use the distinction between lowercase (num) and uppercase (Num) modes as a convention to distinguish between underspecified and overspecified modes, respectively.

### 7.3 Distribution

In the approach we sketch in this chapter, we have special operators to account for morphosyntactic properties instead of more complex basic categories. Some of the advantages of having morphosyntactic information separated from the basic categories will become clear in this section, in which we discuss the distribution of morphosyntactic information in phrase structure. The next chapter will provide more motivation.

#### 7.3.1 Distribution Postulates

The decomposition of syntactic and morphosyntactic information of categories in terms of features structures as developed in theories like GPSG
and HPSG, for instance, offers the possibility to consider the distribution of each feature or group of features in phrase structure separately (head feature principle, foot feature principle, etc.). In the type-logical framework and its application to linguistic description as we have described it above and in previous chapters, the decomposition is taken care of by separating basic category information from morphosyntactic decorations. In feature structure grammars, like HPSG, feature distribution principles are expressed through re-entrancies that require that the values in two parts of the structures have to be the same. We will now show how distribution postulates can take on this role in the type-logical setting.

In a categorial grammar, the phrase structural properties of expressions are projected from the lexicon. More specifically, they are determined by the selectional requirements expressed by functional categories. These categories provide the information that is expressed by phrase structure rules in phrase structure grammars. In the example of the introduction to the first part we therefore presented an example of feature distribution as some kind of constraint on functional categories: agreement in specifier categories is accounted for by the re-entrancy between the value and the argument part of the functor. This idea can be represented for a determiner like *the* as follows.

\[ \forall(\alpha \left[ \begin{array}{c} \text{CAT} \\ \text{AGR} \end{array} \right] \rightarrow \begin{array}{c} \text{NP} \\ \text{AGR} \end{array} \alpha ) \]

In the calculus with first order terms from Morrill (1994) this takes the following form.

\[ \forall x(\text{NP}(x)/\text{NP}(x)) \]

This categorial encoding of a feature distribution principle, however, is not possible in the type-logical language that we have presented because in the formal language that we are working with we have no means to express re-entrancies. Instead of stating distribution principles as constraints on categories, we therefore formulate them as principles that relate morphosyntactic decorations to phrase structural composition. Consider, for instance, the distribution postulate (\(k_1\)), now with specific indices on the logical operators.

\[ k_1 \quad \Diamond_h (A \bullet_{lh} B) \rightarrow \Diamond_h A \bullet_{lh} B \]

From a linguistic point of view, this distribution postulate can be interpreted as fixing the behaviour of head features. The combination mode \(\bullet_{lh}\) should be interpreted as a mode in which the linguistic head appears as the left daughter (\(lh\) for left-headed). A head feature like \(h\) has the property that when it has to check a phrase in \(lh\) mode, it proceeds by checking the head of that phrase. Now consider the following distribution rule (\(k\)):
In this case the mode \( a \) distributes over both parts of an \( m \)-composed structure (\( m \) for modifier). Linguistically speaking this captures the essence of an agreement configuration. Let us illustrate this with a simple Italian fragment. We use binary modes \( s \) and \( m \) to represent specifier-head and head-modifier combinations respectively. Both these modes allow strong distribution of the agreement modality \( (pl) \).

\[
K \quad \diamond_a (A \bullet_m B) \rightarrow \diamond_a A \bullet_m \diamond_a B
\]

With the help of the structural rules we can now distribute the check for \( [pl] \) over both parts so that agreement on this feature is forced on the parts. (We substitute the lexically assigned categories by the words to prevent the formulas from cluttering up the page.)

\[
\begin{align*}
\langle pl \rangle (A \bullet_i B) & \rightarrow \langle pl \rangle A \bullet_i \langle pl \rangle B \ (i \in \{m, s\}) \\
\langle pl \rangle & \rightarrow [pl] \langle NP / N \rangle \\
pomodori & \rightarrow \text{tomatoes} \ [pl] / N \\
rossi & \rightarrow \text{red} \ [pl] / (N \setminus m N)
\end{align*}
\]

7.3.2 Distribution and Inclusion

We will now consider the interaction between distribution and inclusion postulates. The example we used in the introduction to Part II provides us with a good starting point. There we showed that the re-entrancies (or variable sharing) in the entry for the determiner reduced the number of assignments. The determiner can combine with both a singular and a plural noun, but the properties of the resulting noun phrase co-vary with each choice.

There are two aspects about the use of the modalities that have to be considered. First to note is that co-variation will be enforced in our account of agreement by having the agreement postulates defined only for fully specified morphosyntactic values. Second is that we need the morphosyntactic decorations to take global scope to enforce re-entrancy as we saw above (compare \( \forall x (A / B) \) and \( \Box (A / B) \)).

The first point is illustrated by the following derivations: the first is a derivation of a grammatical sentence in which we distribute the fully specified mode and the second is a derivation of an ungrammatical noun phrase, which becomes derivable as soon as we allow distribution of an underspecified mode.
A logic for feature checking

The agreement postulate takes care of the correct distribution of the morphosyntactic information in phrase structure. The use of special decorations, i.e. separate logical operators for the morphosyntactic properties, makes this kind of treatment possible.

\[
\text{pela} \quad \text{peels} \quad \text{IV}/\text{num} | \text{NP} \\
\text{i} \quad \text{the (pl)} \quad [\text{pl}] (\text{NP} / _s \text{N}) \\
\text{pomodoro} \quad \text{tomato} \quad [\text{sg}] | \text{N}
\]

The grammar overgenerates, if we would also allow the general feature \textit{num} to distribute, as the following derivation for the ungrammatical \textit{i pomodoro} shows.

\[
\begin{align*}
(i)_{\text{pl}} & \circ (\text{pomodoro})_{\text{sg}} \Rightarrow \text{NP} \\
(i)_{\text{pl}} & \circ (\text{pomodoro})_{\text{num}} \Rightarrow \text{NP} \\
(i)_{\text{num}} & \circ (\text{pomodoro})_{\text{num}} \Rightarrow \text{NP} \\
(i \circ \text{pomodoro}) & \circ \text{num} \Rightarrow \text{NP} \\
i \circ \text{pomodoro} & \Rightarrow \text{num} | \text{NP}
\end{align*}
\]

The second point regarding the modal decorations involves their scope. To enforce agreement, we have given the modal decoration global scope \((i)(X/Y))\). Consider what would happen if we had an ordering on the basic categories \((i)[X/i][Y])\) only, as in the extended Lambek calculus proposed by Bayer and Johnson (1995). Remember that in their analysis, the basic categories are replaced by simple propositional formulas and the axiom schema is changed so that \(A \Rightarrow B\) is derivable in case it is a theorem of propositional logic. In the following assignments, the morphosyntactic decorations only appear on basic categories.

\[
\text{the} \quad (\text{SG} \land \text{PL} \land \text{NP}) | _s \text{N} \quad [\text{Num}] | \text{NP} / _s \text{num} | \text{N} \\
\text{boy} \quad (\text{SG} \land \text{NP}) \quad [\text{sg}] | \text{NP} \\
\text{boys} \quad (\text{PL} \land \text{NP}) \quad [\text{pl}] | \text{NP}
\]
There is only a single assignment to *the* as it combines with both singular and plural nouns and it gives rise to both singular and plural noun phrases. However this assignment does not relate the choices of argument and result. We can, of course, derive the expression *the boy* as a singular noun phrase.

\[
\begin{align*}
\text{SG} \land \text{PL} \land \text{NP} &\Rightarrow \text{SG} \land \text{NP} \\
(\text{SG} \land \text{PL} \land \text{NP})/_{gN} &\circ (\text{SG} \land \text{N}) \Rightarrow \text{SG} \land \text{NP}
\end{align*}
\]

But we can also derive *the boys* to be a singular noun phrase.

\[
\begin{align*}
\text{SG} \land \text{PL} \land \text{NP} &\Rightarrow \text{SG} \land \text{NP} \\
(\text{SG} \land \text{PL} \land \text{NP})/_{gN} &\circ (\text{PL} \land \text{N}) \Rightarrow \text{SG} \land \text{NP}
\end{align*}
\]

The same goes for the alternative analysis with unary modalities.

\[
\begin{align*}
|\text{Num}| \text{NP} &\Rightarrow |\text{sg}| \text{NP} \\
|\text{pl}| \text{N} &\Rightarrow |\text{num}| \text{N} \\
|\text{Num}| \text{NP}/_{gN} &\circ |\text{pl}| \text{N} \Rightarrow |\text{sg}| \text{NP}
\end{align*}
\]

On the other hand, the use of distribution postulates and the fact that we can mark morphosyntactic information on complex categories allows us to account for these cases of co-variation without variable-sharing or re-entrancy technique, as demonstrated above.

### 7.4 Multiple Modalities

In each of the examples above, we have used just one modal decoration to carry morphosyntactic information. In most cases we have simplified the analysis considerably by taking this decoration to refer to a single morphosyntactic property, say number, ignoring all the other properties like person, case, or gender. Of course, this is not a principled restriction. Because our intention was to make it possible to classify expressions along multiple dimensions, we certainly do not want to restrict the number of attributes to just one extra. There are several options available to arrive at a more realistic description.

The first option is simply to let the indices stand for complexes of information. In fact, this is exactly what we did in the example of the introduction to Part II which we took up in Section 7.1.2, where we used categories like $\square_1 \text{NP}$, to stand for 'third person, singular, and nominative'. As an alternative, we could make this structure more transparant and replace the set of atomic symbols $T$ by simple feature structures or, perhaps more accurately, by formulas from some feature description language:

\[
\begin{bmatrix}
\text{NUMBER} & 3 \\
\text{PERSON} & \text{sg} \\
\text{CASE} & \text{nom}
\end{bmatrix}
\]

where $\phi = ((\text{NUMBER} 3 \land \text{PERSON} \text{sg} \land \text{CASE} \text{nom})$
Note that the square brackets \([\cdot]\) represent our residuated \(\Box\) operator, whereas the \((\cdot)\) brackets represent \(\Diamond\) operators of a Blackburn-type modal feature logic. Of course, this use of a logical language for the feature modes suggests further exploitation of its potential in the context of a hybrid or layered logic. This can be visualised as follows (see also the next chapter and Heylen (1997c)). \(\Rightarrow_{\text{FL}}\) refers to derivability in the feature logic.

\[
x \Rightarrow_{\text{FL}} y \\
\Box_x A \Rightarrow \Box_y A
\]

Our formula language allows another option to decorate categories along multiple dimensions. Instead of complicating the modes on the operators, we can also multiply the number of operators, one for each dimension, and use simplex modes as before.

\[
\Box_3 \Box_{sg} \Box_{nom}^{\text{NP}}
\]

This is a very simple solution, comparable to the list of features of the minimalist analysis above. It does not require any extension to our basic set-up. This option allows the descriptive grammarian to differentiate between the behaviour of individual features with respect to their distribution in phrase structure. We can thus have a collection of distribution postulates that express the ways in which the morphosyntactic properties distribute (per feature and construction mode). Some may act as head features (identity between the whole and the head daughter) others as agreement features (identity between the whole and daughters) depending on the kind of construction that is involved. This separation provides one more way to factorise the information in a grammar.

Without further structural rules, the modal decorations cannot permute. This makes it possible use the order in which the features appear in the string of modal operators. We provide an example in the next chapter.

Obviously, it is possible to mix the two options and to group together in one complex mode all the features that behave in the same way ('head' vs. 'agreement', or 'government' features for instance). There are many alternatives here, and the choices may be different for individual grammars. Different features and different constructions might require different behaviour in this respect. We will provide several illustrations of the various possibilities in the next chapter.

**Conclusion**

In this chapter we have proposed a system of unary modal operators to account for the morphosyntactic aspects of linguistic expressions in the spirit of resource-sensitive grammars. Such an system has the following characteristics.

The decomposition of categorial information that is usually taken care of by means of feature structures, is implemented by modal decorations.
The base logic for the unary operators defines a feature checking mechanism that accounts for the elementary matching of morphosyntactic information between different parts in a linguistic structure.

Inclusion postulates are used to define an information ordering on the morphosyntactic features, thereby providing a mechanism for underspecification of lexical assignments. Feature distribution principles are defined by means of distribution postulates that allow feature checkers to move through phrase structure. Fine-grained distribution rules can be defined that are sensitive to both the type of feature modality and to the mode of composition.

We have also shown that we do not need variable sharing or re-entrancy to define distributional information. Structural rules can define a regime of feature percolation that enforces the sharing of information, or more precisely, that identical information is checked in different parts of the structure. The specification of morphosyntactic properties as modal operators, governed by their own logic, allows us to factor the information in a grammar. It is possible to fix the distributional behaviour of each property or group of properties in a separate component of the grammar (the structural rule package). Not only does this have advantages for expressing generalisations, it also makes the distribution of information less dependent on the function/argument structure of the functional categories $A/B, A\backslash B$ and more on the mode of composition.

So far we have only presented the basic machinery and illustrated its use. In the next chapter we provide further illustrations, propose some refinements and alternatives that deal with underspecification and distribution for linguistic fragments in which selectional and other dependencies may be different. We also discuss the motivation for choosing between the options for particular fragments.
8

The Use of Feature Checking

In the previous chapter we have discussed the outlines of the way in which unary residuated operators can be used to add morphosyntactic decorations to the familiar category structure and how inclusion and distribution postulates may take care of the hierarchical ordering (underspecification) and feature distribution. In this companion chapter we want to point out further refinements to these proposals and motivate their use by looking at some basic grammar fragments that illustrate the effect of the different options for different types of constructions and feature-checking configurations. Important parameters in this respect are the interaction between categorial selection, underspecification and differences in distributional behaviour (feature checking in local trees or in non-local contexts for instance).

In Section 8.1, we concentrate on the linguistic aspects of the logic governing the unary connectives, looking at the options for stating the polarity opposition between checker and checkee that is needed in our account of morphosyntactic description. Given that there are various options, the question is how the modal decorations should be anchored in the lexical assignments given that they will appear in different structural configurations and dependencies and will have different distribution requirements.

A major part of the discussion involves the interaction between the unary (morphosyntactic) modalities and the binary (syntactic) connectives. The latter are important in this respect because they give rise to polarity differences and because the construction modes interact with those for feature checking through the distribution principles.

In Section 8.2, we turn to the decorations on the logical operators and discuss the structure of the resource management modes. We have already discussed the option of either choosing for a restricted set of modal operators with complex indices or having simplex indices with multiple operators in the previous chapter. Now we will discuss this in more detail and illustrate this issue means of several fragments.

8.1 Checking and Polarity

We divide this section into two parts. We start with a presentation of the technical options for implementing the modal checking procedure. This is followed by a series of analyses of small fragments to show the type of motivation that determines the choice between the various options.
8.1.1 Logical Options for Checking

**Polarities** The analysis of morphosyntactic description which we outlined in the previous chapter is framed as a resource-conscious, derivational feature checking theory. Morphosyntactic markings have to be checked parallel to the duality between selection and composition for syntactic construction. The logical core for both is given by the logic of residuation. In order for a construction to be grammatical on the morphosyntactic level there has to exist a balance between decorations of opposite polarities in the same way as there has to be a balance between selection and composition operations on the syntactic level of linguistic description. In the examples so far, the checking modality (indicated as $\Box^-$ below) was either provided as a decoration on the domain subcategory of a functor or as a decoration on the succedent (goal) category:

$$\Box^+ A \circ (\Box^- A) \nmid B \Rightarrow B,$$

$$\Gamma \Rightarrow \Box^- A$$

This does not, however, cover all the options. In the cases above, we have used pairs of the same operator $\Box$ in the sequent each in an opposite position. However, it is also possible to exploit the opposition between the operators. Schematically, this can be represented as follows.

$$\Box A \circ \Diamond (A \nmid B) \Rightarrow B$$

Here are the essential steps in a derivation for this sequent.

$$\langle \Box A \circ A \nmid B \Rightarrow B \rangle \Rightarrow B$$

$$\Box A \circ \langle A \mid B \Rightarrow B \rangle$$

$$\Box A \circ \Diamond (A \nmid B) \Rightarrow B$$

In this derivation we have used a distribution postulate that transfers the checking configuration from one daughter to another.

$$\Gamma[\langle \Delta_1 \circ \Delta_2 \rangle \Rightarrow C]$$

$$\Gamma[\Delta_1 \circ \langle \Delta_2 \rangle] \Rightarrow C$$

Note that with this postulate, it is also possible to derive $\Diamond (A \nmid B) \vdash \Box A \nmid B$.

We will provide some fragments below that make use of this type of distribution postulate.

**Structural behaviour** Our modal analysis of morphosyntactic information made use of the residuated operators $\Diamond$ and $\Box$ with the semantics and rules of inference as below.

$$A \rightarrow \Box_i B \text{ iff } \Diamond_i A \rightarrow B$$
We have used the logic of these operators to define feature cancellation in a linguistic setting. It is, however, also possible to consider more specific structural resource management options for the unary modalities that might be of interest for their function as features. Here we will consider reflexivity (T) of the modalities.

In (condensed) axiomatic form, this behaviour can be formulated as follows.

\[(T) \quad [i]A \rightarrow A \rightarrow \langle i \rangle A\]

Semantically, the frame constraints on the accessibility relation can be written down as \(R_i(x, x)\) for the reflexive case. This allows us to derive sequents of the following form.

\[\Box_i A \Rightarrow A\]

The introduction of this type of sequents has important effects on the resource-sensitive character of the feature modality, because now markings can disappear without being checked explicitly (or checkers can appear out of the blue, so to speak). If we assume that the feature modalities show this behaviour then this will influence the type assignments to the lexical items, of course. We will illustrate the use of these reflexive modalities to mark underspecification below.

In Versmissen (1996) modal decorations with reflexive behaviour are used as well. He makes use of the possibility of compiling this behaviour in the familiar residuated modalities \(\Diamond\) and \(\Box\). It is easily seen that if we choose a feature decoration \(\langle i \rangle [i]\) instead of simply \([i]\), we get the reflexive behaviour of this feature complex by the logical rule: \(\langle i \rangle [i]C \rightarrow C\).

### 8.1.2 Lexical Anchoring of Checking Configurations

The feature checking procedure is defined in terms of the application of logical rules. This requires a matching of operators of different polarities. We will now provide a number of fragments that show some of the possible variation in implementing the required opposition.

Ultimately, the morphosyntactic operators have to be provided by the lexicon in the category assignments to words. The specific choice of which operator to use and at what position in a category depends on the morphosyntactic paradigm (which forms can be underspecified), and the selectional or compositional dependencies of the words.

The polarity issue is tightly connected to the structure of complex categories because the binary operators define positions that differ with respect to polarity: \(+/\,-, + \bullet +, -\backslash +\). This has important effects on the interaction between (i) the inclusion relations (overspecification or underspecification), (ii) the feature checking logic and (iii) the distribution of features in the syntactic structure. Feature distribution is regulated through the interplay of two mechanisms:
• The decorations on functional categories and arguments.
• The distribution principles.

We will illustrate the options below. First we look at basic feature distribution configurations in local trees and the distinction between endocentric and exocentric constructions. Next, we turn to subject-verb agreement (combining exocentric projection with agreement) and basic clitic constructions. These examples focus on variation in expressing distribution in interaction with underspecification. The last fragment of this section involves the use of reflexive modal operators for underspecification.

**Local trees** The relation between the morphosyntactic decorations in phrasal structures is only in part determined by the logic of combination and selection expressed by $\cdot, \cdot, \\cdot$. The inclusion and distribution postulates make it possible to define further options for manipulation.

As far as distribution is concerned, the distribution of morphosyntactic features relies on the structures licensed by the binary connectives and more particularly on the modes of combination. We start with a schematic representation of the basic combination of a functor and an argument, to see how checker-checkee relations are defined in such structures.

$$[i]A \Rightarrow [l]D \Rightarrow [k]C \Rightarrow [j]B$$
$$[i]A/ [j]B \circ [k]C \Rightarrow [l]D$$

In linguistic terms, this schema tells us that the morphosyntactic properties of the complex expression $\square_i$ must include all the properties expressed on the range part of the functor $\square_i$. In terms of the phrase-structure configuration, we could say that the functor daughter (or the range part) must match its features with those of the mother (a kind of projection). We could call these properties the head features, and the functor daughter the head daughter. The head features appear on the range subcategory whereas the features on the argument daughter are checked by the features on the domain part of the functor.

In terms of specificity, we see that the head features on the mother must be the same as, or more specific than, those on (the range part of) the head daughter, whereas the features on the argument must be the same as, or less specific than, the features on the domain part of the functor.

In Bach (1983b) a distinction between three types of functors is made that is related to the classic (Bloomfieldian) distinction between endocentric and exocentric constructions and the structures defined by X-bar theory. Exocentric functors project other properties than what they select for, so they are of (the schematic) type $X/Y$ ($Y\backslash X$). Typically, heads in head-complement constructions are of this type. Endocentric functors project the same properties as they select for, so they are of type $X/X$ (or $X\backslash X$). Modifiers are typically endocentric. It is possible to refine this classification, by decomposing the information in categories and looking at each
component separately. Specifiers, for instance, are typically of the form XP/X. Instead of saying that they are exocentric (because the domain and the range part of the functor are different), we can say that they are endocentric with respect to the major category (the X part) and exocentric with respect to 'bar' level.

This distinction is also reflected in the morphosyntactic properties. Bach argues that functors that are exocentric with respect to syntactic information are also typically exocentric with regards to morphosyntactic information, whereas endocentric categories (including specifiers) also behave endocentrically with respect to morphosyntax. The typical schematic endocentric type would be of one of the following forms (modifier or specifier):

\[[i]P[/i]X or [i]XP[/i]X\]

We have already seen in the previous chapters why this endocentric schema does not work in type-logical grammars when underspecification of the modes is used to reduce lexical assignments. This concerns cases where co-variation is required like the *guten Frau* or *the boy - boys* examples. The endocentric morphosyntactic behaviour can be encoded by decorating the whole category with the modal operator instead of both the domain and range separately. Schematically, this takes the form:

\[[i](XP/\text{spec}X)\]

The same goes for completely endocentric categories like modifiers.

\[[i](X/\text{mod}X)\]

The appropriate checking and percolation is now regulated by means of distribution postulates. The combination of the postulates and the placement of the morphosyntactic decoration ensure agreement, in this sense taking over the function of re-entrancy in the unification-based formalisms.

By defining feature distribution principles as interaction principles we become less dependent on the logical function/argument structure of the categories as we can rely on the sortal decorations of the composition operator. In other words, what becomes important for feature distribution are the sortal (resource management) modes on the composition connective in combination with the specific features.

Besides the classification of constructions and categories as endocentric, exocentric and head, specifier and modifier, there are also other distribution facts that have to be accounted for. We will now consider some further examples.
Subject-Verb Agreement  Consider first the case of a subject-verb combination. In this case the exocentric head agrees with the subject argument. Adapting the principles of feature distribution and construction type that we presented above, we have two options to describe this. The first is to treat the subject selection similar to other complements as a head-complement combination. The schematic form of the verb then looks like this (the brackets are only meant to emphasise the scope of the unary modality):

\[(i|NP)\backslash S\]

The second option involves treating the agreeing morphological features in the same way as the features in a modifier or specifier construction.

\[(i)(NP\backslash S)\]

If we assume the first analysis then we have to take into account the asymmetry between argument and domain in the derivability relation: the subject must be at least as specific as is required by the verb:

\[
\begin{align*}
[i]NP & \Rightarrow [j]NP \\
[j]NP \Rightarrow S & \Rightarrow S
\end{align*}
\]

In case a verb is not marked for certain agreement features (underspecified), then we can use overspecified modes (as we have illustrated before) to define the appropriate checking.

Now consider the case where we want certain properties of the verb to agree with the subject and also to percolate to the mother. In this case, we have to take the same precautions about distribution and underspecification as we did with the endocentric specifier and modifier constructions.

\[
\begin{align*}
[i]NP \circ_c [j](NP\backslash cS) & \Rightarrow S \\
[i]NP \circ_c [j](NP\backslash cS) & \Rightarrow S
\end{align*}
\]

For this derivation to succeed we also need inclusion relations as follows.

\[
\begin{align*}
[i]A & \Rightarrow [k]A \\
[j]A & \Rightarrow [k]A
\end{align*}
\]

When we look at underspecification in the second analysis, we see that both the subject and the verb can be underspecified for certain features. The distribution postulate will enforce agreement. It should be noted that the schematic derivation is slightly simplified: in general there could also be inclusion postulates refining the mode [k] to some mode [k'] that is distributed over the parts.
Another complication in this respect arises in the the case of a transitive verb. We have to take care that the subject-verb agreement features are checked only on the verb and the subject but not on the object. For this we need a distribution postulate like the following.

$$k_1 \langle i \rangle (A \bullet_{h-c} B) \rightarrow \langle i \rangle A \bullet_{h-c} B$$

This postulate transfers the checking of the whole configuration to the checking of one part, the head. In this case, the head is to the left. For right-headed constructions it will be necessary to add another postulate. A distinction should therefore be made between the two composition modes (say $c-h$ and $h-c$ for complement-head and head-complement, respectively).

Schematically, a derivation of this type proceeds along the following lines.

$$\begin{align*}
\langle \text{SUBJ} \rangle_i \circ_{c-h} (\langle \text{VERB} \rangle_i \circ_{h-c} \text{OBJ}) & \Rightarrow S \\
\langle \text{SUBJ} \rangle_i \circ_{c-h} ((\langle \text{VERB} \circ_{h-c} \text{OBJ} \rangle)_i & \Rightarrow S \\
\langle \text{SUBJ} \circ_{c-h} (\text{VERB} \circ_{h-c} \text{OBJ}) \rangle & \Rightarrow_{[i]} S \\
\text{SUBJ} \circ_{c-h} (\text{VERB} \circ_{h-c} \text{OBJ}) & \Rightarrow_{[i]} S \\
K_1 & R\Box
\end{align*}$$

It is also important to note that in the endocentric analysis of the subject-verb agreement, the features that check the subject and the verb percolate to the mother. For each constellation of features that has to be checked we need a checker: for a box (or group of boxes) in positive positions we need boxes in negative positions, as we indicated in the previous section. In the analysis of clitics below we will develop an alternative for this. We will also provide another exocentric analysis of the subject-verb combination.

**Clitics (1)** We can illustrate further possibilities of the multimodal approach to feature checking by looking at possible ways to treat some aspects of agreement involved in (French) clitic constructions.

<table>
<thead>
<tr>
<th>Clitic</th>
<th>English</th>
<th>French</th>
<th>Analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>elle</td>
<td>she</td>
<td>NP</td>
<td>$\text{NP}/\text{NP}$</td>
</tr>
<tr>
<td>les</td>
<td>them</td>
<td>$\text{NP}/\text{NP}$</td>
<td>$\text{NP}/\text{NP}$</td>
</tr>
<tr>
<td>fume</td>
<td>smokes</td>
<td>$\text{NP}/\text{NP}$</td>
<td>$\text{NP}/\text{NP}$</td>
</tr>
</tbody>
</table>

In analyses of clitics, we want a single entry for transitive verbs that is appropriate for combinations with full object noun phrases and with clitic objects. The clitic is therefore assigned a higher order type (see Kraak (1998), for instance). Here, we are interested in what happens to the agreement features. The object clitic *les* combines with a transitive verb and then with the subject. It is possible to let the subject-verb agreement be fixed through the category of the clitic. But this means that we would have to double the assignments to the clitics.
1.52

CHAPTER B

tes ([ss]Nn\slash )/(([ss]rur\slash )Nr)
tes ([pl]\slash )/(([pl]Nn\slash )Nn)

Note that the generic ([Num]\slash )/(([Num]\slash )/[pl]\slash ) does not work, because this makes the ungrammatical 'ellesg les fument_pl' or 'elles_pl les fume_sg' derivable.

We will present two analyses making use of distribution postulates that can reduce the assignments to a single one. In the first case we assume that the agreement features project to the sentence. In the second case we will make use of the duality between the operators $\diamond$ and $\Box$ to define the polarity opposition.

For the first analysis we assume binary modes $sh$ (subject-head), $ch$ (clitic-head) and $ho$ (head-object).

<table>
<thead>
<tr>
<th>Word</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>elle</td>
<td>[sg]NP</td>
</tr>
<tr>
<td>les</td>
<td>([np]\slash sh)\slash ch([np]\slash sh)\slash ho \slash NP</td>
</tr>
<tr>
<td>fume</td>
<td>[sg]([np]\slash sh)\slash ho \slash NP</td>
</tr>
</tbody>
</table>

We can see how these distribution principles guide the checker feature to the appropriate positions for checking in the following schematic derivation.

\[
\begin{align*}
\langle elle \rangle_{sg} \circ_{sh} \langle les \circ_{ch} \langle fume \rangle_{sg} \rangle_{sg} & \Rightarrow s & K_2 \\
\langle elle \rangle_{sg} \circ_{sh} \langle \langle les \circ_{ch} fume \rangle_{sg} \rangle_{sg} & \Rightarrow s & K \\
\langle elle \circ_{sh} \langle les \circ_{ch} fume \rangle \rangle_{sg} & \Rightarrow s & R \Box \\
elle \circ_{sh} \langle les \circ_{ch} fume \rangle & \Rightarrow [sg]S &
\end{align*}
\]

**Clitics (2)** We will now present a second analysis of the example to illustrate some further points. First, the type of analysis of the clitic construction above requires us to percolate the agreement information to the mother's node. This may not always be desirable. Secondly, the analyses presented above show a tendency to put morphosyntactic information on the outermost position of any category, including functor categories. This is not just a technical matter to make the distribution work, but it can also be seen as a reflex of a general idea that the morphosyntactic information refers to properties of the expression as such and not to information about the arguments an expression takes or the expression it is a part of. This echoes part of the motivation of Bouma (1993) to let morphosyntactic features appear on complex categories. Carrying this idea further would result in putting morphosyntactic information in front of all the other categorial information for all words.
A major problem with this idea is, however, that we no longer have symmetrical checker - checkee oppositions. In all the examples above we put for every modal decoration in positive position a corresponding modal decoration in negative position. To solve the problem we can make use of the fact that the polarity oppositions can also be expressed in a different way, namely by introducing diamond decorations next to box decorations.

The basic procedure thus consists in defining positive and negative feature modes by the duality between the $\Diamond$ and $\Box$ modalities. The cancellation of features proceeds as before, but in this case the distribution behaviour is completely determined by the interaction of the unary and binary modes. From this perspective, the grammar writer has to decide on two things:

- Choose between positive and negative markings of the feature information.
- Account for the correct distribution of the modes in phrase structure by distribution postulates.

In our clitic example, we limit the decorations to a number feature for the subject, for expository purposes. We could make it slightly more complex by also adding the number information for the object. Obviously, the clitic is plural and because the verb does not care about the number of the object we leave this requirement on the verb underspecified. Note that we are not forced to leave this underspecified. There are constructions and languages with agreement between verb and object agreement which has to be accounted for.

The lexical assignments for the nominal expressions can be fixed as follows:

$$\text{elle} \quad [sg|NP]$$
$$\text{les} \quad [pl]((NP|S)/((NP|S)/NP))$$

The syntactic part of the category for the verb is the familiar one for transitive verbs: $(NP\backslash S)/NP$. Because we have to take care that both the subject and the object noun phrase are checked for number, we have to mark the verb for both and distinguish between the two.

$$\text{fume} \quad \langle s : sg\rangle\langle o : num\rangle((NP\backslash S)/NP)$$

Of course, we have to define the distribution correctly. We therefore decorate the binary operators with phrasal sorts as in the previous analysis.

$$(\langle\text{elle}\rangle sg \circ_{sh} (\langle\text{les}\rangle o_{num} \circ_{ch} \text{fume}) \Rightarrow s)$$
$$(\langle\text{elle}\rangle sg \circ_{sh} (\text{les} \circ_{ch} (\text{fume}) o_{num}) \Rightarrow s)$$
$$(\langle\text{elle}\rangle sg \circ_{sh} (\text{les} \circ_{ch} o_{num} \text{fume}) \Rightarrow s)$$
$$(\text{elle} \circ_{sh} ((\text{les} \circ_{ch} o_{num} \text{fume}) s_{sg}) \Rightarrow s)$$
$$(\text{elle} \circ_{sh} (\text{les} \circ_{ch} (s : sg) o_{num} \text{fume}) \Rightarrow s)$$
$$(\text{elle} \circ_{sh} (\text{les} \circ_{ch} (s : sg)(o : num) \text{fume}) \Rightarrow s)$$
The derivation uses the following distribution postulates:

\[ A \cdot_{ch} (s : sg) B \rightarrow (s : sg)(A \cdot_{ch} B) \]
\[ A \cdot_{sh} (s : sg) B \rightarrow (s : sg)A \cdot_{sh} B \]
\[ A \cdot_{ch} (o : sg) B \rightarrow (o : sg)A \cdot_{ch} B \]

The advantage of this way of fixing opposite polarities is that the relation between morphosyntactic distribution and selectional requirements is loosened even further. In fact, in this case it is no longer required to put checking features on the argument position of functional categories or on the goal sequent. This means that it is not so much the logic of the binary connectives nor the functor-argument structure of the categories in a sequent that determine the checker-checkee relations but rather the mode of composition.

We can summarise the characteristics of this approach as follows.

- Categories that are not functors can carry checking modalities.
- Categories can be distinguished that differ only with respect to the modes labeling the binary connectives to trigger different patterns of feature distribution.
- Morphosyntactic features of lexical items can be specified on the top level of functor categories instead of on the domain or the range subcategory (for motivation see Bouma (1993)).
- The use of distribution postulates that allow checkers to travel through the structure guided by the modes of construction, allows for non-local checking.

**A note on phrasal sorts** Although we focus on the morphosyntactic features, it will have become clear from the examples above that the sortal decorations on the composition and selection operators are important to define the adequate distribution principles as well. In a realistic grammar the phrasal sorts will be structured themselves into cross-cutting hierarchies. The examples suggest some of the parameters along which they may be classified: endocentric versus exocentric, left-headed versus right-headed, grammatical role of arguments (subject, object,...), phonology and prosody (clitic - host versus complement - functor), lexical versus phrasal (verb-clusters) etc.

The distribution principles as we have specified them state the distribution of individual morphosyntactic features along individual phrasal sorts. It will be clear from the examples above that several generalisations can be made that group together sets of features and collections of phrasal sorts as sharing the same behaviour.

So far, we have looked at some options for stating the checker-checkee duality and how this affects the distribution and underspecification relations. We finish this section on the variation in logical checking operations
by changing the structural properties of the modal operators and looking at the effect on the specification of underspecified information.

**Reflexive modals and Underspecification** Let us now see what happens if we change the structural behaviour of the modal operators. We consider the case of the reflexive operator. From the sequent rule we can see what its potential is.

\[
\frac{\Gamma[(\Delta)_i] \Rightarrow C}{\Gamma[\Delta] \Rightarrow C} T
\]

Reading the rule bottom-up we see that we can introduce a checking configuration somehow out of the blue in the antecedent. This disturbs the resource-sensitivity somewhat and care has to be taken with respect to the lexical anchoring of modalities to ensure correct grammars. Note that the situation is not symmetric: to remove the checking configuration we have to apply a logical rule. In this case we have to use the \( R \) right rule, but this requires the presence of a boxed formula. The typical behaviour of the feature modes as we have used them so far is to have a functor \( X/\{i\} Y \) that looks for an argument with the appropriate markings \( \{i\} Y \), resulting in a step in the derivation like \( \{i\} Y \Rightarrow \{i\} Y \). The feature on the argument is checked by a left \( \{i\} \) rule followed by a right \( \{i\} \) rule. In any case, the occurrence of the feature on the argument requires a matching occurrence of opposite polarity: a checkee needs a checker. When we allow reflexive operators this is no longer the case. We illustrate this with a simple example.

The simple lexicon contains the following entries, where \( IV \) abbreviates \( NP \)'s (with the appropriate decorations).

- cuts: \( IV/\{acc\} NP \)
- them: \( [pl]/\{acc\} NP \)

Note that the verb does not provide feature decorations to check the number feature of the object complement. The derivation for the verb phrase *cuts them* shows the use of the reflexive modal to mark underspecification. As an alternative we could also have used an underspecified mode. We only show the relevant premise of the derivation.

\[
\begin{align*}
[acc] NP \Rightarrow [acc] NP \\
(\langle [pl]/\{acc\} NP \rangle_{pl} \Rightarrow [acc] NP, R \Box) \\
[pl]/\{acc\} NP \Rightarrow [acc] NP, T
\end{align*}
\]

The use of feature modalities that display this reflexive behaviour would typically be useful for those expressions that must be marked for certain properties in certain contexts, but that can also appear in contexts where there is no explicit checker.
In this section, we have shown different options to define checking configurations and we have indicated the kind of motivation that influences the choice in different linguistic contexts.

In the next section we turn to the structure of the indices on the logical operators.

8.2 Sortal Structure

We have now presented various ways to specify checker-checkee relations and discussed the interaction of these options with underspecification and feature distribution. Whereas we focused on the logical behaviour of the modal operations before, we will now turn to the structure of the sorts that index the modal operators.

In Chapter 7, we proposed two strategies to encode multiple morphosyntactic attributes. It is possible to stack multiple operators each representing a single attribute, or to use a single operator and use complex structures for the indices. Of course, one could also use multiple modal operators with complex indices. In this section, we will illustrate the various options and point out some of the motivation that influences the choice in a descriptive linguistic setting. In the first part we focus on the choice between simplex indices on multiple modalities versus complex indices on a single modality. In the second part we focus on the possible structures of complex indices.

8.2.1 Multiple Attributes

In most of the examples above, we have tried to keep the grammar simple for expository purposes by restricting the number of morphosyntactic properties under consideration and by grouping together multiple properties as complex indices on single modal operators. This has not been a matter of principle but rather a matter of presentation. In this section we want to point out a number of situations in which a choice in favour of distributing information over different operators may be desirable.

Distribution Distribution principles as we have used them are defined with respect to specific unary and binary modes. If two properties behave differently with respect to distribution they should be spread over different modal operators (at some point). As an illustration of this idea we could elaborate on the endocentric analysis of subject-verb agreement by refining the morphosyntactic information on the verb by the attribute TENSE.

In the endocentric analysis, the nominal features that are shared by the verb and the subject are projected onto the mother. Suppose we want the verbal attribute TENSE to project to the mother as well but do not want it present on typical nominal expressions. There are several ways to achieve this.
\[ [agr]_{NP} \circ [agr]_{(NP \setminus tns)S} \Rightarrow [agr]_{tns} \circ [agr]_{NP} \circ [agr]_{(NP \setminus S) \Rightarrow [agr]_{tns}S} \]

In the first case we assume that \( agr \) distributes over both parts of the subject-verb combination to check the features on the subject and the verb. After the cancellation of the \( NP \) complement on the verb, the features on the range (s) of the verbal category match the features on the sentence. In the second case, we must define different distribution principles for \( agr \) and \( tns \) in order to take care that the first feature distributes over both parts and the latter only checks the verbal head.

**Ordering** One of the issues that arises when multiple modal operators are used is that of their linear ordering. It is possible, when writing a grammar to use the same consistent ordering which can be fixed in a signature. One can also define general or specific permutation options by postulates like the following.

\[ \langle i \rangle \langle j \rangle A \leftrightarrow \langle j \rangle \langle i \rangle A \]

This offers the grammar writer some freedom of expression but may lead to unwanted growth of the search space in the context of automatic theorem provers for this type of grammar logics. Although these matters of implementation are certainly important, they are as such not vital to the descriptive grammarian who is interested in the ordering insofar as it distinguishes grammatical from ungrammatical structures.

An illustration of such an application of the multi-modal framework can be found in reconstructions of the minimalist program. In the previous chapter we mentioned the parallel between the idea of feature checking in the multimodal framework and the minimalist program. In type-logical reconstructions of the minimalist program (Heylen (1997d)) we can decompose the operation of *move* into a combination of structural rules and a feature checking procedure. We will here present a rather rough reconstruction which captures the general idea but not the precise detail of the move operation as we have defined it. We first repeat the simple version of move and then provide the postulates of the reconstruction.

\[
(+i\Delta_1 \circ (\Delta_2 \circ (-i\Delta_3 \circ \Delta_4))) \text{ move } ((\Delta_3 \circ \Delta_4) \circ (\Delta_1 \circ \Delta_2))
\]

\[
\begin{align*}
+ & \ A \bullet (+i)B \rightarrow (+i)B \bullet A \\
- & \ (-i)A \bullet B \rightarrow A \bullet (-i)B \\
I & \ (+i)A \rightarrow (-i)A
\end{align*}
\]

This reconstruction captures the following aspects of *move*. The first postulate allows movement to the left of structures that are marked for some (licensor) feature \(+i\). The second postulate takes care of positioning a marker \(-i\) from left-daughter to right-daughter so that checking can take
place. The third postulate changes $+i$ markings into $-i$ ones. A more precise implementation should also take into consideration the sortal structure (headedness) of the composition relation to constrain the operations. There should also be a postulate that transfers markings from heads to their maximal projections.

What is more important, however, is that in this type of grammar the feature-checking and feature-manipulation operations have structural effects. In the case of multiple modal operators the ordering of the modal operators is important in fixing the linear ordering of expressions. Consider, for instance, the following tiny grammar where the checking of the unary modalities determines to a large extent the ordering.

\[
\begin{align*}
\text{she} & \quad [-i]NP \\
\text{sinks} & \quad (+i)(S/NP)
\end{align*}
\]

\[
\begin{array}{c}
S/NP \circ \text{NP} \Rightarrow S \\
S/NP \circ ([-i]NP) \Rightarrow S \quad \text{L} \\
(S/NP)_{-i} \circ [-i]NP \Rightarrow S \quad I \\
(S/NP)_{+i} \circ [-i]NP \Rightarrow S \\
[-i]NP \circ (S/NP)_{+i} \Rightarrow S \quad \text{L} \\
[-i]NP \circ (+i)(S/NP) \Rightarrow S
\end{array}
\]

### 8.2.2 Internal structure

Our discussion in this section centers around a small example of agreement within the Dutch noun phrase (see also Heylen (1997a), Heylen (1997c)). In Part II we discussed how Johnson and Bayer (1995), defend the view that a theory modelling agreement phenomena in terms of the requirement that arguments must be *subsumed* by, or logically imply, the corresponding argument specification of a predicate or functor category, is superior to a theory that assumes *unification* (see also Heylen (1996b)). Bouma (in postings to the CG-mailing list, January 1997) challenges this position by arguing that some agreement phenomena in Dutch cannot be treated in a subsumption-based setting without missing generalisations. Bouma is right as far as a Bayer and Johnson type treatment is concerned. In Heylen (1997a) we took up Bouma’s challenge and provided a subsumption-based analysis of the Dutch constructions using the mixed multimodal calculus in which the generalisations are not lost. The re-entrancies that are required in the unification-based analysis are replaced by distribution postulates, similar to the examples we have provided earlier. Here, we will repeat this analysis, but now we focus on the internal structure of the modal decorations.

We first present the aspects of Dutch agreement in the noun phrase that we will be concerned with. Next, we present the type-logical analysis and
then discuss the structure of the decorations.

**Agreement** The paradigm of Dutch agreement phenomena that concerns us here is illustrated by the following data.

- *de deur* 'the door'
- *het huis* 'the house'
- *de goede deur* 'the good door'
- *een goede deur* 'a good door'
- *het goede huis* 'the good house'
- *een goed huis* 'a good house'

Dutch nouns bear grammatical gender. This is reflected in the choice of the definite determiner (*de* or *het*). Neuter nouns combine with the definite determiner *het*, non-neuter nouns with *de*. The indefinite determiner *een* can combine with both. As the above examples show, the form of the adjective varies with the context in a particular way. If the determiner is indefinite and the noun is neuter, the adjective is not inflected. In all other cases the adjective is inflected.

<table>
<thead>
<tr>
<th></th>
<th>Neuter (h)</th>
<th>Non-neuter (d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definite (b)</td>
<td>-e</td>
<td>-e</td>
</tr>
<tr>
<td><em>het goede huis</em></td>
<td><em>de goede deur</em></td>
<td></td>
</tr>
<tr>
<td>Indefinite (o)</td>
<td>∅</td>
<td>-e</td>
</tr>
<tr>
<td><em>een goed huis</em></td>
<td><em>een goede deur</em></td>
<td></td>
</tr>
</tbody>
</table>

**First Analysis** We first provide a failing attempt to analyse this fragment to point out the peculiarity of this construction. In this analysis we use two separate modal operators for the gender and the definiteness distinction.

The modal operator for gender is indexed by the sorts *d* and *h* (for non-neuter, combining with the determiner *de*, and neuter, combining with the determiner *het*) and *Gndr* and *gndr* for the overspecified and the underspecified mode, respectively. The modal operator for definiteness is indexed by the sorts *o* and *b* (for indefinite, Dutch *onbepaald*, and definite, Dutch *bepaald* respectively) and *Dness* and *dness* for the overspecified and underspecified mode. We can summarise the inclusion relations as follows.

\[ [\text{gndr}]A \Rightarrow [i]A \Rightarrow [\text{Gndr}]A \quad (i \in \{d, h\}) \]
\[ [\text{dness}]A \Rightarrow [i]A \Rightarrow [\text{Dness}]A \quad (i \in \{o, b\}) \]

We assume as before that all specific features can distribute over modifier-head structures but no underspecified or overspecified mode can do so.

\[ (i)(A \bullet B) \rightarrow (i)A \bullet (i)B \quad (i \in \{o, b, h, d\}) \]

We assume furthermore the following assignments.
The problem with this analysis is that it does not seem possible to reduce the number of assignments to \textit{goede} to just one. Although both \([h]\) and \([d]\) are possible values for gender and \([b]\) and \([o]\) are possible values for definiteness, it is not appropriate to assign \textit{goede} the overspecified type: \([Gndr][Dness](N/N)\). Because both overspecified features may be instantiated independently this can give rise to the instantiation \([h][o](N/N)\) which we should not allow. This assignment makes it possible to derive the ungrammatical \textit{een goedehuis}. The combination of \textit{een} \((NP/[Gndr][o]N)\) with \textit{goede huis} is possible if we can show that \textit{goede huis} is in the type \([gndr][o]N\).

This makes it possible to derive the ungrammatical \textit{een goedehuis}. The combination of \textit{een} \((NP/[Gndr][o]N)\) with \textit{goede huis} is possible if we can show that \textit{goede huis} is in the type \([gndr][o]N\).

We provide part of the derivation.

\[
\begin{align*}
\text{goede} \circ \text{huis} & \Rightarrow [o]N \\
\langle \text{goede}\rangle_{\text{Gndr}} \circ \langle \text{huis}\rangle_{\text{h}} & \Rightarrow [o]N & L \Box \\
\langle \text{goede}\rangle_{\text{h}} \circ \langle \text{huis}\rangle_{\text{h}} & \Rightarrow [o]N & I \\
\langle \text{goede} \circ \text{huis}\rangle_{\text{h}} & \Rightarrow [o]N & K \\
\langle \text{goede} \circ \text{huis}\rangle_{\text{gndr}} & \Rightarrow [gndr][o]N & I \\
\text{goede} \circ \text{huis} & \Rightarrow [gndr][o]N & R \Box 
\end{align*}
\]

If we look closely at this example we see that the morphosyntactic properties of the category corresponding to the adjective \([Gndr]\) and \([Dness]\) are 'removed' or 'checked' by the modes \([h]\) and \([o]\) respectively. But this combination of the modes has to be excluded as it corresponds to the assignment \([h][o](N/N)\) to \textit{goede}. The problem with the analysis thus seems to be that the generalisations expressed by the two properties are not independent.

**Second Analysis** We now propose a multimodal analysis of this construction that combines the information on both morphosyntactic properties into a single modal operator and we provide modes and lexical entries that are more successful. We then look at the logical structure of the relations between the modes. Our analysis makes use of the following elements.

1. A selection of resource modes on unary modalities is used to express morphosyntactic information: \(d, h, o, db, do, hb, ho, -ho\).
2. Inclusion postulates define an ordering on the feature modes. The inclusion postulates are summarised by the following graph. When two modes are connected by a line, then the one on top \((i)\) and the one below \((j)\) are related by an inclusion postulate: \([i]A \Rightarrow [j]A\).
3. Interaction postulates express the distribution of features over combination modes. In this case the specific (resolved) modes that distribute are: \( db, do, hb \) and \( ho \).

4. Lexical assignments are as follows.

\[
\begin{array}{c|c|c|c}
\text{de} & \text{NP}/[db]N & \text{een} & \text{NP}/[o]N \\
deur & [d]N & \text{het} & \text{NP}/[hb]N \\
\text{goed} & [ho](N/N) & \text{huis} & [h]N \\
\text{goede} & [-ho](N/N) & & \\
\end{array}
\]

For the time being, the resource modes should all be considered atomic, although their names seem to suggest otherwise. This serves mnemonic purposes only. Let us review the features, alias sorts, used briefly.

\([d]\) is used for a word like \( deur \), to mark its grammatical non-neuter gender: it combines with the definite determiner \( de \). Similarly, \([h]\) marks neuter words, those that combine with the definite determiner \( het \).

The definite determiners require their complements to be ‘definite’ and have the appropriate gender. As before, we use \( b \) for definite and \( o \) for indefinite. \([db]\) is the requirement put on nouns and their projections by the definite non-neuter determiner \( de \), \([hb]\) is the requirement put on nouns by the definite neuter determiner \( het \).

The adjectives divide into inflected and uninflected ones. The latter type occurs only in indefinite neuter environments and is marked \([ho]\) whereas the former occurs in all other environments, hence \([-ho]\), to be thought of as “not neuter and indefinite”.

With the assignments and postulates as above we cannot derive the ungrammatical *een goede huis. The adjective-noun combination goede huis is not in type \([o]N\) as required by the type for een, as can be seen from the following partial partial derivation.
The crucial step in this derivation involves the choice of the mode that distributes over the adjective and the noun. At this step, marked (*) in the derivation, we have the choice to instantiate the general checking mode $o$ to either $do$ or $ho$. In this derivation we chose $do$. Neither this option nor the other succeeds in checking the parts: $do$ can check the adjective mode -$ho$ but not the noun mode $h$. For $ho$ the case is reversed.

**Sortal Structure** In the second analysis of Dutch adjective-noun agreement, we used a single modal operator to encode different morphosyntactic properties like definiteness and gender. We used names that we said should be interpreted as atomic but that appeared to have some structure. We will now look at this structure more detail.

As a starting point, we decompose the information about the various attributes analogously to the decomposition in feature structures. In other words, we rephrase resource modes as simple feature structures. So instead of the atomic modes as before we assume that each resource mode is a complete feature structure. The following four feature structure modes represent the resolved feature-value combinations (the maximal structures) for this example.

$$
\begin{align*}
\langle \text{goed} \rangle_{ho} \circ \langle h \rangle_{N} & \Rightarrow N \\
\langle \text{goed} \rangle_{ho} \circ \langle \text{huis} \rangle_{do} & \Rightarrow N \\
\langle \text{goed} \rangle_{do} \circ \langle \text{huis} \rangle_{do} & \Rightarrow N \\
\langle \text{goed} \circ \text{huis} \rangle_{do} & \Rightarrow N \\
\langle \text{goed} \circ \text{huis} \rangle_{o} & \Rightarrow N \\
\text{goed} \circ \text{huis} & \Rightarrow [o]_{N} \\
\text{goed} \circ \text{huis} & \Rightarrow [o]_{N}
\end{align*}
$$

We can abbreviate these structures by leaving out the attribute names and write the structures simply as $[d, b]$, $[d, o]$, $[h, b]$, $[h, o]$. Contrary to the tradition in feature structure theories we will assume that we not only have underspecified but also overspecified feature structure modes. Notice that modes like $d$ and $o$ that we presented earlier now appear as values of an attribute in the feature structure representation. Corresponding to the overspecified and underspecified modes like $\text{Gndr}$ and $\text{gndr}$ we can introduce overspecified and underspecified values for the attributes. However, in this case we will just represent overspecification by mentioning all the values and underspecification by leaving them out.
Suppose we take these feature structures as our modal feature decorations. The modes of our second analysis correspond to the following feature-structure modes.

\[
\begin{align*}
    \text{db} & = [d, b] \\
    \text{do} & = [d, o] \\
    \text{hb} & = [h, b] \\
    \text{ho} & = [h, o] \\
    o & = [o] \\
    h & = [h, b, o] \\
    d & = [d, b, o]
\end{align*}
\]

It is immediately clear that there is no position in the feature structure subsumption lattice that corresponds to the mode -ho. There is no feature structure that is subsumed by \([d, b]\), \([d, o]\), and \([h, b]\) but not by \([h, o]\). This relates to the unification-based analysis of Bouma, which does not provide a simple feature structure analysis for the inflected adjective but uses negation in this case. We can represent these categories in attribute-value matrix notation extended with a negation operator as follows. Note that we must consider these attribute-value matrices here as formulas from the attribute-value matrix description language and not as feature structures.
To see in which space the mode -ho resides we continue adding some lines that parallel the lines already drawn. The names on the nodes indicate the set of maximal feature structures that are below that node and the singleton sets of maximal feature structures on the base level.

If we add all the further lines then we get, geometrically speaking, a part of a four-dimensional cube (each resolved feature structure representing a dimension) — with one corner missing. If we define FS as the set of maximal feature structures, i.e. \( \text{FS} = \{ [d, b], [d, o], [h, b], [h, o] \} \), then the graph represents the structure \( \langle \mathcal{P}(\text{FS}) \setminus \emptyset, \subseteq \rangle \), i.e. the powerset of FS without the empty set, ordered by set inclusion. We will refer to \( \mathcal{P}(\text{FS}) \setminus \emptyset \) as \( \mathcal{O} \). But this graph does not exactly represent the structure we want for our linguistic descriptions as the information ordering only deals with overspecification. For underspecified information we have to mirror the structure we already have. We will refer to this mirrored set as \( \mathcal{U} \). In this set-theoretic interpretation of the structure, the nodes in \( \mathcal{O} \) are decorated by sets that are the union of the sets decorating the nodes that are below it and connected by it.
Note that the nodes immediately above the sets mentioned in the middle represent the same nodes as the ones immediately below these sets. To refer to the elements below the middle we will also use a set-like notation but replace the comma by a semicolon. The lowest element in the graph then bears the name \( \{[d, b]; [d, o]; [h, b]; [h, o]\} \).

The correspondence between the morphosyntactic modes that we used in our second analysis, the feature structure modes with subsumption and with the modes of the graph above is as follows.

<table>
<thead>
<tr>
<th></th>
<th>([d, b, o])</th>
<th>([d, b]; [d, o])</th>
</tr>
</thead>
<tbody>
<tr>
<td>(d)</td>
<td>([h, b, o])</td>
<td>([h, b]; [h, o])</td>
</tr>
<tr>
<td>(h)</td>
<td>([o])</td>
<td>([d, o]; [h, o])</td>
</tr>
<tr>
<td>(o)</td>
<td>([d, b])</td>
<td>([d, b])</td>
</tr>
<tr>
<td>(db)</td>
<td>([d, o])</td>
<td>([d, o])</td>
</tr>
<tr>
<td>(do)</td>
<td>([h, b])</td>
<td>([h, b])</td>
</tr>
<tr>
<td>(hb)</td>
<td>(-ho)</td>
<td>([d, b]; [d, o]; [h, b])</td>
</tr>
</tbody>
</table>

Now that we have a systematic representation of the inclusion relations we can define the inclusion postulates also schematically as follows:

\[
[Y_U]A \Rightarrow [X_U]A \quad (Y_U \subseteq X_U) \\
[Y_O]A \Rightarrow [X_O]A \quad (X_O \subseteq Y_O)
\]
In this schematic presentation inclusion postulates do not only hold between modes that are immediately connected by a line in the diagram but also between modes that are connected by several steps (transitive closure). To illustrate the use of these modes we present the derivation for *een goed huis*. For notational convenience we leave out the outermost square brackets (the brackets indicating the modal operator \(\Box\)) from decorations like \([\{[h, b], [h, o]\}]\) and write \([\{h, o\}, [h, b]\}]\) instead. The derivation for *een goed huis* requires us to derive that *goed huis* is of type \(\{[d, o]; [h, o]\}N\).

\[
\begin{align*}
N \Rightarrow N & \quad N \Rightarrow N \\
N/N \circ N \Rightarrow N & \\
N/N \circ \{\{[h, b], [h, o]\}\}N_{\{[h, b], [h, o]\}} \Rightarrow N \\
N/N \circ \{\{[h, b], [h, o]\}\}N_{\{h, o\}} \Rightarrow N \\
\{\{[h, o]\}\}N/N \circ \{\{[h, b], [h, o]\}\}N_{\{h, o\}} \Rightarrow N \\
\{\{[h, o]\}\}N/N \circ \{\{[h, b], [h, o]\}\}N_{\{h, o\}} \Rightarrow N \\
\{[h, o]\}\}N/\circ \{\{[h, b], [h, o]\}\}N_{\{h, o\}} \Rightarrow N \\
\{[h, o]\}\}N/\circ \{[d, o]; [h, o]\}N \Rightarrow \{[d, o]; [h, o]\}N
\end{align*}
\]

In this simple fragment, we have used complex feature modes. First we organised basic modes in a kind of simple feature structure and then we made more complex modes out of this by considering the powerset of the set of feature structures as possible modes. The operators were used to define the appropriate inclusion relations. This analysis suggests that the modal decorations and the sortal structure can be structured in useful ways.

**Complex versus Simplex** The analysis of Dutch NP internal agreement was not merely introduced to argue that it is possible to provide a subsumption-based type-logical grammar in which we do not need multiple type assignments. It was also meant to indicate to what degree the morphosyntactic modes can be structured internally. This may provide another opportunity to explore combinations of the type-logical and the constraint-based framework by defining an appropriate logic for the modes, possibly inspired by a feature description logic.

In the example, the major function served by the complex decorations is to define a hierarchical classification of the modes. Only one modality is used with a complex mode that is used to account for both cross-classification and inclusion relations. The example does not tell us much about distributional behaviour of morphosyntactic features.

In more complicated cases, it might be necessary to distinguish the behaviour of different modes with respect to their distribution in phrase structure. In this case, it might not be feasible to group the information.
together as a complex index on a single modality. Mixed solutions may be better suited.

One option would be to define a system in which complex indices on one modality can be spread over several modalities, when the need arises for distribution. A source of inspiration for the appropriate logic might be a system like dynamic predicate logic (Goldblatt (1987)). To illustrate this, we mention the Comp schema as defined by Goldblatt (1987, p. 88). Note that the use of ; here should not be confused with our use above.

Comp: \([\alpha; \beta]A \leftrightarrow [\alpha][\beta]A\)

Allowing this kind of schema may be useful in structuring and manipulating feature information. The Comp schema allows a single composed mode to be decomposed into its components (and components can be assembled again). One could imagine an application in which the composed mode is used to structure the information while at some point in the derivation the modes must be split into parts that have different distributional characteristics.

Summary

In this chapter we have discussed and illustrated different options for using unary modalities to account for morphosyntactic description in grammars. We have discussed variants for the implementation of the matching or checking operation and for the structure of combining different classification attributes. We have provided illustrations to show how cross-classification, feature checking, inclusion and distribution relations can be defined in different linguistic settings.

We have not argued in favour of specific variants as general solutions but the choice of examples indicates the respective benefits the various options offer. In a typical grammar for a language, several of these options could be used side-by-side, depending on the construction types and the characteristics of the various morphosyntactic features.

Some general factors that influence the choice for the analysis of a particular language or construction can be summarised as follows.

- The preferred notion of elegance or simplicity (e.g. simplex decorations on multiple modal operators or complex decorations on single operators).
- The fact whether different attributes behave differently distributionally.
- The need to maximise the reduction of assignments.

In general, one might want to define different distribution postulates relative to the modes on the unary and binary connectives. Because different morphosyntactic properties need not share the same distributional
patterns, it may not be useful to combine them in a complex mode. It appears that when writing actual grammars a different modal operator is warranted for each group of morphosyntactic properties that shares the same distributional behaviour. Notice that this may give rise to conflicts in the sense that optimising the reduction of lexical assignments may require combining specific properties into one mode, whereas the distributional behaviour of these properties might require that they decorate different modal operators.
Summary of the third part

In this part we have looked at the application of the basic apparatus assumed in current versions of type-logical categorial grammars in order to deal more adequately with the description of morphosyntactic properties of expressions. This framework meets the objectives we pointed out at the end of Chapter 6 and which we repeat here.

- Connectives are treated as grammatical constants with a complete logic (both elimination and introduction rules).
- Underspecification is polarity-sensitive, respecting the logic of the connectives.
- The framework provides a way to deal with co-variation.
- Morphosyntactic decorations can appear on both basic and complex types.
- A resource-conscious treatment for morphosyntactic information is defined that parallels the treatment of syntactic composition and selection.

We have used the residedated unary operators $\diamond$ and $\Box$ to decorate categories with morphosyntactic information and have used their logic to define a checking procedure. We have also looked at the issue of how underspecification can reduce the amount of information that has to be stated in the lexicon and the number of lexical assignments that need to be entered in it. Besides the specification of morphosyntactic properties on lexical items it is also important to see how such properties distribute in grammatical structures (how and where are features checked, how do they move through constituent structure, etc.). The use of the multimodal calculus with inclusion and distribution postulates provides the possibility to fine-tune this relation in a linguistically contentful manner. When using the modal operators to mark morphosyntactic properties, with resource modes indicating the different values, we still have several options to execute this programme.

Our proposals for the use of the multimodal framework can be summarised as follows.

The logic of residuation constitutes the core engine. Syntax, or the description of composition and selection of expressions, is accounted for by the residedated binary connectives $\!,\bullet,\backslash$. Morphosyntactic features are marked and checked on expressions using the residedated unary connectives.

This basic engine is further refined by decorating the logical connectives with indices, or resource management modes, imposing a sortal structure. These sorts decorate binary connectives as phrasal modes $\circ$ to distinguish between different structural (associative, commutative,...) or other options.
The sorts decorating unary connectives mark different morphosyntactic attributes (and/or function as control operators).

The relation between different sorts is expressed through structural postulates. An ordering relation on the sorts is defined through inclusion postulates on the decorations. Interaction between sorts, particularly between the unary and the phrasal sorts, achieves controlled access to structural changes in syntactic constructions on the one hand and controlled distribution of morphosyntactic information through that structure on the other hand.
Part IV

Underspecification Reconsidered
Introduction to Part IV

Our objective to refine the category structure for type-logical grammars to allow cross-classification, underspecification and the factorisation of information was inspired by the use of feature structures in constraint-based grammars like HSPG. Sag (1997) notes that within the tradition of generative grammar, it is commonplace to assess developments in the field as a progression from 'construction-specific rules' to 'general principles'. In the context of HPSG-grammars, currently influenced by ideas from construction grammar, this is implemented as follows.

In particular, the proposal to treat familiar kinds of phrases in terms of multiple dimensions of classification allows general constraints on constructions to be expressed as properties of super-types, while still allowing the idiosyncrasies of individual constructions to be accommodated. [...] The logic of inheritance allows this kind of analysis to achieve a highly deductive conception of universal grammar.

The type-logical perspective on grammatical description can also be characterised as deductive, but clearly in another sense. Moortgat (1997) mentions as one of the objectives for this programme the following:

Design of a specific grammar logic, i.e. a logic with a consequence relation attuned to the resource-sensitivity of grammatical inference — to be contrasted with 'general purpose' specification languages for grammar development, where such resource sensitivity has to be stipulated, e.g., the language of feature logic used in HPSG.

In the chapters comprising Part III, we have investigated how the tools available in the multimodal framework can be used to refine the classification potential of categories. To account for morphosyntactic properties of expressions, we have proposed the use of the residuated unary operators, marking expressions with the relevant property and defining a feature-checking theory that is used to account for the restrictions on combinations and the percolation or distribution of the information in phrase structure. The logical rules provide the basic mechanism for the feature checking procedure which is fine-tuned to linguistic description by structural rules. We defined an ordering on the information provided by the modal decorations using inclusion postulates to reduce the number of assignments to lexical items and to simplify the grammar. The use of these postulates (and others like distribution postulates) help us to factorise the information contained in a grammar into separate principles.

We now want to compare the proposals concerning the organisation of such type logical grammars with the organisation of feature structure
Part IV

grammars as we have discussed them in Chapter 3. In this part we will try to show in what sense the proposals are complementary rather than incompatible. We will point out that their status in a grammar is different and that they are situated on a different level as far as the presentation of the information and the way it is put to use in defining a language is concerned.

The Status of Underspecification An important aspect in the discussion of complementarity concerns the status of underspecification in linguistic description. We will now point out a distinction between two strategies for underspecification or partial information in the use of a grammar. The first we will call the off-line strategy and the second the on-line strategy. An alternative pair of terms could be lexical and derivational strategy, indicating that the underspecification is resolved lexically in the first case (the lexical look-up procedure provides fully specified categories) and that it enters a grammatical derivation unresolved in the second case.

In Part I, we introduced the notion of underspecification in grammatical description by means of a simple example with a phrase structure grammar extended with feature structures. Here, we will explain the different status of generalisations by means of a simple categorial grammar. We consider the simple applicative AB-grammar as presented in a series of papers by Emmon Bach which we reviewed in Chapter 5.

Off-line Strategy Bach (1983b) introduces feature structures to account for morphosyntactic properties of expressions. The basic categories are pairs of an atomic symbol, representing the major category or part of speech information, and a feature bundle representing the morphosyntactic features. For the sake of simplicity we will include this major category as part of the feature bundle. The complex categories are freely generated as usual from the basic categories using the operators \ and \. A language, taken to be a set of pairs of expressions and categories, is defined by a lexicon as in the following paraphrase of Definition 23.

Definition 26 (Language) Given a lexicon Lex, the language \(\cal L\) over this lexicon is specified as the smallest set such that:

(i) \(\text{Lex} \subseteq \cal L\) and

(ii) \((e_1 \circ e_2, A) \in \cal L\) if either

(ii-a) \((e_1, A/B)\) and \((e_2, B)\) are in the language, or

(ii-b) \((e_1, B)\) and \((e_2, B\setminus A)\) are in the language.

What is important to note about this definition is that in the application schema the domain category of the functor has to be identical to the category of the argument it combines with. Note that \(\text{Lex}\) is defined to be a part
of the language. Furthermore, the feature structures that make up the basic categories are assumed to be fully specified in Lex as it appears in the definition of the language (in HPSG terms, they should be ‘totally well-sorted’ and ‘sort-resolved’; see Carpenter (1992b) and Pollard and Sag (1994)). The ‘lexicon’ as it is normally understood in grammatical theories, does not coincide with Lex, but should be taken as a description or a specification of it. This lexicon can use underspecified descriptions, inheritance, lexical rules etc. to specify the set Lex of totally well-typed and sort-resolved elements.

Although Bach requires the categories to be identical, he accommodates underspecification of lexical entries by means of notational conventions. The lexicon is not specified as a list of words with completely specified categories, but by providing a list of pairs of words and partial categories. This list is expanded by using a further set of conventions that fill in the information that is left out. They multiply out the entries for which multiple instantiations are possible. So it is clear that the first lexicon is not the ‘lexicon’ (Lex) that figures in the definition of a language, because it contains partial categories. We could call it a pre-lexicon.

Because the conventions do not simply concern individual items but whole classes, they can be used to capture not just simple underspecification but also other interesting linguistic generalisations. Let us make this more concrete by considering an example. We will assume that the feature structures are appropriately formed according to some signature. Take the following ‘pre-lexicon’ (masculine, singular, accusative, weak).

\[
\begin{align*}
\text{den} & \quad \text{alten} & \quad \text{Mann} \\
\text{NP/N} & \quad \text{N/N} & \quad \text{N} \\
\text{GEN} & m & \text{GEN} & m & \text{GEN} & m \\
\text{PER} & s & \text{PER} & s & \text{PER} & s \\
\text{CASE} & a & \text{CASE} & a & \text{CASE} & a \\
\text{DECL} & w & \text{DECL} & w & \\
\end{align*}
\]

The conventions fill in the information that is missing in the underspecified pre-lexicon as follows.

- If a feature is not specified for a value, although it is appropriate in the structure, this means that we can multiply out the assignments instantiating the structure further for all the appropriate values.

- If a complex category is a modifier type (the \text{cat} value of the domain is identical to the \text{cat} value of the range), then all the values of the other features of domain and range are identical as well.

- If a complex category is a specifier type (the \text{cat} value of the domain is the same as that of the range except for the projection or bar level, e.g. \text{n} versus \text{np}) then all the values of the other features are identical as well, except for those that are explicitly mentioned in the pre-lexicon to be different.
The expanded version that satisfies these constraints looks like this, where *strong* and *mixed* are other values appropriate for decl besides *w* (for *weak*).

\[
\begin{align*}
\text{den} & \quad \text{NP} \\
& \quad \text{GEN } m \\
& \quad \text{PER } s \\
& \quad \text{CASE } a \\
& \quad \text{DECL } w
\end{align*}
\]

\[
\begin{align*}
\text{alten} & \quad \text{N} \\
& \quad \text{CAT } n \\
& \quad \text{GEN } m \\
& \quad \text{PER } s \\
& \quad \text{CASE } a \\
& \quad \text{DECL } w
\end{align*}
\]

\[
\begin{align*}
\text{Mann} & \quad \text{N} \\
& \quad \text{GEN } m \\
& \quad \text{PER } s \\
& \quad \text{CASE } a \\
& \quad \text{DECL } w
\end{align*}
\]

On a technical level, the conventions by Bach take their inspiration from theories about feature structures that use so-called feature specification defaults, feature co-occurrence restrictions, etc. like Chomsky and Halle (1968) and Gazdar et al. (1985) which have been developed and implemented in a number of systems (Gazdar et al. (1988), Ritchie et al. (1987), Evans and Gazdar (1989), to name just a few).

The only step defined by Definition 26 is application, it does not mention the conventions. This means that in the actual definition of a language or of compositional structures, the underspecification has already been compiled out, so to speak. The use of feature structures, the possibility of underspecification and the specification of the conventions is independent from the mechanisms (application) that say how a grammar defines a language.

This way of looking at underspecification as a kind of abbreviation is also found in other approaches. In Carpenter (1992a) this position is taken on the use of variables in a predicational version of the basic categories: a lexical assignment of an expression to \(n(X)\) (where \(X\) ranges over possible values for number) is shorthand for two assignments: \(n(sg)\) and \(n(pl)\). Carpenter also motivates the introduction of lexical rules "to account for additional categorisations which are predictable on the basis of core lexical assignments". He furthermore assumes that the rules are strictly lexical,
they apply before syntactic applications. This is clearly a kind of off-line use of lexical rules.

Also in Hepple (1990, p. 74ff) we find this idea. There it is assumed that lexical assignments to words are constructed in several stages. “This notion of ‘stages in construction’ is handled by allowing the lexicon to consist of a number of distinct ‘compartments’ or subdomains [...]”. As Hepple points out, not all the types that arise during the stages become available to syntax.

In the case of the phrase structure grammar a similar strategy can be used. However, in this case the expansion does not only pertain to the lexicon but also to the grammar rules.

<table>
<thead>
<tr>
<th>Pre-Grammar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-lexicon</td>
</tr>
<tr>
<td>+ Pre-rules</td>
</tr>
<tr>
<td>(\downarrow) (Expansion) (\downarrow)</td>
</tr>
<tr>
<td>Grammar</td>
</tr>
<tr>
<td>Lexicon</td>
</tr>
<tr>
<td>+ Rules</td>
</tr>
</tbody>
</table>

We will call this approach to the organisation of grammatical information the off-line strategy. It should be noted, though, that we are not referring here to processing strategies for parsing or generation.

**On-line** The analysis of morphosyntactic information using unary modalities and inclusion postulates which we presented in the previous part belongs to an alternative conception of partiality in the grammatical framework. Here underspecification is dealt with (on-line) in the calculus. We can also define an alternative to the Bach grammar in which the underspecified structures are used as such, on-line, in the calculus. For this we could change the definition of language by replacing the condition that the domain of the functor has to be identical to the argument by a subsumption check. This leads to the categorial equivalent of the subsumption-based grammar from Borsley (1991) that we presented in Chapter 3. As we will see in the next chapter, these choices have an effect on the language that is generated by the grammar.

**Definition 27 (Language)** Given a lexicon \(\text{Lex}\), the language \(\mathcal{L}\) over this lexicon is specified to be the smallest set such that:

(i) \(\text{Lex} \subseteq \mathcal{L}\) and

(ii) \((e_1 \circ e_2, A) \in \mathcal{L}\) if either

(ii-a) \((e_1, A/B)\) and \((e_2, C)\) are in the language and either \(B \subseteq C\) or \(C \subseteq B\), or

(ii-b) \((e_1, C)\) and \((e_2, B \setminus A)\) are in the language and either \(B \subseteq C\) or \(C \subseteq B\).
In this case we can let the underspecified feature structures enter in the derivation of grammatical structure. This involves a simple change to the Bach-style grammar. We could also extend it to allow for re-entrancies between feature structures that make up the domain and range subcategories.

**Combination** What becomes immediately obvious from the way we presented the difference between the strategies, is that they are not mutually incompatible, also in a type-logical setting. For instance, we could combine an off-line strategy to organise the grammatical (=lexical) information (using some implementation of a logic of inheritance to define a version of the 'hierarchical lexicon' (Pollard and Sag (1987), Gazdar and Daelemans (1992a), Gazdar and Daelemans (1992b)) or some other techniques to state abbreviations) with an on-line strategy in which inclusion postulates are used in the derivation of phrases. Other types of combination are possible. Note that restricting the constraint-based techniques to off-line underspecification only, does not lead to the problems of mixing this type of underspecification with the type-logical inference procedure that were presented in Part II.

If we assume such a combination, then we have two different techniques that can be used for the same or at least similar goals (underspecification off-line by constraint-based types or on-line by postulates). The question that arises immediately is whether this does not lead to an unnecessary duplication of techniques or whether an appropriate division of labour can be defined.

In the next chapter we will look at this question in some depth and point out areas for which an on-line strategy is necessary and an off-line strategy is insufficient and other areas for which an off-line strategy is suited best and an on-line strategy is less appropriate.

It is already clear now, that off-line strategies or rather, constraint-based techniques are particularly useful to capture global patterns across a whole class of categories, whereas the inclusion postulates we presented before were used to reduce the number of assignments to individual items. In the following chapter, we expand on the benefits and limits of both techniques, looking specifically at the motivation for underspecification, for which both seem appropriate.

It should be noted that from a processing point of view (parsing or generation) this does not entail that all lexical inferences should precede all grammatical inferences. In fact one can define processing strategies in which the two kinds of inferences are interwoven (see, for instance, Bouma and van Noord (1994)).

**Conclusion**

In this introduction to the final part, we have introduced the terms *on-line* and *off-line* to refer to two ways in which underspecification can
be used in a grammatical framework. An off-line approach to the organisation of grammatical information seems compatible with a type-logical framework. In particular this means that the logic of inheritance and the other techniques to express generalisations in HPSG and similar frameworks is complementary to the proposals we made before.

Such a combination of the constraint-based approach and the type-logical approach leads to a mix of different perspectives on the 'deductive conception of grammar' that was referred to in the quotes of Sag and Moortgat we presented above. We can distinguish between three different notions of deduction in the hybrid architecture that results from the kind of marriage we indicated above.

- Logical inferences pertaining to the constants of grammatical reasoning: the logic of residuation for /, •, \ and □, ◊.

- Logical inferences for inheritance as defined by the feature description logic for the lexical structure.

- Structural inferences based on inclusion and distribution postulates for underspecification and structural operations.

Note that the third type of inferences in a type-logical context can be used for the same purposes as the typical use of the second type in a constraint-based context. This duplication of options for the same purpose is further investigated in the next chapter, in which coordination constructions play an important role.
Coordination and the Limits of Underspecification

In the introduction we put our proposals for treating cross-classification of expressions using modal operators for the morphosyntactic features and the use of inclusion postulates for underspecified structures in a wider perspective. We made a distinction between two strategies for expressing generalisations and defining underspecification in grammars. We also made a start with indicating the possible complementary use of each of the two types. In this chapter we will discuss in more detail the benefits and drawbacks of both the on-line and off-line strategy to underspecification. Important in this respect is the analysis of some typical coordination constructions in which the notion of underspecified categories plays a crucial role.

In Bayer and Johnson (1995) the same type of construction has been investigated to argue in favour of a type-logical approach and to explain some of the shortcomings of the standard constraint-based approaches (see also Johnson and Bayer (1995), and Bayer (1996)). We will summarise that argument here.

We show that the analysis relies on the use of a polarity-sensitive version of the on-line (derivational) strategy to underspecification for certain cases. However, we will also show how and when this use of underspecified categories must be restricted to avoid overgeneration. For these cases it is possible to use the off-line strategy to reduce lexical type-assignments.

We will first present a sketch of the coordination constructions that we are interested in, followed by their type-logical analysis. We illustrate the argument favouring the ‘asymmetric’ type-logical analysis (see below) but also provide more details on why an off-line strategy to underspecification falls short for these constructions. Next, we discuss some problems with the on-line strategy to show the limits of its use.

9.1 Coordination of unlike types

In the following paragraphs we distinguish between the types of coordination constructions which are relevant for our discussion.

Following Pullum and Zwicky (1986) we will use the term ‘Chomsky’s generalisation’ to refer to the familiar statement about coordination that conjuncts must at least belong to the same syntactic category, which we will interpret as including morphosyntactic properties, (Chomsky (1957)),
and the term 'Wasow's generalisation' for the constraint on coordination that says that an element in construction with a coordinate constituent must be syntactically construable with each conjunct separately (see also Sag et al. (1985)). We will call the element in construction with the coordination the factor (again following Pullum and Zwicky). In a construction like remained wealthy and a republican, the factor is remained.

There are some well-known apparent counter-examples to Chomsky's generalisation. It seems to rightfully exclude (9.1) and (9.3) but to wrongfully exclude (9.2). Note that judgments are taken from various authors referred to in this chapter.

*I want another beer and to have a good time

I want another beer
I want to have a good time

He is a republican and proud of it

He is a republican
He is proud of it

*A Republican and proud of it was elected President

A Republican was elected President

*Proud of it was elected President

Wasow's generalisation, on the other hand, rightfully includes (9.2) and excludes (9.3) but seems to include (9.1) wrongfully. Although the verb want apparently can combine with each of the conjuncts separately, the case is different from (9.2) because in the case of want we are dealing with two distinct lexical items 'wish to possess something' and want 'wish to do something' (Pullum and Zwicky (1986, p. 753)). This means that want is similar to can in (9.4).

*I can tuna and work for a living

If we assume this analysis, Wasow's generalisation also holds for (9.1). The interpretation of what at first sight seems a detail of the generalisation as formulated above becomes important here. When we apply the test of construing 'the element in construction with the coordinate constituent' with each conjunct separately we must be sure to use the same element each time and not pick another one (albeit one that is homophonous). A reflex of this constraint is the Anti-Pun Ordinance from Zaenen and Karttunen (1984, p. 316) that says that "A phrase cannot be used in two different senses at the same time." The distinction between same and different is essential to a correct treatment of coordination.
However, there are other cases of coordination, such as (9.5), which violate Wasow's generalisation. These are cases of so-called principled resolution (see Corbett (1983)). Here, the coordination is grammatical despite the fact that the factor cannot combine with each of the conjuncts separately.

\[ \text{John and Bill have been to the beach} \] (9.5)

\[ \text{*John have been to the beach} \]

\[ \text{*Bill have been to the beach} \]

Summarising this short discussion, we distinguish four options for coordination constructions with unlike types which are important for our discussion. Typical examples for each of these are the following.

(I) Neutral factor Er findet und hilft Frauen (9.6)

(II) Ambiguous factor *Sie singt und singen (9.7)

(III) Principled Resolution un père et une mère excellents (9.8)

(IV) Unfit factor *Kim grew wealthy and a republican (9.9)

Class I (also 9.2) contains grammatical coordinations of unlike types with a so-called neutral factor. They are grammatical despite the fact that they seem to violate Chomsky's generalisation. In the example, findet selects an accusative and hilft a dative noun phrase. We will argue that on-line underspecification (or overspecification) of the neutral factor is needed for these cases. They also form an argument for the asymmetric or polarity-sensitive subsumption-based treatments and against the unification-based ones (if used on-line). These arguments will be presented and illustrated in Section 9.2.

Class II (also 9.1), (9.4) contains ungrammatical coordinations despite the fact that the factor, being ambiguous, can combine with each conjunct separately. They show that Wasow's generalisation is not a sufficient condition for grammaticality. They will be used to point out the limits of on-line underspecification. These can be analysed by either not assigning underspecified types to the ambiguous factor that can enter the derivation, or taking the off-line (lexical) strategy in which the underspecification is resolved before lexical look-up.

Class III (also 9.5) involves principled resolution. Although Wasow's generalisation is violated, the coordination is grammatical. They are not important for the discussion about the types of underspecification. We will therefore consider these only briefly below.

Class IV (also 9.3) contains ungrammatical coordinations, violating both Chomsky's generalisation (unlike types are coordinated) and that of Wasow (the factor is unfit for at least one conjunct). Kim grew a republican is ungrammatical. The factor is neither neutral nor ambiguous. They can be
ignored in this discussion because they do not need special consideration. In the analyses we present below, we will assume that Chomsky's generalisation has to hold. This can be taken care of by assuming a polymorphic category like $X \backslash X / X$ for the coordinator. The challenge then is to provide an analysis for those cases, like Class I, where this restriction on the coordination of like types does not seem to hold.

In Pullum and Zwicky (1986) a fifth class of coordinations is presented which involves 'phonological resolution'. Technically speaking, they can be treated the same way as the class with neutral factors. We will therefore be concerned mainly with class I and class II type of coordinations. The first we call neutralisation cases and the second cases of ambiguity.

We will show that a proper use of underspecification can account for all of the cases. The case of principled resolution (class III) is slightly different because it does not involve underspecification. For the sake of completeness we will briefly sketch some suggestions on how to analyse them in a type-logical setting first.

**Principled Resolution** Corbett (1983) provides many examples of cases of person, number and gender resolution. In Sag et al. (1985), a few words are devoted to a treatment in a GPSG analysis. In this analysis the solution to the problem lies in choosing appropriate features and values. The rules for French gender resolution can be stated as in Corbett (1983).

1. if all conjuncts are feminine (syntactically), the feminine form is used;
2. otherwise the masculine is used.

The preferred treatment of these cases depends on the exact analysis of coordination. We will outline two proposals. The first involves coding the rules in the options for the conjunction. Even when one opts for a highly polymorphic category for *and* as something like $X \backslash X / X$, this does not mean that it cannot be further restrained. To take care of the French gender resolution one could refine the category assignment and say that *and* is ambiguous between the following assignments:

$$[f]X \backslash [f]X / [f]X \quad [m]X \backslash [m]X / [m]X \quad [m]X \backslash [m]X / [f]X \quad [f]X \backslash [m]X / [m]X$$

This is a very simple way to spell out all the options. Another way to define the feature computation rules is by making use of special distribution postulates. We assume that coordination constructions involve a special mode of combination. This means that we can make the distribution of the gender feature sensitive to the coordination context. For *un père et une mère* we would have something like the following sequent, where the exact type for the conjunction is not important.

$$[m]NP \circ_c \text{conj} \circ_c [f]NP \Rightarrow [m]NP$$
With distribution postulates like the following encoding the resolution rule, we can account for the French examples.

\[
\begin{align*}
\langle i \rangle (A \bullet B) & \rightarrow \langle i \rangle A \bullet \langle i \rangle B \quad (i \in \{m, f\}) \\
\langle m \rangle (A \bullet B) & \rightarrow \langle m \rangle A \bullet \langle f \rangle B \\
\langle m \rangle (A \bullet B) & \rightarrow \langle f \rangle A \bullet \langle m \rangle B
\end{align*}
\]

In the following section we will turn to an analysis of the other coordination types which are more relevant for our discussion on the status of underspecification. We will first discuss the neutral cases (Case I) which apparently violate Chomsky's generalisation (conjunction of unlike type) but not Wasow's generalisation (both conjuncts can combine with the factor separately). We argue that on-line underspecification is needed for these cases. Next we will discuss the cases for which Wasow's generalisation fails, i.e. the ambiguous cases (both conjuncts can combine with the factors separately, they are of unlike types, but they cannot be conjoined). For these cases on-line underspecification should be avoided or carefully controlled (Heylen (1997b)).

### 9.2 On-line underspecified neutral factors

In this section we look at the type of coordinate structures that involves coordinates of unlike types with a neutral factor. This class has been used by Johnson and Bayer (1995) to argue in favour of a type-logical, subsumption-based analysis and against a constraint-based analysis. The crucial factor contrasting these types of analyses that makes the difference between success and failure is the polarity sensitivity of underspecification in the first type of analysis, which is lacking in the second. The analysis of this class of constructions also sheds light on the distinction between off-line (lexical) versus on-line (derivational) underspecification. Our analysis will make crucial use of the latter type. Before we present these arguments in more detail, we will provide a type-logical analysis of the constructions, both along the lines of a Bayer and Johnson framework and along the lines of the framework we have presented in Part III.

#### 9.2.1 Type-Logical analysis

**Bayer and Johnson** In a verb phrase such as remained wealthy and a republican, an adjective phrase and a noun phrase are conjoined. In other words we see a conjunction of unlike categories apparently violating Chomsky's generalisation. In a Lambek-style categorial grammar extended with boolean operators this verb phrase is derivable if we assume that the verb is assigned the type \( VP/(AP \lor NP) \). The logic of \( \lor \) allows us to weaken the specification of the class of items to which wealthy and a republican belong to a shared type denoting the union of adjective and noun phrases.
Following Johnson and Bayer (1995), the coordinator \textit{and} is assigned the type \textit{conj} in this derivation and some kind of conjunction rule is used such that $A \textit{conj} A \vdash A$. At this point we do not want to go into the precise treatment. We could replace it with a polymorphic category like $X \setminus X/X$ as we will occasionally do in other examples. Both of these capture Chomsky’s generalisation that coordinates must be of the same type.

\[
\begin{array}{c|c|c}
\text{reained} & \text{wealthy} & \text{a republican} \\
\hline
\text{VP/(AP \lor NP)} & \text{AP} & \text{NP} \\
\text{AP \lor NP} & \text{AP \lor NP} & \text{AP \lor NP} \\
\text{VP} & \text{AP \lor NP} & \text{AP \lor NP} \\
\end{array}
\]

Another illustration of a similar construction involves overspecification. In Johnson and Bayer (1995) the phrase \textit{findet und hilft Frauen} (\textit{finds and helps women}) is shown to be derivable when the following assignments are assumed.

\[
\begin{align*}
\text{Frauen} & \quad \text{NP \land acc \land dat} \\
\text{findet} & \quad \text{VP/(NP \land acc)} \\
\text{hilft} & \quad \text{VP/(NP \land dat)}
\end{align*}
\]

Note that \textit{findet} requires its object to be accusative and \textit{hilft} requires it to be dative. \textit{Frauen} in this case fits both requirements.

The following derivation shows how the coordination of the verbs is in $\text{VP/(NP \land acc \land dat)}$ which can combine with the overspecified \textit{Frauen} in $\text{NP \land acc \land dat)}$.

\[
\begin{align*}
\text{findet} & \quad \text{VP/(NP \land acc)} \\
\text{NP \land acc} & \quad \text{VP} \\
\text{und} & \quad \text{VP/(NP \land acc \land dat)} \\
\text{conj} & \quad \text{VP} \\
\text{VP} & \quad \text{VP/(NP \land acc \land dat)} \\
\text{VP} & \quad \text{VP/(NP \land acc \land dat)}
\end{align*}
\]

**Modal Analysis** The type-logical analysis of coordination along the lines of Bayer and Johnson can be reconstructed in the multimodal framework we discussed in the previous parts.

The terms making up the basic categories of the Bayer and Johnson language combine two functions. First of all, they are used to underspecify ($\text{VP/(AP \lor NP)}$) and overspecify ($\text{acc \land dat}$) information. Secondly, they are also used for cross-classification purposes as a simple kind of feature description language ($\text{NP \land acc}$). In the multimodal language that we have been using in the previous chapters, this aspect of category decomposition is taken over by decorating basic categories with unary modalities indexed
by morphosyntactic sorts. An information ordering on these sorts is defined by inclusion postulates. An ordering on the basic categories can be defined by non-logical axioms.

The German example can be described as follows. To encode the values for case we use as modalities: [acc], [dat], [Case]. Here [Case] represents the overspecified values which we will also call general features. Their behaviour is fixed by the following inclusion postulates, which allow us to infer specific values from the general features.

\[ [\text{Case}] A \Rightarrow [i] A \quad (i \in \{\text{acc}, \text{dat}\}) \]

We can use the following type assignments for the lexical entries.

- Frauen: $[\text{Case}] \text{NP}$
- findet: $\text{VP}/([\text{acc}] \text{NP})$
- hilft: $\text{VP}/([\text{dat}] \text{NP})$

We can show that findet und hilft derives $\text{VP}/[\text{Case}] \text{NP}$ if we can show that both findet and hilft derive $\text{VP}/[\text{Case}] \text{NP}$. The derivation for findet is given below. The derivation for hilft is completely parallel.

\[
\begin{align*}
\text{VP} & \Rightarrow \text{VP} \quad [\text{Case}] \text{NP} \Rightarrow [\text{acc}] \text{NP} \\
\text{VP}/[\text{acc}] \text{NP}, [\text{Case}] \text{NP} & \Rightarrow \text{VP} \\
\text{VP}/[\text{acc}] \text{NP} & \Rightarrow \text{VP}/[\text{Case}] \text{NP}
\end{align*}
\]

The type-logical analysis of these constructions has a number of benefits. One has to do with the logical basis and the issue of polarity-sensitivity of underspecification. The discussion of this advantage is the major topic of the papers by Bayer and Johnson. The second advantage is a motivation for having underspecification not merely off-line but also during the grammatical derivations. We will now look at both of these advantages in turn.

### 9.2.2 Subsumption versus unification

Bayer and Johnson claim that their Lambek style analysis is superior to a constraint-based analysis. Why is this so? In the discussion about categorial unification grammars in Part II, we pointed out the difference between the way agreement is handled in a unification-based approach and what we called a subsumption-based approach as in the case of the Lambek calculus. In Bayer and Johnson (1995) this is put as follows:

"This [...] brings out one of the fundamental differences between the standard treatment of agreement in 'unification-based' grammar and this treatment of agreement in LCG [=Lambek Categorial Grammar]. In the 'unification-based' accounts agreement
is generally a symmetric relationship between the agreeing constituents: both agreeing constituents impose constraints on a shared agreement value, and the construction is well-formed iff these constraints are consistent. However, in the LCG treatment of agreement proposed here agreement is inherently asymmetric, in that an argument must logically imply, or be subsumed by, the antecedent of the predicate it combines with.

This difference becomes important when features are not completely specified. Otherwise subsumption and consistency requirements are equivalent. Coordination cases play an important role in arguing for an asymmetric approach to agreement checking, because in these cases underspecification seems "to play a crucial linguistic role, and cannot be regarded merely as an abbreviatory device for a disjunction of fully-specified agreement values." (Bayer and Johnson (1995)).

In unification-based analyses of coordination, subsumption often replaces unification. In Shieber (1992) there is a short discussion about the treatment of coordination of unlike-type constructions (Sag et al. (1985) and Dörre (1994)). He proposes the following coordination rule "to give a flavor for the proposal":

\[
E \rightarrow C \text{ Conj } D \\
E \subseteq C \\
E \subseteq D
\]

This should be read as a phrase structure rule in which \(E, C, D\) represent the feature structure categories with the restriction that the category of the mother must subsume the category of each conjunct. Note that one would probably want \(E\) to be the most informative category that subsumes the others (generalisation). This analysis works out nicely for simple examples like became wealthy and a Republican. We could, for instance, assume the following categories for wealthy and a Republican:

\[
\begin{align*}
\text{wealthy} & \quad \left[ \begin{array}{c}
\text{VERB} & + \\
\text{NOUN} & + \\
\end{array} \right] \\
\text{a Republican} & \quad \left[ \begin{array}{c}
\text{VERB} & - \\
\text{NOUN} & + \\
\end{array} \right]
\end{align*}
\]

If we combine these with the conjunction rule to produce wealthy and a Republican, the category of the combination will be the generalisation of these two:

\[
\text{wealthy and a Republican} \quad \left[ \begin{array}{c}
\text{NOUN} & + \\
\end{array} \right]
\]

The difference in grammaticality between remained wealthy and a Republican and *grew wealthy and a Republican can be explained if we assume that the verb remained selects complements of this category, whereas the verb grew selects complements of the category that are +verb and +noun.
If grew combines in the usual HPSG way with the complement wealthy and a Republican, via the subcategorisation, or the valence principle, then this specification of the complement unifies with the category of the combination of wealthy and a Republican. This means that the combination of the verb with its object leads us to the assumption that the object also bears the category above. However, this assumption violates the restriction on the coordination rules and this excludes the ungrammatical sentence.

So far so good, but Bayer and Johnson point out that when we also co-ordinate grew and remained (into grew and remained), the subsumption-based analysis as proposed by Shieber would result in the category:

\[
grew \text{ and } remained \quad [\text{OBJECT} \quad [\text{NOUN} + ]] \\
\]

This then, makes it also possible for this grammar to derive the ungrammatical *grew and remained wealthy and a Republican.

In the Lambek categorial grammar the ungrammatical sentence is not derivable. The category for grew will be V_P/A_P, the category for remained will be V_P/(A_P \lor N_P). In order to derive the sentence, we should be able to derive the category for grew to be V_P/(A_P \lor N_P) as well. However, this is not possible as the following attempt shows, because the inference step marked \( \perp \) is not valid. So this grammar does not overgenerate at this point.

\[
\frac{\text{grew} \quad [(A_P \lor N_P)^1]}{V_P/A_P \quad A_P \quad \perp \quad 1^1} \quad \frac{V_P}{V_P/(A_P \lor N_P)}
\]

Underspecification is sensitive to the polarity of the (sub)categorization: from A/B we can derive (A \lor C)/B, but not A/(B \lor C).

So the advantages of the Lambek-style approach to the unification-based approach can be summarised as follows.

- Subsumption defines the appropriate condition for the correct treatment of coordination cases.
- Type-logical grammars provide the correct logic: with positive and negative occurrences in formulas providing a correct derivational treatment \( \Rightarrow \) of the inclusion relations \( \subseteq \) that hold between expressions.

In Part II we saw that the motivation of authors like Dörre et al. (1996) and Francez (1997) was to build systems in which the capacities of unification-based reasoning were maximised in a Lambek style grammar. In the
previous part, we have shown that some typical functions of the unification-based tradition that these authors want to introduce in a type-logical setting can also be fulfilled by techniques already within the type-logical framework. This means that for these functions we do not need a unification-based account. What the analyses of Bayer and Johnson argue for is that in these cases they are not even appropriate.

### 9.2.3 The limits of off-line underspecification

We have just seen the benefits of having polarity-sensitive underspecified categories in the analysis of coordination constructions. These have been used in an on-line (derivational) strategy. We will now show that this is an essential feature of the analysis. As we already said in the introduction, in these coordination constructions underspecification has “to play a crucial linguistic role, and cannot be regarded merely as an abbreviatory device for a disjunction of fully-specified agreement values.” (Bayer and Johnson (1995)).

We now want to show that having underspecification only as an abbreviatory device blocks the derivation of the grammatical sentences of unlike type coordination we have been considering. We first discuss the case for the analysis using the boolean connectives and then for the multimodal case.

In the boolean analysis of the examples above we used the following lexical assignments.

\[
\begin{array}{lcl}
\text{remained} & : & \text{VP}/(\text{AP} \lor \text{NP}) \\
\text{wealthy} & : & \text{AP} \\
\text{a republican} & : & \text{NP} \\
\text{findet} & : & \text{VP}/(\text{NP} \land \text{acc}) \\
\text{hilft} & : & \text{VP}/(\text{NP} \land \text{dat}) \\
\text{Frauen} & : & \text{NP} \land \text{acc} \land \text{dat}
\end{array}
\]

Interpreting the underspecified entries as abbreviations we consider these assignments as a kind of pre-lexicon that has to be expanded first to all the fully specified entries (showing neither underspecification, nor overspecification).

<table>
<thead>
<tr>
<th>Pre-Lexicon</th>
<th>Lexicon</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\downarrow) (Expansion) (\downarrow)</td>
<td></td>
</tr>
</tbody>
</table>

\[
\begin{array}{lcl}
\text{remained} & : & \text{VP}/\text{AP} \\
\text{remained} & : & \text{VP}/\text{NP} \\
\text{wealthy} & : & \text{AP} \\
\text{a republican} & : & \text{NP} \\
\text{findet} & : & \text{VP}/(\text{NP} \land \text{acc}) \\
\text{hilft} & : & \text{VP}/(\text{NP} \land \text{dat}) \\
\text{Frauen} & : & \text{NP} \land \text{acc} \\
\text{Frauen} & : & \text{NP} \land \text{dat}
\end{array}
\]

However, none of the options we now have enable us to derive \textit{remained wealthy and a republican} and \textit{findet und hilft Frauen}.
COORDINATION AND THE LIMITS OF UNDERSPECIFICATION 191

\[
\begin{align*}
VP/AP & \circ AP \circ conjunction \circ NP /\ VP \\
VP/NP & \circ AP \circ conjunction \circ NP /\ VP \\
VP/(NP \land acc) & \circ conjunction \circ VP/(NP \land dat) \circ (NP \land acc) /\ VP \\
VP/(NP \land acc) & \circ conjunction \circ VP/(NP \land dat) \circ (NP \land dat) /\ VP
\end{align*}
\]

We show an abbreviated attempt to derive the German example in Prawitz notation.

\[
\begin{array}{c}
\text{findet und hilft} \\
\hline
\text{Frauen} \\
\text{findet und hilft} \\
\hline
\text{Frauen}
\end{array}
\]

\[
\begin{array}{c}
VP/(NP \land acc \land dat) \\
NP \land acc \\
fail \\
VP/(NP \land acc \land dat) \\
NP \land dat \\
fail
\end{array}
\]

The same argument that off-line underspecification does not lead to the correct analysis can also be shown for the multimodal analysis. Here, an abbreviatory perspective on underspecification means that we have to apply the inclusion postulates as lexical inferences deriving a lexicon with fully instantiated types from a pre-lexicon. We consider only the German example here. The lexicon we gave earlier now becomes a pre-lexicon.

\[
\begin{array}{l}
\text{Frauen} \quad \text{[Case]}NP \\
\text{findet} \quad VP/(\text{[acc]}NP) \\
\text{hilft} \quad VP/(\text{[dat]}NP)
\end{array}
\]

Applying the inclusion postulates we get:

\[
\begin{array}{l}
\text{Frauen} \quad \text{[dat]}NP \\
\text{Frauen} \quad \text{[acc]}NP \\
\text{findet} \quad VP/(\text{[acc]}NP) \\
\text{hilft} \quad VP/(\text{[dat]}NP)
\end{array}
\]

The following attempts show how this lexicon fails to derive the grammatical sentences.

\[
\begin{align*}
VP & \Rightarrow VP \quad [\text{dat}]NP \nRightarrow [\text{Case}]NP \\
VP/(\text{[Case]}NP) \circ [\text{dat}]NP & \Rightarrow VP \\
VP & \Rightarrow VP \quad [\text{acc}]NP \nRightarrow [\text{Case}]NP \\
VP/(\text{[Case]}NP) \circ [\text{acc}]NP & \Rightarrow VP
\end{align*}
\]

The problem with resolving 'ambiguity' already in the lexicon is that we do not have access to the ambiguous forms in the derivation. In the analysis of the neutral (Class I) constructions, on the other hand, it is essential that we leave the choice open.

**Summary** We can draw the following conclusions from the analysis of coordination cases with neutral factors.
• The subsumption-based, polarity-sensitive account of underspecification and overspecification provided by the type-logical grammars makes it possible to analyse the coordination cases with neutral factors.
• On-line underspecification of neutral factors is required for an appropriate analysis.

However, as we will see next, this does not mean that on-line underspecification is appropriate to reduce multiple lexical assignments for all cases. We will now turn to an analysis of the coordination cases with ambiguous factors. Despite the fact that the factor can combine with both conjuncts (Wasow's requirement) the coordination is ungrammatical.

9.3 Ambiguity and underspecification

We will first show that on-line underspecification of ambiguous factors leads to a lexicon that overgenerates, i.e. it accepts ungrammatical sentences. Next, we will discuss a way to treat these ambiguous factors to avoid this overgeneration (see Heylen (1997b) for an alternative).

9.3.1 Overgeneration

As we have seen above, the boolean type forming connectives ($\land$, $\lor$) can be used to deal with multiple type assignments to lexical items. Such a proposal was already made in Lambek (1961). Other authors that have made similar proposals include Morrill (1990) and Hendriks (1995), for instance. An example from the former is the word square which is ambiguous between an adjective and a noun and is assigned the type $(N/N) \land N$. This type assignment is a case of overspecification. Other examples provided by Morrill are with: $((N \land N) \land ((NP \land (NP \land S))) / NP$ and wants: $(NP \land S) / (VP \lor (NP \land VP))$.

Above we pointed out that the use of underspecified (or overspecified) categories can be applied fruitfully to the analysis of certain coordination constructions, namely those involving the coordination of unlike types. We provided two examples, one in which the argument position of a functor is underspecified and another one in which the argument itself is overspecified. But for both examples we will now give an apparent counter-example to indicate the problem with such an analysis.

The first example was remained wealthy and a republican, where remained is assigned an underspecified type to indicate that it combines with both adjective and noun phrases. Now let us consider the verb can which is ambiguous between the types $VP / NP$ (I can tuna) and $VP / VP$ (I can work for a living) and therefore seems to be a candidate for the categorisation $VP / (NP \lor VP)$. However, this category makes it possible to derive the sentence I can tuna and work for a living.
Another illustration of the same problem involves overspecification. Above we have shown how the phrase *findet und hilft Frauen* is derivable in the system using the following assignments: *findet*: $VP/\overline{(NP \land acc)}$, *hilft*: $VP/\overline{(NP \land dat)}$, *Frauen*: $NP \land acc \land dat$. Again, we run into problems if we assume, analogously, that the expression *the sheep* is ambiguous between singular and plural and consequently is assigned the type $NP \land sg \land pl$. We cannot prevent *the sheep walks and graze* from becoming derivable.

Similarly, the sentence *John wants to go and Susan to go* becomes derivable if we follow Morrill’s suggestion to assign *wants* the type $(NP\backslash S)/(VP \lor (NP \bullet VP))$.

The same problem arises with the multimodal analysis. We can illustrate this with postulates and a lexic for the German words *Sie, singt* and *singen* parallel to the assignments for *findet, hilft* and *Frauen*. We use the following unary modalities to encode the values for number and case: $[sg]$, $[pl]$, $[acc]$, $[dat]$, $[Num]$, $[Case]$. Here $[Num]$ and $[Case]$ represent the overspecified or general values. Their behaviour is fixed by the following inclusion postulates, which allow us to infer specific values from the general features.

$[Case]A \Rightarrow [i]A \ (i \in \{acc, dat\})$

$[Num]A \Rightarrow [i]A \ (i \in \{sg, pl\})$

With these assignments, we can also derive the ungrammatical *Sie singt und singen* using the lexical assignments above. In other words, we meet the same overgeneration problem as we did with the boolean overspecification.
9.3.2 No underspecification

Basically, the solution we suggest is to not to use underspecified categories for the factors involved in violations of Wasow's generalisation. Following Pullum and Zwicky (1986) one should make a distinction between ambiguous and neutral expressions. If we restrict the use of underspecified categories to the neutral forms this solves the problem. We can make the distinction between neutral and ambiguous expressions visible by considering the models for some examples.

First consider the case of the singular, nominal noun phrase *Kim: NP \and \text{sg} \and \text{nom}*. This expression is modelled by a linguistic resource (a form-meaning pair) that can be pictured as follows. Note that we identify the element by its orthographic representation.

This seems to be a plausible interpretation for the decomposition of a category into features where each feature denotes a property of the same linguistic resource. The question is whether we want the same interpretation to hold for the booleans in the cases of multiple type assignment to *square, (N/N)\and N, or can, (VP/NP)\and (VP/VP)*. The answer is that we should not be using \and here with this interpretation because it fails to express that there are actually two expressions *square* which happen to be homophonous. By this we mean that there are no occurrences of *square* which are both adjective and noun at the same time. In an expression like the *square square*, the first occurrence is an adjective and the second a noun. It is not the case that each occurrence is both an adjective and a noun at the same time. Because linguistic resources are form-meaning pairs, we should assume the existence of at least two expressions *square*, that differ in their semantic component and their syntactic category. The picture for *square* looks as follows:
The point here is that the element identified by its orthographic representation \textit{square} in \( N/N \) is different from the element \textit{square} in the set \( N \). For \textit{Frauen} with the assignment \( NP \land acc \land dat \) the situation is rather like that for \textit{Kim}, as we want this noun phrase to be both accusative and dative at the same time in the expression \textit{findet und hilft Frauen}.

These examples show that we do not want to assign overspecified categories to ambiguous words (of type \textit{want}) but only to neutral words.

In Morrill (1994), it is proposed to distinguish between a semantically active, \( \land \), and a semantically non-active, \( \sqcap \), connective. The latter corresponds to the connective \( \land \) as we have been using it. An expression is assigned the type \( A \land B \) if it is in \( A \) with semantics \( x \) and it is in \( B \) with semantics \( y \). An expression is assigned the type \( A \sqcap B \) if it is in \( A \) and in \( B \) with semantics \( x \) in both cases. This distinction relates to the difference between ambiguous and neutral forms. But this distinction does not allow the use of boolean types to ambiguous forms either. At least not to those that cannot figure in the coordination constructions. For instance, if we would assign the German word \textit{Sie} a semantically potent boolean type with the semantics being a pair of terms (one for the singular and one for the plural interpretation) then we can still not prohibit the combination with the conjunction of a singular and a plural verb. In fact, the semantics of \textit{Sie} would enable us to provide a semantic reading for the ungrammatical sentence.

The idea that in a specific construction a word cannot be used in two different senses at the same time, i.e. the Anti-Pun Ordinance from Zaenen and Karttunen (1984) can be used as further evidence against the introduction of semantically potent boolean types.

Some discussion on the explanation of why some forms are neutral whereas others are ambiguous can be found in Ingria (1990), Pullum and Zwicky (1986) and Zaenen and Karttunen (1984). Not surprisingly, attempts to capture the distinction are often sought in a difference between purely formal features and those that have semantic import. Ingria (1990) points out:

- Syntactic features which have semantic ramifications, such as number on nouns, tense on verbs, degree on adjectives, are never neutralised (underspecified). They are always fully specified and items which seem to be underspecified with regard to them are, in fact, ambiguous items with distinct, fully specified representations.

- Purely formal syntactic features, on the other hand, can be neutralised, producing truly underspecified representations [...].

There are however some complications with these semantic explanations. We give an example here.

Above we saw that the feature \([\text{Num}]\) in German as assigned to \textit{Sie} is ambiguous and therefore not conjoinable. Following the discussion on
neutral and ambiguous features as summarised in the conclusions by Ingria above, we might expect that number bears semantic content and can therefore typically not be neutralised. Now consider the following noun phrases (also discussed in Ingria (1990)).

\[
\begin{align*}
&\text{der Antrag des oder der Lehrer/Lehrers} \\
&\text{the petition (of) the}_{\text{gen/sg}} \text{ or the}_{\text{gen/pl}} \\
&\text{teacher}_{\text{gen/sg}}/\text{teachers}_{\text{gen/pl}} \\
&\text{der Antrag des oder der Dozenten} \\
&\text{the petition (of) the}_{\text{gen/sg}} \text{ or the}_{\text{gen/pl}} \text{ lecturers}_{\text{gen/sg/pl}}
\end{align*}
\]

These phrases show that Dozenten in German is neutral between singular and plural and should therefore be assigned a neutral feature. This cannot be \([\text{Num}]\) as above because this is ambiguous and not conjoinable. From a technical point of view the solution to this conflict is rather straightforward. We can simply assume a feature theory in which for each morphosyntactic distinction (number, person, gender, etc) there are both ambiguous and neutral generalisations; say \([\text{Num}]\) and \([\text{NUM}]\) for the case at hand. The correct distribution is fixed by the lexicon. For more discussion on which features are prone to neutralise and which ones are not see Bayer (1996) and Pullum and Zwicky (1986). It is clear from this examples that the issue of semantics is quite a tricky one: why is \textit{Sie} ambiguous but \textit{Dozenten} neutral? For some further suggestions see Bayer (1996).

**Another problem solved** This diagnosis of the problem and the remedy of not using underspecification for the ambiguous cases also solves the problem discussed in Moortgat (1997).

In the analysis of the coordination cases of Johnson and Bayer (1995) we can distinguish between two functions of the boolean operators: overspecification or ambiguity marking and ‘feature decomposition’ (see also Kanazawa (1992)).

Now consider the following example. The pronoun \textit{her} can be considered to be ambiguous between a personal (NP) and a possessive (NP/N) pronoun. In the former reading it bears accusative case (acc). Combining this information into a single type leads to the assignment to \textit{her} of the type (NP \land acc) \land (NP/N). However, now the problem arises of how to prevent the acc marking, which is true of the personal pronoun reading, to reassociate with the possessive type: NP/N. Moortgat (1997) therefore warns against the use of the boolean operator as two functions: feature decomposition and ambiguity marking. As we hope to have made clear by now, this problem is not so much caused by mixing the two functions of the boolean connectives, but rather by using them to reduce the number of
assignments to (certain) ambiguous words. Even assigning her the category \( \text{NP} \land (\text{NP}/\text{N}) \) is problematic because it suggests that her can occur in contexts in which it acts both a noun phrase and as a possessive pronoun. Similarly, the assignment to \( \text{wants:}\  (\text{NP}\backslash\text{S})/(\text{VP} \lor (\text{NP} \bullet \text{VP})) \) taken from Morrill leads to overgeneration problems in the cases of coordination constructions. The relevant constructions here are those that are apparent violations of Wasow's generalisation. The same holds for her, where the picture looks like this.

![Diagram](image)

However, this is not the picture that fits the interpretation of the category \( (\text{NP} \land \text{acc}) \land (\text{NP}/\text{N}) \). It is clear that her is not in \( v(\text{NP}) \land v(\text{acc}) \land v(\text{NP}/\text{N}) \) and should therefore be assigned two types.

### 9.3.3 Off-line underspecification and ambiguity

What we have demonstrated above is that the ambiguous factors should not be assigned underspecified categories in an on-line strategy similar to the neutral factors.

The main reason that we are discussing coordination constructions in this chapter is to evaluate the status of underspecification in grammars (on-line/off-line, polarity sensitive/insensitive, subsumption/unification based). The grammatical cases with coordinates of unlike types involve a neutral factor and show how we need on-line, polarity-sensitive, underspecification in grammatical derivations. Underspecification as abbreviation falls short in these cases. The ungrammatical cases with coordinates of unlike types involve an ambiguous factor and show the limits of allowing underspecification within grammatical derivations. What we now want to show is that underspecification as abbreviation is fine for these cases.

**Off-line strategy for Ambiguity** The reason that we can use off-line abbreviations for ambiguous factors is essentially the same as the reason why we cannot use it for neutral factors. Both times it blocks the derivation of coordination constructions with unlike types. In the first case this is
exactly what we want, but in the second case we do not want this. So consider the following pre-lexicon.

\[
\begin{array}{ccc}
\text{Frauen} & \text{[Case]}\text{NP} & \text{Sie} & \text{[Num]}\text{NP} \\
\text{findet} & \text{VP}/([\text{acc}]\text{NP}) & \text{singt} & ([\text{sg}]\text{NP})/s \\
\text{hilft} & \text{VP}/([\text{dat}]\text{NP}) & \text{singen} & ([\text{pl}]\text{NP})/s \\
\end{array}
\]

Applying the inclusion postulates we get the following expanded lexicon.

\[
\begin{array}{ccc}
\text{Frauen} & \text{[dat]}\text{NP} & \text{Sie} & \text{[sg]}\text{NP} \\
\text{Frauen} & \text{[acc]}\text{NP} & \text{Sie} & \text{[pl]}\text{NP} \\
\text{findet} & \text{VP}/([\text{acc}]\text{NP}) & \text{singt} & ([\text{sg}]\text{NP})/s \\
\text{hilft} & \text{VP}/([\text{dat}]\text{NP}) & \text{singen} & ([\text{pl}]\text{NP})/s \\
\end{array}
\]

Above we saw that this incorrectly prohibits the derivation for \textit{findet und hilft Frauen}. However, by the same kind of reasoning it correctly prohibits the derivation for \textit{Sie singt und singen}.

What we need for these cases is an analysis in which ambiguous underspecification is distinguished from neutral underspecification.

**Keeping them Apart** The simplest way to account for both cases is to distinguish between the behaviour of the different forms of underspecification. This could easily be done by introducing two types of overspecified or underspecified features: the neutral and the ambiguous ones. The former are not expanded in the move from the pre-lexicon to the lexicon whereas the latter must be expanded. This requires us to distinguish between two types of inclusion postulates: those that apply off-line to the pre-lexicon (lexical inferences) and those that apply on-line during the grammatical derivations. We will mark the 'off-line' as bold-face. For the German example, we can use the following assignments.

\[
\begin{array}{ccc}
\text{Frauen} & \text{[Case]}\text{NP} & \text{Sie} & \text{[Num]}\text{NP} \\
\text{findet} & \text{VP}/([\text{acc}]\text{NP}) & \text{singt} & ([\text{sg}]\text{NP})/s \\
\text{hilft} & \text{VP}/([\text{dat}]\text{NP}) & \text{singen} & ([\text{pl}]\text{NP})/s \\
\end{array}
\]

Applying the off-line inclusion postulates to the boldface sorts we get the following expanded lexicon.

\[
\begin{array}{ccc}
\text{Frauen} & \text{[Case]}\text{NP} & \text{Sie} & \text{[sg]}\text{NP} \\
\text{findet} & \text{VP}/([\text{acc}]\text{NP}) & \text{Sie} & \text{[pl]}\text{NP} \\
\text{hilft} & \text{VP}/([\text{dat}]\text{NP}) & \text{singt} & ([\text{sg}]\text{NP})/s \\
\text{singen} & ([\text{pl}]\text{NP})/s \\
\end{array}
\]
9.4 The use of underspecification

In the previous sections we have dealt with the issue of multiple type assignments to lexical expressions in the context of coordination constructions. The coordination cases we have presented point out that a distinction has to be made between neutral and ambiguous expressions. There is another case that has to be considered as regards underspecification which we illustrated by underspecification for number of the object noun phrase with (English) transitive verbs. This class provided the original motivation for the simplification of lexical assignments by underspecification. Prepositions also do not care whether they combine with singular or plural noun phrases (and do not care about many other properties either). Having no underspecification for these cases results in a massive increase in the number of categories that have to be assumed and the number of assignments to lexical items. Typically, this kind of pattern belongs to a whole class of words and involves a generalisation across all possible values for some attribute. Having no way to account for this kind of generalisation in a grammar requires making distinctions which are not really there (which is worse than merely increasing the size of the lexicon).

A typical difference between neutralisation and the "don't care" cases is that neutralisation is rather idiosyncratic; it often involves a particular word instead of a whole class of words. The phenomenon seems therefore to be more exceptional, for instance the fact that Frauen can be checked for both accusative and dative at the same time. The constructions in which such neutral items have to be assumed are also special. It may be that some lexical item must be underspecified for only a few values for some attribute instead of the whole range. In this case, it is not simply possible to multiply out the possible category assignments to the neutral word because then it will have no assignment that fits the special context in which the word appears in two functions at the same time.

Underspecification for the ambiguous words is problematic as we have pointed out because of possible contexts like coordination. As with the neutral words, these cases seem idiosyncratic as well. Expanding the number of lexical assignments and not having underspecification for these cases will not involve a big cost to the grammar. The linguistic status of an underspecified entry is also unclear. What is the status of the entry sheep as something that is both singular and plural? In any utterance in which the word occurs it will either be taken as singular or plural but not as both at the same time. The underspecified entry here is then simply an artificial construct, so it seems.

The question that comes to mind at this point is how the difference between the on-line versus the off-line strategies relates to the differences in the cases of underspecification that we have distinguished: don't care, neutral, ambiguous.

We can allow both ambiguous, neutral and don't care forms to be un-
derspecified in the pre-lexicon. However, to avoid overgeneration in the coordination cases, the ambiguous forms should not appear underspecified in the final lexicon that results. For the neutral forms the underspecified forms should be able to enter the derivation to account for the coordination constructions with neutral factors. It seems that the don’t care entries can be treated either way.

Notice that this means that a combination of both strategies seems to be optimal. In any case, we cannot choose completely for the off-line strategy because than we miss out on the correct treatment of the neutral forms. If we opt for a strategy in which all underspecified forms can also appear in the grammatical derivations, then we should take care not to assign underspecified categories to ambiguous forms.

**Conclusion**

In general, we draw the following conclusions from these coordination cases.

- As is argued in the work by Ingria and Bayer and Johnson, unification strategies can be sub-optimal: "unification — variable-matching combined with variable substitution — is the wrong mechanism for effecting agreement."

- Instead of unification, subsumption checking provides a better means to analyse certain constructions. This claim is also found in the literature on unification grammars in analyses of coordination constructions (Shieber (1992) and Sag et al. (1985)).

- The problem with the subsumption based approaches in the feature-structure theories is that there is no appropriate interaction between the procedure of subsumption checking on the one hand and the logic of combination and selection on the other.

- More specifically, a polarity-sensitive logic is required and is defined by the type-logical grammars.

- The use of on-line underspecification for ambiguous words may lead to overgeneration and should therefore be restricted to neutral items.

In the previous sections, we hope to have made clear that combining a type-logical approach and a constraint-based approach requires a precise division of labour for the underspecification of linguistic information.
Summary of the fourth part

In this last part, we have looked at the status of underspecification with respect to the general organisation of grammatical information. In the previous part we defined a resource-sensitive, deductive perspective on cross-classification and organisation and opposed this to the constraint-based perspective. However, in this part we wanted to point out how the two traditions can be used in a complementary way.

Because the constraint-based framework provides a general purpose logic it can also be used to encode and organise a type-logical grammar. In this way we can define a hierarchical type-logical lexicon in the spirit of the hierarchical lexicons for grammars like HPSG.

With respect to underspecification, we distinguished between underspecification in the lexicon (pre lexical insertion) that is resolved off-line and the use of underspecified types in a grammatical derivation. From the discussion of some cases of co-ordination it appears that polarity-sensitive on-line underspecification is needed for adequate linguistic description. This can be implemented along the lines we developed in the previous part using inclusion postulates. The discussion also showed that a purely constraint-based approach fails to provide an adequate analysis of certain cases because it does not have polarity-sensitive underspecification. However, a constraint-based approach may still be useful to organise grammatical information, particularly when it is used off-line, to express lexical and other generalisations.
Conclusion

We set out to define a grammatical framework in which the type-logical perspective on grammatical composition is combined with a refined system of classification.

We have demonstrated how the multi-modal version of the Lambek calculus provides the basis not just for the definition of compositional structure using the binary connectives, but also for refined cross-classification using unary connectives. The resource-conscious logic governing the connectives constitutes both a dedicated logic of linguistic composition and selection on the one hand and a logic of feature marking and checking on the other. This use of the unary operators obviates the need to introduce feature structures or similar techniques for cross-classification into the formalism.

We have focused on morphosyntactic aspects of linguistic description because techniques for other aspects are already well established. We have proposed to use unary connectives as (morphosyntactic) feature decorations to decompose the information that is often encoded by basic categories. The logical rules define the basic feature checking procedure. These are complemented by inclusion postulates defining a partial ordering that is used to express generalisations through underspecification and by interaction postulates defining the distribution of morphosyntactic information through the compositional structure.

Our analysis shows the expressivity of current multimodal categorial grammars as no extra devices need to be introduced to the formalism. Moreover, the kind of decomposition of categorial information as we propose it and the way it is used and manipulated in the grammar has a number of additional advantages which, taken together, distinguish this approach from others proposed in the literature. We list three advantages here.

- Modal (morphosyntactic) decorations are allowed on both basic and complex categories.
- Underspecification is sensitive to the polarity of (sub)categories and modal decorations.
- Distribution postulates can be used instead of techniques like structure or variable-sharing.

These properties make it possible to maximise the use of underspecification. Polarity-sensitivity is also important for an adequate account of co-ordination constructions. These constructions constitute an important motivation for the subsumption-based, asymmetric account of underspecification as it is defined in type-logical grammars.

In the discussion of these constructions we have emphasised that it is important to distinguish between different cases of underspecification.
An analysis of these distinctions led us to reconsider the combination of the type-logical framework and constraint-based approaches. Although we have defined a framework in which many of the functions that feature structures serve in constraint-based theories are taken over by unary modalities and their logical and structural properties, this does not mean that a constraint-based perspective on the organisation of lexical information or the organisation of the postulates and the resource management modes cannot be useful as a notational tool for type-logical grammars.
Samenvatting in het Nederlands

In dit proefschrift wordt betoogd dat het overbodig is om (multimodale) type-logische grammatica's uit te breiden met kenmerkstructuren zoals die in vele computationeel taalkundige theorieën gebruikt worden. We laten zien hoe de voordelen van kenmerkstructuren in taalkundige beschrijvingen bereikt kunnen worden met andere middelen die reeds in het multimodale kader voorhanden zijn. Dit instrumentarium biedt zelfs extra voordelen. Zo maken ze een analyse van bepaalde nevenschikkingen mogelijk die niet voorhanden is in de theorieën die uitsluitend gebruik maken van kenmerkstructuren en kenmerklogica's. Hoewel betoogd wordt dat uitbreidingen van type-logische grammatica's met kenmerkstructuren niet nodig zijn, kunnen deze laatste wel gebruikt worden als notationeel instrument om de informatie in het type-logische lexicon te representeren en te structureren.

Het betoog is opgebouwd uit vier delen. In het eerste deel geven we een karakterisering van de probleemstelling en verstrekken we de achtergrondinformatie die nodig is voor een goed begrip van het type-logische kader dat centraal staat in dit proefschrift. In het tweede deel bespreken we de kwaliteiten en gebreken van de diverse voorstellen die eerder gedaan zijn om het categoriale model uit te breiden met kenmerkstructuren of met vergelijkbare technieken. Vervolgens stellen we in het derde deel onze eigen benadering voor. Ten slotte vergelijken we in het vierde deel onze voorstellen opnieuw met kenmerkstructuren. Door een analyse van de beschrijvingstechnieken voor bijzondere nevenschikkingsconstructies komen we tot twee conclusies. De eerste is dat het verfijnde type-logische instrumentarium een betere beschrijving van deze constructies mogelijk maakt dan de (niet type-logische) theorieën die gebruik maken van kenmerkstructuren. De tweede is dat we de twee modellen niet volledige antagonistisch tegenover elkaar moeten stellen maar dat ze een complementaire positie kunnen innemen. Kenmerkstructuren kunnen gebruikt worden om type-logische grammatica's te noteren en te structureren.

Deel I Achtergrond en Probleemstelling (Hoofdstuk 1-4)

In het eerste hoofdstuk geven we aan welke taken van de grammatica in dit proefschrift besproken zullen worden en op welke manier verschillende soorten grammatica's deze taken vervullen. Onder de elementaire taken van een grammatica vallen: een specificatie van de uitdrukkingen die tot een taal behoren, een indeling van deze uitdrukkingen in categorieën en een beschrijving van de opbouw van de uitdrukkingen. Over een andere elementaire taak, een specificatie van de manier waarop betekenissen toegekend worden aan uitdrukkingen wordt verder niet uitgeweid. We stellen
in dit hoofdstuk ook twee basissystemen voor om talen te beschrijven. Het eerste systeem is een context-vrije herschrijfgrammatica en het tweede een categoriale grammatica.

In het tweede hoofdstuk presenteren we het grammatica-model dat centraal staat in het proefschrift: type-logische grammatica's, ook wel multimodale categoriale grammatica's genoemd. Zonder in detail te treden kunnen we de werking van dit type grammatica als volgt schetsen. Een categoriale grammatica kent aan iedere basisuitdrukking (woord) in een taal een categorie (ook wel type genoemd) toe in een lexicon. Een categorie is een formule van een logische (modale) taal die als volgt kan worden gespecificeerd.

\[ C ::= B \mid C \cdot_i C \mid C /_i C \mid C\setminus_i C \mid \circ_i C \mid \square_i C \]

\( B \) is een verzameling atomaire symbolen van basiscategorieën. Met de subscripts \( i \) worden verschillende instanties van de logische connectieven \( /,\cdot,\setminus,\circ,\square \) onderscheiden.

Een woord behoort tot de taal wanneer het in het lexicon staat. De grammaticaliteit van complexe uitdrukkingen wordt bepaald door een logische afleiding. Deze afleiding maakt gebruik van de logische regels voor de connectieven.

Willen we weten of \textit{Jamie kookt een kreeft} een welgevormde zin is, dan zoeken we de categorieën van de individuele woorden op in het lexicon en vormen hiermee een gestructureerde term. In het algemeen zijn termen \( T \) van de volgende vorm.

\[ T ::= C \mid (T \circ_i T) \mid \langle T \rangle_i \]

De structuur van de term weerspiegelt de structuur van de zin. Wanneer we het volgende lexicon gebruiken dan blijkt dat met de gepaste afleidingsregels \textit{Jamie kookt een kreeft} als zin (s) kan worden afgeleid. In dit voorbeeld laten we de subscripts bij de connectieven achterwege.

\textbf{Lexicon}

\begin{tabular}{ll}
\textit{Jamie} & NP \\
\textit{kookt} & (NP\setminus S)/NP \\
\textit{een} & NP/N \\
\textit{kreeft} & N
\end{tabular}

Wanneer we in de derivatie de categorieën in de term \( T \) systematisch vervangen door de gepaste woorden dan ziet de afleiding voor deze zin er als volgt uit.
Voor deze afleiding gebruiken we de volgende afleidingsregels.

\[
\begin{align*}
\Delta_1 & \vdash A/B & \Delta_2 & \vdash B \\
\Delta_1 \circ \Delta_2 & \vdash A
\end{align*}
\]

\[
\begin{align*}
\Delta_1 & \vdash B & \Delta_2 & \vdash B/A \\
\Delta_1 \circ \Delta_2 & \vdash A
\end{align*}
\]

In het derde hoofdstuk bespreken we het nut van kenmerkstructuren in een grammatica. We noemen de volgende drie voordelen.

- Ze leveren de mogelijkheid om uitdrukkingen op basis van meerdere criteria in te delen (cross-classificatie).
- Ze ordenen uitdrukkingen in een taxonomie met subklassen en superklassen. Dit maakt het mogelijk om generalisaties door onderspecificatie uit te drukken.
- Door informatie te decomponeren in verschillende onderdelen is het mogelijk om algemene uitspraken te doen over de onderdelen afzonderlijk.

In het vierde hoofdstuk vragen we ons af voor welke onderdelen van de taalkundige beschrijving een uitbreiding van het type-logische kader met kenmerkstructuren gewenst zou zijn. Meer bepaald willen we een karakterisering geven van die onderdelen die in het type-logische kader nog geen adequate behandeling krijgen. Hiervoor beschouwen we de verschillende kenmerken die in de theorie van Head-Driven Phrase structure Grammar worden gebruikt. We stellen ons telkens de vraag bij ieder kenmerk of type-logische grammatica's baat zouden vinden bij uitbreidingen met dit kenmerk. Onze conclusie is dat voornamelijk voor de beschrijving van morfosyntactische eigenschappen een uitbreiding van het type-logische kader gewenst is.

**Deel II** Literatuur (Hoofdstuk 5-6)

In het vijfde hoofdstuk bespreken we de technieken die in de literatuur zijn voorgesteld om categoriale grammatica's uit te breiden zodat cross-classificatie, onderspecificatie en het generaliseren over afzonderlijke kenmerken mogelijk wordt.

In het zesde hoofdstuk wordt gekeken naar de voor- en nadelen van deze systemen. Op deze manier komen we tot een lijst van eigenschappen waaraan een optimaal systeem zou moeten voldoen.

**Deel III** Resource logica als oplossing (Hoofdstuk 7-8)

In hoofdstuk zeven stellen we voor om de operatoren $\boldsymbol{\diamond}_1$ en $\square_1$ (die reeds in type-logische grammatica's gebruikt worden om bijvoorbeeld structurele manipulaties te controleren en om lokale domeinen te definiëren) aan te wenden voor de beschrijving van morfosyntactische eigenschappen. Deze techniek voldoet aan de eisen die we in het tweede deel hebben gesteld aan een optimaal systeem. We noemen hier drie voordelen van deze techniek.
Het laat morfosyntactische decoraties toe op zowel atomaire categorieën als op complexe categorieën.
Het is polariteits-gevoelig.
Het laat onderspecificatie met co-variatie toe.

Deze laatste eigenschap lichten we toe met een voorbeeld. Het Nederlandse lidwoord *de* combineert zowel met enkelvoudige (*kreeft*) als met meervoudige naamwoorden (*kreeften*). De resulterende naamwoordengroepen dragen hetzelfde kenmerk. Wanneer we het lidwoord onderspecifice- ren voor getal moeten we er wel voor zorgen dat de waarde voor het getal van de groep overeenstemt met de waarde van het nomen. Dit noemen we een probleem van co-variatie. In onze beschrijving wordt dit opgelost met behulp van distributiepostulaten.

In het algemeen bestaat onze beschrijving van morfosyntactische eigenschappen uit verschillende componenten. De logische regels voor de operatoren $\lozenge$ en $\boxdot$ controleren of kenmerken al dan niet aanwezig zijn en de structurele regels regelen onderspecificatie en de distributie van morfosyntactische informatie in complexe uitdrukkingen. In hoofdstuk 7 introduceren en illustreren we de basistechniek. Deze kan met een eenvoudig voorbeeld worden verduidelijkt. We schrijven daarbij $\boxdot_i$ als $[i]$.

**Lexicon**

- *de* | $\mathbb{N}u$m$\langle NP/sN \rangle$
- *kreeft* | $\mathbb{E}nkv$ | $N$
- *kookt* | $\mathbb{E}nkv$ | $NP\setminus cS$

Gegeven dit lexicon, leiden we de zin *de kreeft kookt* als volgt af.

**Afleiding**

$$
\begin{align*}
\text{de} & \vdash [\mathbb{N}um]\langle NP/sN \rangle \\
\langle \text{de}\rangle_{\mathbb{N}um} & \vdash NP/sN \\
\langle \text{de}\rangle_{\mathbb{E}nkv} & \vdash NP/sN \\
\langle \text{de}\rangle_{\mathbb{E}nkv} \circ_s \langle \text{kreeft}\rangle_{\mathbb{E}nkv} & \vdash NP \\
\langle \text{de} \circ_s \text{kreeft}\rangle_{\mathbb{E}nkv} & \vdash NP \\
\langle \text{de} \circ_s \text{kreeft}\rangle & \vdash [\mathbb{E}nkv]\langle NP \rangle \\
\text{kookt} & \vdash [\mathbb{E}nkv]\langle NP \setminus cS \rangle \\
\langle \text{de} \circ_s \text{kreeft} \rangle \circ_c \text{kookt} & \vdash s
\end{align*}
$$

We gebruiken hiervoor de volgende regels.

$$
\begin{align*}
\Delta & \vdash \boxdot_i C \\
\langle \Delta\rangle_i & \vdash C \\
\Delta & \vdash \boxdot_i C \\
\langle \Delta\rangle_{\mathbb{N}um} & \vdash C \\
\langle \Delta\rangle_{\mathbb{E}nkv} & \vdash C \\
\langle \Delta\rangle_{\mathbb{E}nkv} \circ_s \langle \Delta_2\rangle_{\mathbb{E}nkv} & \vdash C \\
\langle \Delta_1 \circ_s \Delta_2\rangle_{\mathbb{E}nkv} & \vdash C
\end{align*}
$$
De eerste twee regels zijn de logische regels voor $\square$. Ze gelden voor alle indices $i$. De andere regels gelden voor specifieke indices. De eerste is een inclusieregel die zegt dat de markering $Num$ vervangen kan worden door de markering $sg. Num$ is een onderspecificeerde markering. De tweede regel is een distributieregel die zegt dat wanneer twee delen verbonden door $\circ_s$ allebei een $enkv$ markering dragen, we deze kunnen vervangen door een markering op het geheel. Met deze regel dwingen we co-variatie af omdat de eis is dat zowel het geheel als de delen identiek gemarkerd worden.

In het achtste hoofdstuk laten we de taalkundige gebruiksmogelijkheden zien van diverse opties en verfijningen die aan het basissysteem kunnen worden aangebracht.

**Deel IV** Onderspecificatie opnieuw bekeken (Hoofdstuk 9)

In het laatste hoofdstuk bespreken we de analyse van nevenschikkingsconstructies die voor kenmerk-gebaseerde theorieën problemen op leveren. We laten zien dat het fijngevoelige logische apparaat van de type-logische grammatica's wel de goede analyse levert. Vooral van belang in dit verband is het polariteits-gevoelige redeneren in de type-logische aanpak die in de kenmerkgrammatica's ontbreekt.

Een ander punt dat belicht wordt in dit hoofdstuk is het verschil in de status van onderspecificatie voor verschillende gevallen. Het gegeven dat het Engelse woord *sheep* zowel in het enkelvoud als in het meervoud kan optreden is van een andere aard dan het gegeven dat het Duitse *Frauen* zowel een accusatief als een datief vorm is. Immers, *the sheep walk and graze* is niet grammaticaal terwijl *Er findet und hilft Frauen* dat wel is (waarbij *findet* een accusatief en *hilft* een datief object naast zich wil hebben). De analyse van deze verschillen in status leiden ons ertoe om de relatie tussen het type-logische kader en het kenmerk-gebaseerde vanuit een ander perspectief te beschouwen. Behalve op het contrast tussen de twee theorieën, wijzen we in dit hoofdstuk ook op hun complementariteit en schetsen de contouren van een kader waarin de voordelen van beide gecombineerd worden.
Index

/, 16
d, 30
\diamond, 30
c, 27
\lambda-calculus, 21
\lambda-abstraction, 29
\lambda-term, 21
\lambda-types, 21
\exists, 52
\forall, 101
\land, 101

AB, 16

grammars, 16
language, 16
phrase structure, 18
string set, 17
trees, 17
abbreviation, 94
abstraction, 45
accessibility relation, 26, 30
additive, 102
agreement, 150
agreement postulate, 141
antecedent, 27
anti-pun ordinance, 182
application, 64, 71
appropriateness conditions, 49
ARG, 96
argument, 16
argument composition, 77
argument lowering, 28
arity, 29
associative, 12
associative Lambek calculus, 27
associativity, 33
attribute, 46
attribute-value matrix, 47
axiomatic form, 30
axiomatic presentation, 26
axiomatic rule, 37

basic category, 16
boolean, 102, 185
categorial grammar, 15
categorial unification grammar, 93
categorisation, 15
category, 12
category label, 13
CFG-language, 12
check, 41
checkee, 145
checker, 145
checking procedure, 133
Chomsky’s generalisation, 181
classification, 15
clitics, 151
covariation, 56, 113, 140
complex category, 16
complex type, 16
concatenation, 12
conditionalisation, 25
connective, 16
constant, 70, 103
costituent structure, 13, 34, 73
CONTENT, 79
context-free grammar, 11
contraction, 33
cross-classification, 53
cross-serial dependencies, 39
Curry-Howard, 29

DATR, 66
declarative unit, 106
denotation
category, 13
derivalational, 181
DIR, 96
disjunction, 188
distinguished occurrence, 27
distribution postulates, 37, 138–140
domain, 16
dominance, 15
DTRS, 70
Dutch, 136, 158
dynamic predicate logic, 167
elimination rule, 27
endocentric, 148
exocentric, 148
extension, 52

factor, 182
ambiguous, 183
neutral, 183
factorisation, 45
factorise, 53
feature, 46
bundle, 45
cancellation, 147
checking, 129
description, 48
description language, 46, 48
distribution principle, 59
non local, 77
percolation, 118
structure, 45, 46
feature distribution, 113
first-order term, 103
first-order terms, 93
four-dimensional cube, 164
frame, 21, 22, 26
frame condition, 38
frame constraints, 147
framework, 70
function composition, 77
functional type, 21
functionality constraint, 46
functor, 16

gender resolution, 184
general features, 187
general purpose, 53
generalised Lambek calculus, 38
generating functions, 133
Gentzen sequent calculus, 40
Gentzen-style, 40
global constraint, 106
grammar formalism
constraint-based, 45

head feature principle, 64

hierarchical classification, 53
homophonous, 182
hybrid system, 102
hypothetical reasoning, 25, 41, 77
hypothetical term, 28

inclusion postulates, 37
inheritance, 51, 65
inheritance hierarchy, 99
interaction postulates, 36
interpretation
category, 13, 17
complex term, 31
multimodal types, 38
introduction rule, 27

Kripke model, 47
Kripke-style semantics, 26

labelled deductive system, 80
lambda term, 21
Lambek calculus, 28
layering, 93
left rule, 40
leftward application, 16
lexicon, 13
linear logic, 37
linguistic resource, 26
logic
fibred, 105
first-order, 104
layered, 105
propositional, 101
long-distance dependencies, 77
lowercase, 138
LP, 33

macro, 66, 100
markings, 130
meaning, 29, 65
merge, 133
minimal recursion semantics, 80
minimalist program, 133, 157
mixed multimodal Lambek calculus, 38
model, 22
<table>
<thead>
<tr>
<th>Term</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>model-theoretic</td>
<td>30</td>
</tr>
<tr>
<td>modes of composition</td>
<td>25</td>
</tr>
<tr>
<td>MODIFIER</td>
<td>75</td>
</tr>
<tr>
<td>modus ponens</td>
<td>19</td>
</tr>
<tr>
<td>MOR</td>
<td>96</td>
</tr>
<tr>
<td>move</td>
<td>133</td>
</tr>
<tr>
<td>multimodal</td>
<td>25</td>
</tr>
<tr>
<td>multiple inheritance</td>
<td>66</td>
</tr>
<tr>
<td>multiplicity</td>
<td>32</td>
</tr>
<tr>
<td>natural deduction</td>
<td>37</td>
</tr>
<tr>
<td>Gentzen style</td>
<td>20, 27</td>
</tr>
<tr>
<td>Prawitz style</td>
<td>19</td>
</tr>
<tr>
<td>negative occurrences</td>
<td>119</td>
</tr>
<tr>
<td>neutralisation</td>
<td>184</td>
</tr>
<tr>
<td>nominal</td>
<td>46</td>
</tr>
<tr>
<td>non-associative</td>
<td>34</td>
</tr>
<tr>
<td>non-commutative</td>
<td>33</td>
</tr>
<tr>
<td>non-logical vocabulary</td>
<td>136</td>
</tr>
<tr>
<td>non-monotonic</td>
<td>66</td>
</tr>
<tr>
<td>non-terminal symbol</td>
<td>12</td>
</tr>
<tr>
<td>object</td>
<td>70</td>
</tr>
<tr>
<td>off-line</td>
<td>181</td>
</tr>
<tr>
<td>on-line</td>
<td>181</td>
</tr>
<tr>
<td>order</td>
<td>32</td>
</tr>
<tr>
<td>order-sensitive</td>
<td>20</td>
</tr>
<tr>
<td>overspecification</td>
<td>117, 137</td>
</tr>
<tr>
<td>overspecified mode</td>
<td>138</td>
</tr>
<tr>
<td>part-of-speech</td>
<td>83</td>
</tr>
<tr>
<td>permutation</td>
<td>33</td>
</tr>
<tr>
<td>permute</td>
<td>33</td>
</tr>
<tr>
<td>PHONOLOGY</td>
<td>79</td>
</tr>
<tr>
<td>phrasal sorts</td>
<td>154</td>
</tr>
<tr>
<td>phrase structure</td>
<td>13</td>
</tr>
<tr>
<td>phrase structure tree</td>
<td>13</td>
</tr>
<tr>
<td>polarity</td>
<td>72, 116, 134, 147</td>
</tr>
<tr>
<td>polarity sensitivity</td>
<td>72</td>
</tr>
<tr>
<td>positive occurrences</td>
<td>119</td>
</tr>
<tr>
<td>postulate</td>
<td>1, 37</td>
</tr>
<tr>
<td>(k_1)</td>
<td>37</td>
</tr>
<tr>
<td>(k_2)</td>
<td>37</td>
</tr>
<tr>
<td>ASSOC</td>
<td>34</td>
</tr>
<tr>
<td>K</td>
<td>37</td>
</tr>
<tr>
<td>MC</td>
<td>39</td>
</tr>
<tr>
<td>Prawitz notation</td>
<td>29</td>
</tr>
<tr>
<td>Prawitz-style</td>
<td>40</td>
</tr>
<tr>
<td>precedence</td>
<td>15</td>
</tr>
<tr>
<td>predicate symbol</td>
<td>103</td>
</tr>
<tr>
<td>principled resolution</td>
<td>183</td>
</tr>
<tr>
<td>propositional logic</td>
<td>19, 141</td>
</tr>
<tr>
<td>prosodic interpretation</td>
<td>79</td>
</tr>
<tr>
<td>QSTORE</td>
<td>79</td>
</tr>
<tr>
<td>quantification</td>
<td>104</td>
</tr>
<tr>
<td>range</td>
<td>16</td>
</tr>
<tr>
<td>re-entrant</td>
<td>48</td>
</tr>
<tr>
<td>reflexivity</td>
<td>147</td>
</tr>
<tr>
<td>relative clause</td>
<td>77</td>
</tr>
<tr>
<td>residuation</td>
<td>26</td>
</tr>
<tr>
<td>resource management mode</td>
<td>36</td>
</tr>
<tr>
<td>resource-conscious logic</td>
<td>33</td>
</tr>
<tr>
<td>resource-sensitive</td>
<td>32</td>
</tr>
<tr>
<td>rewrite rule</td>
<td>17</td>
</tr>
<tr>
<td>rewriting system</td>
<td>12</td>
</tr>
<tr>
<td>right rule</td>
<td>40</td>
</tr>
<tr>
<td>rightward application</td>
<td>16</td>
</tr>
<tr>
<td>rule-to-rule</td>
<td>22</td>
</tr>
<tr>
<td>semantic form</td>
<td>21</td>
</tr>
<tr>
<td>semantic interpretation</td>
<td>79</td>
</tr>
<tr>
<td>sequent</td>
<td>27</td>
</tr>
<tr>
<td>sign</td>
<td>10, 11, 20, 73</td>
</tr>
<tr>
<td>signature</td>
<td>46, 49, 94</td>
</tr>
<tr>
<td>HPSG</td>
<td>60</td>
</tr>
<tr>
<td>signs</td>
<td>21</td>
</tr>
<tr>
<td>SLASH</td>
<td>78</td>
</tr>
<tr>
<td>sort</td>
<td>46</td>
</tr>
<tr>
<td>sort constraint</td>
<td>50, 65</td>
</tr>
<tr>
<td>sort-hierarchy</td>
<td>49</td>
</tr>
<tr>
<td>sort-resolved</td>
<td>50, 63</td>
</tr>
<tr>
<td>sortal hierarchy</td>
<td>51</td>
</tr>
<tr>
<td>SPEC</td>
<td>75</td>
</tr>
<tr>
<td>stand-alone formalism</td>
<td>53</td>
</tr>
<tr>
<td>start symbol</td>
<td>12</td>
</tr>
<tr>
<td>string</td>
<td>11</td>
</tr>
<tr>
<td>Term</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>string set</td>
<td>13</td>
</tr>
<tr>
<td>strong distribution</td>
<td>37</td>
</tr>
<tr>
<td>structural operator</td>
<td>30</td>
</tr>
<tr>
<td>structurally complete</td>
<td>35</td>
</tr>
<tr>
<td>structure building operation</td>
<td>27</td>
</tr>
<tr>
<td>structure-sensitive</td>
<td>32</td>
</tr>
<tr>
<td>structure-sharing</td>
<td>48</td>
</tr>
<tr>
<td>SUBCAT</td>
<td>70</td>
</tr>
<tr>
<td>subcategorisation</td>
<td>71</td>
</tr>
<tr>
<td>subcategorisation principle</td>
<td>64</td>
</tr>
<tr>
<td>SUBJECT</td>
<td>70</td>
</tr>
<tr>
<td>subsort</td>
<td>49</td>
</tr>
<tr>
<td>substructural logic</td>
<td>33</td>
</tr>
<tr>
<td>subsume</td>
<td>52</td>
</tr>
<tr>
<td>subsumption</td>
<td>52</td>
</tr>
<tr>
<td>subsumption-based grammar</td>
<td>54</td>
</tr>
<tr>
<td>succedent</td>
<td>27</td>
</tr>
<tr>
<td>supersort</td>
<td>49</td>
</tr>
<tr>
<td>tag</td>
<td>49</td>
</tr>
<tr>
<td>taxonomic classification</td>
<td>45, 49</td>
</tr>
<tr>
<td>term</td>
<td>27, 30</td>
</tr>
<tr>
<td>term constructor</td>
<td>27</td>
</tr>
<tr>
<td>terminal symbol</td>
<td>12</td>
</tr>
<tr>
<td>theory</td>
<td>70</td>
</tr>
<tr>
<td>totally well-typed</td>
<td>50, 63</td>
</tr>
<tr>
<td>tree</td>
<td>13</td>
</tr>
<tr>
<td>turnstile</td>
<td>20</td>
</tr>
<tr>
<td>type constraint, see sort constraint</td>
<td></td>
</tr>
<tr>
<td>type skeleton</td>
<td>106</td>
</tr>
<tr>
<td>type-constraint</td>
<td>100</td>
</tr>
<tr>
<td>type-driven</td>
<td>23</td>
</tr>
<tr>
<td>typed feature structure</td>
<td>49</td>
</tr>
<tr>
<td>types</td>
<td>16</td>
</tr>
<tr>
<td>unary connective</td>
<td>25</td>
</tr>
<tr>
<td>unary product</td>
<td>29</td>
</tr>
<tr>
<td>unbounded dependency</td>
<td>77</td>
</tr>
<tr>
<td>underspecification</td>
<td>59</td>
</tr>
<tr>
<td>underspecified mode</td>
<td>138</td>
</tr>
<tr>
<td>unification</td>
<td>53</td>
</tr>
<tr>
<td>unification-based grammar</td>
<td>56</td>
</tr>
<tr>
<td>uppercase</td>
<td>138</td>
</tr>
<tr>
<td>valuation function</td>
<td>26</td>
</tr>
<tr>
<td>variable</td>
<td>103</td>
</tr>
<tr>
<td>verb raising</td>
<td>39</td>
</tr>
<tr>
<td>vocabulary</td>
<td>12</td>
</tr>
<tr>
<td>Wasow's generalisation</td>
<td>182</td>
</tr>
<tr>
<td>weak distribution</td>
<td>37</td>
</tr>
<tr>
<td>weak equivalence</td>
<td>17</td>
</tr>
<tr>
<td>weakening</td>
<td>33</td>
</tr>
<tr>
<td>well-typed</td>
<td>23</td>
</tr>
<tr>
<td>word</td>
<td>12</td>
</tr>
<tr>
<td>X-bar</td>
<td>58</td>
</tr>
<tr>
<td>VAL</td>
<td>96</td>
</tr>
</tbody>
</table>
Bibliography


215


BIBLIOGRAPHY


Curriculum Vitae

Dirk Heylen was born on October 4, 1963 in Bouwel (Belgium). He received his masters degree (licentiaat) Germaanse filologie in 1985 at the University of Antwerp and two other degrees (getuigschriften) in computational linguistics and computer science in 1987 and 1988, respectively. He worked as a computational linguist in Leiden at the Institute of Dutch Lexicology from 1986 till 1989 and at the Uil-OTS in Utrecht from 1989 till 1999. At present he is assistant professor at the University of Twente as a member of the ParleVink research group.