An Inheritance-Based Theory of the Lexicon in Combinatory Categorial Grammar

Mark McConville

Doctor of Philosophy

Institute for Communicating and Collaborative Systems
School of Informatics
University of Edinburgh

2007

Abstract

This thesis proposes an extended version of the Combinatory Categorial Grammar (CCG) formalism, with the following features:

1. grammars incorporate inheritance hierarchies of lexical types, defined over a simple, feature-based constraint language

2. CCG lexicons are, or at least can be, *functions* from forms to these lexical types

This formalism, which I refer to as 'inheritance-driven' CCG (I-CCG), is conceptualised as a partially model-theoretic system, involving a distinction between category descriptions and their underlying category models, with these two notions being related by logical satisfaction. I argue that the I-CCG formalism retains all the advantages of both the core CCG framework and proposed generalisations involving such things as multiset categories, unary modalities or typed feature structures. In addition, I-CCG:

1. provides non-redundant lexicons for human languages

2. captures a range of well-known implicational word order universals in terms of an acquisition-based preference for shorter grammars

This thesis proceeds as follows:

Chapter 2 introduces the 'baseline' CCG formalism, which incorporates just the essential elements of category notation, without any of the proposed extensions. Chapter 3 reviews parts of the CCG literature dealing with linguistic competence in its most general sense, showing how the formalism predicts a number of language universals in terms of either its restricted generative capacity or the prioritisation of simpler lexicons. Chapter 4 analyses the first motivation for generalising the baseline category notation, demonstrating how certain fairly simple implicational word order universals are not formally predicted by baseline CCG, although they intuitively do involve considerations of grammatical economy. Chapter 5 examines the second motivation underlying many of the customised CCG category notations — to reduce lexical redundancy, thus allowing for the construction of lexicons which assign (each sense of) open class words and morphemes to no more than one lexical category, itself denoted by a non-composite lexical type.

Chapter 6 defines the I-CCG formalism, incorporating into the notion of a CCG grammar both a type hierarchy of saturated category symbols and an inheritance hierarchy of constrained lexical types. The constraint language is a simple, feature-based, highly underspecified notation, interpreted against an underlying notion of category *models* — this latter point is crucial, since it allows us to abstract away from any particular inference procedure and focus on the category notation itself. I argue that the partially model-theoretic I-CCG formalism solves the lexical redundancy problem fairly definitively, thereby subsuming all the other proposed variant category notations. Chapter 7 demonstrates that the I-CCG formalism also provides the beginnings of a theory of the CCG lexicon in a stronger sense — with just a small number of substantive assumptions about types, it can be shown to formally predict many implicational word order universals in terms of an acquisition-based preference for simpler lexical inheritance hierarchies, i.e. those with fewer types and fewer constraints. Chapter 8 concludes the thesis.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Mark McConville*)

Acknowledgements

I owe a particular debt of gratitude to three people, each of whom has played a major role in helping me formulate the ideas that constitute this thesis.

- My advisor, Mark Steedman, whose book *Surface Structure and Interpretation* was my first introduction to the whole field of non-transformational approaches to syntax. Over the past five years, Mark has been remarkably patient with me as the direction of my research gradually became clear, encouraging my good ideas, disabusing me of (most of) my bad ones, and supporting me over the final period when I was juggling full-time work, childcare, and writing up.

- My friend and former officemate, Jason Baldridge, whose work on syntactic asymmetries in Tagalog was crucial to my decision to work on CCG. During the initial stages of the research, Jason was a great sounding board for many of the ideas I was playing around with.

- My occasional colleague Cem Bozsahin, who has been a regular visitor to Edinburgh over the past four years. My thinking on what a theory of the CCG lexicon might look like and how important CCG is for theoretical syntax in general, as well as my appreciation for lowland malts, has been formed in the course of countless conversations with Cem.

The former Centre for Cognitive Science at the University of Edinburgh has turned out to be an immensely stimulating place to do postgraduate study, full of intelligent people doing interesting things. My various lab- and office-mates over the last six years have made life as a grad student much more bearable, in chronological order: Beata Klebanov, Colin Fraser, Malte Huebner, Ger Keohane, Alex McCauley, Dimitra Tsovaltsi, Jason Baldridge, Peter Dienes, Colin Bannard, Chris Callison-Burch, Nicole Kim, Jochen Leidner and Viktor Tron.

Parts of this thesis (specifically chapter 3 and bits of chapters 5 and 6) were presented as my contribution to the course "CCG and linguistic diversity", which I co-taught with Cem Bozsahin at the 17th European Summer School in Logic, Language and Information held at Heriot-Watt University, Edinburgh during August 2005. Since the planning and delivery of this course provided me with an invaluable and timely opportunity to sharpen up both the content and presentation of many of the ideas I was

trying to get across, I am particularly grateful to Cem for encouraging me to teach it with him, to the programme committee for selecting us, and to the thirty or so participants who bore with us for all five days, despite the other attractions on offer. Particular thanks to Ash Asudeh, Wilfred Hodges and Shravan Vasishth, who pointed out a number of problems that needed resolved and made some highly relevant suggestions.

I have also discussed the ideas contained within this thesis at the 14th Postgraduate Conference in Linguistics at the University of Manchester (March 2005), the 12th Postgraduate Conference of the Department of Theoretical and Applied Linguistics at the University of Edinburgh (April 2005), and the 8th (and sadly final) Durham Postgraduate Conference in Theoretical and Applied Linguistics (June 2005). I'd like to take this opportunity to thank both the organisers and all those who attended. Feedback from Miriam Bouzouita, Anders Holmberg, Simon Kirby and Bob Ladd, was particularly useful and its effects are evident in many details of presentation throughout this thesis.

The ideas underlying chapters 5 and 6 were presented at the 11th Conference of the European Chapter of the Association for Computational Linguistics in Trento, Italy in April 2006, and published as McConville (2006).

The research reported in this thesis was funded by an EPSRC doctoral scholarship. Over the last two years, I have had particular cause for gratitude to the people who have kept me in paid work: Anna Hobbes, Dave Robertson, Helen Pain, Jean Carletta and Jonathan Sher.

Finally, while writing up a PhD thesis inevitably involves a large dose of personal sacrifice, no one suffers from this more than one's family. With this in mind, I would like to finish by thanking my partner Elaine and son Finn for putting up with my absences over the last few years, and for providing the inspiration to keep going. I hereby promise we'll take lots more holidays over the next couple of years than we have managed in the past two.

To Elaine and Finn

roll down the window and let the wind blow back your hair
the night's bustin' open these two lanes will take us anywhere

Table of Contents

1 Introduction **1**

1.1 Background 1

1.1.1 Formal languages and grammar formalisms 2

1.1.2 Grammar formalisms as theories of linguistic competence . . 6

1.1.3 The model-theoretic perspective 9

1.2 The main ideas 11

1.3 Thesis outline 29

2 Combinatory Categorial Grammar **33**

2.1 Categories 34

2.1.1 A brief history of category notation in CCG 34

2.1.2 Category frames 40

2.1.3 Category structures 41

2.1.4 Category models 41

2.1.5 Category descriptions 42

2.1.6 Satisfaction 43

2.1.7 Summary 44

2.2 Combinatory operations 46

2.2.1 Results and arguments of category models 47

2.2.2 Forward application 48

2.2.3 Backward application 49

2.2.4 Forward composition 50

2.2.5 Backward composition 52

2.2.6 Forward raising 53

2.2.7 Backward raising 53

2.2.8 Raising and composition in the grammar of English 53

2.2.9 Other combinatory rules in CCG 55
2.2.10 Summary 57
2.3 The CCG formalism 59
2.3.1 CCGs 59
2.3.2 Combinatory projection of a CCG lexicon 60
2.3.3 CCG generation 61
2.4 Semantics 62
2.4.1 Denotational semantics in CCG 62
2.4.2 Representational semantics in CCG 65
2.5 CCG derivations as models 69
2.5.1 Generalised category frames 70
2.5.2 Generalised category structures 71
2.5.3 Generalised category models 71
2.5.4 Category descriptions and satisfaction 71
2.5.5 Binary ordered trees 72
2.5.6 Derivation models 73
2.5.7 Lexical descriptions 74
2.5.8 Derivation satisfaction 75
2.5.9 Summary 76
2.6 Summary 77

3 CCG and linguistic competence **81**
3.1 Unbounded dependencies in CCG 82
3.1.1 Unbounded nested dependencies 82
3.1.2 Unbounded cross-serial dependencies 84
3.1.3 Doubly unbounded scrambling 89
3.1.4 Summary 92
3.2 CCG and implicational language universals 92
3.2.1 Basic word order and gapping 95
3.2.2 Free inversion and subject extraction in Romance 101
3.2.3 Basic word order and gapping again 118
3.3 Summary 120

4 Some problematic data **123**
4.1 Clausal order, prepositions and postpositions 125
4.2 Direct objects, indirect objects and modifiers 134
4.3 Summary 140

5 Type-hierarchical CCG **143**
5.1 CCG and lexical redundancy 144
5.2 Type hierarchies 150
5.3 The T-CCG formalism 153
5.3.1 T-CCG category models 153
5.3.2 T-CCG category descriptions 157
5.3.3 T-CCG satisfaction 158
5.3.4 Type-hierarchical CCGs 158
5.3.5 Combinatory projection of a T-CCG lexicon 159
5.3.6 T-CCG generation 160
5.4 A proof system for T-CCG 161
5.4.1 Lexical insertion 161
5.4.2 Compatibility of T-CCG category descriptions 161
5.4.3 Application 162
5.4.4 Raising 162
5.4.5 Composition 162
5.4.6 Properties of the proof system 163
5.5 T-CCG and the functionality condition on lexicons 164
5.5.1 T-CCG and morphosyntactic CCG 166
5.5.2 T-CCG and typed feature structures 174
5.6 Conclusion 182

6 Inheritance-driven CCG **185**
6.1 A new description language for CCG categories 187
6.1.1 Flexible category descriptions 189
6.1.2 T-CCG satisfaction 190
6.2 T-CCG with flexible category descriptions 192
6.2.1 T-CCGs 193
6.2.2 Combinatory projection of a T-CCG lexicon 195
6.2.3 T-CCG generation 195

6.2.4 The T-CCG proof system . . . 196
6.3 Inheritance hierarchies . . . 198
6.4 The I-CCG formalism . . . 201
6.4.1 I-CCGs . . . 201
6.4.2 I-CCG category models . . . 204
6.4.3 I-CCG satisfaction . . . 205
6.4.4 Combinatory projection of an I-CCG lexicon . . . 207
6.4.5 I-CCG generation . . . 209
6.4.6 An I-CCG proof system . . . 209
6.5 I-CCG and non-redundant lexicons . . . 210
6.6 Summary . . . 215

7 I-CCG and linguistic competence 219
7.1 I-CCG with argument hierarchies . . . 222
7.1.1 Redefining I-CCG grammars . . . 222
7.1.2 I-CCG category descriptions . . . 223
7.1.3 I-CCG argument descriptions . . . 223
7.1.4 An example I-CCG . . . 224
7.1.5 I-CCG category models . . . 226
7.1.6 Redefining I-CCG satisfaction . . . 226
7.1.7 The combinatory projection of an I-CCG lexicon . . . 228
7.1.8 I-CCG generation . . . 229
7.2 Explaining the data . . . 229
7.2.1 Clausal order, prepositions and postpositions . . . 230
7.2.2 Direct objects, indirect objects and modifiers . . . 234
7.2.3 Other word order universals . . . 237
7.2.4 Comparison with processing-based explanations . . . 245
7.3 Summary . . . 248

8 Conclusion 251

Bibliography 261

List of Figures

1.1 The Chomsky hierarchy of formal languages 4
1.2 The CF^{+} languages on the Chomsky hierarchy 5

2.1 A taxonomy of expressions 38
2.2 Two category frames 40
2.3 A category structure 41
2.4 A category model over $\{\mathsf{S}, \mathsf{NP}\}$ 42
2.5 A generalised category frame 70
2.6 A generalised category structure 71
2.7 A generalised category model over alphabet $\{\mathsf{S}, \mathsf{NP}, \mathsf{N}\}$ 72
2.8 A binary ordered tree 73
2.9 A derivation model 75

5.1 A type hierarchy 151
5.2 A type hierarchy of saturated category symbols for the language in Table 5.1 154
5.3 A partition diagram representing the type hierarchy in Figure 5.2 . . . 156
5.4 A T-CCG category model and a non-T-CCG category model over the type hierarchy in Figure 5.2 156
5.5 A type hierarchy for Turkish nouns 172

6.1 The type hierarchy from Figure 5.2 repeated 187
6.2 The lexical inheritance hierarchy for the language in Table 5.1 202
6.3 An I-CCG derivation 214

7.1 A redefined lexical inheritance hierarchy for the language in Table 5.1 225
7.2 An inheritance hierarchy of argument types for the language in Table 5.1 226

7.3 A partial argument inheritance hierarchy for human language 231
7.4 A partial lexical inheritance hierarchy for human language 232
7.5 Accusative and ergative languages 243

List of Tables

4.1 Baseline CCG lexical fragments for languages in VO ∩ PO, VO ∩ OP, etc. 128
4.2 Baseline CCG lexical fragments for languages in VX ∩ VI, VX ∩ IV, etc. 137

5.1 Schematisation of a fragment of English 145
5.2 A CCG lexicon for the language in Table 5.1 146
5.3 A CFG for the language schematised in Table 5.1 151
5.4 A T-CCG lexicon for the language in Table 5.1 159
5.5 Some accusative case noun-forms in Turkish 166
5.6 A typed feature structure-based CCG lexicon for the language schematised in Table 5.1 178
5.7 A T-CCG lexicon for the language schematised in Table 5.1, with dependency semantics 181

6.1 An I-CCG lexicon for the language in Table 5.1 assuming the hierarchies in Figures 6.1 and 6.2 203

7.1 I-CCG constraints for Travis' language types 236

Chapter 1

Introduction

The main idea defended in this thesis is the following: if we take a partially model-theoretic perspective on the Combinatory Categorial Grammar (CCG) formalism of Steedman (2000), whereby a theoretical distinction is drawn between category descriptions and the underlying category models which satisfy them, then we can construct a hybrid formalism which:

1. provides non-redundant lexicons for human languages
2. captures some well-known implicational universals of human language as *formal* universals, without requiring much in the way of substantive stipulation

In defence of this thesis, I present an extension of the CCG formalism where:

1. grammars incorporate inheritance hierarchies of lexical types, defined over a simple, feature-based constraint language
2. lexicons are, or at least can be, *functions* from morphemes to 'atomic' lexical types, i.e. consisting of a single, non-composite symbol

This chapter will proceed as follows: **section 1.1** fills in a bit of necessary background, **section 1.2** elaborates on the central idea, and **section 1.3** presents the outline of the thesis.

1.1 Background

Subsection 1.1.1 introduces some of the basic tenets and concepts of the discipline of formal language theory, focusing on formal languages as sets of strings, grammar

formalisms as theories of classes of formal language, and the Chomsky hierarchy of classes of formal language. **Subsection 1.1.2** then turns to the study of human linguistic competence, presenting Chomsky's foundational assumption that human languages are formal languages, i.e. sets of strings of morphemes, and hence that the knowledge underlying linguistic competence can be modelled as a formal grammar, and the language faculty as a grammar formalism. Finally, **subsection 1.1.3** discusses two different perspectives on grammar formalism design, which Pullum and Scholz (2001) call the 'generative-enumerative' and the 'model-theoretic' perspectives.

1.1.1 Formal languages and grammar formalisms

I start by assuming a number of standard notions from formal language theory. An 'alphabet' is any finite set of symbols. The set of 'strings' over alphabet Σ, often written Σ^*, is defined as the closure of Σ under concatenation. And the class of 'formal languages' over Σ is defined as the set of all subsets of Σ^* i.e. a formal language is simply a set of strings of symbols.

A 'grammar formalism' is a theory of a particular subclass of the formal languages over alphabet Σ. A grammar formalism, parameterised relative to alphabet Σ, specifies two things: (a) a set of 'grammars' over Σ; and (b) a 'generates' relation between grammars over Σ and strings in Σ^* i.e. for every grammar G over Σ and every string $s \in \Sigma^*$, either G generates s or G does not generate s.[1] Overloading the terminology slightly, we can say that in addition to generating strings, each grammar also generates a formal language. For every grammar G over alphabet Σ, the formal language generated by G is the set of all strings over Σ^* which are generated by G. For every grammar formalism F and every alphabet Σ, the 'generative capacity' of F with respect to Σ is the set of formal languages over Σ which are generated by some F-grammar over Σ. Thus, every grammar formalism F characterises one particular subclass of formal language, known as the class of F-languages.

Study of the formal properties of different grammar formalisms was initiated by Chomsky (1959). Chomsky started by defining a grammar formalism known as 'Phrase Structure Grammar' (PSG): a PSG over alphabet Σ is an ordered triple $\langle A, S, \Phi \rangle$, where: (a) A is an alphabet of non-terminal symbols; (b) S is a distinguished element of A; and (c) Φ is a finite set of string-rewriting rules, each of the form $\alpha \rightarrow \beta$, where α and β

[1] A probabilistic grammar formalism will also assign a probability to each generated string.

are both strings of symbols drawn from either Σ or A. PSG $\langle A, S, \Phi \rangle$ over alphabet Σ generates string $s \in \Sigma^*$ just in case there is a derivation from symbol S to string s, using only the string rewriting rules in Φ. Two important results were quickly established for the PSG formalism: (a) for any alphabet Σ, the generative capacity of PSG is a proper subset of the set of formal languages over Σ; (b) the languages over alphabet Σ which are generated by some PSG are exactly those languages which are accepted by a Turing machine. In other words, the unrestricted string-rewriting grammar formalism PSG defines a 'natural class' of formal language, consisting of all and only the languages whose membership can be enumerated algorithmically, i.e. the 'recursively enumerable' languages.

Three further, progressively more restricted natural classes of formal language were identified in Chomsky (1963), defined as the generative capacities of three PSG-style grammar formalisms which place increasingly stronger restrictions on the class of possible string-rewriting rules:

context-sensitive grammar (CSG) all rules are of the form $\alpha \rightarrow \beta$, where β contains at least as many symbols as α[2]

context-free grammar (CFG) all rules are of the form $X \rightarrow \alpha$, where α is a non-empty string of symbols and X is a single non-terminal

regular grammar (RG) all rules are of the form $X \rightarrow aY$ where X and Y are both single non-terminals and a is a terminal symbol

Since every RG is also a CFG, every CFG is a CSG, and every CSG is a PSG, the generative capacities of each of these formalisms constitute a containment hierarchy. In other words, where R, CF, CS and RE denote the classes of regular, context-free, context-sensitive and recursively enumerable languages over alphabet Σ:

$$\text{R} \subset \text{CF} \subset \text{CS} \subset \text{RE} \subset \wp(\Sigma^*) \tag{1.1}$$

Note that $\wp(\Sigma^*)$ denotes the set of all subsets of Σ^*, which in this case simply means the set of all formal languages over Σ. This containment hierarchy, known as the Chomsky hierarchy, is represented in Figure 1.1.

[2]The appellation 'context-sensitive' is derived from a particular normal form where each rule is of the form $\alpha X \beta \rightarrow \alpha\gamma\beta$, stating that non-terminal X can be rewritten as string γ in context α_β

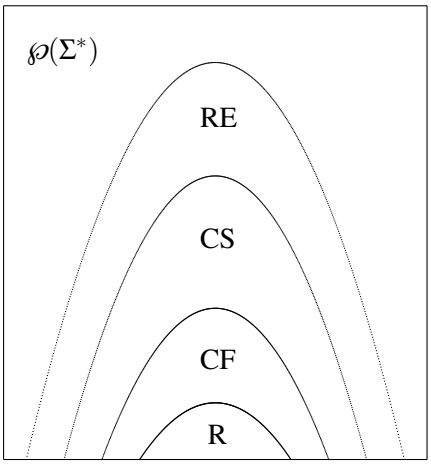


Figure 1.1: The Chomsky hierarchy of formal languages

There are a number of reasons why formal language theorists regard each of the four classes of language on the Chomsky hierarchy as a 'natural class' of formal language. First of all, each of the four classes corresponds to a particular family of string-accepting automata — just as the recursively enumerable languages are those accepted by Turing machines, so it has been proved that the regular languages are those accepted by finite state machines, the context-free languages by non-deterministic pushdown automata, and the context-sensitive languages by non-deterministic Turing machines whose tape is bounded proportionate to the length of the input string. Secondly, these four classes of formal language share interesting algebraic closure properties, i.e. R, CF, CS and RE are all closed under union, concatenation, unbounded iteration (Kleene star), and intersection with R. A third reason why these sets of formal languages are considered to be natural classes involves the fact that each constitutes the generative capacity of a range of different grammar formalisms (loosely defined). To take just one example, the context-free languages can be defined in at least the following ways: (a) the languages generated by some CFG; (b) the languages accepted by some non-deterministic pushdown automaton; (c) the languages generated by either the basic categorial grammars of Ajdukiewicz (1935) or Bar-Hillel (1953) or the associative categorial calculi of Lambek (1958); or (d) the string yields of the finite trees which satisfy some set of monadic second order formulas (Rogers, 2003).

Subsequent work has identified many other 'levels' in the Chomsky hierarchy.

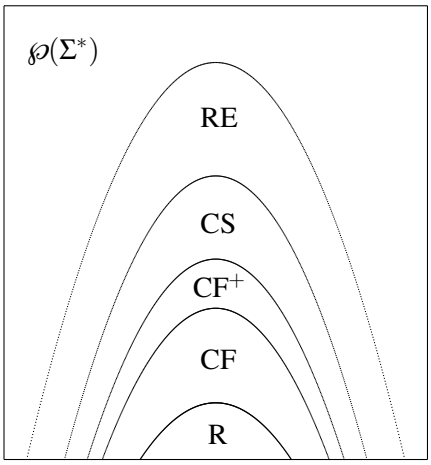


Figure 1.2: The CF^+ languages on the Chomsky hierarchy

From the perspective of the study of human language, one class which is particularly significant is the *least inclusive* natural class of formal language which is a *proper superset* of the class of context-free languages. Let's call this class CF^+ for convenience:

(1.2) $\quad CF \subset CF^+ \subset CS$

CF^+ can be defined in any of the following ways: (a) the languages generated by a Linear Indexed Grammar (Gazdar, 1988); (b) the languages accepted by an embedded pushdown automaton; (c) the languages generated by a Combinatory Categorial Grammar (Weir and Joshi, 1988); (d) the languages generated by a Tree Adjoining Grammar (Joshi et al., 1975); or (e) the string yields of the classes of finite three-dimensional trees which satisfy some set of monadic second order formulas (Rogers, 2003). The place of the CF^+ languages in the Chomsky hierarchy is shown in Figure 1.2.

CF^+ is the most restrictive class of language we know about which both properly includes the context-free languages and is itself included in the class of 'mildly context-sensitive' languages. This class is of particular interest in that it is the lowest complexity class on the Chomsky hierarchy which contains languages exhibiting unbounded cross-serial dependency constructions. Note finally that CF^+ is also closed under union, concatenation, unbounded iteration, and intersection with R.

1.1.2 Grammar formalisms as theories of linguistic competence

Now, formal language theory is an extremely interesting subject, and there is a lot more that can be said about it. However, this is a thesis about human languages like English, Flemish and Greenlandic Eskimo rather than such artifical constructs as the set of all bit strings consisting of some finite number of 0's followed by exactly the same number of 1's, the assembly language for 8-bit microprocessors, and so on. But the two may not be as different as they appear on the surface — one of the foundational assumptions of modern formal linguistics is that the study of formal languages has profound implications for the study of human languages, the former perhaps even subsuming certain aspects of the latter.

Chomsky (1957) famously suggested that *human* languages such as English can be modelled as formal languages (p.13):

> From now on I will consider a *language* to be a set (finite or infinite) of sentences, each finite in length and constructed out of a finite set of elements. All natural languages in their spoken or written form are languages in this sense, since each natural language has a finite number of phonemes (or letters in its alphabet) and each sentence is representable as a finite sequence of these phonemes (or letters), though there are infinitely many sentences.

In other words, a human language can be considered to be simply a set of strings of morphemes, generated by some formal grammar.

Chomsky (1965) clarified this in two important ways. Firstly, he defined human languages to be properties of individual language-users, rather than of speech communities. In the terminology of the sociolinguists, linguistic theory is concerned solely with 'idiolect' rather than 'language' or 'dialect'. As he put it himself, somewhat counterintuitively: "linguistic theory is concerned primarily with an ideal speaker-listener, in a completely homogeneous speech community, who knows its language perfectly" (p.3). Thus, whenever we talk about the class of 'human languages', we are to be understood as referring to the set of all possible idiolects.

Secondly, Chomsky (1965) distinguished between two important aspects of the idiolect of a language-user: (a) linguistic 'performance', or the utterances actually produced by the language-user on particular occasions; and (b) linguistic 'competence', or the strings which the language-user knows intuitively to be grammatical sentences of his/her language. These sets are partially disjoint, since actual utterances are not always grammatical sentences, and not every sentence judged by a language-user to be

grammatical will be uttered in the course of his or her lifetime. Chomsky points out that linguistic theory is centrally concerned with competence rather than performance, although evidence from the study of linguistic performance may be admissable in evaluating different competence theories. In this respect, when we talk about the class of 'human languages', we are to be understood as referring to the set of all possible linguistic competences.

Thus, Chomsky proposes that a human language (in the sense of the linguistic competence which underlies the idiolect of some language-user) can be modelled as a set of strings of morphemes, although the language-user may not have conscious knowledge of the precise membership of his or her language. This implies that the knowledge which underlies this competence can be modelled as a formal grammar, of the kind discussed in subsection 1.1.1. The aim of formal linguistics is then to find the grammar formalism whose grammars best correspond to the ones in our heads, and which can thus be said to model the human language faculty.

Going back to the Chomsky hierarchy in Figure 1.2, Chomsky (1956) proved that English is not a regular language (i.e. $\mathit{English} \notin \mathrm{R}$), owing to the existence of unbounded nested dependency structures such as 'the man who said that ϕ is arriving today' in the linguistic competence of any English speaker. Thus, any grammar formalism whose generative capacity is the class of regular languages, for example RG or the family of finite state automata, is inadequate as a theory of human language competence, since there are human languages which cannot be generated by one of its grammars.[3] In other words, where NL denotes the class of human, or 'natural', languages, $\mathrm{NL} \not\subseteq \mathrm{R}$.

Moving on up to the context-free languages, Shieber (1985) provided the first proof that there is at least one human language, Swiss German, which is not (even weakly) context-free, owing in part to it having unbounded cross-serial dependency constructions in subordinate clauses. Several other human languages have also been shown to have such cross-serial dependency constructions (e.g. Dutch, Bambara). Thus, grammar formalisms whose generative capacity is the class of context-free languages are also inadequate as theories of human language competence, since $\mathrm{NL} \not\subseteq \mathrm{CF}$.

[3] Although this is not to say that substantial fragments of particular human languages cannot be effectively modelled as regular languages, as much recent work in natural language engineering has shown.

The next level in the Chomsky hierarchy is CF^+, the most restrictive class of mildly context sensitive language which properly includes the CF languages. Languages such as Swiss German and Dutch, which manifest unbounded cross-serial dependency constructions, fall within this class. Furthermore, nobody has to date presented any conclusive evidence of a human language which is not in CF^+, the efforts of Rambow (1994) and Hoffman (1995) notwithstanding. Thus, a substantial body of opinion in formal linguistics has it that $\text{NL} \subseteq \text{CF}^+$, and thus that any grammar formalism whose generative capacity is this class, for example Linear Indexed Grammar, Tree Adjoining Grammar or Combinatory Categorial Grammar, is a potential candidate for a theory of human language competence.

Before going on to discuss two different ways of thinking about grammar formalisms, a couple of further methodological remarks are in order. First of all, it should be noted that not even the most autonomous of syntacticians would suggest that all the interesting aspects of a human being's linguistic competence could be modeled as a set of strings of speech sounds. From this perspective, it is more appropriate to model linguistic competence as a relational structure of the form $\langle A, R_1, \ldots, R_n \rangle$, where A is a set of strings of symbols, and each R_i is a relation on A corresponding to some aspect of the linguistic intuition of a native speaker of the language. Such relations can be syntactic (e.g. sentence s_1 is the *passive* of sentence s_2) or semantic (e.g. sentence s_1 is *consistent* with sentence s_2, in the sense that both sentences can be true in the same model). This generalised model of human language competence is the basis for a distinction, first formulated in Chomsky (1964), between grammars which are merely *observationally adequate* theories of a human language (i.e. they generate the correct set of sentences), and those which are *descriptively adequate* as well (i.e. they generate the correct set of sentences and also account for the various relations on the sentences which are part of the native speaker's intuitions about his/her language).

Secondly, given that human languages are *acquired* over a considerable period of time, rather than emerging spontaneously and fully-formed, a more accurate model of linguistic competence will seek to capture its dynamic nature. In this respect, competence is more like a sequence of languages, one for each stage of linguistic development, and the job of formal syntax includes accounting for observed patterns of acquisition as well. Finally, I make an obvious simplification about the sound-structure of human language, assuming that the algebra underlying grammatical composition is simply that of string concatenation. This ignores important aspects of 'suprasegmen-

tal' phonology such as stress and intonation. Since most of the focus of this thesis is on generalisations involving basic word order in human languages, this idealisation does not seem inappropriate.

1.1.3 The model-theoretic perspective

As we have seen, one of the central aims of the formal study of linguistic competence is to find a grammar formalism which somehow encapsulates the human language faculty. For most grammar formalisms, there are at least two ways of interpreting any given grammar, i.e. either as a set of instructions for constructing sentences, or as a set of constraints that must be satisfied by sentences. We can illustrate this distinction by considering what is possibly the best known and most obviously intuitive grammar formalism, i.e. CFG, characterised by Chomsky (1957) as the generative system which underlies the traditional notion of immediate constituent analysis, or 'parsing'. The standard definition of the class of CFGs runs as follows, adapted slightly from Hopcroft and Ullman (1979):

> A CFG over alphabet Σ is an ordered triple $\langle A, S, \Phi \rangle$, where A is an alphabet of non-terminals, $S \in A$, and Φ is a finite set of context-free rewrite rules, each of the form $X \rightarrow \alpha$, where $X \in A$ and $\alpha \in (\Sigma \cup A)^{+}$.

The customary definition of the 'generates' relation is something like:

> CFG $\langle A, S, \Phi \rangle$ over Σ generates string $s \in \Sigma^{*}$ just in case there is a derivation leading from the distinguished symbol S to string s, using only the rules in Φ.

The notion of generation assumed here is explicitly *derivational*. Whether or not a string is generated by a CFG is dependent on the outcome of a non-deterministic procedure which takes the initial symbol and the rules as inputs, and then applies the rules to the initial symbol, until a string of terminals is arrived at. This view of a grammar formalism, according to which generation is understood derivationally, is termed the 'generative-enumerative' perspective by Pullum and Scholz (2001).

There is however another way of thinking about the 'generates' relation for a CFG, first noticed by McCawley (1968), which interprets the rules as *constraints* rather than as instructions:

> CFG $\langle A, S, \Phi \rangle$ over Σ generates string $s \in \Sigma^{*}$ just in case s is the string yield of some labelled tree whose root node is labelled S and where every local subtree satisfies some rule in Φ.

From this standpoint, the CFG formalism constitutes a *model theory*, in the sense that it assumes a class of 'models', a class of 'descriptions', and a 'satisfaction' relation between the two. Given alphabets Σ of terminal symbols and A of non-terminals: (a) the models are finite ordered trees where every leaf node is labelled with an element of Σ, and every other node with an element of A; (b) the descriptions are the context-free rules over Σ and A; and (c) the satisfaction relation tells us first of all which nodes in which tree satisfy which descriptions, and secondly which trees globally satisfy which *sets* of descriptions (Blackburn et al., 1993).

When we conceptualise the 'generates' relation in terms of logical satisfaction, then we are assuming what Pullum and Scholz (2001) call a 'model-theoretic' perspective on the grammar formalism.

Although most, if not all, grammar formalisms can be thought about either as generative-enumerative or as model-theoretic systems, some formalisms are most naturally considered from one particular perspective. For example, the Head-Driven Phrase Structure Grammar (HPSG) formalism of Pollard and Sag (1987) and Pollard and Sag (1994) is generally considered to be a prime example of a model-theoretic (i.e. 'constraint-based') formalism, or more accurately a formalism which is more easily understood as constraint satisfaction system than as a derivation procedure. In HPSG, a 'language' is explicitly defined not as a set of strings or even of trees, but rather as a set of 'totally well-typed, sort-resolved, typed feature structures', which are a particular kind of directed acyclic graph with labels on both edges and vertices. These are the *models* of the logic of HPSG, and contain information about the phonology, syntax and semantics of linguistic signs. An HPSG grammar is fundamentally a set of formulas of feature logic (King, 1989), generally normalised into one big inheritance hierarchy of linguistic types constrained by attribute-value matrices. The language generated by such a grammar is the (string yield of the) set of feature structures which satisfy *all* the formulas in the grammar.

Another class of grammar formalisms which are most naturally considered from the model-theoretic perspective are the so-called 'optimality theories' of, for example, Prince and Smolensky (1993) and Grimshaw (1997). The twin pillars of any optimality-theoretic formalism are firstly a class of *models* (e.g. $\bar{\text{X}}$-theoretic trees) and secondly a finite set of universal, violable, *atomic constraints* on these models (e.g. OP-SPEC, STAY). Linking these is of course a *local satisfaction* relation telling us which atomic constraints are satisfied at which bits (e.g. nodes) of which model.

An optimality-theoretic grammar consists of a lexicon and a partial ordering relation on the atomic constraints, ranking them according to relative inviolability. Given some set α of lexical items, model $\mathcal{M}_1$ over α is *better* than model $\mathcal{M}_2$ over α just in case the highest-ranked constraint not satisfied by $\mathcal{M}_1$ is ranked below some constraint not satisfied by $\mathcal{M}_2$. Model $\mathcal{M}$ *globally* satisfies a particular constraint-ranking just in case $\mathcal{M}$ is better than every other model built up from the same set of lexical items. The language generated by an optimality-theoretic grammar is thus the set of all models which globally satisfy the grammar's constraint-ranking in this way.

In this thesis, I argue that it is useful to take a model-theoretic, constraint-based perspective on the category notation of the CCG formalism, distinguishing between category descriptions and underlying category models, the two concepts related by logical satisfaction. In this way, it will be possible to hold constant the underlying models and deconstruct the category descriptions, so as to come up with a notation which provides non-redundant lexicons for human languages and captures a range of word order universals in terms of an acquisition-based preference for shorter grammars.

1.2 The main ideas

The CCG formalism

Having covered these paradigmatic preliminaries, it is time to come back to the central idea defended in this thesis. The starting point is the Combinatory Categorial Grammar (henceforth CCG) formalism of Steedman (2000) and Baldridge (2002). CCG is a principled, linguistically motivated generalisation of the categorial grammars of Ajdukiewicz (1935) and Bar-Hillel (1953), incorporating three of the combinators which Curry and Feys (1958) use to define applicative systems such as the λ calculus. As a first approximation, the class of grammars specified by the CCG formalism can be defined as follows:

> A CCG over alphabet Σ is an ordered triple $\langle A, S, L\rangle$ where A is an alphabet of saturated category symbols, $S \in A$, and L is a lexicon — a finite mapping from Σ to categories over A, the latter defined as the closure of A under the binary connectives $/$ and $\backslash$.

There are two main points of note with this definition. Firstly, CCG, like all categorial grammar formalisms, provides a countably infinite set of grammatical types,

constructed using the directional slash operators / and \. Secondly, to all intents and purposes, a CCG is a *lexicon*, since individual grammars do not contain any language-particular rules of combination. Thus, CCG is a particularly strong kind of *lexicalised* formalism, where all language-specific grammatical knowledge concerns the syntactic and semantic properties of minimally functioning syntactic units (i.e. words or morphemes). The 'generates' relation for CCG can be given the following provisional definition:

> CCG $\langle A, S, L \rangle$ over alphabet Σ generates string $s \in \Sigma^*$ just in case the ordered pair $\langle s, S \rangle$ is in the *combinatory projection* of lexicon L, the latter defined as the closure of L under the CCG combinatory operations.

The CCG formalism provides a finite number of combinatory operations. These include not only the familiar forward and backward application operations common to all categorial grammars, but also a range of others based on the functional composition, type raising and functional substitution combinators of Curry and Feys (1958). **Chapter 2** presents a more detailed introduction to the CCG formalism.

The task of formal linguistics

In section 1.1.2, I identified one of the main concerns of theoretical linguistics as the search for a grammar formalism whose generative capacity *includes* the class of human languages. To the best of current knowledge, CCG constitutes such a formalism, since there are CCGs for even the most automata-theoretically complex constructions attested in human languages, i.e. the unbounded cross-serial dependency constructions in languages like Dutch and Swiss German (Steedman, 1985).

If the task of formal linguistics is simply to come up with a suitably expressive grammar formalism, then that's really all there is to say. Many formalisms have been proposed whose generative capacity is known to include the human languages, for example the CSG formalism discussed in subsection 1.1.1. However, there are obvious grounds to prefer CCG over CSG as a theory of human language competence, as a quick glance at the Chomsky hierarchy in Figure 1.2 on page 5 makes clear. Recall that the generative capacity of the CSG formalism is the class of context-sensitive languages, or CS. In contrast, the generative capacity of CCG is the class CF^+, i.e. the most restrictive natural class of formal language which properly includes the context-free languages (Weir and Joshi, 1988). Note that CF^+ is a *proper subset* of the context-sensitive languages, i.e. every language in CF^+ is also in CS, but not vice versa. Thus,

general considerations of scientific parsimony suggest that CCG is a better theory of human linguistic competence than CSG, since it is a more restrictive, but still adequate, theory.

Such considerations suggest that a more careful delineation of the task of formal linguistics would be to say that we are seeking not simply a grammar formalism whose generative capacity includes the class of human languages, but rather the *most restrictive such formalism*. Near context-free formalisms like Linear Indexed Grammar, Tree Adjoining Grammar and CCG are therefore better theories of human linguistic competence than more permissive formalisms like CSG.

Possibly the simplest idea about what the study of human linguistic competence should be aiming for is characterised by Lyons (1991) as follows (p.225):

> The goal of theoretical linguistics can be described as that of constructing a class of formal languages, all of whose members share certain general properties and each of whose members can be put into correspondence with some actual or potential natural language.

In other words, the task is to characterise the class of possible human languages over alphabet Σ as a natural subclass of the formal languages over Σ, by specifying a grammar formalism whose generative capacity is exactly the class of possible human languages. Such a formalism would constitute the strongest possible theory of human linguistic competence, or what since Chomsky (1965) has usually been called a 'universal grammar', by providing theories for all human languages, but not for any non-human ones.

With regard to his own characterisation of the task of formal linguistics, Lyons himself concludes that "it is as yet unclear whether this goal can be achieved." One reason for this lack of clarity involves the following paradox of sufficiency. Recall that we are assuming that every human language is a CF^+ language. Moreover, CF^+ is the closest automata-theoretic fit to the class of human languages, i.e. there is no known natural class of formal language which is both a strict subset of the former and contains the latter. The problem arises from the fact that the converse does not hold, i.e. it is not the case that every CF^+ language over alphabet Σ corresponds to a possible human language over Σ.

A fairly trivial illustration of this involves the regular languages, which are all in CF^+ but none of which corresponds to any known human language. There are, however, many non-trivial examples of non-regular context-free languages which do not correspond to any known human language. To cite just one famous example from

the language universals literature, there are many theoretically possible context-free languages where: (a) constituent order in matrix declarative clauses is verb-subject-object (VSO); and (b) there are postpositions but not prepositions. However, according to Comrie (1989), there is *no* clearly attested human language which has both these properties, i.e. VSO languages are always prepositional and never postpositional.

So we find ourselves faced with a problem. On the one hand, we have an intuition that the human languages constitute some kind of natural class. On the other hand, this class is not directly specifiable in terms of automata-theoretic complexity *alone*. The theory of human linguistic competence obviously needs to take into account factors other than just automata-theoretic complexity. What these factors are has been an open question in formal linguistics since the discipline began back in the 1950s.

Substantive universals?

Perhaps the most widespread approach adopted in generative grammar involves a theory of 'substantive universals' of human language, conceptualised as some kind of add-on to the basic grammar formalism (the 'formal' universals). The idea here is that a theory of human linguistic competence is a bipartite system, consisting of: (a) a grammar formalism, specifying a class of formal grammars; and (b) a set of constraints on the form of these grammars, such that only the grammars which correspond to human language grammars satisfy all the constraints.

The HPSG framework discussed briefly in section 1.1.3 is a notable example of a system which embodies this formalism-theory dichotomy. In HPSG the underlying grammar formalism is the Typed Feature Logic of Carpenter (1992),[4] according to which a grammar is essentially any set of constraints on typed feature structures. In addition to this underlying formalism, HPSG proposes a 'theory', which is essentially an alphabet of types and a finite set of constraints on objects of these types, common to all and only the human languages. In more extreme approaches to generative grammar, for example the tradition inspired by Chomsky (1981), this distinction between formal grammars and substantive universals is taken to its logical conclusion, i.e. the assumption that the (core) human language grammars constitute a *finite set*, and can simply be listed.

Interestingly, the designers of the formalism which was the direct precursor of

[4]Or the Speciate Reentrant Logic of King (1989)

HPSG, i.e. Generalised Phrase Structure Grammar (GPSG), appear to have advised against such a 'formalism-theory' distinction. As Gazdar et al. (1985) state in their introduction (pp. 2-3):

> The most interesting contribution that generative grammar can make to the search for universals of language is to specify formal systems that have putative universals as *consequences*, as opposed to merely providing a technical vocabulary in terms of which autonomously stipulated universals can be expressed.
>
> If the fact [about language] needs a special statement, as opposed to following from the very form in which the theoretical reconstruction of the notion 'natural language' has been cast, the job is not done.

Obviously, the notion of 'formal system' implied here does not merely involve the kind of string rewriting systems discussed in previous sections. In addition, the authors are honest enough to admit that the theory of linguistic competence proposed in their book fails to live up to their ideal in a number of ways. However, these comments appear to clearly imply that a theory of human linguistic competence which starts off by specifying a large, general class of grammars, and then rules out all those which are not supersets of some lowest common denominator 'universal grammar', cannot be seen as the ultimate goal of linguistic theorising.[5]

Grammar ranking?

Distinguishing between formalism and substantive theory is not the only approach to the problem discussed above. Indeed, it was not even the first to be considered in the development of generative grammar. Early studies considered an alternative, which involves taking a grammar formalism and then specifying a 'ranking' of the class of grammars provided. This ranking is determined by some 'evaluation measure', defined by Chomsky (1965) as "a function m such that $m(i)$ is an integer associated with the grammar G_i as its value (with, let us say, lower value indicated by higher number)" (p.31). Chomsky goes on to propose the following (p.42): "the obvious numerical measure to be applied to a grammar is length, in terms of numbers of symbols". In other words, *shorter* grammars are better.

[5]To be fair to the designers of the HPSG formalism, it does not strive to be a theory of human linguistic competence. Rather the aim is to come up with a formal system rich enough to encode all the facts which a person knows about his/her language. In other words, the aims of HPSG are explicitly descriptive, rather than explanatory.

The original motivation for this notion of grammar-ranking was the problem of language acquisition, or more accurately the question as to how a child goes about choosing one particular grammar from the many available which account for the primary data he/she has been exposed to. The obvious idea is that the evaluation measure underlying the ranking function is innate and the child can simply select the most highly ranked grammar which covers the data. However, the uses of such a ranking function are not limited to considerations of how languages are acquired. The idea can also be applied to the theory of human linguistic competence itself. Given some grammar formalism F with some ranking function on the class of F-grammars, if F-grammars of possible human languages are consistently ranked more highly than those of non-human languages, then we can argue that F constitutes a successful theory of human linguistic competence. If we assume that the evaluation measure is simply one of 'shortness', then this boils down to the search for a grammar formalism where grammars ofprobable languages are shorter than those of improbable ones.

One of the main aims of this thesis will be to investigate to what extent considerations of 'grammar-length' can be utilised in an explanatory theory of the CCG lexicon, in tandem with the formal universals which characterise CCG within the class of grammar formalisms (the 'principles' of CCG — adjacency, consistency, inheritance, and combinatory type transparency). The idea is that the CCG formalism itself characterises a strict subset of the formal languages (CF^{+}), and this subset is itself ranked (according to the shortest CCG which generates the language) such that the probable languages are highly ranked (i.e. have shorter grammars) compared to the improbable languages.

Implicational universals

Support for this grammar-ranking approach to the theory of human linguistic competence comes from the study of 'implicational universals' of natural language, a body of work which constitutes the main data source for the study of linguistic competence. An implicational universal is a statement of the following form:

> All human languages which have property P also have property Q.

A famous example of an implicational universal which was discussed above and which ultimately derives from Greenberg (1963), the seminal work on implicational universals, is the following: all human languages which have VSO ordering in main declar-

ative clauses also have prepositions rather than postpositions. Implicational universals can be schematised set-theoretically as subset statements i.e. $P \subseteq Q$, and can be interpreted as making the prediction that $P \cap Q' = \emptyset$, where Q' is the complement of set Q, i.e. $\text{NL} - Q$. For example, where PO is the set of languages with prepositions but not postpositions, we can state the above implicational universal as either $\text{VSO} \subseteq \text{PO}$ or equivalently $\text{VSO} \cap \text{PO}' = \emptyset$.

The most obvious way in which a grammar formalism can be argued to 'capture' an implicational universal is if languages of the unattested type lie *outwith* the generative capacity of the formalism. Thus, formalism F predicts implicational universal $P \subseteq Q$ if the following conditions both hold:

1. there are F-grammars for languages in $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$
2. there is *no* F-grammar for a language in $P \cap Q'$

This mode of explanation underlies many treatments of linguistic universals in the generative grammar literature. Indeed this approach forms the foundation of some of the most interesting CCG analyses of linguistic phenomena, for example the explanation in Steedman (1990) for the observation of Ross (1970) that there is a correlation between basic word order and the existence of forward- and/or backward-gapping constructions in a human language. This example is discussed in detail in **subsection 3.2.1**.

Unfortunately, as the $\text{VSO} \subseteq \text{PO}$ example makes clear, not all empirically motivated implicational universals can be captured so easily, in terms of raw generative power. In particular, most interesting implicational universals are not 'absolute' but are rather 'statistical', in the sense that they posit a distributional tendency rather than a categorical statement. A typical statistical implicational universal is a statement of the following form, where the universal quantifier 'all' becomes the generalised quantifier 'most':

> *Most* human languages which have property P also have property Q.

We can denote such statements set-theoretically as $P \subseteq^* Q$, where the $\subseteq^*$ symbol is intended to mean something like 'is almost a subset of'.

Such statistical universals cannot be fully explained in terms of generative capacity, since they do not involve statements that some language type is completely unattested, but rather declare that it is comparatively rare with respect to other equivalent types. Any formalism whose generative capacity completely excludes languages of a rare

but attested language type cannot be argued to constitute a complete theory of human linguistic competence.

Statistical implicational universals thus require a different mode of explanation. The idea of grammar ranking by means of an evaluation measure is useful in this regard. For instance, we can argue that grammar formalism F captures implicational universal $P \subseteq^* Q$ if the following conditions both hold:

1. there are equivalently ranked F-grammars for comparable languages in $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$

2. the *highest ranked* F-grammar for a language in $P \cap Q'$ is *less highly ranked* than some F-grammar of one of its $P \cap Q$, $P' \cap Q$ or $P' \cap Q'$ equivalents

This general approach can be shown to be implicit in many treatments of implicational universals in the generative grammar literature. Such analyses are generally couched in terms of 'simplicity' or 'elegance' of a particular grammar compared to other more complex or less elegant counterparts. To all intents and purposes, such notions of simplicity and elegance come down to basic considerations of the *size* of either the grammar as a whole, or of important aspects of it, for example the number of entries in the lexicon, the size of the rule component, the number of parameters which need to be set to non-default values, the number of structural transformations required, and so on.

This assumption that the ranking value of a grammar is related to its shortness is made explicit in the CCG formalism. One of the basic methodologies of CCG design is what Steedman (2000) calls the *Principle of Head Categorial Uniqueness* (p.33):

> A single nondisjunctive lexical category for the head of a given construction specifies both the bounded dependencies that arise when its complements are in canonical position and the unbounded dependencies that arise when these complements are displaced under relativization, coordination, and the like.

What this means in practice is that, when we are extending a CCG for some fragment of a language, we should try to avoid the temptation of capturing some *unbounded* syntactic construction by adding an additional lexical category for some word or morpheme already in the lexicon. Steedman justifies this principle in terms of lexical economy (ibid. p.32-33):

> The size of the lexicon involved is ... an important aspect of a grammar's complexity. Other things being equal, one lexical grammar is simpler than another if it captures the same pairings of strings and interpretations using a smaller lexicon.

Subsection 3.2.2 discusses an example from the CCG literature of an implicational universal which is explained in terms of the comparative simplicity of different lexicons. The universal concerned relates the existence of embedded subject relativisations in a language to the possibility of free subject inversion (i.e. the '*pro*-drop parameter'). The argument basically says that, since the smallest CCG lexicon for a language with embedded subject relativisation but without free subject inversion is larger than some CCG lexicon for both the language which is exactly the same but lacks embedded subject relativisation and the language which is exactly the same but has free subject inversion, the correlation is predicted by the CCG formalism.

CCG and implicational universals

Summing up the discussion so far, the aim of linguistic theory can be described in the following manner: to construct a grammar formalism whose generative capacity includes the class of human languages and such that grammars of more likely human languages are more highly ranked than those of less likely ones. Following Chomsky (1965) we may also assume that the obvious numerical evaluation measure to be applied to a grammar is length, in terms of number of symbol tokens. Hence, we want grammars of human languages from statistically common types to be *shorter* than grammars of languages from rare or unattested types. It is possible to interpret the length of a grammar in at least three ways, depending on how accurate we want to be: (a) the number of lexical items in the lexicon; (b) the total number of symbol tokens that make up the lexicon; or (c) the length of the smallest bit string which encodes the lexicon. I shall in general apply the first of these in order to keep things simple, although **subsection 3.2.2** presents an explicit definition of the bit length of an arbitrary CCG using some simple information-theoretic notions.

The main argument pursued in this thesis proceeds from the following observation:

> Although the CCG formalism captures a number of implicational universals in terms of a basic preference for shorter grammars, there are many well-known constituent order universals which are not so easily explained.

This observation obviously assumes some kind of *baseline* definition of the CCG formalism similar to the one presented at the start of this section and discussed more fully in **chapter 2**. This definition incorporates all the essential elements that are accepted by the whole CCG community and assumed in all descriptive CCG work. The baseline CCG formalism incorporates: (a) the binary category connectives $/$, $\backslash$, and $|$; (b) directionally-sensitive versions of functional application, composition, substitution and type raising, lexically controlled by means of the slash modalities of Baldridge (2002), and satisfying the 'principles' of adjacency, consistency and inheritance. Baseline CCG does *not* include innovations like syntactic features, lexical rules or 'multiset' categories, since I regard the inclusion of such descriptive mechanisms in the formalism as an open research question, although I do not deny the inherent rationale behind them.

One example of an implicational universal which does not have a straightforward explanation from the perspective of baseline CCG is a generalisation of the Greenbergian observation mentioned above relating VO/OV clausal order with the existence of prepositions rather than postpositions in a language, i.e. VO languages are generally prepositional (PO) and OV languages are generally postpositional (OP), or VO $\approx$ PO. I argue in **chapter 4** that there is no sense in which baseline CCG lexicons for VO $\cap$ PO or OV $\cap$ OP languages can be argued to be shorter than the shortest lexicons for OV $\cap$ PO or VO $\cap$ OP languages, and hence that the formalism does not capture this implicational universal in any kind of intrinsic manner.

Thus, if CCG is to account for the VO $\approx$ PO implicational universal in terms of a preference for shorter grammars, we need to make some changes to the baseline definition of the formalism. We can take inspiration from the following passage in Chomsky (1965) (p.42):

> But if [length] is to be a meaningful measure, it is necessary to devise notations and to restrict the form of rules in such a way that significant considerations of complexity and generality are converted into considerations of length, so that real generalizations shorten the grammar and spurious ones do not. Thus it is the notational conventions used in presenting a grammar that define "significant generalization", if the evaluation measure is taken as length.

In defence of this approach, the VO $\approx$ PO universal does indeed appear to be motivated by considerations of grammatical economy. Let's first of all make the traditional assumption that the finite verb is the *head* of a clause, and that it selects the subject

and object as *dependents*. Secondly, we assume that adpositions are heads, selecting a single noun phrase dependent. With this in mind, the relevant part of the grammar of a VO ∩ PO language can be summarised using a single statement: *dependents follow their heads*. Similarly, languages in OV ∩ OP will likewise have a single statement for the corresponding phenomena: *dependents precede their heads*. Take however the case of a VO ∩ OP language. If we want to express the relevant generalisation for this kind of language in the same kind of terms, then we need *two* corresponding statements: (a) dependents *follow* verbal heads; and (b) dependents *precede* adpositional heads. Thus, assuming the appropriate notational metalanguage, it is clear that the grammar of a language of the unattested type will require more statements than an equivalent language of one of the other types — in other words, it will be 'longer'.

A model-theoretic perspective on CCG category notation

This thesis argues that the best way to go about formulating a version of CCG which allows the expression of such generalisations is to take a *model-theoretic*, or 'constraint-based', perspective on the notation for the categories which words and morphemes are assigned to in the lexicon. Recall the definition of baseline CCG category notation:

> Given some alphabet A of atomic category symbols, the set of CCG categories over A is the closure of A under the infix operators $/$ and $\backslash$.

A model-theoretic treatment of CCG categories involves making a crucial distinction between the traditional notion of a category label as a kind of metalinguistic *formula* and that of a category as a kind of mathematical *model* which underlies such formulas. Thus, the theory of CCG categories will have the following aspects:

1. a class of *models* representing category notation as labelled tree-like structures
2. a set of *formulas* for describing the models
3. a *satisfaction* relation specifying which models satisfy which descriptions

Section 2.1 presents a model-theoretic definition of baseline CCG category notation in these terms.

Whilst the theory of CCG categories is expressed model-theoretically, I retain a traditional conception of the CCG combinatory system as a generative-enumerative mechanism. In other words, I define CCG combinatory rules like forward application as *operations* on category models, rather than structural constraints on the trees

underlying CCG derivations. Although such a *purely model-theoretic* conception of CCG is possible (see **section 2.5**), and is an interesting subject of research in itself, I have chosen not to go down this route. Rather I stick with a *partially model-theoretic* conception, basically because my focus of interest is not the combinatory system of CCG, which I accept pretty much exactly as defined in Steedman (2000) and Baldridge (2002).

One immediate consequence of having a distinction between category models and category descriptions is that it permits us to develop more flexible metalanguages for category description, while retaining the rest of the CCG formalism as is. In other words, we can hold constant both the underlying notion of category models and the combinatory rules which operate upon them, whilst experimenting with different kinds of description language in order to try and come up with one which fulfils Chomsky's criterion of capturing significant linguistic generalisations in terms of length.

This methodology is not strictly speaking new to the CCG community, since it arguably underlies the introduction of 'multiset' CCG notation in Hoffman (1995) and Baldridge (2002), if we assume that a multiset category formula is simply a metalinguistic means of specifying a *set* of distinct categories. What is proposed in this thesis is that we can take the idea a lot further and make it do a lot more work, capturing more significant linguistic generalisations in terms of lexical economy. In order to achieve this, I propose a category description metalanguage based on the attribute-value logic underlying unification-based grammar formalisms (Shieber, 1986). This language includes statements of three important types, among others:

- statements of the form ARG ϕ denote the set of expressions which select an argument of category ϕ
- statements of the form SLASH / and SLASH \ denote the sets of expressions which select respectively a following and a preceding argument
- statements of the form RES ϕ denote the set of expressions which yield an expression of category ϕ when combined with an argument

This method of notation is inspired by previous attempts to implement categorial grammars in unification-based grammar engineering environments, for example Uszkoreit (1986), Zeevat et al. (1987), Villavicencio (2002), Beavers (2004). In addition, it is inspired particularly by the treatment in Blackburn (1993) of attribute-value notations

as applications of multimodal logic, interpreted on Kripke structures, and where unification is a matter of logical satisfiability.

Lexical inheritance in CCG

A second aspect of the generalised CCG formalism proposed in this thesis is that I propose that the definition of a CCG grammar be revised to include a *lexical inheritance hierarchy* (Flickinger, 1987). In other words, a grammar over alphabet Σ is not simply an ordered triple $\langle A, S, L \rangle$, where A is an alphabet of atomic category symbols, $S \in A$, and L is a mapping from Σ to CCG categories over A. The kind of inheritance hierarchy I propose is based on the 'type hierarchies' of Carpenter (1992), i.e. ordered pairs of the form $\langle T, \sqsubseteq \rangle$ where T is an alphabet of types and $\sqsubseteq$ is a particular kind of weak ordering relation denoting the notion of 'subtype'. Such hierarchies are usually expressed as tree-like structures in, for example, HPSG grammars or descriptions of programs written in 'object-oriented' programming languages like C++ or Java. **Chapter 5** discusses type hierarchies and their application to CCG at greater length.

Inheritance hierarchies are basically type hierarchies where types are associated with sets of formulas from some constraint language. The intuition behind an inheritance hierarchy is that a given type is subject to not only its own constraints but also to all those belonging to all of its supertypes as well.[6] I assume that types are associated with constraints from the language discussed above. Formally, an inheritance hierarchy over constraint language Φ is characterised as an ordered triple $\langle T, \sqsubseteq, f \rangle$ where $\langle T, \sqsubseteq \rangle$ is a type hierarchy and f is a function from T to subsets of Φ.

With all this in mind, I assume a definition of the CCG formalism where a grammar over alphabet Σ is at least an ordered 6-tuple $\langle A, S, B, \sqsubseteq_B, b, L \rangle$, where:

1. A is an alphabet of saturated category symbols, e.g. S, NP
2. $S \in A$ i.e. the distinguished, sentential symbol
3. B is an alphabet of lexical types, e.g. *transitive-verb*, *adposition*
4. $\langle B, \sqsubseteq_B, b \rangle$ is an inheritance hierarchy over the constraint language discussed above

[6] Thus I assume that inheritance is 'monotonic' and that a type may have more than one immediate supertype.

5. lexicon L is a function from Σ to B, i.e. all lexical assignments are to categories denoted by non-composite symbols

Non-redundant CCG lexicons

The primary reason for generalising the definition of the CCG formalism in this way is to develop an explanatory theory of human linguistic competence, where significant linguistic generalisations are converted into considerations of economy. However, it is also the case that the particular innovations assumed here, especially the notion of a lexical inheritance hierarchy over an underspecified constraint language, have an independent motivation. Specifically, as Pollard and Sag (1987) show, they allow for the formulation of *non-redundant* lexicons for human languages within a lexicalised grammar formalism.

The requirement of non-redundancy is a direct consequence of the sheer *size* of a typical human language lexicon. This can contain upwards of 50,000 distinct items, which will need to be stored in a reasonably efficient manner, at least as far as the efficiency of the language acquisition process is concerned. I interpret this general requirement of efficiency as imposing two independent ideals on human language lexicons, construed as mappings from morphemes to categories:

ideal of functionality a lexicon is ideally a *function* from forms to categories

ideal of atomicity a lexicon is a mapping from forms ideally to an alphabet of *non-composite* symbols

In other words, a human language lexicon should map phonetic forms to at most one lexical category, and lexical categories should be represented by atomic symbols, possibly macro-like abbreviations for the traditional 'functional' types of categorial grammar.

Of course, these two conditions should be considered as ideals rather than as necessities, the motivating principle being that a lexicon should satisfy them as much as it can, other things being equal. In particular, the ideal of functionality may be overridden by considerations of genuine semantic ambiguity. Also, atomicity is only really an issue for open class lexical items like nouns and verbs, of which a lexicon will contain thousands of instances, and not necessarily for functional elements like prepositions or auxiliary verbs. Together, the ideals of functionality and atomicity prioritise

space-efficient lexicons, of the kind assumed by other, unification-based, lexicalised grammar formalisms like HPSG and the Lexical-Functional Grammar (LFG) formalism (Bresnan, 1982).

The ideals of functionality and atomicity pose a particular problem for CCG due to the extreme violations of them that baseline CCG analyses of diverse syntactic phenomena appear, at least on the surface, to imply. In the first place, the accepted way of handling local, bounded dependency constructions like case, agreement, binding, pro-drop or local scrambling in a CCG involves assigning certain lexical forms (usually verbs) to a plethora of lexical categories. For example, Baldridge (1998) argues that a satisfactory baseline CCG account of local scrambling in Tagalog would involve assigning every ditransitive verb to *six* distinct lexical categories, one for each possible postverbal ordering of subject, direct object and indirect object. Secondly, CCG lexicons are inherently non-atomic, in that the sole responsibility for regulating syntactic combination lies with the lexical categories assigned to heads. For example, a verb may be assigned to a lexical category like $(\mathsf{S}/\mathsf{NP})/\mathsf{PP}$, signalling that it subcategorises for exactly two arguments, one NP and one PP.

The ideal of functionality has been the underlying motivating factor in the development of certain innovative CCG category notations. For example, the set-based categories introduced by Hoffman (1995) and Baldridge (2002) for languages with local scrambling constructions were designed so as to allow CCG lexicons where each verb is mapped to a single lexical category. Other violations of functionality are handled in Steedman (2000) by, variously, lexical rules and feature structure underspecification. In addition, considerations of lexical atomicity have played a role as well, especially in the development of computational environments implementing CCG grammars, where a number of ad-hoc engineering solutions have been utilised in order to attain a greater degree of atomicity, for example category macros and XTAG-style lexical families.[7]

The advantage of the present proposal, where the generalised notion of a CCG grammar incorporates an inheritance hierarchy of lexical categories, is that it involves a unified solution to the problem of lexical redundancy. In other words, all the other suggested notations can be replaced by a single descriptive mechanism. **Chapter 6** spells out in greater detail how CCG with lexical inheritance allows us to formulate human language lexicons which satisfy both the ideals of functionality and atomicity.

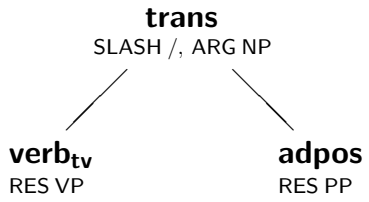
[7] See `http://www.cis.upenn.edu/~xtag`

Inheritance-driven CCG and linguistic competence

The incorporation of lexical inheritance hierarchies into CCG grammar, as well as providing for non-redundant lexicons, also gives rise to a formalism which constitutes a better theory of human linguistic competence. Recall the problem discussed above, involving the fact that there are many well-known implicational universals which are not captured by baseline CCG in terms of a basic preference for shorter grammars. One example given was the VO $\approx$ PO universal, relating clausal word order and the existence of prepositions in a human language. I mentioned above that this universal has a kind of economy-driven flavour to it, in that the unattested language type appears to require a greater number of generalisations than the attested types do. This intuition can be formalised using lexical inheritance hierarchies over the simple feature-based constraint language discussed above.

Here is the simplest lexical inheritance hierarchy for the relevant fragment of a VO $\cap$ PO language:

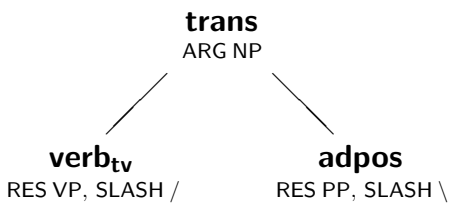
(1.3)



The hierarchy consists of three lexical types: transitive verbs, adpositions, and a supertype 'trans' of which these are exclusive subtypes. This supertype captures the features that transitive verbs and adpositions have in common, i.e. in the case of a VO $\cap$ PO language they both take following NP arguments. The respective subtypes encode the properties which are peculiar to each, in this case the incompatible result categories. Note that this fragment of the lexical inheritance hierarchy contains a total of *four* constraints, including just the one for argument directionality.

In contrast, the simplest lexical inheritance hierarchy for the corresponding fragment of a VO $\cap$ OP language is as follows:

(1.4)



In this case, the only property shared by transitive verbs and adpositions is that they take NP arguments. The directionality must be specified separately for each type. The resulting hierarchy contains *five* constraints, two of which express argument directionality.

Thus, it is clear that the simplest lexical inheritance hierarchy for the relevant fragment of a VO ∩ OP language is *larger* than that for the corresponding VO ∩ PO language, in that it contains more constraints. Recall that a CCG is now defined as an ordered 6-tuple $\langle A, S, B, \sqsubseteq_B, b, L\rangle$. Can we claim that the size of such a CCG is determinedly either solely or mainly by the size of the lexical inheritance hierarchy $\langle B, \sqsubseteq_B, b\rangle$? Note that the definition above stated that the lexicon L should be a *function* from the alphabet of terminals to the alphabet B of lexical types. In other words, this kind of CCG lexicon satisfies the ideals of functionality and atomicity *by definition*. One consequence of this is that the lexicons for the relevant fragments of all four language types in our discussion of the VO ≈ PO universal will be identical:

(1.5) $\mathsf{love}, \mathsf{kill}, \mathsf{devour}, \ldots \vdash \mathsf{verb_{tv}}$

$\mathsf{beyond}, \mathsf{during}, \mathsf{within}, \ldots \vdash \mathsf{adpos}$

Thus, we can conclude that the size of a particular CCG is mainly determined by the size of its lexical inheritance hierarchy, This suggests that, other things being equal, the fewer constraints in the hierarchy, the more likely it is to be part of the grammar of a human language.

Therefore, it appears reasonable to argue that the revised CCG formalism, where grammars incorporate inheritance hierarchies of lexical types constrained by formulas of the particular constraint language discussed above, predicts the implicational universal VO ≈ PO in terms of a general preference for shorter grammars and hence smaller lexical hierarchies. **Chapter 7** shows how this simple example can be generalised to account for more complex implicational universals involving constituent ordering.

The proposal

To sum up this section, in this thesis I will set out to justify the following hypothesis: if we take a partially model-theoretic, or 'constraint-based', perspective on the CCG formalism, whereby a theoretical distinction is drawn between category descriptions and the underlying category models which satisfy them, then we can construct a hybrid formalism which:

1. provides non-redundant lexicons for natural languages i.e. lexicons which satisfy the ideals of functionality and atomicity

2. captures some well-known implicational universals of human language as predominantly formal rather than substantive universals, based on a general preference for 'shorter' grammars

I present an extension of the CCG formalism, which I informally call 'inheritance-driven' CCG (I-CCG), where:

1. grammars incorporate inheritance hierarchies of lexical categories, defined over a simple, feature-based constraint language

2. lexicons are functions from morphemes to atomic lexical types

Before I conclude this introduction, there are two points left to discuss. Firstly, in the previous discussion I mentioned that mainstream generative grammar largely abandoned the 'grammar-ranking' approach to explaining language acquisition and hence implicational universals, and focused overwhelmingly on positing substantive constraints on grammars, ultimately leading to the idea that there is a finite number of human grammars. This paradigm-shift is defended by van Riemsdijk and Williams (1986) in the following terms (p.13):

> Clearly "shortness" is one component of simplicity — all else being equal, short grammars are preferred over longer ones — but it was quite clear from the start that "shortness" did not exhaust the technical notion of simplicity.
>
> This approach, including development of general rule-writing systems and formal evaluation of grammars, has largely been abandoned in favor of less formal and more "substantive" ideas about how grammars are composed. This has happened largely because nothing of interest emerged beyond the initial idea of "shortness" that helped to define how the language learner or the linguist is to determine the correct grammar from finite data.

I contend that this negative conclusion is largely an artifact of the dominant grammar formalism of the era i.e. (Extended) Standard Theory transformational grammar. By the end of this thesis I hope to have demonstrated that, at least for a range of constituent ordering universals, we can construct a non-transformational, non-phrase-structural theory where shortness does indeed "exhaust the technical notion of simplicity".

Secondly, even if we do accept the grammar-ranking paradigm, it is not necessarily the case that we have to assume that the relevant evaluation measure involves some formal property of the grammar itself, such as its length. For example, Hawkins (1990)[8] proposes that phrase structure grammars be ranked according to the average amount of effort required to parse the strings they generate, given some formal model of sentence processing. The idea is that the more easily parsable a grammar is, the more likely it is to be a human language grammar. **Chapter 7** briefly considers such parsing complexity accounts of grammar-ranking from a CCG perspective. For the moment, it suffices to say that the two methods of ranking grammars, according to either length or parsability, are by no means incompatible.

1.3 Thesis outline

This thesis will proceed as follows:

Chapter 2: Combinatory Categorial Grammar introduces and formally defines the baseline CCG formalism — basically the core concepts which everyone working within CCG accepts as being crucial. CCG is defined as a partially model-theoretic system, in that the theory underlying CCG category notation makes a crucial distinction between categories as models and categories as descriptions of models. This distinction forms the basis for the methodological approach pursued in the remainder of this thesis, where I hold constant the category models and experiment with different description languages, in order to try and find a notation capable of capturing significant linguistic generalisations as considerations of lexical economy.

Chapter 3: CCG and human language competence discusses the CCG formalism as a theory of human linguistic competence. I start by reviewing a number of arguments from the CCG literature concerning the relationship between the generative capacity of CCG and predictions it makes about the permitted complexity of unbounded dependency constructions in human languages. Then I turn to the subject of implicational universals and consider two examples which CCG has been argued to capture formally. The first of these is the observation that human languages with backward-gapping constructions are generally verb-final, and the second involves a correlation between the extractability of embedded subjects in a language and the possibility of free subject

[8] Work that has recently been revisited by Newmeyer (2005).

inversion. I pay particular attention to the underlying rationale behind the claim that CCG predicts these universals, and conclude that, at least in the second case, the CCG analysis is implicitly based on the notion of grammar-ranking in terms of an evaluation measure of grammar length.

Chapter 4: Some problematic data draws attention to a couple of implicational universals which are not captured by the baseline CCG formalism in terms of lexical economy. The first of these is the familiar Greenbergian correlation between basic clausal ordering in a human language and the existence of prepositions or postpositions in its lexicon. The second is a lesser-known observation from the Principles and Parameters literature which posits a link between verb-object and verb-modifier serialisation in human languages. I argue that, although there is no real sense in which baseline CCG can be argued to predict these universals formally, the generalisations do appear to be based on some notion of grammatical economy, i.e. the number of descriptive statements required to characterise the phenomena. This provides the motivation for the next three chapters — to extend the definition of the baseline CCG formalism so as to be able to express the relevant kinds of generalisation.

Chapter 5: Type-hierarchical CCG presents a redefinition of the CCG formalism, where the alphabet of saturated category symbols is organised into a type hierarchy. The motivation for this extra machinery derives from the requirement that the information contained within a human language lexicon be stored in a reasonably efficient manner, i.e. children acquiring human languages aim to construct lexicons which satisfy the ideals of functionality and atomicity. I call the redefined formalism 'type-hierarchical CCG' (T-CCG), and argue that it renders redundant a number of other proposed generalisations of the baseline CCG category notation, where saturated categories either are conceptualised as typed feature structures or are prefixed by unary modalities.

Chapter 6: Inheritance-driven CCG presents another redefinition of the CCG formalism, incorporating an alphabet of lexical category symbols organised into an inheritance hierarchy over a simple attribute-value-style category description language. I call this formalism 'inheritance-driven CCG' (I-CCG), and show that it allows for the construction of space-efficient human language lexicons, satisfying both the ideals of functionality and atomicity.

Chapter 7: I-CCG and linguistic competence argues that the I-CCG formalism

also captures, as primarily *formal* universals, those implicational universals that were shown to be problematic for the baseline CCG formalism, as well as number of others involving consistent operator-operand ordering. This explanation takes the form of a basic preference for lexical inheritance hierarchies with fewer constraints.

Chapter 8: Conclusion brings the thesis to a close.

Chapter 2

Combinatory Categorial Grammar

The aim of this chapter is to introduce the Combinatory Categorial Grammar (CCG) formalism of Steedman (2000) and Baldridge (2002). CCG is a principled, linguistically motivated, computationally tractable and psycholinguistically realistic generalisation of the categorial grammars of Ajdukiewicz (1935) and Bar-Hillel (1953). This chapter presents the definition of the CCG formalism in considerable detail. In particular, I define CCG as an *partially model-theoretic* (or 'constraint-based') system (Pullum and Scholz, 2001), where a crucial distinction is drawn between linguistic structures and descriptions of these structures. I have found this approach to be useful in order to prepare the ground for the proposals which I make in this thesis.

This chapter will proceed as follows:

Section 2.1 introduces the notion of a 'category label', denoting a 'category' or set of expressions. I formally distinguish the conception of a category label as a kind of abstract mathematical structure from that of a category label as a statement in some formal language. These two different conceptions are related by a 'satisfaction' definition, telling us which structures are licensed by which descriptions. The distinction between structure and description, though familiar from explicitly model-theoretic grammar formalisms like HPSG, is not normally made in expositions of categorial frameworks like CCG, although I contend that it is implicit in much descriptive and computational work.

Section 2.2 introduces the combinatory operations of CCG, as operations on the structures underlying category descriptions, rather than on category descriptions themselves. I discuss how the combinatory operations are not in themselves primitives of

the CCG formalism, but rather are to be seen as theorems of the underlying 'principles' of CCG.

Section 2.3 provides a formal definition of the 'baseline' CCG formalism on which the work in this thesis will build. This involves a specification of a set of grammars over arbitrary alphabets of terminal symbols, along with a 'generates' relation, telling us which grammars generate which strings of terminals. This definition of the baseline CCG formalism is the main contribution this chapter will make to the overall argumentation of this thesis.

Section 2.4 contains a brief discussion of semantics in CCG , in particular how every CCG of a language is also a compositional theory of its semantics. Both denotational and representational approaches are discussed.

Section 2.5 presents a brief sketch of how the entire CCG formalism, and not just that part of it dealing with category notation, can be defined model-theoretically, in terms of constraints on derivation structures. This section is rather more speculative than the previous ones, and will not be pursued in the remainder of this thesis.

2.1 Categories

2.1.1 A brief history of category notation in CCG

In general linguistic theory, 'categories' are sets of linguistic expressions which have a similar distribution in larger expressions. Category 'labels', for example 'sentence' or 'noun', are used to denote particular categories. In the original phrase structure-based grammar formalisms such as context-free grammar, category labels are understood as being atomic symbols (i.e. categories *partition* the set of well-formed expressions, modulo ambiguity), and categories are related to one another by means of language-particular phrase structure rules. The family of categorial grammar formalisms departs from this approach in two ways. First of all, category labels may be complex, containing other category labels; and secondly, partly as a result of this, there are no language-particular rules of syntactic combination.

Ajdukiewicz (1935) identified two distinct kinds of category of expression in formal languages. The 'saturated' categories, also known as primitive, basic, atomic or non-functional categories, are denoted by atomic symbols, just like the categories of

phrase structure grammar. On the other hand, the 'unsaturated' categories, alternatively the functor, complex, functional or operator categories, are denoted by fractions of the form $\alpha/\beta_1 \ldots \beta_{n\geq 1}$ where α and β_i are category labels. Intuitively, an unsaturated category label of the form $\alpha/\beta_1 \ldots \beta_n$ is to be understood as denoting the set of expressions which, when combined with expressions $x_1, \ldots, x_n$ of categories $\beta_1, \ldots, \beta_n$, give rise to an expression of category α. For example, the conjunction connective & of propositional logic can be assigned to the category denoted by S/SS, where the symbol 'S' stands for 'sentence', encoding the fact that it combines with two sentences (i.e. the conjuncts) to form another, more complex, sentence (i.e. the conjunction).

In Ajdukiewicz' system there is a denumerably infinite number of unsaturated category labels for any alphabet of saturated category symbols — "an unbounded and ramified ascending hierarchy of functor categories characterised in two ways: first by number and semantic category of their arguments taken in order; second by the semantic category of the whole composite expression formed by them together with their arguments" (p.210). In particular, an unsaturated category label can itself be the numerator or part of the denominator of an unsaturated category label. For example, Ajdukiewicz denotes the category of the English adverb 'strongly' as $(\mathsf{S}/\mathsf{NP})/(\mathsf{S}/\mathsf{NP})$, with NP standing for 'noun phrase', where both the numerator and the denominator consist of unsaturated category labels.

In addition, every unsaturated category label of the form $\alpha/\beta_1 \ldots \beta_n$ can be converted into a 'curried' equivalent where all functors take exactly one argument, in other words $((\alpha/\beta_n)/\ldots)/\beta_1$. For example the category of the conjunction connective & in propositional logic can be denoted as $(\mathsf{S}/\mathsf{S})/\mathsf{S}$ i.e. it combines with a sentence to yield an expression which itself combines with a sentence to yield a sentence. The use of such curried representations of unsaturated categories has since become standard in categorial grammar formalisms, although variants such as HPSG have retained the uncurried format to allow for the possibility of 'flat' derivations. Curried representations of unsaturated category labels can be understood as binary trees, where: (a) every non-leaf node dominates two other nodes, one of which is the 'result' and the other the 'argument'; and (b) every leaf node carries a saturated category symbol.

Bar-Hillel (1953) generalises the unsaturated category labels of Ajdukiewicz to distinguish arguments which precede a functor from those which follow it. In this notation, an unsaturated category is represented by a fraction of the following form,

where α, β_i and γ_i are category labels (saturated or unsaturated) and $n+m \geq 1$:

(2.1) $\alpha/(\beta_1)\ldots(\beta_n)[\gamma_1]\ldots[\gamma_m]$

Arguments enclosed in curved brackets must precede the functor and those enclosed in square brackets must follow. Thus a fraction of the form in (2.1) is used to denote a class of expressions which, when combined with a preceding sequence of expressions $x_1,\ldots,x_n$ of categories $\beta_1,\ldots,\beta_n$, and a following sequence of expressions $y_1,\ldots,y_m$ of categories $\gamma_1,\ldots,\gamma_m$, give rise to an expression of category α. Thus, the 'infix' nature of the conjunction connective & in standard logical notation can be captured by assigning it to the category denoted by $\mathsf{S}/(\mathsf{S})[\mathsf{S}]$ — it combines with one preceding sentence (the left conjunct) and one following sentence (the right conjunct) to yield a complex sentence.

The standard notation for categories in categorial grammar formalisms, including CCG, involves applying the currying convention to Bar-Hillel's ordered categories. Given some alphabet A of saturated category labels, the set of all category labels over A is defined as follows:

1. every element of A is a category label over A
2. if α and β are both category labels over A, then so are (α/β) and $(\alpha\backslash\beta)$

A category label of the form α/β denotes the class of expressions which, when combined with a *following* expression of category β, yield an expression of category α. On the other hand, a category label of the form $\alpha\backslash\beta$ denotes the class of expressions which, when combined with a *preceding* expression of category β, yield an expression of category α. Thus, the conjunction connective & in standard infix logical notation can assigned to either of the categories denoted as $(\mathsf{S}\backslash\mathsf{S})/\mathsf{S}$ or $(\mathsf{S}/\mathsf{S})\backslash\mathsf{S}$. Again, curried representations of such directional category labels can be understood as binary trees, where: (a) every non-leaf node is labelled by either / or \; (b) every non-leaf node dominates two other nodes, one of which is the result and the other the argument; and (c) every leaf node is labelled by a saturated category symbol.

Some categorial grammarians follow Lambek (1958) in interpreting the backslash differently i.e. they understand a category label of the form $\alpha\backslash\beta$ as denoting the class of expressions which, when combined with a preceding expression of category α, yield an expression of category β. This interpretation is not without its advantages, particularly

when conceptualising category labels as formulas of some logical calculus. However, I will follow the alternative 'result-first' convention throughout this thesis.

In addition, some categorial grammars utilise a third slash | representing unspecified directionality. Thus, a category label of the form $\alpha|\beta$ denotes the class of expressions which, when combined with either a preceding or a following expression of category β, yield an expression of category α. In other words:

(2.2) $\alpha|\beta = \alpha/\beta \cup \alpha\backslash\beta$

A number of other abbreviatory conventions are also of interest. First of all, one occasionally sees *category variables* used in category labels in CCG analyses. For example, a category label such as $(\mathsf{S}\backslash\mathsf{NP})/X$ is intended to represent the *set union* of all / categories whose result is denoted by $\mathsf{S}\backslash\mathsf{NP}$ i.e.

(2.3) $(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} \cup (\mathsf{S}\backslash\mathsf{NP})/\mathsf{S} \cup (\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP}) \cup \ldots$

Multiple occurences of the same variable in a category label are meant to indicate token identity, for example $(\mathsf{S}/X)/((\mathsf{S}\backslash\mathsf{NP})/X)$. This kind of token identity in category labels means that they can no longer be seen as binary *trees*, but rather as a particular kind of rooted directed acyclic graph which allows reentrancy.

Secondly, \$ variables are sometimes used to schematise over sequences of arguments together with their associated directionality. Thus, a category label such as $(\mathsf{S}\backslash\mathsf{NP})/\$$ is intended to represent the set union of all / categories whose *penultimate* result is denoted $\mathsf{S}\backslash\mathsf{NP}$. This includes not only all the expressions in $\mathsf{S}\backslash\mathsf{NP}$ and $(\mathsf{S}\backslash\mathsf{NP})/X$, but also all those in $((\mathsf{S}\backslash\mathsf{NP})/X)/Y$, $(((\mathsf{S}\backslash\mathsf{NP})/X)/Y)/Z$, and so on. Similar conventions exist for $\backslash\$$ and $|\$$. Again, such variables may occur more than once in a single category label, thus indicating token identity. For example, $(\mathsf{S}/\$)/((\mathsf{S}\backslash\mathsf{NP})/\$)$ denotes the following sets of expressions:

(2.4) $\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP}) \cup (\mathsf{S}/X)/((\mathsf{S}\backslash\mathsf{NP})/X) \cup ((\mathsf{S}/X)/Y)/(((\mathsf{S}\backslash\mathsf{NP})/X)/Y) \cup \ldots$

Thirdly, following the work of Hoffman (1995) it has become commonplace to use a *set-based* unsaturated category notation, particularly for CCG analyses of languages with scrambling of arguments within the local domain of the verb (e.g. Turkish, German, Japanese, Tagalog). For example, a category label such as $\mathsf{S}\{/\mathsf{NP}_{\mathsf{obj}}, \backslash\mathsf{NP}_{\mathsf{sbj}}\}$ can be seen as a kind of shorthand representing two distinct CCG category labels: $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$ and $(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}})\backslash\mathsf{NP}_{\mathsf{sbj}}$. In other words, the use of set-based category

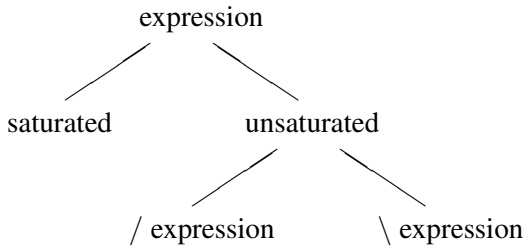


Figure 2.1: A taxonomy of expressions

labels underspecifies the particular order in which a functor seeks its arguments, whilst optionally preserving their directionality (forwards or backwards).

The standard definition of category labels presented above is essentially the one I will assume throughout this thesis. Note that it involves first of all partitioning the set of expressions into two subclasses, the saturated expressions and the unsaturated expressions, and then partitioning the latter into two further subclasses, the / expressions and the \ expressions. Such a taxonomy is represented in Figure 2.1. As categorial grammars have been applied to a wider range of syntactic phenomena in human languages, it has become necessary to partition the set of expressions in different ways, in order to allow the syntactic behaviour of categories to be distinguished in finer detail. For example, Baldridge (2002) proposes to partition the unsaturated expressions along two further dimensions, corresponding to the syntactic combinatory properties of associativity and permutativity.

This extended taxonomy is necessary for a full specification of the CCG formalism, especially as it has been applied to the nitty-gritty of extraction phenomena in English and Dutch.[1] However, for the purposes of this thesis, I will more often than not assume the simple taxonomy of CCG categories in Figure 2.1, since the focus is not on the syntax of extraction but the basic principles of lexical organisation. All of the definitions in this thesis can be easily extended to incorporate the full taxonomy of categories discussed in Baldridge (2002) and Baldridge and Kruijff (2003).

One important innovation which requires mention at the outset is the use of *slash modalities* to permit lexical control over combinatory operations, as advocated by

[1] See Chapter 5 of Baldridge (2002) for details.

Baldridge (2002). The simplest version of CCG which incorporates modalities involves the following definition of category labels:

1. every element of A is a category label over A
2. if α and β are both category labels over A, then for all $m \in \{\star, \diamond, \times\}$ so are $(\alpha /_m \beta)$ and $(\alpha \backslash^m \beta)$

Expressions belonging to a $\star$ category are assumed to be neither associative nor permutative, and thus can only enter into the most basic combinatory operations i.e. forward and backward application. Expression assigned to $\diamond$ and $\times$ are more permissive: $\diamond$ expressions are associative but non-permutative, and can thus participate in *harmonic* composition and substitution operations; $\times$ expressions are permutative but non-associative, and can participate in *disharmonic* composition and substitution operations. The use of slash modalities in CCG categories was inspired by work in the related categorial grammar framework known as 'Categorial Type Logic' (Morrill, 1994), (Hepple, 1995), (Moortgat, 1997).

I note in passing that one important dimension of variation has been ignored in this brief history of category notation in CCG — I have said nothing about the structure of saturated category labels themselves, assuming them to be simply atomic symbols. I will return to this question in section 5.5.

The following subsections present the syntax and semantics of the baseline category notation I will assume in the rest of this thesis.

Subsections 2.1.2 - 2.1.4 define the mathematical structures, or 'category models', undelying both the baseline CCG category notation and its variants discussed above.

Subsection 2.1.5 formalises the baseline CCG category notation itself as a formal language, the language of 'category descriptions'.

Subsection 2.1.6 relates these two notions by means of a 'satisfaction' relation, specifying which category models can be seen as underlying which category descriptions.

Subsection 2.1.7 concludes with a brief discussion of some of the advantages of taking an explicitly model-theoretic perspective on CCG category notation.

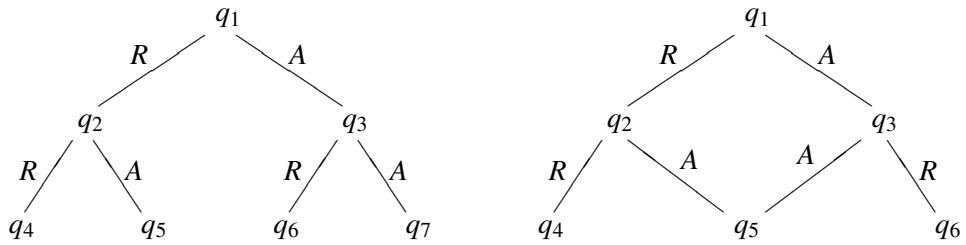


Figure 2.2: Two category frames

2.1.2 Category frames

Figure 2.2 presents two examples of the kind of feature structure I call 'category frames'. A category frame consists of a set of points, along with two relations $R(esult)$ and $A(rgument)$ on those points. By convention, the arcs denoting relational connections point downwards. Formally, a category frame is an ordered triple $\langle Q, Res, Arg \rangle$, where:

1. Q is a finite, non-empty set of points
2. Res is the 'result' relation on Q i.e. if $\langle q_i, q_j \rangle \in Res$ then point q_j is a 'result' of point q_i
3. Arg is the 'argument' relation on Q i.e. if $\langle q_i, q_j \rangle \in Arg$ then point q_j is an 'argument' of point q_i
4. Res is a functional mapping from Q to Q i.e. every point has at most one result, although not every point has a result
5. Arg is a functional mapping from Q to Q i.e. every point has at most one argument, although not every point has an argument
6. the domains of Res and Arg are identical i.e. every point with a result has an argument and vice versa
7. where $\prec$ is the transitive closure of $Res \cup Arg$, $\langle Q, \prec \rangle$ is a strict (partial) order with a first/least element i.e. no point is its own result or argument, and no point is the result or argument of a point contained within its own result or argument

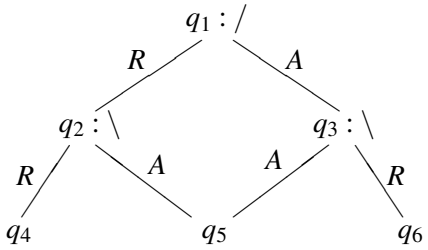


Figure 2.3: A category structure

As Figure 2.2 shows, category frames can be, but are not obliged to be, trees. In other words, reentrancy is allowed. All category frames are rooted, all and only non-end points have exactly one result and exactly one argument, and category frames are acyclic.

The 'root' point of category frame $\langle Q, Res, Arg \rangle$ is the first element of $\langle Q, \prec \rangle$, where $\prec$ is the transitive closure of $Res \cup Arg$. The set of 'end' points of category frame $\langle Q, Res, Arg \rangle$ is defined as $Q - R$ where R is the domain of Res.

2.1.3 Category structures

Figure 2.3 presents an example of the kind of feature structure I call a 'category structure', based on the righthand category frame in Figure 2.2. A category structure is a category frame, where every non-end point is annotated by a directional slash, i.e. $/$ or $\backslash$. Formally, a category structure is an ordered 4-tuple $\langle Q, Res, Arg, V_S \rangle$, where:

1. $\langle Q, Res, Arg \rangle$ is a category frame
2. V_S is a function from the non-end points of $\langle Q, Res, Arg \rangle$ to $\{/, \backslash\}$ i.e. every non-end point is labelled with exactly one directionality symbol

2.1.4 Category models

Figure 2.4 contains a feature structure representing a 'category model' over alphabet $\{\mathsf{S}, \mathsf{NP}\}$, based on the category structure in Figure 2.3. A category model over alphabet A is a category structure, where every end point is annotated with a symbol in A. Formally, a category model over alphabet A of saturated category symbols is an ordered

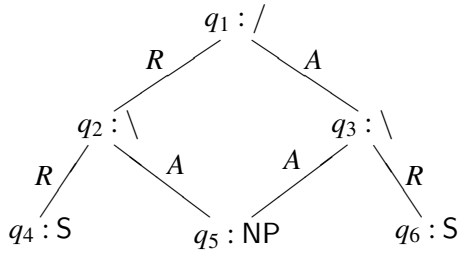


Figure 2.4: A category model over $\{\mathsf{S}, \mathsf{NP}\}$

5-tuple $\langle Q, Res, Arg, V_S, V_A \rangle$, where:

1. $\langle Q, Res, Arg, V_S \rangle$ is a category structure
2. V_A is a function from the end points of $\langle Q, Res, Arg \rangle$ to A i.e. every end point in the structure is labelled by exactly one saturated category symbol

The root of category model $\langle Q, Res, Arg, V_S, V_A \rangle$ over some alphabet A is defined as the root of the underlying category frame $\langle Q, Res, Arg \rangle$.

2.1.5 Category descriptions

In subsection 2.1.4, I defined the structures underlying the category notation of the CCG formalism as a particular kind of feature structure. In this section, I introduce a simple description language for categories, based on the traditional linear notation for unsaturated categories discussed in subsection 2.1.1.

The set of 'category descriptions' over alphabet A of saturated category symbols is defined as the smallest set Φ such that:

1. $A \subseteq \Phi$ i.e. every saturated category symbol is itself a description
2. for all $\phi, \psi \in \Phi$, (ϕ/ψ), $(\phi\backslash\psi)$ and $(\phi|\psi)$ are all in Φ

The directional slashes used in the category descriptions include a slash $|$ denoting unspecified directionality. This symbol was not found in the definition of category structures in subection 2.1.3. According to this definition, the following are all category descriptions over $\{\mathsf{S}, \mathsf{NP}\}$:

(2.5) S, NP, $\mathsf{S/NP}$, $(\mathsf{S|NP})\backslash\mathsf{S}$, $(\mathsf{NP}\backslash\mathsf{NP})/(\mathsf{S|NP})$, $((\mathsf{S}\backslash\mathsf{NP})/\mathsf{S})|\mathsf{NP}$

2.1.6 Satisfaction

In subsection 2.1.4 I defined a class of formal structures embodying the intuitive notion of CCG category notation. In subsection 2.1.5, I presented a simple formal language for describing such structures. In this section I introduce the 'satisfaction' relation linking the two, determining which category models satisfy which descriptions, or alternatively which descriptions are true in which category models.

Formally, category model $\mathcal{M}$ over alphabet A satisfies category description ϕ over A, written $\mathcal{M} \models \phi$, if and only if $\mathcal{M}, q \models \phi$, where q is the root of $\mathcal{M}$. In other words, a category description is *globally* satisfied by a category model just in case it is *locally* satisfied from the root point of the category model. The definition of local satisfaction is as follows: Category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over alphabet A locally satisfies category description ϕ over A, from point $q \in Q$, written $\mathcal{M}, q \models \phi$, if and only if:

1. where $\phi \in A$: $V_A(q) = \phi$
2. where $\phi = (\psi_1/\psi_2)$: $V_S(q) = /$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
3. where $\phi = (\psi_1 \backslash \psi_2)$: $V_S(q) = \backslash$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
4. where $\phi = (\psi_1 \,|\, \psi_2)$: $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$

The first clause states that descriptions consisting of a single saturated category symbol are only satisfied from a point if that point is labelled with the particular symbol. In other words, this kind of statement is only satisfied from end points. For example, where $\mathcal{M}$ is the category model represented in Figure 2.4: $\mathcal{M}, q_5 \models \mathsf{NP}$, $\mathcal{M}, q_5 \not\models \mathsf{S}$ and $\mathcal{M}, q_2 \not\models \mathsf{NP}$.[2]

The second and third clauses state the satisfaction conditions for descriptions of the forms ϕ/ψ and $\phi\backslash\psi$. These kinds of expression are only satisfied from those non-end points which are labelled with the correct slash, whose result point satisfies ϕ and whose argument point satisfies ψ. For example, where $\mathcal{M}$ is again the category model represented in Figure 2.4:

(2.6) $\mathcal{M}, q_2 \models \mathsf{S}\backslash\mathsf{NP}$

$\mathcal{M}, q_2 \not\models \mathsf{S}\backslash\mathsf{S}$ because $\mathcal{M}, Arg(q_2) \not\models \mathsf{S}$

[2]Note that $\not\models$ denotes the negation of the satisfaction relation $\models$.

$$\begin{aligned}
\mathcal{M},q_2 &\not\models \mathsf{NP}\backslash\mathsf{NP} \text{ because } \mathcal{M},Res(q_2) \not\models \mathsf{NP}\\
\mathcal{M},q_2 &\not\models \mathsf{S}/\mathsf{NP} \text{ because } V_S(q_2) \neq /\\
\mathcal{M},q_1 &\models (\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP})
\end{aligned}$$

The fourth clause states the satisfaction conditions for descriptions of the form $\phi|\psi$. This kind of description is satisfied from all non-end points whose result and argument points satisfy ϕ and ψ respectively, regardless of the slash on the point. For example:

$$\begin{aligned}
\mathcal{M},q_3 &\models \mathsf{S}|\mathsf{NP}\\
\mathcal{M},q_1 &\models (\mathsf{S}|\mathsf{NP})|(\mathsf{S}|\mathsf{NP})
\end{aligned} \tag{2.7}$$

Finally, note that the following descriptions are all satisfied from the root point q_1 of the category model represented in Figure 2.4:

$$\begin{aligned}
\mathcal{M},q_1 &\models (\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP})\\
\mathcal{M},q_1 &\models (\mathsf{S}\backslash\mathsf{NP})|(\mathsf{S}\backslash\mathsf{NP})\\
\mathcal{M},q_1 &\models (\mathsf{S}|\mathsf{NP})/(\mathsf{S}|\mathsf{NP})
\end{aligned} \tag{2.8}$$

Thus, according to the definition of global satisfaction above, they are all satisfied by the category model as a whole:

$$\begin{aligned}
\mathcal{M} &\models (\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP})\\
\mathcal{M} &\models (\mathsf{S}\backslash\mathsf{NP})|(\mathsf{S}\backslash\mathsf{NP})\\
\mathcal{M} &\models (\mathsf{S}|\mathsf{NP})/(\mathsf{S}|\mathsf{NP})
\end{aligned} \tag{2.9}$$

2.1.7 Summary

In subsection 2.1.4, I defined the class of category models over some alphabet A of saturated category symbols as one particular kind of relational structure, or graph. Subsection 2.1.5 presented a formal language of category descriptions for talking about sets of category models. These two notions were linked by the satisfaction definition in subsection 2.1.6, specifying which descriptions constitute true statements about which category models.

My reasons for insisting upon and persisting with this distinction between category models and category descriptions will become clear in later chapters of this thesis, where I attempt to integrate ideas from unification-based grammar formalisms into a

theory of the CCG lexicon. Since unification-based formalisms are themselves inherently model-theoretic (or 'constraint-based'), having a model-theoretic conception of (at least parts of) CCG is a necessary prerequisite for such a convergence.

There are a number of advantages of taking a model-theoretic approach to human language grammar (Pullum and Scholz, 2001). One principal motivation is that it allows architects of grammar formalisms to make use of the full range of weaponry available to the contemporary computational logician. One such example is the familiar notion of truth relative to a variable assignment function — recall that the standard Tarskian semantics of first order logic states that sentence ϕ is satisfied in model $\mathcal{M}$, i.e. $\mathcal{M} \models \phi$, if and only if *for all functions g from variables to individuals*, $\mathcal{M}, g \models \phi$. This technique comes in useful when it comes to providing a semantics for the category variables and \$ variables in CCG category notation mentioned in subsection 2.1.1.

Let's start with category variables. Incorporating these into the CCG formalism involves adding a third kind of category description to those defined in section 2.1.5, i.e. for all $i \in \mathbb{N}$, X_i is a category description over alphabet A.[3] Satisfaction of category descriptions then needs to be defined relative to an assignment function: $\langle Q, Res, Arg, V_S, V_A \rangle \models \phi$ iff. for some function g from $\mathbb{N}$ to Q, we can determine that $\langle Q, Res, Arg, V_S, V_A \rangle, g, q \models \phi$, where q is the root point. Finally we just need to add a clause to the local satisfaction definition from subsection 2.1.6 for the new class of description: $\mathcal{M}, g, q \models X_i$ iff. $g(i) = q$.

A more elaborate application of the same technique provides a semantics for the use of \$ variables over sequences of arguments, without having to assume an underlying uncurried representation involving lists of arguments-slash pairs. Again we add a new kind of category description to the definition in subsection 2.1.5, i.e. for all category descriptions ϕ over alphabet A and all $i \in \mathbb{N}$, $(\phi/\$_i)$, $(\phi\backslash\$_i)$ and $(\phi\,|\,\$_i)$ are also category descriptions over A. Satisfaction is again made relative to an assignment function, this time from $\mathbb{N}$ to *sets of points*, rather than to individual points. Finally, the following recursive local satisfaction clause captures the intuitive behaviour of a /\$ variable in schematising over an infinite set of categories: $\mathcal{M}, g, q \models (\phi/\$_i)$ iff. either: (a) $\mathcal{M}, g, q \models \phi$; or (b) $V_s(q) = /$, $Arg(q) \in g(i)$, and $\mathcal{M}, g, Res(q) \models (\phi/\$_i)$. A more precise formalisation of this is left to future work — in particular, certain structural conditions must be placed on the assignment function to get the desired effect.

Another advantage of taking a model-theoretic perspective on CCG categories, dis-

[3] $\mathbb{N}$ denotes the set of natural numbers $\{0, 1, 2, 3, \ldots\}$.

tinguishing the category notation from its underlying category models, is that it allows us to experiment with a range of variations on the baseline language defined in section 2.1.5, whilst holding constant the underlying models. Thus, for example, the set-based category notation of Hoffman (1995) can be interpreted on the same structures as the baseline CCG notation, by means of a satisfaction definition according to which category description $\mathsf{S}\{/\mathsf{NP}_{\mathsf{obj}}, \backslash\mathsf{NP}_{\mathsf{sbj}}\}$ is satisfied by all the category models which satisfy either $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$ or $(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}})\backslash\mathsf{NP}_{\mathsf{sbj}}$. This method will be pursued rigorously in this thesis, where I hold constant the definition of underlying category models from subsection 2.1.4 and gradually develop a more flexible description language for use in descriptive, explanatory and computational grammars of human languages.

2.2 Combinatory operations

In section 2.1, I defined the category notation of CCG, taking particular care to distinguish between underlying category models and category descriptions, the two notions being related by the concept of satisfaction. For the remainder of this thesis, I am going to assume that, in CCG, a lexical item pairs a string with a *category description*[4], the latter related to a particular set of underlying *category models* by means of satisfaction. This assumption, standard in explicitly model-theoretic grammar formalisms such as HPSG, will allow us to take advantage of the underspecification inherent in our choice of description language so as to formulate non-redundant, explanatory lexicons for human languages.

In CCG, lexical items are built up into complex constructions by means of a range of universal combinatory operations. It is these which constitute the subject of this section. An important question which must be considered at the outset is the following: Since we are distinguishing between two notions of a category label, should combinatory operations be defined as applying to the underlying category models or to the category descriptions?

Recall the method to be pursued in this thesis — holding the category models constant while experimenting with progressively more expressive kinds of category description. From this perspective it is obvious that the combinatory operations of CCG need to be defined with respect to the underlying category models, since otherwise they would have to be redefined each time the description language was modified.

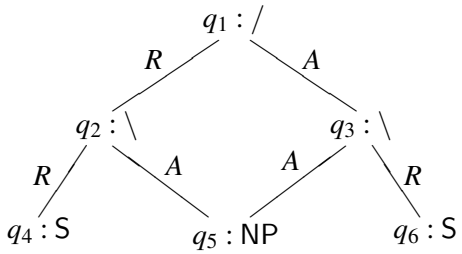
[4]Later a *set* of category descriptions.

This is the approach that will be taken in this section, where I introduce the combinatory operations made available by the CCG formalism. Two caveats should be noted however. Firstly, in defining combinatory operations over category models, I'm in no way suggesting that the human beings directly manipulate category models when parsing an utterance, or indeed that a computational implementation of the CCG formalism should do so. On the contrary, as the relation between the formulas of the description language and the underlying category models becomes more opaque, I will provide an inference system for performing derivations with category descriptions themselves, similar to the natural deduction proof system for classical logic. Secondly, a properly model-theoretic formulation of the entire CCG formalism, rather than just that of CCG categories as proposed here, will need to define CCG combinatory operations as geometric constraints on the mathematical structures representing derivations. This kind of approach is sketched out briefly in section 2.5.

2.2.1 Results and arguments of category models

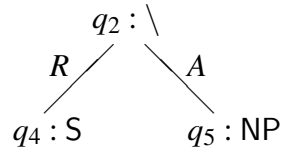
I start with a couple of preliminary definitions. Consider again the category model represented in Figure 2.4 repeated here:

(2.10)



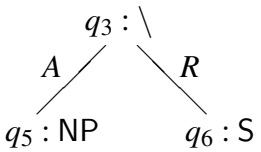
This is an unsaturated category model, so it has both a 'result' and an 'argument'. The result is the category model rooted at the point which can be reached by following the R arc from the root i.e. the category model rooted at q_2:

(2.11)



Similarly, the argument of the category model represented in Figure 2.4 is the category model rooted at the point which can be reached by following the A arc from the root i.e. the category model rooted at q_3:

(2.12)



Now armed with these concepts, we can precede to the combinatory operations themselves.

2.2.2 Forward application

Recall the following baseline definition of the category labels of the CCG formalism from section 2.1.1, assuming some alphabet A of saturated category symbols:

1. every element of A is a category label over A
2. if α and β are both category labels over A, then so are (α/β) and $(\alpha\backslash\beta)$

Recall also that a category label of the form α/β denotes the class of expressions which, when combined with a following expression of category β, yield an expression of category α. This implies that the categories of a language are related in the following way:

(2.13) $\{s_1 s_2 \mid s_1 \in \alpha/\beta,\ s_2 \in \beta\} \subseteq \alpha$

In other words, if $s_1 \in \alpha/\beta$ and $s_2 \in \beta$ then $s_1 s_2 \in \alpha$. This relationship is enshrined in the most basic combinatory operation of CCGknown as *forward application*, or $>$ for short:[5]

(2.14) $X/Y \quad Y \stackrel{>}{\Rightarrow} X$

As a simple example, take the following lexical assignments:

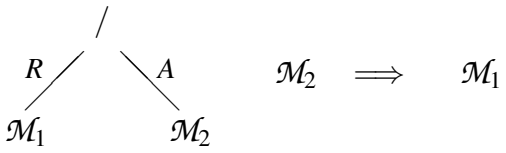
(2.15) $\mathsf{John} \vdash \mathsf{NP}$
$\mathsf{Mary} \vdash \mathsf{NP}$
$\mathsf{loves} \vdash (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$

The transitive verb ‘loves’ can be combined with its direct object ‘Mary’ by means of forward application, as follows:

[5] When presenting combinatory operations I will follow standard practice and use X, Y, Z as variables over categories, rather than α, β, γ.

(2.16)
$$\dfrac{\dfrac{\text{loves}}{(\text{S}\backslash\text{NP})/\text{NP}} \quad \dfrac{\text{Mary}}{\text{NP}}}{\text{S}\backslash\text{NP}}\,{>}$$

Recall that in this thesis, I will be assuming that CCG combinatory operations work on category models, as defined in section 2.1.4. Thus, the following diagram represents forward application graphically:

(2.17)


Formally, where:

1. $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ is a category model over alphabet A such that, where q is the root of $\mathcal{M}$, $V_S(q) = /$
2. $\mathcal{M}'$ is a category model over A
3. $\mathcal{M}'$ is identical (or at least isomorphic) to the argument of $\mathcal{M}$

Then forward applying $\mathcal{M}$ and $\mathcal{M}'$ yields the result of $\mathcal{M}$.

All of the combinatory operations in CCG, including those discussed in the following sections, can be defined formally in this way, as operations on category models. In general however, I will omit such definitions, unless the formalisation is interesting for some reason.

2.2.3 Backward application

A category label of the form $\alpha\backslash\beta$ denotes the class of expressions which, when combined with a *preceding* expression of category β, yield an expression of category α. This implies that the categories of a language are related in the following way:

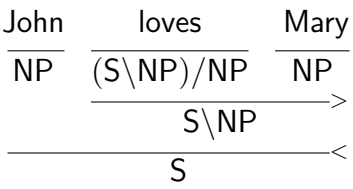
(2.18) $\{s_2 s_1 \mid s_1 \in \alpha\backslash\beta,\ s_2 \in \beta\} \subseteq \alpha$

In other words, if $s_1 \in \alpha\backslash\beta$ and $s_2 \in \beta$ then $s_2 s_1 \in \alpha$. This relationship yields another basic combinatory operation in CCG, known as *backward application*, or $<$ for short:

(2.19) $Y \quad X\backslash Y \ \stackrel{<}{\Rightarrow}\ X$

The derivation in (2.16) can then be continued as follows:

(2.20)



The forward and backward application operations, as defined in sections 2.2.2 and 2.2.3, are used in CCG for deriving bounded dependency constructions in human languages. Thus they are crucially involved in the analysis of phenomena such as reflexivation, agreement, case, local scrambling, basic word order, and so on. In particular, forward application is crucially implicated in the analysis of head-initial languages such as English, Irish and Tagalog, while backward application finds its role in the grammars of head-final languages such as Japanese, Lakhota and Tamil.

The categorial grammar formalism which includes only the two combinatory operations of forward and backward application is often termed 'AB' categorial grammar (after Ajdukiewicz and Bar-Hillel), or alternatively 'application-only' categorial grammar.

2.2.4 Forward composition

The combinatory operations of forward and backward application, as defined in sections 2.2.2 and 2.2.3, are common to all variations on the AB categorial grammar formalism of Ajdukiewicz (1935) and Bar-Hillel (1953). However, as soon as categorial grammarians turned their attention to the unbounded constructions in human languages, for example relativisation and topicalisation, it became clear that the combinatory potential of AB categorial grammar is insufficient.

The first proposal to increase the combinatory potential of categorial grammars was made by Lambek (1958), who proposed generalising the meaning of the unsaturated category notation as follows:

(2.21) $\alpha/\beta = \{s \mid \forall s' \in \beta,\ ss' \in \alpha\}$

$\alpha\backslash\beta = \{s \mid \forall s' \in \beta,\ s's \in \alpha\}$

This interpretation forms the basis of an inference system known as the 'associative Lambek calculus'. A 'sequent' is defined as a structure of the form $\alpha_1 \ldots \alpha_{n\geq 1} \vdash \beta$, where α_i and β are category labels. Sequent $\alpha_1 \ldots \alpha_n \vdash \beta$ is 'valid' just in case for every interpretation function from alphabet A of saturated category symbols to strings of terminals, and all $s_i \in \alpha_i$, $s_1 \ldots s_n \in \beta$.

CCG adds a number of theorems (i.e. valid sequents) of the associative Lambek calculus to the basic AB combinatory machinery in order to provide more elegant analyses of unbounded dependency and coordination constructions in natural languages. One example is the following combinatory operation, known as *forward composition*, or $>\mathbf{B}_1$ for short:

(2.22) $X/Y \quad Y/Z \stackrel{>\mathbf{B}_1}{\Rightarrow} X/Z$

Consider the following lexical category assignments:

(2.23) $\mathsf{will} \vdash (\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP})$

$\mathsf{marry} \vdash (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$

The auxiliary and the transitive infinitive may be combined by means of forward composition, as follows:

(2.24)
$$\dfrac{\dfrac{\mathsf{will}}{(\mathsf{S}\backslash\mathsf{NP})/(\mathsf{S}\backslash\mathsf{NP})} \quad \dfrac{\mathsf{marry}}{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}}}{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}}\,{>}\mathbf{B}1$$

This operation is required in order to derive the following kind of coordination construction, where a lexical transitive verb is conjoined with an auxiliary plus lexical infinitive cluster:[6]

(2.25) John $[\text{loves}]_{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}}$ and $[\text{will marry}]_{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}}$ Mary

The following related example shows that further generalisation of forward composition is necessary:

(2.26) John $[\text{showed}]_{((\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP})/\mathsf{NP}}$ and $[\text{will give}]_{((\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP})/\mathsf{NP}}$ Mary a ring

Assuming that ditransitive infinitives are assigned to category $((\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP})/\mathsf{NP}$, the relevant operation is the following, which we can call 'second order' forward composition, or $>\mathbf{B}_2$ for short:

(2.27) $X/Y \quad (Y/Z_1)/Z_2 \stackrel{>\mathbf{B}_2}{\Rightarrow} (X/Z_1)/Z_2$

This version of forward composition is used to combine an auxiliary and a ditransitive infinitive:

[6]Note that CCG follows standard practice in assuming that coordination can apply to any two adjacent expressions belonging to the same category.

(2.28)
$$\dfrac{\dfrac{\text{will}}{(\text{S}\backslash\text{NP})/(\text{S}\backslash\text{NP})} \quad \dfrac{\text{give}}{((\text{S}\backslash\text{NP})/\text{NP})/\text{NP}}}{((\text{S}\backslash\text{NP})/\text{NP})/\text{NP}}\,{>}\mathbf{B}2$$

In fact, the CCG formalism makes available an infinite number of forward composition operations, all of which are theorems of the associative Lambek calculus. These generalised forward composition operations can be defined using the following recursive definition:

(2.29) $X/Y \;\; Y/Z \overset{>\mathbf{B}_1}{\Rightarrow} X/Z$

if $X/Y \;\; Y' \overset{>\mathbf{B}_n}{\Rightarrow} X'$ then $X/Y \;\; Y'/Z \overset{>B_{n+1}}{\Rightarrow} X'/Z$

This recursive definition is equivalent to the following schematisation:

(2.30) $X/Y \;\; Y/Z_1 \ldots/Z_{n\geq 1} \overset{>\mathbf{B}_n}{\Rightarrow} X/Z_1 \ldots/Z_n$

Although the CCG formalism makes available an infinite number of forward composition operations, it is not the case that all of these can be used in a given CCG. In fact, forward composition must necessarily be *lexically bounded* in a particular CCG. What this means in practice is that the range of composition operations a CCG can make use of is determined by the largest category in its lexicon. If the largest lexical category has n arguments, then the CCG can avail itself of operations $\mathbf{B}_1, \ldots, \mathbf{B}_n$, but not $\mathbf{B}_{m>n}$. This detail will prove to be important in section 3.1, when I discuss generative capacity results for the CCG formalism.

2.2.5 Backward composition

Just as there is a backward counterpart to the forward application operation, so CCG provides a range of *backward composition* operations, or $<\mathbf{B}_n$, as well:

(2.31) $Y\backslash Z \;\; X\backslash Y \overset{<\mathbf{B}_1}{\Rightarrow} X\backslash Z$

$(Y\backslash Z_1)\backslash Z_2 \;\; X\backslash Y \overset{<\mathbf{B}_2}{\Rightarrow} (X\backslash Z_1)\backslash Z_2$

$((Y\backslash Z_1)\backslash Z_2)\backslash Z_3 \;\; X\backslash Y \overset{<\mathbf{B}_3}{\Rightarrow} ((X\backslash Z_1)\backslash Z_2)\backslash Z_3$

$\ldots$

Again all of these are theorems of the associative Lambek calculus. In addition, the backward composition operations can be recursively schematised similar to the definition of $>\mathbf{B}_n$ in (2.29). Backward composition in the CCG formalism is lexically bounded, just like forward composition.

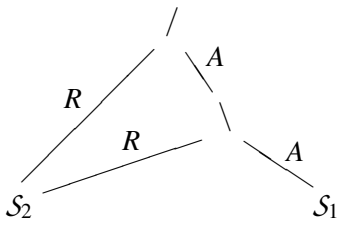
2.2.6 Forward raising

Another theorem (more accurately 'class' of theorems) of the associative Lambek calculus which has been incorporated into the CCG formalism is the following, which is known as *forward (type) raising*, or $>\mathbf{T}$ for short:

(2.32) $X \overset{>\mathbf{T}}{\Rightarrow} Y/(Y\backslash X)$

This is to be read as follows: if s is a string of category X, then for any other category Y, s also belongs to category $Y/(Y\backslash X)$. The category model formed as a result of forward type raising category model $\mathcal{S}_1$ over category model $\mathcal{S}_2$ is represented graphically as follows:

(2.33)



Note that defining forward raising in terms of category models makes the token identity of the two result categories explicit, by means of reentrancy.

2.2.7 Backward raising

The corresponding 'backward raising' operation is defined as follows:

(2.34) $X \overset{<\mathbf{T}}{\Rightarrow} Y\backslash(Y/X)$

Again, this operation represents a related class of theorems of the associative Lambek calculus.

2.2.8 Raising and composition in the grammar of English

In sections 2.2.2 and 2.2.3, I introduced the forward and backward application combinatory operations, common to all variants of the basic AB categorial grammar formalism. Sections 2.2.4 through 2.2.7 discussed the associative Lambek calculus and then defined those valid sequents which have been incorporated into CCG, specifically forward and backward composition, and forward and backward raising.

The interaction between type raising and the forward and backward composition rules is crucial to the most celebrated CCG analyses of unbounded dependency and coordination constructions. For example, assume the following mini-lexicon:

(2.35) $\mathsf{John, Mary, Bill} \vdash \mathsf{NP}$
$\mathsf{loves, admires} \vdash \mathsf{(S\backslash NP)/NP}$

A transitive verb with the right-branching category $\mathsf{(S\backslash NP)/NP}$ can combine directly with its subject using forward type raising and forward composition, as follows:

(2.36)
$$\dfrac{\dfrac{\dfrac{\mathsf{John}}{\mathsf{NP}}\ \dfrac{\mathsf{loves}}{\mathsf{(S\backslash NP)/NP}}}{\mathsf{S/(S\backslash NP)}}\,{>}\mathbf{T}}{\mathsf{S/NP}}\,{>}\mathbf{B}1$$

Thus we can derive a right-node raising construction such as that in (2.37), without needing to add an additional, left-branching lexical category $\mathsf{(S/NP)\backslash NP}$ for transitive verbs:

(2.37) [John loves]$_{\mathsf{S/NP}}$ and [Bill admires]$_{\mathsf{S/NP}}$ Mary

We can derive unbounded right-node raising constructions as well, assuming the following additional lexical categories:

(2.38) $\mathsf{thinks} \vdash \mathsf{(S\backslash NP)/S'}$
$\mathsf{that} \vdash \mathsf{S'/S}$

Using type raising and composition, derivations like the following are possible:

(2.39)
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathsf{Bill}}{\mathsf{NP}}\,{>}\mathbf{T}}{\mathsf{S/(S\backslash NP)}}\ \dfrac{\mathsf{thinks}}{\mathsf{(S\backslash NP)/S'}}}{\mathsf{S/S'}}\,{>}\mathbf{B}1\ \dfrac{\mathsf{that}}{\mathsf{S'/S}}}{\mathsf{S/S}}\,{>}\mathbf{B}1}{\mathsf{S/(S\backslash NP)}}\,{>}\mathbf{B}1}{\mathsf{S/NP}}\,{>}\mathbf{B}1$$
$$\dfrac{\dfrac{\mathsf{Tom}}{\mathsf{NP}}\,{>}\mathbf{T}}{\mathsf{S/(S\backslash NP)}}\ \dfrac{\mathsf{admires}}{\mathsf{(S\backslash NP)/NP}}$$

Thus we can derive unbounded right-node raising examples like:

(2.40) [John loves]$_{\mathsf{S/NP}}$ and [Bill thinks that Tom admires]$_{\mathsf{S/NP}}$ Mary

Further examples of the way in which type raising and forward and backward composition interact so as to provide elegant, intuitive analyses of relativisation constructions like (2.41) and argument cluster coordinations like (2.42) can be found in Steedman (1996) and Steedman (2000).

(2.41) woman $[\text{whom}]_{(\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP})}[$ Bill thinks that Tom admires$]_{\mathsf{S}/\mathsf{NP}}$

(2.42) John gave [Bill a book] and [Mary a flower]

Another benefit of including the raising and composition combinatory operations in the formalism is that it allows for a straightforward model of incremental parsing. The derivational flexibility these operations provide means that, for example, the sentences of a language defined by a *right-branching* lexicon have possible incremental, left-branching derivations, as in example (2.39) above. See chapter 9 of Steedman (2000) for a detailed discussion of this issue.

2.2.9 Other combinatory rules in CCG

In sections 2.2.2 and 2.2.3 I discussed the basic combinatory operations of forward and backward application, common to all categorial grammar formalisms, including CCGSections 2.2.4 - 2.2.7 presented four classes of theorem of the Lambek calculus which have been incorporated into the definition of the CCG formalism so as to provide a better account of unbounded dependency and coordination constructions in human language — forward and backward composition, and forward and backward raising. The resulting formalism, which can usefully be called 'CCG-lite' forms the core of the theory of natural language, in the sense that it is an approximation to the associative Lambek calculus, and hence to the logic of concatenation.

However, the existence of discontinuities in natural language constructions necessitates that the theory of linguistic competence involves something more than this simple logic of concatenation. To this end, CCG generalises forward and backward composition to include operations which are *not* theorems of the associative Lambek calculus, but rather make use of existing degrees of freedom in the formalism. For example, the CCG formalism provides an infinite range of 'forward crossed composition' operations, or $>\mathbf{B}^{\times}_{n}$:

(2.43) $X/Y \quad Y\backslash Z \quad \overset{>\mathbf{B}^{\times}_{1}}{\Rightarrow} \quad X\backslash Z$

$X/Y \quad (Y\backslash Z_1)\backslash Z_2 \quad \overset{>\mathbf{B}^{\times}_{2}}{\Rightarrow} \quad (X\backslash Z_1)\backslash Z_2$

$$X/Y \quad (Y\backslash Z_1)/Z_2 \quad \overset{>\mathbf{B}^{\times}_2}{\Rightarrow} \quad (X\backslash Z_1)/Z_2$$
$$X/Y \quad ((Y\backslash Z_1)\backslash Z_2)\backslash Z_3 \quad \overset{>\mathbf{B}^{\times}_3}{\Rightarrow} \quad ((X\backslash Z_1)\backslash Z_2)\backslash Z_3$$
$$X/Y \quad ((Y\backslash Z_1)/Z_2)\backslash Z_3 \quad \overset{>\mathbf{B}^{\times}_3}{\Rightarrow} \quad ((X\backslash Z_1)/Z_2)\backslash Z_3$$
$$X/Y \quad ((Y\backslash Z_1)\backslash Z_2)/Z_3 \quad \overset{>\mathbf{B}^{\times}_3}{\Rightarrow} \quad ((X\backslash Z_1)\backslash Z_2)/Z_3$$
$$X/Y \quad ((Y\backslash Z_1)/Z_2)/Z_3 \quad \overset{>\mathbf{B}^{\times}_3}{\Rightarrow} \quad ((X\backslash Z_1)/Z_2)/Z_3$$
$$\ldots$$

Again these operations can be schematised reursively:

(2.44) $$X/Y \quad Y\backslash Z \quad \overset{>\mathbf{B}^{\times}_1}{\Rightarrow} \quad X\backslash Z$$
$$\text{if } X/Y \quad Y' \quad \overset{>\mathbf{B}^{\times}_n}{\Rightarrow} \quad X' \text{ then } X/Y \quad Y'\backslash Z \quad \overset{>B^{\times}_{n+1}}{\Rightarrow} \quad X'\backslash Z$$
$$\text{if } X/Y \quad Y' \quad \overset{>\mathbf{B}^{\times}_n}{\Rightarrow} \quad X' \text{ then } X/Y \quad Y'/Z \quad \overset{>B^{\times}_{n+1}}{\Rightarrow} \quad X'/Z$$

The forward crossed composition operations are used in the CCG analysis of cross-serial dependencies in languages like Dutch and Swiss German. Subsection 3.1.2 discusses this in detail. There is an equivalent set of *backward* crossed composition operations, which are also non-theorems of the Lambek calculus. Since I do not use these operations in this thesis, I omit discussion of them for reasons of space. It should be noted that the forward and backward crossed composition operations are *lexically bounded* in a particular CCG, just like the forward and backward composition operations defined in subsections 2.2.4 and 2.2.4. Finally, a brief terminological point is in order — the forward and backward composition operations which are theorems of the Lambek calculus are often called 'harmonic'or 'order-preserving' composition, whereas forward and backward crossed composition are termed 'disharmonic' or 'non-order-preserving'.

The last set of combinatory operations included within the standard definition of the CCG formalism are the 'substitution' operations, for example 'forward substitution':

(2.45) $$(X/Y)/Z \quad Y/Z \quad \overset{>\mathbf{S}}{\Rightarrow} \quad X/Z$$

Again, since substitutions operations are not crucial to the discussion in this thesis, I will say no more about them, except to direct the interested reader towards Steedman (2000).

2.2.10 Summary

This section has introduced the full range of combinatory operations which make up the standard, 'baseline' CCG formalism. Subsections 2.2.2 and 2.2.3 discussed forward and backward application, which are common to all properly 'categorial' frameworks based on the work of Ajdukiewicz (1935) and Bar-Hillel (1953). Subsections 2.2.4 and 2.2.5 presented the 'harmonic' forward and backward composition operations, and subsections 2.2.6 and 2.2.7 discussed forward and backward raising. The categorial formalism which uses just these six operations ('CCG-lite') is an approximation to the associative Lambek calculus. Subsection 2.2.9 briefly discussed the CCG combinatory operations which are not theorems of the Lambek calculus but instead rely on other degrees of freedom which the theory provides, for example forward crossed composition.

At the start of this chapter I stated that the CCG formalism is not simply a *generalisation* of AB categorial grammar but rather a *principled* generalisation. What is meant by this is that the various combinatory operations in CCG are not theoretical primitives, but are in some sense theorems of four underlying 'principles'. The four principles of CCG are as follows (Steedman, 2000):

adjacency combinatory rules may only apply to finitely many phonological realized and string-adjacent entities

combinatory type transparency all syntactic combinatory rules are type-transparent versions of one of a small number of simple semantic operations over functions

consistency all syntactic combinatory rules must be consistent with the directionality of the principle function

inheritance if the category that results from the application of a combinatory rule is a function category, then the slash defining directionality for a given argument in that category will be the same as the one(s) defining directionality for the corresponding argument(s) in the input function(s)

The Principle of Adjacency imposes a general ban on things like empty categories or the non-concatenative 'wrapping' rules of Bach (1979). The Principle of Combinatory Type Transparency rules out any combinatory operations that are not versions of application, composition, raising or substitution. The set of possible operations which

satisfy adjacency and combinatory type transparency are further restricted by the other two principles. For example, the Principle of Consistency rules out the following version of composition, since since the principle expression X/Y requires a *following* secondary expression:

(2.46) $\quad Y/Z \quad X/Y \;\Rightarrow\; X/Z$

Finally, the following version of composition satisfies the Principle of Consistency but violates the Principle of Inheritance, since argument Z has different slashes in the input and output:

(2.47) $\quad X/Y \quad Y\backslash Z \;\Rightarrow\; X/Z$

Baldridge (2002) suggests that these four principles may not in themselves be primitives of the theory, but rather corollaries of a Lambek-style calculus according to which all and only the CCG combinatory operations are provable sequents. He goes on to define a relevant set of axioms, based on the slash modalities discussed below. However, whilst the Lambek calculus has been proved to be complete with respect to the logic of concatenation, there is no equivalent categorial type logic, in the sense of Moortgat (1997), for which CCG is a complete system.

I conclude this section with a brief mention of the role played by the slash modalities in allowing lexical items to control the combinatory operations in which they may participate. Recall from subsection 2.1.1 that Baldridge (2002) proposes at least the following three modalities: $\star$, $\diamond$, $\times$. These modalities are attached to slashes in unsaturated category descriptions, for example $(\mathsf{NP}\backslash^{\star}\mathsf{NP})/_{\diamond}(\mathsf{S}/_{\times}\mathsf{NP})$.

The function of the modalities is to allow lexical control over what combinatory operations a word or morpheme may enter into. Thus, an unsaturated category label carrying any of the three modalities may function as the principle expression in an application operation. Thus, the forward and backward application operations from (2.14) and (2.19) need to be redefined as follows:[7]

(2.48) $\quad X/_m Y \quad Y \;\overset{>}{\Rightarrow}\; X$ where $m \in \{\star, \diamond, \times\}$

(2.49) $\quad Y \quad X\backslash^m Y \;\overset{<}{\Rightarrow}\; X$ where $m \in \{\star, \diamond, \times\}$

[7]Note that this definition differs minimally form that in Baldridge (2002), since I am not assuming that the modalities are related by a 'casting' hierarchy.

Secondly, the forward and backward harmonic composition operations require that both input expressions carry the associative $\diamond$ modality. For example, the 'second order' forward harmonic composition from (2.27) needs to be defined as follows:

(2.50) $X/_{\diamond}Y \quad (Y/_{\diamond}Z_1)/_{\diamond}Z_2 \quad \overset{>\mathbf{B}_2}{\Rightarrow} \quad (X/_{\diamond}Z_1)/_{\diamond}Z_2$

Finally, the forward and backward crossed composition operations require that both input expressions carry the permutative $\times$ modality. For example, the two versions of second order forward crossed composition from (2.43) are now as follows:

(2.51) $X/_{\times}Y \quad (Y\backslash^{\times}Z_1)\backslash^{\times}Z_2 \quad \overset{>\mathbf{B}_2^{\times}}{\Rightarrow} \quad (X\backslash^{\times}Z_1)\backslash^{\times}Z_2$

$X/_{\times}Y \quad (Y\backslash^{\times}Z_1)/_{\times}Z_2 \quad \overset{>\mathbf{B}_2^{\times}}{\Rightarrow} \quad (X\backslash^{\times}Z_1)/_{\times}Z_2$

Note finally, that Baldridge (2002) also allows for underspecified modalities, which can enter into both harmonic and disharmonic operations.

2.3 The CCG formalism

In section 2.1, I defined the notions of CCG category models and category descriptions, along with a satisfaction relation. In section 2.2, I discussed the CCG combinatory operations on categories models — forward and backward application, forward and backward harmonic composition, forward and backward raising, forward and backward cross composition, and the related range of substitution operations. In this section, I provide a brief formal definition of the baseline CCG formalism itself.

As noted in chapter 1, a grammar formalism must provide at least a specification of a set of grammars over arbitrary alphabets of terminal symbols, and a relation specifying for every grammar over Σ and every string in Σ^*, whether or not the string belongs to the language generated by the grammar. This section provides the relevant definitions for baseline CCG. **Subsection 2.3.1** defines the class of CCG grammars, **subsection 2.3.2** defines the 'combinatory projection' of a CCG lexicon, and **subsection 2.3.3** specifies the 'generates' relation itself.

2.3.1 CCGs

A CCG over alphabet Σ is an ordered triple $\langle A, S, L\rangle$ where:

1. A is an alphabet of saturated category symbols

2. S is a distinguished element of A

3. L is a lexicon i.e. a finite mapping from Σ to category descriptions over A

Take for example the following CCG:

(2.52) $G_1 = \langle \{\mathsf{S}, \mathsf{NP}\}, \mathsf{S}, \{\langle \mathsf{John}, \mathsf{NP}\rangle, \langle \mathsf{Mary}, \mathsf{NP}\rangle, \langle \mathsf{loves}, (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\rangle\}\rangle$

In this CCG just two saturated category symbols are listed, 'S' and 'NP', the former being the distinguished one. The lexicon consists of three lexical items:

(2.53) $\mathsf{John} \vdash \mathsf{NP}$

$\mathsf{Mary} \vdash \mathsf{NP}$

$\mathsf{loves} \vdash (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$

One desirable result of providing a formal definition of the set of grammars specified by the CCG formalism is that it is clear exactly *what is* and *what is not* a (baseline) CCG.

2.3.2 Combinatory projection of a CCG lexicon

Given CCG $\langle A, S, L\rangle$ over alphabet Σ, the 'combinatory projection' of lexicon L is defined as the smallest set L', such that:

1. for all $\langle s, \phi\rangle \in L$, and all category models $\mathcal{M}$ over A such that $\mathcal{M} \models \phi$, $\langle s, \mathcal{M}\rangle \in L'$

2. for all $\langle s, \mathcal{M}_1\rangle \in L'$ and all category models $\mathcal{M}_2$ over A, where $\mathcal{M}_3$ is the result of either forward or backward raising $\mathcal{M}_1$ over $\mathcal{M}_2$, then $\langle s, \mathcal{M}_3\rangle \in L'$

3. for all $\langle s_1, \mathcal{M}_1\rangle, \langle s_2, \mathcal{M}_2\rangle \in L'$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to $\mathcal{M}_3$ by means of one of the binary CCG combinatory operations, then $\langle s_1 s_2, \mathcal{M}_3\rangle \in L'$

Recall from subsection 2.3.1 that a CCG lexicon is a set of pairs $\langle s, \phi\rangle$, where s is a terminal *symbol* and ϕ is a category *description*. The combinatory projection of a lexicon is, on the other hand, a set of pairs $\langle s, \mathcal{M}\rangle$, where s is a *string* of terminal symbols and $\mathcal{M}$ is a category *model* i.e. the combinatory projection maps strings to categories as structures rather than to category descriptions.

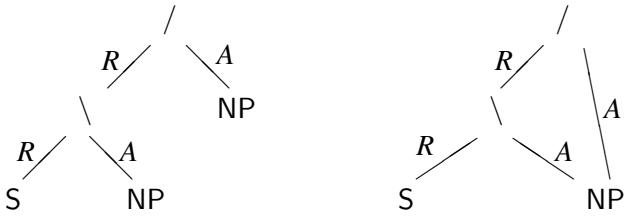
The first clause of this definition specifies the 'base' of the combinatory projection. According to this, if form s is assigned to category description ϕ in the lexicon,

and category model $\mathcal{M}$ satisfies ϕ according to the definition in subsection 2.1.6, then $\langle s, \mathcal{M} \rangle$ is in the combinatory projection of the lexicon. For example, the CCG in (2.52) contains the following lexical entry:

(2.54) $\mathsf{loves} \vdash (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$

In addition, the category description $(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$ is satisfied by the following two distinct, non-isomorphic category models:

(2.55)



Thus, the combinatory projection of the CCG lexicon in (2.53) will map the form 'loves' to at least these two category models.

The second clause in the above definition of the combinatory projection of a CCG lexicon encodes the forward and backward raising operations defined in subsections 2.2.6 and 2.2.7. The third clause is the recursive one — if both $\langle s_1, \mathcal{M}_1 \rangle$ and $\langle s_2, \mathcal{M}_2 \rangle$ are in the combinatory projection of a lexicon, and category models $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to category models $\mathcal{M}_3$ by means of one of the CCG combinatory operations defined in subsections 2.2.2, 2.2.3, 2.2.4, 2.2.5 and 2.2.9, then the pair $\langle s_1 s_2, \mathcal{M}_3 \rangle$ is also in the combinatory projection of the lexicon.

2.3.3 CCG generation

CCG $\langle A, S, L \rangle$ over alphabet Σ generates string $s \in \Sigma^*$ if and only if the ordered pair $\langle s, S \rangle$[8] is an element of the combinatory projection of lexicon L.

In other words, a CCG generates a string just in case the combinatory projection of its lexicon maps the string onto a saturated category model whose single point is labelled with the grammar's distinguished symbol. This definition encapsulates another important aspect of the CCG formalism — it is clear exactly *which strings are* and *which are not* generated by a given CCG.

[8] More precisely, the ordered pair $\langle s, \langle \{q\}, \emptyset, \emptyset, \emptyset, \{\langle q, S \rangle\} \rangle \rangle$ for some point q

2.4 Semantics

In section 2.3, I presented a formal definition of baseline CCG, thus fulfilling the two most basic desiderata of a grammar formalism: (a) a specification of a class of grammars over an arbitrary alphabet of terminal symbols (subsection 2.3.1); and (b) a 'generates' relation between the grammars over Σ and strings over Σ (subsection 2.3.3). The third necessary component of a grammar formalism is a compositional theory of the semantics of the languages generated by its grammars. The link between CCGs and compositional semantics is the subject of this section.

As has been noted many times in the categorial grammar literature, the ties between categorial grammars (e.g. CCGs) and compositional semantics is extremely close. In a nutshell, *every CCG of language L is also a compositional theory of the non-lexical semantics of L.* In this section I examine and exemplify this statement from two distinct perspectives. In subsection 2.4.1 I discuss how to turn a CCG into a theory of the *denotational* semantics of the language generated by the grammar. Subsection 2.4.2, on the other hand, presents a brief discussion of *representational* semantics in the CCG literature, focusing on the 'hybrid logic dependency semantics' (HLDS) of Baldridge and Kruijff (2002).

2.4.1 Denotational semantics in CCG

To turn a CCG $\langle A, S, L\rangle$ over alphabet Σ into a compositional theory of the denotational semantics of the language generated by the grammar, we first need to define a semantic domain function δ mapping each saturated category symbol in A onto its semantic domain. So, for example, it is customary (but of course not obligatory) to define $\delta(\mathsf{S})$ as the set of truth values, and $\delta(\mathsf{NP})$ as the set of individuals. The semantic domain of unsaturated categories is then determined according to the following recursive definition:

$$\delta(X/Y) = \delta(X\backslash Y) = \delta(X)^{\delta(Y)} \tag{2.56}$$

Recall that $\delta(X)^{\delta(Y)}$ denotes the set of all possible functions from $\delta(Y)$ to $\delta(X)$. So for example, assuming the standard semantic domains for S and NP, the semantic domain of the unsaturated category $\mathsf{S}\backslash\mathsf{NP}$ is the set of functions from individuals to truth values (i.e. sets of individuals), and the semantic domain of $(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$ is the set of all functions from individuals to sets of individuals (i.e. relations on individuals).

The second step in turning a CCG $\langle A, S, L\rangle$ into a compositional semantic theory involves pairing lexical categories in L with appropriate denotata. In other words, for every lexeme $\langle s, X\rangle \in L$, we select an element of $\delta(X)$ as the denotatum of the lexeme. For example, imagine the lexicon contains a lexeme $\langle \mathsf{loves}, (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\rangle$. Recall that the customary semantic domain of the unsaturated category $(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$ is the set of relations on individuals. Thus, the denotation of the lexeme $\langle \mathsf{loves}, (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\rangle$ will be one particular relation on individuals — the set of all and only the pairs $\langle x, y\rangle$ of individuals such that x loves y.

Once we have assigned appropriate denotata to all lexemes, then the denotata of complex expressions can be determined automatically, as a function of the denotata of the component parts and the derivational history. Recall that the simplest combinatory operations in the CCG formalism are forward and backward application, as defined in subsections 2.2.2 and 2.2.3 respectively. The semantic operation associated with these rules is simply the application of a function to its argument i.e. where f and x are the denotata of the relevant expressions:

(2.57) $\quad X/Y : f \quad Y : x \quad \overset{>}{\Rightarrow} \quad X : fx$

$\qquad\quad Y : x \quad X\backslash Y : f \quad \overset{<}{\Rightarrow} \quad X : fx$

Take for example the following derivation:

(2.58)
$$\dfrac{\dfrac{\mathsf{John}}{\mathsf{NP} : j} \quad \dfrac{\mathsf{loves}}{(\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} : \{\langle j, \{\langle j,t\rangle, \langle m,t\rangle\}\rangle, \langle m, \{\langle j,t\rangle, \langle m,f\rangle\}\rangle\}} \quad \dfrac{\mathsf{Mary}}{\mathsf{NP} : m}\,{>}}{\mathsf{S}\backslash\mathsf{NP} : \{\langle j,t\rangle, \langle m,f\rangle\}}\,{<}$$
$$\mathsf{S} : t$$

In the first step, lexemes $\langle \mathsf{loves}, (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\rangle$ and $\langle \mathsf{Mary}, \mathsf{NP}\rangle$ combine by means of forward application. The denotatum of the former is:

(2.59) $\quad \{\langle j, \{\langle j,t\rangle, \langle m,t\rangle\}\rangle, \langle m, \{\langle j,t\rangle, \langle m,f\rangle\}\rangle\}$

Note that j and m are individuals and t and f are truth values. This function is equivalent to the following relation on $\{j,m\}$: $\{\langle j,j\rangle, \langle m,j\rangle, \langle j,m\rangle\}$ i.e. j loves both j and m, but m just loves j. In tandem with the forward application operation which combines $\langle \mathsf{loves}, (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}\rangle$ and $\langle \mathsf{Mary}, \mathsf{NP}\rangle$ yielding $\langle \mathsf{loves\ Mary}, \mathsf{S}\backslash\mathsf{NP}\rangle$ the denotatum of the former is applied to that of the latter:

(2.60) $\quad \{\langle j, \{\langle j,t\rangle, \langle m,t\rangle\}\rangle, \langle m, \{\langle j,t\rangle, \langle m,f\rangle\}\rangle\}\ m = \{\langle j,t\rangle, \langle m,f\rangle\}$

Thus the denotatum of the complex expression *loves Mary* is equivalent to the following set of individuals: $\{j\}$. In the same way, the denotatum of the entire sentence *John loves Mary* is worked out as follows:

(2.61) $\{\langle j,t\rangle, \langle m,f\rangle\}\, j = t$

In other words, the sentence is true.

All of the combinatory operations licensed by the underlying principles of the CCG formalism have a corresponding semantics, based on a small number of simple operations on functions. Take for example the 'first order' composition operations $>\mathbf{B}_1$, $<\mathbf{B}_1$, $>\mathbf{B}_1^{\times}$ and $<\mathbf{B}_1^{\times}$, as defined in subsections 2.2.4, 2.2.5 and 2.2.9. The corresponding semantic operation in this case is the *composition* of the two functions serving as the denotata of the input expressions, for example:

(2.62) $X/Y : f \quad Y/Z : g \quad \overset{>\mathbf{B}_1}{\Rightarrow} \quad X/Z : \mathbf{B}_1 f g$

Note that $\mathbf{B}_1$ is the symbol used in combinatory logic to denote functional composition i.e.

(2.63) $\mathbf{B}_1 f g = g \circ f = \lambda x.f(gx)$

By extension, the semantic operation corresponding to nth order composition operations is $\mathbf{B}_n$. The forward and backward raising operations, defined in subsections 2.2.6 and 2.2.7 correspond to the operation on functions known as 'type raising', or $\mathbf{T}$:

(2.64) $\mathbf{T}x = \lambda f.fx$

And finally, the substitution operations discussed briefly in subsection 2.2.9 are linked to the operation on functions known as 'functional substitution', or $\mathbf{S}$:

(2.65) $\mathbf{S}fg = \lambda x.(fx)(gx)$

See Steedman (2000) for further details. On important point to note involves the 'derivational flexibility' of CCGs — for any given sentence generated by a CCG there is usually a number of distinct derivations of that sentence. For example, assume the following CCG lexicon again:

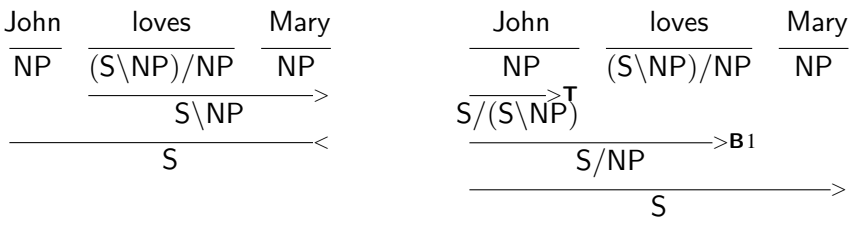
(2.66) $\mathsf{John} \vdash \mathsf{NP}$

$\mathsf{Mary} \vdash \mathsf{NP}$

$\mathsf{loves} \vdash (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP}$

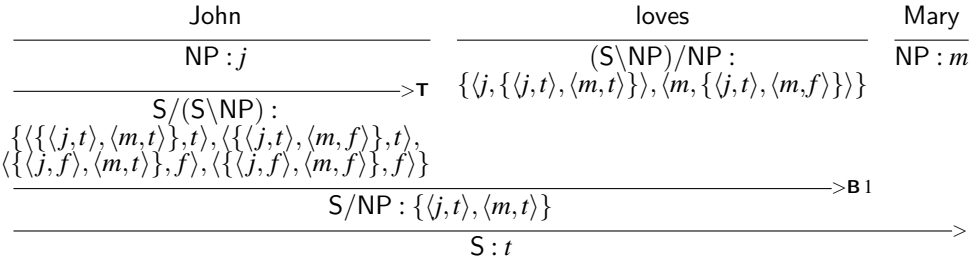
Based on this lexicon, the sentence *John loves Mary* can be derived in the following two ways:

(2.67)



Although these two derivations are formally distinct, they form a semantic equivalence class, as is clear by comparing the semantics of the derivation in (2.58) with the following:

(2.68)



The end result of the two derivations is the same i.e. the sentence is true. In addition, the denotatum of the subject is j, and when type raised this becomes the set of all sets of which j is a member.

2.4.2 Representational semantics in CCG

In the previous section I discussed the denotational approach to semantics in CCG, where every lexical expression is paired with a function representing its denotation, and the denotation of phrasal expressions is determined by a small number of operations on functions. An alternative approach which has been applied in recent years is a representational approach based on the 'hybrid logic dependency structures' of Baldridge and Kruijff (2002).

In fact, the presentation of the CCG formalism in both Steedman (1996) and Steedman (2000) also argues for a representational approach to semantics, in order to capture phenomena such as binding and control as constraints on a level of representation called 'predicate-argument structure'. The representation language assumed there is essentially the simply typed λ calculus, where the potentially powerful operation of λ abstraction is strictly local in effect. An example derivation is given here, complete with β reductions:

(2.69)

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{John}}{\text{NP}:j'}\quad\dfrac{\text{loves}}{(\text{S}\backslash\text{NP})/\text{NP}:\lambda y\lambda x.love'yx}\quad\dfrac{\text{Mary}}{\text{NP}:m'}}{\text{S}/(\text{S}\backslash\text{NP}):\lambda P.Pj'}\,{>}\mathbf{T}}{\text{S}/\text{NP}:\lambda z.(\lambda P.Pj')((\lambda y\lambda x.love'yx)z)}\,{>}\mathbf{B}1}{\text{S}/\text{NP}:\lambda z.love'zj'}\,\beta}{\text{S}:love'm'j'}\,{>}$$

Even this strictly local use of λ abstraction can be avoided by using unification to bind arguments in semantic representations, as follows:

(2.70)

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{John}}{\text{NP}:j'}\quad\dfrac{\text{loves}}{(\text{S}:love'yx\backslash\text{NP}:x)/\text{NP}:y}\quad\dfrac{\text{Mary}}{\text{NP}:m'}}{\text{S}:p/(\text{S}:p\backslash\text{NP}:j')}\,{>}\mathbf{T}}{\text{S}:love'yj'/\text{NP}:y}\,{>}\mathbf{B}1}{\text{S}:love'm'j'}\,{>}$$

Kruijff (2001) proposes using a representation language based on the hybrid logic of Blackburn (2000) to encode the notion of semantic dependency roles in categorial grammars. The resulting representation scheme, names 'hybrid logic dependency semantics' (HLDS) is incorporated compositionally into the CCG system in Baldridge and Kruijff (2002). White and Baldridge (2003) presents a 'flattened' HLDS system, designed for use in a CCG-based natural language generation system, similar to the Minimal Recursion Semantics of Copestake et al. (2005). It is this flattened version of the HLDS representation language which will be presented here.

White and Baldridge (2003) call the flattened hybrid logic terms 'elementary predications'. Given alphabets P of propositional symbols and M of modal symbols, and assuming a countably infinite set N of so-called 'nominals' (which for present purposes can be thought of as equivalent to first order variables):

1. for all $x \in N$ and $p \in P$, $@_x p$ is an elementary predication over P and M
2. for all $x, y \in N$ and $m \in M$, $@_x\langle m\rangle y$ is an elementary predication over P and M
3. for all $x, y \in N$, $@_x y$ is an elementary predication over P and M

Hybrid logic corresponds to a fragment of first order logic, and hence that every hybrid logic term can be translated into an equivalent first order formula. Thus the elementary predications can be redefined as first order formulas i.e. given alphabets P of unary predicate symbols and M of binary relation symbols, and assuming a countably infinite set V of first order variables:

1. for all $x \in V$ and $p \in P$, $p(x)$ is an elementary predication over P and M
2. for all $x, y \in V$ and $m \in M$, $m(x,y)$ is an elementary predication over P and M
3. for all $x, y \in V$, $x = y$ is an elementary predication over P and M

Since many more people are familiar with the syntax and semantics of first order logic than hybrid logic, and since the usefulness of any representation is directly proportional to the number of people who can understand it, I shall persist with the first order representations of elementary predications henceforth.

To add HLDS representations to a CCG, we first of all have to add first order variables representing DRT-style referential indices to each of the saturated category symbols which make up some category in the lexicon. For example, take the following CCG lexicon once more:

(2.71)
$$\begin{array}{lll} \mathsf{John} & \vdash & \mathsf{NP} \\ \mathsf{Mary} & \vdash & \mathsf{NP} \\ \mathsf{loves} & \vdash & (\mathsf{S}\backslash\mathsf{NP})/\mathsf{NP} \end{array}$$

We can add referential indices as follows:

(2.72)
$$\begin{array}{lll} \mathsf{John} & \vdash & \mathsf{NP}_x \\ \mathsf{Mary} & \vdash & \mathsf{NP}_x \\ \mathsf{loves} & \vdash & (\mathsf{S}_x\backslash\mathsf{NP}_y)/\mathsf{NP}_z \end{array}$$

Secondly, every lexical category needs to be supplemented with a set of elementary predications, which will generally make reference to the indices in the category label itself. For example, the above lexicon can be further elaborated as follows:

(2.73)
$$\begin{array}{lll} \mathsf{John} & \vdash & \mathsf{NP}_x : \mathit{john}'(x) \\ \mathsf{Mary} & \vdash & \mathsf{NP}_x : \mathit{mary}'(x) \\ \mathsf{loves} & \vdash & (\mathsf{S}_x\backslash\mathsf{NP}_y)/\mathsf{NP}_z : \mathit{loving}'(x),\ \mathit{sbj}'(x,y),\ \mathit{obj}'(x,z) \end{array}$$

Note that the predication $\mathit{sbj}'(x,y)$ is to be read as stating that the entity denoted by y is a 'subject' dependent of that denoted by x.

Finally, semantic composition of the HLDS structures in CCG is simply a matter of set union. In other words, the combinatory operations of forward application and

first order forward harmonic composition can be redefined as follows:

(2.74) $X/Y : A \quad Y : B \stackrel{>}{\Rightarrow} X : A \cup B$

$X/Y : A \quad Y/Z : B \stackrel{>\mathbf{B}_1}{\Rightarrow} X/Z : A \cup B$

Note that the variables X, Y and Z in these rule schemata are to be understood as including the referential indices in category labels. An example derivation involving HLDS representations is given here:

(2.75)

$$\dfrac{\dfrac{\dfrac{\text{John}}{\mathsf{NP}_x : \mathit{john}'(x)} \quad \dfrac{\text{loves}}{(\mathsf{S}_y\backslash\mathsf{NP}_x)/\mathsf{NP}_z : \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z)} \quad \dfrac{\text{Mary}}{\mathsf{NP}_z : \mathit{mary}'(z)}}{\mathsf{S}_y/(\mathsf{S}_y\backslash\mathsf{NP}_x) : \mathit{john}'(x)}\,{>}\mathbf{T}}{\mathsf{S}_y/\mathsf{NP}_z : \mathit{john}'(x),\ \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z)}\,{>}\mathbf{B}1}{\mathsf{S} : \mathit{john}'(x),\ \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z),\ \mathit{mary}'(z)}\,{>}$$

Before concluding this section on the application of a compositional semantics in CCG, it should be noted that the main focus of this thesis is on syntax. As such I will generally ignore the semantics of the constructions I deal with, although in all cases either a denotational or representational compositional semantics could be defined straightforwardly. Whenever focus on the semantics of a construction is important, however, I will freely avail myself of either of the above discussed representation language i.e. either the simply typed λ calculus or the HLDS structures. In particular, the latter will come in useful when it comes to dealing with processing aspects of CCG i.e. in trying to formalise processing preferences like 'get as much semantic information as early as possible', which appear to provide intuitive explanations for certain language universals involving case, agreement and word order.

The HLDS representations also have two further advantages. Firsty, they allow a formalisation of the notion of 'focus' in natural language sentences. For example, a simple sentence such as *John loves Mary* may have three different focus readings depending on which word is most heavily stressed i.e. the focus of *John loves MARY* is different from that of *JOHN loves Mary*, and that of *John LOVES Mary* is different again. Now it is clear that the focus of a sentence cannot be identified with one of its discourse referents, but rather corresponds to some minimal unit of information i.e. an elementary predication. So for example, the focus of sentence *John loves MARY* is the elementary predication $\mathit{mary}'(z)$ and the background is the set of elementary predications $\mathit{john}'(x)$, $\mathit{loving}'(y)$, $\mathit{sbj}'(y,x)$, $\mathit{obj}'(y,z)$. Similarly, the focus of *John LOVES Mary* is $\mathit{loving}'(y)$ and the background is $\mathit{john}'(x)$, $\mathit{sbj}'(y,x)$, $\mathit{obj}'(y,z)$, $\mathit{mary}'(z)$.

A second advantage of the HLDS representations over λ terms involves the need for a natural language lexicon to be as non-redundant as possible. In later chapters I will introduce lexical inheritance hierarchies into the CCG formalism, and the kind of underspecified lexical categories this machinery entails gives rise to the need for similarly underspecified, flattened semantic representations like the HLDS structures.

Note finally that a number of proposals for different semantic representations in CCG and related categorial formalisms have been passed over here: (a) Indexed Languages (Zeevat et al., 1987); (b) Minimal Recursion Semantics (Villavicencio, 2002), (Beavers, 2004); and (c) Discourse Representation Structures (Traat and Bos, 2004).

2.5 CCG derivations as models

Sections 2.3 and 2.4 provided a formal introduction to the CCG formalism, including a discussion of compositional semantics. We have seen that this baseline CCG formalism has two crucial properties: (a) it is simple, intuitive and precisely defined; and (b) it is semantically transparent, and compatible with both denotational and representational approaches to semantics.

Previously it was claimed that the above definition of the CCG formalism was 'partially model-theoretic', and it is time now to spell out in a little more detail what I meant by that. Recall that the definition of the formalism came in two distinct parts. Section 2.1 introduced the category labels used in CCG; and section 2.2 presented the combinatory operations. The definition of the former was presented in explicitly model-theoretic terms i.e. involving a crucial distiction between structures (category models) and descriptions (category descriptions), these two notions related by means of logical satisfaction. The definition of the combinatory operations, on the other hand, was presented in a more traditional, 'proof-theoretic' or algebraic manner.

This formulation of CCG is sufficient for the purposes of this thesis, where the focus is on enriching the theory of the lexicon by progressively more expressive category description languages interpreted on a fixed underlying class of category models. However, the question as to what a purely model-theoretic formulation of CCG would look like, and more importantly, what such a formulation could teach us about the nature of grammatical composition in CCG, is of considerable interest. In this section, I provide a sketch of how such an investigation might proceed. Note that this section can safely be skipped, since the definitions presented here are not presupposed by the

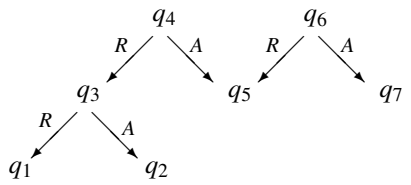


Figure 2.5: A generalised category frame

remainder of this thesis.

2.5.1 Generalised category frames

Recall that subsection 2.1.2 defined a class of relational structures known as 'category frames': ordered triples $\langle Q, Res, Arg \rangle$, where: (a) Q is a set of points; (b) Res is the result relation on Q; and (c) Arg is the argument relation on Q. A number of conditions were placed on Res and Arg: (a) both are functional mappings with identical domains; (b) the transitive closure of $Res \cup Arg$ is a strict partial ordering relation on Q; and (c) the transitive closure of $Res \cup Arg$ has a first/least element i.e. *category frames are rooted or 'point-generated'*.

For the purposes of formulating a purely model-theoretic definition of CCG, it is useful to work with a slightly generalised notion of category frame, where only condition (c) is relaxed. In other words, generalised category frames are exactly like category frames as defined in subsection 2.1.2, except that they need not be point-generated. An example of a generalised category frame is given in Figure 2.5.

Although generalised category frames need not be rooted at a single point, there is one important condition that they must still obey. Every point must be reachable from every other point by means of some sequence of arcs, where arcs can be travelled in either direction. In other words, for all generalised category frames $\langle Q, Res, Arg \rangle$, the reflexive, transitive closure of $Res \cup Arg \cup Res^{-1} \cup Arg^{-1}$ is $Q \times Q$.

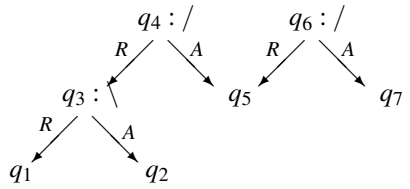


Figure 2.6: A generalised category structure

2.5.2 Generalised category structures

Subsection 2.1.3 defined a class of relational structures called 'category structures' as ordered 4-tuples of the form $\langle Q, Res, Arg, V_S \rangle$, where: (a) $\langle Q, Res, Arg \rangle$ is a category frame; and (b) V_S is a function which assigns either $/$ or $\backslash$ to every non-end point in the frame.

Generalised category structures are defined similarly, except that the underlying frame is a *generalised* category frame, as defined in subsection 2.5.1. An example of a generalised category structure is given in Figure 2.6.

2.5.3 Generalised category models

Subsection 2.1.4 defined another class of relational structures called 'category models' relative to an alphabet A of saturated category labels. A category model over alphabet A was defined as an ordered 5-tuple $\langle Q, Res, Arg, V_S, V_A \rangle$, where: (a) $\langle Q, Res, Arg, V_S \rangle$ is a category structure; and (b) V_A is a function which assigns exactly one element of A to every end point in the structure.

Generalised category models are defined in the same way, again with just a single exception — the underlying structure is a *generalised* category structure as defined in subsection 2.5.2. An example of a generalised category model is given in Figure 2.7.

2.5.4 Category descriptions and satisfaction

Although the class of category models has been generalised slightly, so as to include instances which are not point-generated, we retain the same conception of category

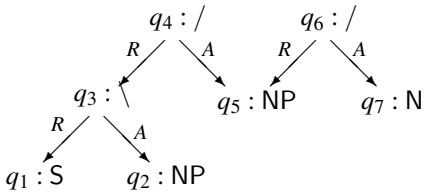


Figure 2.7: A generalised category model over alphabet $\{\mathsf{S}, \mathsf{NP}, \mathsf{N}\}$

descriptions as defined in subsection 2.1.5. We also assume the same version of the 'local' satisfaction relation from subsection 2.1.6:

Generalised category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over alphabet A locally satisfies category description ϕ over A, from point $q \in Q$, written $\mathcal{M}, q \models \phi$, if and only if:

1. where $\phi \in A$: $V_A(q) = \phi$
2. where $\phi = (\psi_1 / \psi_2)$: $V_S(q) = /$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
3. where $\phi = (\psi_1 \backslash \psi_2)$: $V_S(q) = \backslash$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
4. where $\phi = (\psi_1 \,|\, \psi_2)$: $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$

2.5.5 Binary ordered trees

Now that we have a generalised notion of category models, we can start to think about what kind of mathematical model might underly the notion of a CCG derivation. The obvious choice is to formalise CCG derivations as a 'tree' of category models, rather like the 'derivation trees' of Tree Adjoining Grammar. However, other options are possible, and the following subsections present one such alternative idea.

I start with the notion of a binary ordered tree, which underlies CCG derivations. A binary ordered tree over alphabet Σ of terminal symbols is an ordered 4-tuple $\langle N, \lhd_1, \lhd_2, V_\Sigma \rangle$, where:

1. N is a finite non-empty set of nodes
2. $\lhd_1$ is the 'first child' relation on N i.e. if $\langle n_1, n_2 \rangle \in \lhd_1$ then node n_2 is the first child of node n_1

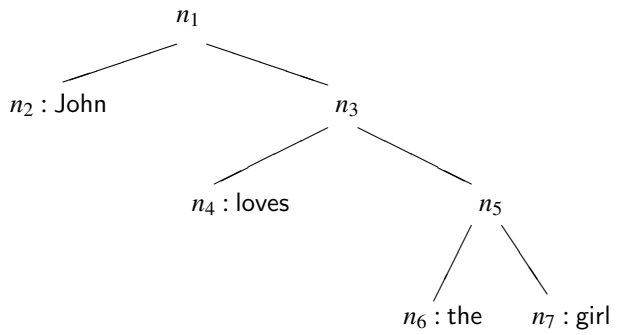


Figure 2.8: A binary ordered tree

3. $\lhd_2$ is the 'second child' relation on N i.e. if $\langle n_1, n_2 \rangle \in \lhd_2$ then node n_2 is the second child of node n_1

4. V_Σ is a function from the leaf nodes in N to Σ

Note also that: (a) both $\lhd_1$ and $\lhd_2$ are functional mappings, since every node has at most one first child and at most one second child; (b) the domains of $\lhd_1$ and $\lhd_2$ are identical i.e. every node with a first child has a second child, and vice versa i.e. this is the set of non-leaf nodes; (c) the union of $\lhd_1$ and $\lhd_2$ constitutes an injective mapping from N to N i.e. every node is a child of at most one parent; and (d) the reflexive transitive closure of $\lhd_1 \cup \lhd_2$ is a reflexive, antisymmetric relation on N with a first element i.e. there is exactly one 'root' node which dominates all other nodes. A simple example of a binary ordered tree is given in Figure 2.8.

2.5.6 Derivation models

Now we are ready for the definition of derivation models themselves, considered as essentially *bipartite* structures consisting of a binary ordered tree and a generalised category model, where the nodes of the tree are linked to the points of the model by means of a function.

A derivation model over alphabets A of saturated category symbols and Σ of terminal symbols is an ordered 10-tuple $\langle N, \lhd_1, \lhd_2, V_\Sigma, Q, Res, Arg, V_S, V_A \delta \rangle$, where:

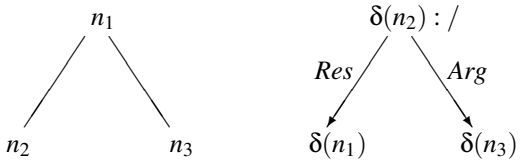
1. $\langle N, \lhd_1, \lhd_2, V_\Sigma \rangle$ is a binary ordered tree over Σ

2. $\langle Q, Res, Arg, V_S, V_A \rangle$ is a generalised category model over A

3. δ is a total function from N to Q

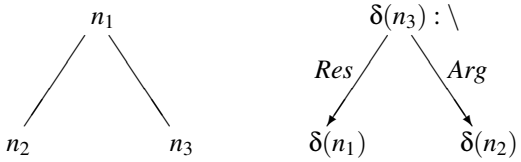
There are a number of conditions on the node-to-point function δ. Firstly, δ must be an injective function i.e. every point is mapped to by at most one node, although some points may not be the value of any node at all. Secondly, every local tree must be 'licensed' by one of the CCG combinatory operations. For example, in a forward application structure, the first child is mapped to a point q labelled $/$, the parent is mapped to the result of q, and the second child is mapped to the argument of q, schematised as follows:

(2.76)



In a backward application structure in contrast, the *second* child is mapped to a point q labelled $\backslash$, the parent is mapped to the result of q, and the *first* child is mapped to the argument of q:

(2.77)



Other combinatory operations can be defined similarly. An example of a derivation model is given in Figure 2.9. Note that every local subtree in this model is licensed by either forward or backward application.

2.5.7 Lexical descriptions

The previous subsection defined the class of mathematical models I will use the represent CCG derivations. Now all we need is a formal language to talk about them. This language is basically the set of possible lexical entries in a CCG grammar.

The set of lexical descriptions over alphabets Σ of terminal symbols and A of saturated category symbols is the set of all expressions of the form $s \vdash \phi$, where $s \in \Sigma$ and ϕ is a category description over A. In other words, lexical descriptions pair forms with category descriptions.

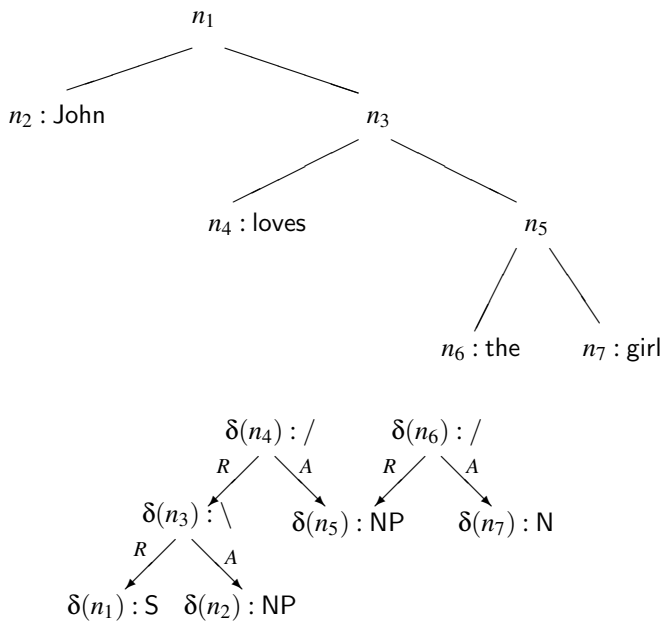


Figure 2.9: A derivation model

2.5.8 Derivation satisfaction

Subsection 2.5.6 specified a class of models encoding the essence of a CCG derivation, and subsection 2.5.7 defined a formal language for talking about them. These two notions are again related by a satisfaction definition:

Derivation model $\mathcal{M} = \langle N, \lhd_1, \lhd_2, V_\Sigma, Q, Res, Arg, V_S, V_A, \delta \rangle$ over alphabets Σ and A satisfies set Φ of lexical descriptions, written $\mathcal{M} \models \Phi$, if and only if for all leaf nodes $n \in N$ there is some $\phi \in \Phi$ such that $\mathcal{M}, n \models \phi$.

In other words, a derivation model $\mathcal{M}$ satisfies a set of lexical descriptions (i.e. lexical entries) Φ just in case some description in Φ is 'locally' satisfied at every leaf node in the binary ordered tree underlying $\mathcal{M}$. This local satisfaction relation is defined as follows:

Derivation model $\mathcal{M} = \langle N, \lhd_1, \lhd_2, V_\Sigma, Q, Res, Arg, V_S, V_A, \delta \rangle$ over alphabets Σ and A satisfies lexical description $(\sigma \vdash \phi)$ over Σ and A at node $n \in N$, written $\mathcal{M}, n \models (\sigma \vdash \phi)$, if and only if $V_\Sigma(n) = \sigma$ and $\langle Q, Res, Arg, V_S, V_A \rangle, \delta(n) \models \phi$.

Thus, a leaf node in the tree of a derivation model is licensed by a particular lexical description $\sigma \vdash \phi$ just in case the node itself carries the terminal symbol σ and is linked to a point in the category model which satisfies category description ϕ. For example, where $\mathcal{M}_1$ is the derivation model in Figure 2.9:

$$\begin{aligned} (2.78) \quad \mathcal{M}_1, n_2 &\models (\mathsf{John} \vdash \mathsf{NP}) \\ \mathcal{M}_1, n_6 &\models (\mathsf{the} \vdash \mathsf{NP/N}) \\ \mathcal{M}_1, n_4 &\models (\mathsf{loves} \vdash (\mathsf{S\backslash NP})/\mathsf{NP}) \end{aligned}$$

And a global satisfaction result is as follows:

$$(2.79) \quad \mathcal{M}_1 \models \{\mathsf{John} \vdash \mathsf{NP},\ \mathsf{girl} \vdash \mathsf{N},\ \mathsf{loves} \vdash (\mathsf{S\backslash NP})/\mathsf{NP},\ \mathsf{the} \vdash \mathsf{NP/N}\}$$

In conclusion, a CCG can be viewed as a finite set of lexical descriptions which constrain derivation models. The string yield of the derivation models which satisfy set Φ of lexical descriptions is the language generated by Φ.

2.5.9 Summary

This section has sketched out how a purely model-theoretic formulation of the CCG formalism might look, based on a conception of the abstract mathematical structures underlying CCG derivations as consisting of a binary ordered tree, a generalised category model, and a function mapping the nodes of the former to the points of the latter.

Pullum and Scholz (2001) presents a number of arguments for favouring such model-theoretic, or 'constraint-based', theories of natural language. For example, since model-theoretic grammars are actually finite sets of constraints, the degree of ungrammaticality of a non-sentence can be simply modelled as the number of constraints violated by its underlying derivation model. Another interesting aspect of model-theoretic formulations of established grammar formalisms involves the ability to study them as logics (Rogers, 1996), (Rogers, 1998). From the perspective of CCG, this would involve an investigation into the graph-theoretic, geometric underpinnings of combinatory operations such as raising, composition and substitution, and hence of the underlying principles of CCG such as adjacency, inheritance and consistency.

Finally, this kind of model-theoretic perspective on categorial derivation provides the best approach to formalising and comparing a number of 'constraint-based' categorial formalisms. Take for example the categorial formalism of Flynn (1985), which

attempts to provide an explanatory account of word order universals. Flynn assumes that unsaturated categories are unordered and that left-to-right ordering of constituents is specified by a language-particular, cross-categorial 'word order convention'. For example, Flynn's word order convention for English is as follows, where a 'major' category is one whose result is S:[9]

> If some phrase ϕ is of category $X|Y$ and ϕ *contains* a major category, then $X|Y$ is to be regarded as $X\backslash Y$. Otherwise, $X|Y$ is to be regarded as X/Y.

Thus, a Flynn-style grammar contains an unordered lexicon and a word order constraint. It should be clear from the use of the word 'contains' in the above that such word order conventions are constraints on *derivations* rather than on *categories*. Thus parsing with a Flynn-grammar is a much more complex affair than parsing with a CCG. Finally, note that Foster (1990) takes Flynn's approach a step further, formalising the notion of a word order convention as a set of 'linear precedence statements'. Again such statements are best conceptualised as constraints on categorial derivations.

2.6 Summary

This chapter introduced the CCG formalism of Steedman (2000) and Baldridge (2002), a principled, linguistically motivated, computationally tractable and psycholinguistically realistic generalisation of the categorial grammars of Ajdukiewicz (1935) and Bar-Hillel (1953).

Section 2.1 introduced the notion of a 'category', making a formal distinction between two different conceptions of what a category is — category models, and category descriptions. These two conceptions were related by a 'satisfaction' definition, telling us which category models are licensed by which descriptions. This aspect of the formalism, influenced by work in such obviously constraint-based frameworks as HPSG, is what gives it its *partially model-theoretic* flavour, which I come back to below.

Section 2.2 introduced the combinatory operations of CCG as operations on the underlying category models licensed by a category description, rather than on the category descriptions themselves. I discussed how the combinatory operations are not in themselves primitives of the CCG formalism, but rather are to be seen as theorems of the underlying 'principles' of CCG

[9] I have converted the notation to that used in this thesis, and the emphasis is my own.

Section 2.3, together with the previous two sections, contains the main contribution this chapter makes to the overall argumentation of this thesis, providing as it does a formal definition of the 'baseline' CCG formalism on which the work in this thesis will build. This definition involved a specification of a set of grammars over arbitrary alphabets of terminal symbols, along with a 'generates' relation, telling us which grammars generate which strings of terminals.

Section 2.4 involved a brief discussion of semantics in CCG in particular how every CCG of a language is also a compositional theory of its semantics. Both denotational and representational approaches, for example the hybrid logic dependency representations of Baldridge and Kruijff (2002) and White and Baldridge (2003), were discussed.

Section 2.5 contained an appendix to the main discussion — a brief sketch of how the entire CCG formalism, and not just that part of it dealing with category labels, can be defined model-theoretically, in terms of constraints on derivation structures.

As mentioned above, this chapter defined the CCG formalism as a partially model-theoretic, partially proof-theoretic system. The system of category labels was defined model-theoretically, with a crucial distinction between category models and category descriptions. On the other hand, the combinatory system was defined proof-theoretically, as a kind of algebra on category models.

Let me briefly reiterate my reasons for this, somewhat curious, division of labour. The main focus of this thesis is on developing a theory of CCG lexicons which is more conducive to providing descriptive and explanatory grammars of natural languages. Previous work on eliminating redundancy in other lexicalised grammar formalisms, such as LFG and HPSG, has shown that constraint-based techniques are the way to go. Since such techniques are essentially model-theoretic in nature (e.g. feature structures vs. feature structure descriptions), I have chosen to take a similar perspective on the system of category labels in CCG. The benefits of this decision will be obvious in later chapters. In contrast, I have nothing substantial to say about the CCG combinatory system, which I simply accept as is, along with the standard conception of grammatical composition as a sequence of operations on form-category pairs.

In defining a 'baseline' CCG formalism in this chapter, I have tried to isolate what I see as the *intersection* of the formal assumptions made by all recent work involving CCG, for example the work reported in Steedman (1996), Steedman (2000), Baldridge (2002), Baldridge and Kruijff (2003), Bozsahin (2002), Trechsel (2000),

Hockenmaier (2003), Hoffman (1995), and Beavers (2004). The baseline formalism includes the modalised bidirectional unsaturated category notation, with no assumptions made about either the possible internal structure of saturated category symbols or ways of imposiing organisation on the lexicon. This baseline formalism is not to be seen as a straw man. Rather it will allow me to draw attention more effectively to the reasons why, as everyone agrees, the basic CCG category notation needs some kind of generalisation, even if there is as yet no general consensus about the particular form this should take.

Finally, for those schooled within the Lambek tradition of categorial grammar, the decision to interpret category notation against underlying model-theoretic structures may seem peculiar. In the CTL tradition, it is customary to interpret category notation directly as denoting sets of strings of terminals. This approach is feasible for systems like the Lambek calculus, which is sound and complete relative to the logic of concatenation. However, since there is no equivalent categorial type logic for which CCG is a sound and complete proof system, this latter approach to interpreting category notation is inappropriate here.

Chapter 3

CCG and linguistic competence

Chapter 2 introduced the CCG formalism of Steedman (2000) and Baldridge (2002), including its baseline category notation and combinatory operations. In this chapter, I turn to the evaluation of CCG as a theory of human linguistic competence.

Section 3.1 examines the claim that CCG is a *restrictive* grammar formalism, in the sense that its generative capacity with respect to alphabet Σ is a proper subset of the context-sensitive languages over Σ, identical to the generative capacities of other restrictive formalisms such as Tree Adjoining Grammar and Linear Indexed Grammar. I discuss some of the implications of this claim for the theory of natural language competence. In particular, I review a number of arguments from the CCG literature concerning the relationship between the generative capacity of CCG and predictions it makes about the permitted complexity of unbounded dependency constructions in human languages.

Section 3.2 then turns to the subject of implicational universals of human language, i.e. statements about non-random correlations between two or more independent grammatical properties. I discuss two particular implicational universals which the CCG formalism has been argued to explain, and attempt to elucidate exactly what methodological approach underlies the intuition that a grammar formalism *explains* an implicational universal. The first universal involves the observation of Ross (1970) that the potential for gapping constructions in a language is determined by basic word order. The second involves the familiar 'null subject parameter' (or '*pro*-drop' parameter) of Government and Binding Theory (Chomsky, 1981), which identifies a correlation in the family of Romance languages between the possibility that the subject may appear

optionally in a post-verbal position, and the ability of a subject to be extracted across an overt complementiser. I argue that these two implicational universals are captured by the CCG formalism in subtly different ways, and tentatively propose a generalised, intrinsic approach to linguistic explanation based on grammatical simplicity.

3.1 Unbounded dependencies in CCG

CCG is a *restrictive* grammar formalism, in the sense that the class of formal languages generated by CCGs constitutes a proper subset of the class of context-sensitive languages, as discussed in chapter 1. The significance of this fact for the study of human language competence is that CCG thus makes strong predictions about what is and what is not a natural language — only a small minority of definable languages are possible human languages. Thus CCG claims that there is a fixed upper bound on the complexity of human language. This section surveys a range of observations and results from the CCG literature which address this issue.

3.1.1 Unbounded nested dependencies

One of the most basic assumptions of generative syntax, dating back to Chomsky (1957), concerns the existence in human language competence of *unbounded nested dependencies*. An unbounded nested dependency construction can be schematised as follows:

(3.1) $n_n \ldots n_1 v_1 \ldots v_n$

Every noun n_i is a dependent of verb v_i, and every verb v_i is a dependent of verb v_{i+1}. Chomsky gives the following example of an unbounded nested dependency construction from the linguistic competence of a native speaker of English:

(3.2) The *man* who said that ... *is* arriving today.

In this example the gap represents an arbitrary embedded sentence. Note that there is a dependency between elements on either side of the embedded sentence i.e. between the common noun 'man' and the finite verb 'is'. If the number of the former is made plural, then the latter must be changed to 'are':

(3.3) The *men* who said that ... *are* arriving today.

Since the embedded sentence can be either of (3.2) or (3.3), it is clear that this construction exemplifies the kind of unbounded nested dependency construction schematised in (3.1):

(3.4) The man$_2$ who said that the men$_1$ who said so are$_1$ arriving today is$_2$ arriving today.

Another example of a nested dependency construction comes from German, where finite verbs are generally positioned at the end of subordinate clauses, following all of their dependents, including infinitival complements. The following sentences provide evidence for the existence of unbounded nested dependencies in the linguistic competence of a German speaker:[1]

(3.5) Jan sagt, daß wir dem Hans$_2$ das Haus$_1$ anstreichen$_1$ halfen$_2$.
Jan says that we the.DAT Hans the.ACC house paint helped
Jan says that we helped Hans paint the house.

(3.6) Jan sagt, daß wir die Kinder$_3$ dem Hans$_2$ das Haus$_1$ anstreichen$_1$ helfen$_2$ liessen$_3$.
Jan says that we the.ACC children the.DAT Hans the.ACC house paint help let
Jan says that we let the children help Hans paint the house.

Note that the assumption is that unbounded nested dependencies are involved in human language *competence*. The fact that such centre-embeddings sharply reduce in unacceptability is argued to be a matter of *performance*, and thus irrelevant to the generative grammar itself (Chomsky, 1965). For example, the fact that German native speakers find sentences like (3.6) hard to process, in general using an alternative construction which avoids such multiple centre-embeddings, does not count as evidence against German having unbounded nested dependencies in the competence grammar.

The assumption that human language competence crucially involves unbounded nested dependency constructions is central to Chomsky's argument that the natural languages cannot be characterised as the regular ('finite-state') languages, and thus that the language faculty cannot be fully modelled by the family of finite state automata. On the other hand, the context-free grammar (CFG) formalism does permit the formulation of grammars for languages containing such constructions. For example, the archetypal unbounded nested dependency language $a^n b^n$, all of whose sentences consist of a sequence of some finite number of tokens of symbol a followed by exactly the

[1] Verb-object dependencies are indicated by subscripts. Note that the verb *helfen* ('help') selects a dative object, while *lassen* ('let') selects an accusative one. Note also that the use of case markers with proper nouns is a feature of spoken South German, rather than the written standard.

same number of tokens of symbol b, can be generated by the following CFG:

(3.7) $\mathsf{S} \rightarrow ab$

$\mathsf{S} \rightarrow a\,\mathsf{S}\,b$

Bar-Hillel et al. (1964) prove that the basic AB categorial grammar formalism, where the only permitted combinatory operations are forward and backward application, discussed in subsections 2.2.2 and 2.2.3 respectively, is weakly equivalent to the CFG formalism. In other words, for every context-free language there is some AB categorial grammar which generates it; and every AB categorial grammar generates a context-free language. Thus, AB categorial grammars, as well as generalisations like CCG, can generate languages with unbounded nested dependency constructions. For example, the language schematised in (3.1) can be generated from the following categorial lexicon using backward application only:

(3.8) $v_1 \vdash \mathsf{S}_1\backslash\mathsf{NP}_1$

$v_{i\geq 2} \vdash (\mathsf{S}_i\backslash\mathsf{NP}_i)\backslash\mathsf{S}_{i-1}$

$n_i \vdash \mathsf{NP}_i$

For example, here a simple derivation using this lexicon:

(3.9)
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{n_3}{\mathsf{NP}_3}\quad\dfrac{n_2}{\mathsf{NP}_2}\quad\dfrac{\dfrac{n_1}{\mathsf{NP}_1}\quad\dfrac{v_1}{\mathsf{S}_1\backslash\mathsf{NP}_1}}{\mathsf{S}_1}<\quad\dfrac{v_2}{(\mathsf{S}_2\backslash\mathsf{NP}_2)\backslash\mathsf{S}_1}}{\mathsf{S}_2\backslash\mathsf{NP}_2}<}{\mathsf{S}_2}<\quad\dfrac{v_3}{(\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{S}_2}}{\mathsf{S}_3\backslash\mathsf{NP}_3}<}{\mathsf{S}_3}<$$

3.1.2 Unbounded cross-serial dependencies

Although any context-free formalism can generate the kind of unbounded nested dependency construction discussed in section 3.1.1, there are other kinds of unbounded dependency which require greater generative power. For example, a typical *unbounded cross-serial dependency* construction is represented as follows:

(3.10) $n_n \ldots n_1 v_n \ldots v_1$

Again, every noun n_i is a dependent of verb v_i, and every verb v_i is a dependent of verb v_{i+1}. Languages containing sentences which manifest this kind of dependency

cannot be generated by a CFG or an application-only categorial grammar. Shieber (1985) argues that the linguistic competence of speakers of Swiss German contains such unbounded intersecting dependency constructions, and moreover that: (a) Swiss German is *not* a context-free language;[2] and thus (b) the natural languages cannot be characterised as (a subset of) the class of context-free languages. The Swiss German translational equivalents of the German sentences in (3.5) and (3.6) are as follows:

(3.11) Jan säit, das mer em Hans_2 es huus_1 hälfed_2 aastriiche_1.

Jan says that we the.DAT Hans the.ACC house helped paint

Jan says that we helped Hans paint the house.

(3.12) Jan säit, das mer d'chind_3 em Hans_2 es huus_1 lönd_3 hälfe_2 aastriiche_1.

Jan says that we the.ACC children the.DAT Hans the.ACC house let help paint

Jan says that we let the children help Hans paint the house.

Note that, although the basic application-only AB categorial grammar formalism does not provide grammars which generate languages with unbounded cross-serial dependency constructions, it is possible to write an AB categorial grammar for a language with *bounded* cross-serial dependencies, for example where the maximal size of a verb cluster is fixed at three verbs, as in the Swiss German examples above. For example, the language consisting of the bounded instantiation of the schema in (3.10) such that $n \leq 3$ can be derived from a categorial lexicon where each $v_{i\geq 2}$ is assigned to a distinct lexical category:

(3.13) $v_2 \vdash ((\text{S}_2\backslash\text{NP}_2)\backslash\text{NP}_1)/(\text{S}_1\backslash\text{NP}_1)$

$v_3 \vdash ((((\text{S}_3\backslash\text{NP}_3)\backslash\text{NP}_2)\backslash\text{NP}_1)/(\text{S}_1\backslash\text{NP}_1))/(((\text{S}_2\backslash\text{NP}_2)\backslash\text{NP}_1)\backslash(\text{S}_1\backslash\text{NP}_1))$

Before moving on, there are two points that need to be made at this point regarding the merits of AB categorial grammar as a theory of natural language competence. Firstly, the AB formalism predicts that human language competence involves unbounded nested dependencies but not unbounded cross-serial dependencies — only bounded cross-serial dependencies are predicted by AB categorial grammar. This would involve rejecting the assumption in Shieber (1985), and would presumably be justified by drawing attention to the relative scarcity of cross-serial dependencies in hu-

[2]Since there is some regular language L such that the intersection of Swiss German and L is not context-free.

man languages compared to nested dependencies. Shieber warns against this however (p.341):

> One could argue that the phenomenon of cross-serial clause structure is bounded by, say, five embeddings or, to be more generous, one hundred. In either case, the language with bounded cross-seriality would be context-free, regardless of case-marking properties. Down this path lies tyranny. Acceptance of this argument opens the way to proofs of natural languages as regular, nay, finite. The linguist proposing this counterargument to salvage the context-freeness of natural language may have won the battle, but has certainly lost the war.

Secondly, whilst the AB formalism does allow for human languages, in the marked case, to have bounded cross-serial dependency constructions, this crucially requires that: (a) the relevant class of verbs are assigned to a plurality of lexical categories; and (b) each of these categories contains at least one argument NP which is not a semantic dependent of the verb itself. These observations have led syntacticians to look at other means of deriving cross-serial dependencies from categorial lexicons.

Let us start by considering the Lambek Calculus, discussed briefly in chapter 2. It was conjectured by Chomsky (1963), and eventually proved by Pentus (1993), that the Lambek calculus generates exactly the class of context-free languages. In other words, the generative capacity of the Lambek formalism is identical to that of basic application-only AB categorial grammar. A corollary of this result is that any generalised categorial grammar formalism which supplements the basic AB calculus with *only* combinatory rules which are theorems of the Lambek calculus remains context-free. So, for example, adding unbounded harmonic composition as defined in sections 2.2.4 and 2.2.5, and forward and backward raising as defined in sections 2.2.6 and 2.2.7 (i.e. the formalism I have called 'CCG-lite') does not increase generative capacity. What this means in terms of the discussion of unbounded dependency constructions above is that CCG-lite makes exactly the same predictions as does basic AB categorial grammar — human language competence involves unbounded nested dependencies but not unbounded cross-serial dependencies; however, *bounded* cross-serial dependencies are available as a marked alternative, by means of lexical category ambiguity.

Adding the crossed composition operations $>\mathbf{B}^n_\times$ and $<\mathbf{B}^n_\times$, discussed briefly in section 2.2.9, leads to a formalism which makes very different predictions about unbounded dependency constructions in natural language, however. Steedman (1985)

shows that the inclusion of $>\mathbf{B}^n_\times$ gives us a formalism which can derive unbounded cross-serial dependencies, of the kind argued to be involved in the linguistic competence of Swiss German and Dutch speakers. Let's assume the following categorial lexicon, which differs from that in (3.8) only in that the S argument of the category for $v_{i\geq 2}$ is forward-seeking rather than backward-seeking:

(3.14)
$$\begin{aligned} v_1 &\vdash \mathsf{S}_1\backslash\mathsf{NP}_1 \\ v_{i\geq 2} &\vdash (\mathsf{S}_i\backslash\mathsf{NP}_i)/\mathsf{S}_{i-1} \\ n_i &\vdash \mathsf{NP}_i \end{aligned}$$

Using this lexicon we can derive a cross-serial dependency construction as follows:

(3.15)
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{n_4}{\mathsf{NP}_4}\ \dfrac{n_3}{\mathsf{NP}_3}\ \dfrac{n_2}{\mathsf{NP}_2}\ \dfrac{n_1}{\mathsf{NP}_1}\ \dfrac{v_4}{(\mathsf{S}_4\backslash\mathsf{NP}_4)/\mathsf{S}_3}\ \dfrac{v_3}{(\mathsf{S}_3\backslash\mathsf{NP}_3)/\mathsf{S}_2}\ \dfrac{v_2}{(\mathsf{S}_2\backslash\mathsf{NP}_2)/\mathsf{S}_1}\ \dfrac{v_1}{\mathsf{S}_1\backslash\mathsf{NP}_1}}{(\mathsf{S}_2\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}{>}\mathbf{B}_\times 1}{((\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}{>}\mathbf{B}_\times 2}{(((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}{>}\mathbf{B}_\times 3}{((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2}{<}}{(\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3}{<}}{\mathsf{S}_4\backslash\mathsf{NP}_4}{<}}{\mathsf{S}_4}{<}$$

Note that this derivation assumes a right-branching verb cluster. An alternative derivation with a left-branching verb cluster is also possible:

(3.16)
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{n_4}{\mathsf{NP}_4}\ \dfrac{n_3}{\mathsf{NP}_3}\ \dfrac{n_2}{\mathsf{NP}_2}\ \dfrac{n_1}{\mathsf{NP}_1}\ \dfrac{v_4}{(\mathsf{S}_4\backslash\mathsf{NP}_4)/\mathsf{S}_3}\ \dfrac{v_3}{(\mathsf{S}_3\backslash\mathsf{NP}_3)/\mathsf{S}_2}}{((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)/\mathsf{S}_2}{>}\mathbf{B}_\times 2\ \dfrac{v_2}{(\mathsf{S}_2\backslash\mathsf{NP}_2)/\mathsf{S}_1}}{(((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)/\mathsf{S}_1}{>}\mathbf{B}_\times 2\ \dfrac{v_1}{\mathsf{S}_1\backslash\mathsf{NP}_1}}{(((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}{>}\mathbf{B}_\times 1}{((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2}{<}}{(\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3}{<}}{\mathsf{S}_4\backslash\mathsf{NP}_4}{<}}{\mathsf{S}_4}{<}$$

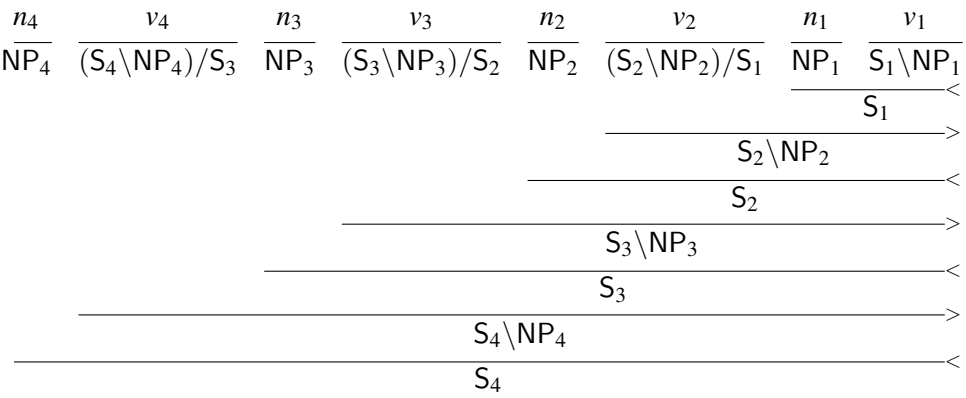
Deriving such cross-serial dependency constructions with uniformly *right*-branching verb-clusters requires the use of the *unbounded* version of $>\mathbf{B}^n_\times$. However, deriving exactly the same strings with uniformly *left*-branching verb clusters can be done just using $>\mathbf{B}^1_\times$ and $>\mathbf{B}^2_\times$. Why is this important? As was briefly mentioned in subsection

2.2.4, the generally accepted definition of CCG assumes that composition operations are *lexically bounded* in the competence grammar. In other words, the grammar may only make use of operations $\mathbf{B}^1$ to $\mathbf{B}^n$, where n is the maximum valency in the lexicon of the language. For example, it is generally assumed that for English $n = 3$.

The reason behind this assumption derives from the proof in Weir and Joshi (1988) that CCG is weakly equivalent to other restrictive grammar formalisms such as Tree Adjoining Grammar, Head Grammar and Linear Indexed Grammar, but only if composition operations are lexically bounded in this way. In other words, unbounded composition raises the generative capacity of the CCG formalism beyond the level of the mildly context-sensitive languages. However, even the weaker version of the CCG formalism, where composition operations are lexically bounded, is sufficient to derive unbounded cross-serial dependency constructions, as long as we assume that the verb clusters implicit in such constructions are left-branching rather than right-branching.

Another important point about this analysis of unbounded cross-serial dependencies is that, as noted in Steedman (1985), the lexicon in (3.14) does not generate *only* sentences involving cross-serial dependencies — non-nested, non-intersecting dependency constructions can be derived as well, by means of the basic application rules:

(3.17)



To date, there is no completely satisfactory solution to this problem, although chapter 6 of Steedman (2000) and chapter 5 of Baldridge (2002) propose using a combination of minor syntactic features and obligatory, restricted type-raising to eliminate the overgeneration. To be sure, this does not cause a problem for the grammar of Swiss German, since the alternative orderings are all grammatical. However, in Dutch the situation is complicated by the fact that with some verbs cross-serial dependencies are obligatory, whereas with others they are merely optional. Note finally that a lexical treatment of *bounded* cross-serial dependencies does not suffer from the overgeneration problem — it is not possible to derive sentence $n_3v_3n_2v_2n_1v_1$ from the categorial

lexicon in (3.13).

In conclusion, the generally accepted version of the CCG formalism, where composition operations are lexically bounded, predicts that human language competence can involve both: (a) unbounded nested dependencies; as well as (b) unbounded cross-serial dependencies.

3.1.3 Doubly unbounded scrambling

Although the CCG formalism provides sufficient combinatory power to derive unbounded cross-serial dependency constructions, it is not the case that it allows for arbitrary unbounded dependency constructions in human language competence. One example of an unbounded dependency construction which has been discussed in the CCG literature, and which CCG predicts *not* to exist in human language competence, is the *doubly unbounded scrambling* construction from Rambow (1994) and Hoffman (1995).

I start with a brief discussion of word order in Turkish, which is generally considered to be a verb-final language. Examples (3.18) and (3.19) illustrate the phenomenon of local scrambling in Turkish — the subject and object of a transitive verb can appear in either order, with no change in truth-conditional meaning:

(3.18) Ayse kitab-i oku-yor.
Ayse book-ACC read-PROG
Ayse reads the book.

(3.19) Kitab-i Ayse oku-yor.
book-ACC Ayse read-PROG
Ayse reads the book.

Turkish also allows local scrambling within embedded clauses:

(3.20) Fatma [Ayse-'nin kitab-i oku-dugunu] bili-yor.
Fatma [Ayse-GEN book-ACC read-GER] know-PROG
Fatma knows that Ayse reads the book.

(3.21) Fatma [kitab-i Ayse-'nin oku-dugunu] bili-yor.
Fatma [book-ACC Ayse-GEN read-GER] know-PROG
Fatma knows that Ayse reads the book.

And also *of* embedded clauses:

(3.22) [Ayse-'nin kitab-i oku-dugunu] Fatma bili-yor.
[Ayse-GEN book-ACC read-GER] Fatma know-PROG
Fatma knows that Ayse reads the book.

In addition, Turkish also allows *long-distance* scrambling, where an argument of an embedded verb is interspersed among the arguments of a higher verb:

(3.23) Kitab-i Fatma [Ayse-'nin oku-dugunu] bili-yor.
book-ACC Fatma [Ayse-GEN read-GER] know-PROG
Fatma knows that Ayse reads the book.

Based on similar data from German, Rambow (1994) argues that long-distance scrambling is a *doubly unbounded* construction, in the sense that: (a) there is no limit on the *distance* an argument can be moved; and (b) there is no limit on the *number* of arguments that can be moved. However, this assumption is problematic, not least because languages with doubly unbounded long-distance scrambling are known not to be mildly context-sensitive — in other words they cannot be generated by Tree Adjoining Grammars, Linear Indexed Grammars or CCGs. In the absence of any more compelling evidence of a human language which is not mildly context-sensitive, recent work, particularly Joshi et al. (2000) and Baldridge (2002), has argued that human language competence does not have doubly unbounded long-distance scrambling constructions — rather long-distance scrambling is restricted for any verb cluster consisting of *four or more* verbs.

Hoffman (1995) schematises doubly unbounded scrambling constructions as follows, where $\sigma(x_1 \ldots x_n)$ denotes an arbitrary permutation of string $x_1 \ldots x_n$:

(3.24) $\sigma(n_1 \ldots n_n)\ v_1 \ldots v_n$

As above, each noun n_i is to be understood as a dependent of verb v_i, and each verb v_i as a dependent of verb v_{i+1}. As Baldridge (2002) has shown, the CCG formalism with

lexically bounded composition is capable of deriving *some* such constructions. Take for example the lexicon in (3.8), repeated here:

(3.25)
$$\begin{aligned} v_1 &\vdash \mathsf{S}_1\backslash\mathsf{NP}_1 \\ v_{i\geq 2} &\vdash (\mathsf{S}_i\backslash\mathsf{NP}_i)\backslash\mathsf{S}_{i-1} \\ n_i &\vdash \mathsf{NP}_i \end{aligned}$$

Note that in this lexicon no lexical item has more than two arguments. The lexically bounded composition assumption thus states that the only composition operations available to this grammar are $\mathbf{B}^1$ and $\mathbf{B}^2$. In other words, for any $n \geq 3$, $\mathbf{B}^n$ *cannot* be used in a derivation from this lexicon.

From the lexicon in (3.25), we can derive certain long distance scrambling constructions. Indeed, any string of the form $\sigma(n_1n_2n_3)\ v_1v_2v_3$ can be derived. An example derivation is given here:

(3.26)
$$\dfrac{\dfrac{n_2}{\mathsf{NP}_2}\quad\dfrac{n_1}{\mathsf{NP}_1}\quad\dfrac{\dfrac{n_3}{\mathsf{NP}_3}}{\mathsf{S}_3/(\mathsf{S}_3\backslash\mathsf{NP}_3)}\,{>}\mathbf{T}\quad\dfrac{\dfrac{v_1}{\mathsf{S}_1\backslash\mathsf{NP}_1}\quad\dfrac{\dfrac{v_2}{(\mathsf{S}_2\backslash\mathsf{NP}_2)\backslash\mathsf{S}_1}\quad\dfrac{v_3}{(\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{S}_2}}{((\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{S}_1}\,{<}\mathbf{B}^2}{((\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}\,{<}\mathbf{B}^1}{(\mathsf{S}_3\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}\,{>}\mathbf{B}_\times 2}{\mathsf{S}_3\backslash\mathsf{NP}_2}\,{<}}{\mathsf{S}_3}\,{<}$$

However, since the combinatory operation $>\mathbf{B}^3_\times$ cannot be used in derivations from this lexicon, the possibilities for generating sentences of the form $\sigma(n_1\ldots n_n)\ v_1\ldots v_{n\geq 4}$ are highly restricted. For example it is not possible to derive any string of the form $\sigma(n_1n_2n_3)\ n_4\ v_1v_2v_3v_4$ from lexicon (3.25):

(3.27)
$$\dfrac{\dfrac{n_4}{\mathsf{NP}_4}\quad\dfrac{\dfrac{n_4}{\mathsf{NP}_4}}{\mathsf{S}_4/(\mathsf{S}_4\backslash\mathsf{NP}_4)}\,{>}\mathbf{T}\quad\dfrac{v_1}{\mathsf{S}_1\backslash\mathsf{NP}_1}\quad\dfrac{\dfrac{v_2}{(\mathsf{S}_2\backslash\mathsf{NP}_2)\backslash\mathsf{S}_1}\quad\dfrac{\dfrac{v_3}{(\mathsf{S}_3\backslash\mathsf{NP}_3)\backslash\mathsf{S}_2}\quad\dfrac{v_4}{(\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{S}_3}}{((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{S}_2}\,{<}\mathbf{B}^2}{(((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{S}_1}\,{<}\mathbf{B}^2}{(((\mathsf{S}_4\backslash\mathsf{NP}_4)\backslash\mathsf{NP}_3)\backslash\mathsf{NP}_2)\backslash\mathsf{NP}_1}\,{<}\mathbf{B}^1}{***}$$

As Baldridge (2002) concludes, although the CCG formalism does *not* allow for the derivation of doubly unbounded long-distance scrambling constructions, it *does* permit all such examples to be derived, for which reliable native-speaker judgments can be obtained.

3.1.4 Summary

In this section, I have discussed some of the implications of the claim that CCG is a *restrictive* grammar formalism, in the sense that its generative capacity with respect to alphabet Σ is a proper subset of the context-sensitive languages over Σ. As Baldridge (2002) notes (p.6): "By using a system with limited generative power, many linguistic predictions come for free since the system simply cannot perform a wide range of potential operations." In particular, I have shown that CCG makes clear, falsifiable predictions about what kinds of unbounded dependency construction are involved in human language competence. These predictions are theorems of the foundational principles of the CCG formalism, and crucially do not require independent stipulation in the form of constraints on structural representations.

Subsection 3.1.1 showed that CCG predicts the existence of unbounded *nested* dependencies in linguistic competence, in common with all other formalisms whose generative capacity is context-free or higher. Subsection 3.1.2 argued that CCG also predicts the existence of unbounded *cross-serial* dependencies in human language competence, unlike context-free formalisms which predict that such constructions must be bounded at best. Subsection 3.1.3 showed that CCG predicts that human language competence *lacks* doubly unbounded long-distance scrambling constructions, although a certain degree of such scrambling is permitted.

3.2 CCG and implicational language universals

In this section, I turn to the study of language typology and universals (Comrie, 1989), in an attempt to evaluate how successful CCG is in predicting *implicational universals* of human language, i.e. those observations gleaned from the comparative, typological study of a large number of languages, preferably from different families and geographical areas, which make a statement of the following form: if a human language has property P then it also has property Q.

A language universal, as the name suggests, ideally involves universal quantification over the class of natural languages i.e. a statement of the following form:

(3.28) *All* languages have property P, or $\forall x.Px$

However universals are typically expressed in less clearcut terms, using generalised

quantifiers such as:

(3.29) *Most* languages have property P

Almost all languages have property Q

In attempting to explain such *statistical* universals, considerations of generative capacity are of limited use, in contrast with the case of the *absolute* universals discussed in section 3.1. What is necessary is some means of defining a *markedness* ordering on the languages generated by a particular formalism; this will turn out to be a major topic of this thesis.

The study of language universals has its roots in Greenberg (1963), who presented forty-five provisional universals, both absolute and statistical, based on a diverse sample of thirty human languages. Greenberg's universal number 1 is as follows:

> In declarative sentences with nominal subject and object, the dominant order is almost always one in which the subject precedes the object.

This is an example of a simple, statistical universal — statistical because it contains a generalised quantifier 'almost always', and simple since it involves only the one property, 'subject precedes object'. However, few of the Greenbergian universals are as simple as this. Most involve a correlation between two independent properties, for example number 3:

(3.30) Languages with dominant VSO order are always prepositional.

This is an example of an 'implicational' universal i.e. one which relates two properties ('being VSO' vs. 'being prepositional') in terms of material implication. An implicational universal is a statement of the following form:

(3.31) All languages having property P also have property Q, or $\forall x.Px \rightarrow Qx$

Such a statement is literally true just in case there is no P language which is not also a Q language i.e. $P \subseteq Q$. However, note that the implicational universals which are of interest in the study of linguistic typology are a strict subset of those that are literally true — they need to be 'meaningful' as well. An implicational universal of the form $P \subseteq Q$ is meaningful just in case the following four conditions all hold:

1. some languages have both properties P and Q, i.e. $P \cap Q \neq \emptyset$
2. some languages have neither property P nor Q, i.e. $P' \cap Q' \neq \emptyset$

3. some languages have property Q but lack property P, i.e. $P' \cap Q \neq \emptyset$

4. no language has property P but lacks property Q, i.e. $P \cap Q' = \emptyset$

Greenberg's universal number 3, quoted in (3.30) above, is an example of a meaningful implicational universal, since: (a) there exist VSO, prepositional languages (e.g. Irish); (b) there exist non-VSO, non-prepositional languages (e.g. Japanese); (c) there exist non-VSO, prepositional languages (e.g. English, Persian); but (d) there are no clearly attested VSO, non-prepositional languages. Comrie (1989), on the other hand, cites the following example of a true, but non-meaningful implicational universal:

(3.32) All languages with nasal vowels also have oral vowels.

This statement is literally true since there are no human languages whose phoneme inventory contains nasal vowels but lacks oral vowels. However, it is not meaningful since the second condition above does not hold i.e. there are no languages which have neither nasal nor oral vowels. A more meaningful statement would simply be: all languages have oral vowels.

A special case involves an implicational universal where the third statement in the definition of meaningful implicational universals does not hold, but the others all do, i.e. $P' \cap Q = \emptyset$. This constitutes a *bidirectional* implication i.e. $P \subseteq Q \;\&\; Q \subseteq P$ or equivalently $P = Q$.

In the following sections I review two arguments that substantiate the claims that CCG is a more successful theory of human linguistic competence than its phrase structure-based competitors.

Subsection 3.2.1 argues, following Steedman (1990), that CCG predicts the correlations found in Ross (1970) between the potential for gapping constructions in a language and its basic word order.

Subsection 3.2.2 makes a similar argument for aspects of the familiar 'null subject parameter' of Government and Binding Theory, which identifies a correlation in the family of Romance languages between the possibility that the subject may appear optionally in a post-verbal position, and the ability of a subject to be extracted across an overt complementiser.

All in all, the aim of this section is not to present novel linguistic explanations of these phenomena, but rather to draw attention to the kind of claim that language universals make and, in particular. the different ways in which CCG goes about explaining them.

3.2.1 Basic word order and gapping

In this subsection, I will be discussing one simple implicational universal deriving from the work of Ross (1970), involving correlations between the basic word order of a language and the existence of gapping constructions. The particular universal I am interested in is the following:

(3.33) All languages with a backward-gapping construction are verb-final.

This kind of implicational universal needs to be seen as the outcome of at least *two* steps of activity. The first step involves formulating *two independent binary partitions* of the class of natural languages. In the current example, the first partition is as follows:

OV in main declarative clauses, the main verb can occur in the final position, after both its subject and object

VO in main declarative clauses, the main verb cannot occur in final position[3]

These two classes are, of course, to be understood as being both: (a) mutually exclusive i.e. no language can belong to both; and (b) exhaustive i.e. every language belongs to at least one. Japanese is an example of an OV language:

(3.34) John-ga hon-o yon-da.

John-NOM book-ACC read-PAST

John read a book.

The OSV ordering is also possible in Japanese as a marked alternative. Turkish is also an OV language:

(3.35) Ayse kitab-i oku-yor.

Ayse book-ACC read-PROG

Ayse reads the book.

Again OSV ordering is possible but marked in Turkish. In fact, all six constituent orders are possible in Turkish, and SVO, OVS, VSO and VOS are considered to be more marked than both SOV and OSV. Another example of an OV language is Tamil:

(3.36) John Mary-ai kadhali-thth-aan.

John Mary-ACC love-PRES-SG

John loves Mary.

[3]Note that, in this context, VO is just another name for OV′, i.e. the complement of OV.

On the other hand, Irish is an example of a VO language. VSO ordering is possible in Irish main declarative clauses:

(3.37) Chonaic Eoghan Siobhán.
saw Eoghan Siobhán
Eoghan saw Siobhán.

But neither kind of OV ordering is possible:

(3.38) *Eoghan Siobhán chonaic.
*Siobhán Eoghan chonaic.

The second binary partition underlying our implicational universal involves the notion of a *backward-gapping* construction. In short, a language has a backward-gapping construction if it is possible to take two or more conjoined transitive main clauses with the same verb as in (3.39), to delete all but the *last* verb (3.40), and still have a grammatical sentence with the same meaning (3.41).

(3.39) John read a book and Bill read a newspaper.

(3.40) John ... a book and Bill read a newspaper.

(3.41) John a book and Bill read a newspaper.

Obviously, English does *not* have a backward-gapping construction, since (3.41) is ungrammatical. However, Japanese does:

(3.42) John-ga hon-o, Bill-ga shimbun-o yon-da.
John-NOM book-ACC Bill-NOM newspaper-ACC read-PAST
John read a book and Bill a newspaper.

As do Turkish and Tamil:

(3.43) Adam dergi-yi kiz kitab-i oku-yor.
man magazine-ACC girl book-ACC read.PROG
The man is reading the magazine and the girl the book.

(3.44) John Mary-ai-um Bill Susan-ai-um kaadhali-thth-aarkal.
John Mary-ACC-CONJ Bill Susan-ACC-CONJ love-PRES-PL
John loves Mary and Bill Susan.

However, like English, Irish does not have backward-gapping constructions:

(3.45) * Eoghan Siobhán agus chonaic Eoghnaí Ciarán.

Eoghan Siobhán and saw Eoghnaí Ciarán

Thus, the second partition involves dividing the class of natural languages into another two disjoint, exhaustive subclasses of language:

B-GAP there are backward-gapping constructions

B-GAP′ there are no backward-gapping constructions[4]

Thus, Japanese, Turkish, Tamil $\in$ B-GAP and English, Irish $\in$ B-GAP′. We now have a two-dimensional binary partition of the natural languages: (a) OV vs. VO; and (b) B-GAP vs. B-GAP′. This gives us a complex typology of *four* mutually exclusive, exhaustive subclasses of language:

1. OV $\cap$ B-GAP
2. OV $\cap$ B-GAP′
3. VO $\cap$ B-GAP
4. VO $\cap$ B-GAP′

The second step in formulating an implicational universal of natural language involves finding as many attested languages as possible, and categorising them into each of the four complex types.[5] In the example I am pursuing here, the relevant data is as follows:

1. OV $\cap$ B-GAP $= \{\text{Japanese}, \text{Tamil}, \text{Turkish}, \ldots\}$
2. OV $\cap$ B-GAP′ $= \{\ldots\}$
3. VO $\cap$ B-GAP $= \emptyset$
4. VO $\cap$ B-GAP′ $= \{\text{English}, \text{Irish}, \text{Tagalog}, \ldots\}$

[4] Note that this does not imply that there are forward-gapping constructions in *all* these languages.

[5] Actually things are rather more complex than this, since the sample must be weighted to take account of genetic and geographical influences (Comrie, 1989).

In other words, whereas Ross was able to find a number of attested examples of both verb-final languages with backward-gapping constructions and verb-nonfinal languages with no backward-gapping constructions, he could find *no clear instances* of verb-nonfinal languages with backward-gapping. Note that Ross is unclear on the existence of verb-final languages *without* backward-gapping constructions. If substantial numbers of these do exist, then the relevant implicational universal is bidirectional i.e. B-GAP = OV. If not it is a straightforward implication i.e. B-GAP $\subseteq$ OV.

We should note in passing that Ross (1970) formulated a number of other universal statements about the relationship between basic word order and gapping constructions in human languages. I have chosen to concentrate on just the one aspect of Ross' observations in order to keep the discussion simple — the focus of this section is on *how* a restrictive formalism such as CCG tries to explain such implicational universals, rather than on any detailed analysis of the gapping construction itself.

Having formulated a language universal in this way, the next task is to *explain* it. Perhaps, the most obvious way in which a grammar formalism F can be argued to explain an implicational universal of the form $P \subseteq Q$, where P and Q are classes of language, is if the following four conditions hold:

1. there are F-grammars for $P \cap Q$ languages
2. there are F-grammars for $P' \cap Q$ languages
3. there are F-grammars for $P' \cap Q'$ languages
4. there is *no* F-grammar for any $P \cap Q'$ language

It is this basic approach that underlies the argument in Steedman (1990) that the implicational universal B-GAP $\subseteq$ OV is predicted by the CCG formalism.

Steedman makes *two* important assumptions. Firstly, transitive verbs in OV languages are assumed to be assigned to at least one of the following lexical categories:

(3.46) $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})\backslash\mathsf{NP}_{\mathsf{obj}}$

(3.47) $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{obj}})\backslash\mathsf{NP}_{\mathsf{sbj}}$

In addition, it is assumed that transitive verbs in VO languages are assigned to *neither* of these two lexical categories. Note that for the purposes of this discussion, I assume that the lexical categories in (3.46) and (3.47) are associative but non-permutative i.e.

using the modalities of Baldridge (2002) they would be written as $(S\backslash^{\diamond}NP_{sbj})\backslash^{\diamond}NP_{obj}$ and $(S\backslash^{\diamond}NP_{obj})\backslash^{\diamond}NP_{sbj}$. In other words, they may participate in harmonic composition operations, but not disharmonic ones, as discussed briefly in subsection 2.2.10.
Secondly, coordination is assumed to be an operation which applies to *two adjacent expressions of the same category*. In other words, the coordination rule in CCG is as follows, where CONJ is a coordinating conjunction, for example 'and' or 'or' in English:

(3.48) $X \text{ CONJ } X \overset{\&}{\Rightarrow} X$

Alternatively, we could take a non-syncategorematic approach to syntactic coordination, whereby a coordinating conjunction would be assigned to a non-associative, non-permutative lexical category of the form $(X\backslash^{*}X)/_{*}X$, where X is a variable over category labels (Baldridge, 2002). Using the lexical entry in (3.46), a simple transitive sentence in an SOV language can be derived as follows:

(3.49)
$$\dfrac{\dfrac{\text{subject}}{NP_{sbj}} \quad \dfrac{\dfrac{\text{object}}{NP_{obj}} \quad \dfrac{\text{verb}}{(S\backslash NP_{sbj})\backslash NP_{obj}}}{S\backslash NP_{sbj}}<}{S}<$$

Assuming the approach to coordination summarised in (3.48), it is clear that we can derive a sentential coordination construction like (3.50), as well as a VP coordination like (3.51), since both sentences and VPs are constituents of the derivation in (3.49):

(3.50) SOV and SOV

(3.51) SOV and OV

In addition, Steedman points out that there is an alternative derivation in CCG for the SOV simple transitive sentence, using the same lexical category in (3.46). This alternative derivation involves type raising and composing the two arguments to form an 'argument cluster' constituent:

(3.52)
$$\dfrac{\dfrac{\dfrac{\text{subject}}{NP_{sbj}}}{S/(S\backslash NP_{sbj})}>\mathbf{T} \quad \dfrac{\dfrac{\text{object}}{NP_{obj}}}{(S\backslash NP_{sbj})/((S\backslash NP_{sbj})\backslash NP_{obj})}>\mathbf{T} \quad \dfrac{\text{verb}}{(S\backslash NP_{sbj})\backslash NP_{obj}}}{S/((S\backslash NP_{sbj})\backslash NP_{obj})}>\mathbf{B}1$$
$$\dfrac{}{S}>$$

Since the subject and object form a constituent in this derivation, they can be coordinated according to the rule schema in (3.48), yielding the following backward-gapping construction:

(3.53) SO and SOV

In other words, assigning transitive verbs to the clause-final category in (3.46) *immediately* gives rise to a CCG which can generate backward-gapping constructions i.e. for a B-GAP $\cap$ OV language. The related OSV category in (3.47) has a similar effect, except here the resultant gapping construction is 'OS and OSV'. Furthermore, under the assumption that coordination only applies to adjacent, like constituents, the fact that CCG disallows empty categories on principle (i.e. the Principle of Adjacency) means that there is *no* CCG which can generate a B-GAP $\cap$ VO language. Thus, the CCG formalism is argued to predict the B-GAP $\subseteq$ OV implicational universal in a very strong sense, since the unattested language type lies outwith the generative capacity of the formalism.

To conclude this section, I discussed one particular implicational universal of natural language, deriving from the work of Ross (1970). Like all simple implicational univerals, this one is based on a two-dimensional binary partition of the domain involving clausal word order and presence or absence of backward-gapping constructions. The implicational universal is as follows: if a language allows backward-gapping constructions, then it has possible verb-final ordering in declarative main clauses i.e. B-GAP $\subseteq$ OV. I then reviewed the claim in Steedman (1990) that the CCG formalism predicts this universal, and argued that this prediction is a result of the fact that the three attested language types can all be generated by some CCG but the unattested one cannot. Note in particular that this universal thus turns out to be a *theorem* of the underlying principles of CCG i.e. the Principles of Adjacency, Inheritence, Consistency etc. and does not require any kind of independent stipulation.

Before moving on a final observation is in order. The analysis of Steedman (1990) and chapter 6 of Steedman (2000) actually covers a significantly larger *superset* of the data presented here. Explanations are also provided for other universals like: (a) why languages with forward-gapping constructions have post-verbal direct objects; and (b) why SOV languages with alternative OSV word orders exhibit the following gapping possibilities: SO&SOV, OS&OSV, *SO&OSV, *OS&SOV.

3.2.2 Free inversion and subject extraction in Romance

In the previous section, I discussed a simple implicational universal involving a correlation between basic word order and the existence of backward-gapping in a language, for which the CCG formalism provides a rather straightforward explanation — the three attested language types are all within the generative capacity of CCG, but the unattested type is not. In this section, I turn to another equally well-known simple implicational universal, this time involving a correlation between the existence of free subject inversion and extractability of subjects from subordinate clauses in Romance languages. Steedman (1996) suggests that this universal is also predicted by the CCG formalism, a claim which I examine here.

The fixed subject constraint

The four complex noun phrases from English in (3.54) to (3.57) exemplify a phenomenon first noticed by Perlmutter (1971), and nicknamed the 'fixed subject constraint' by Bresnan (1977):

(3.54) the woman whom John loves

(3.55) the man who loves Mary

(3.56) the woman whom Bill thinks that John loves

(3.57) *the man who Bill thinks that loves Mary

Examples (3.54) and (3.55) are instances of matrix object and subject relativisation respectively. In example (3.56) it is the object of an embedded (complement) clause which is relativised. Example (3.57), on the other hand, shows that the same does not go for embedded subjects. The subject of a subordinate clause (introduced by the complementiser 'that') cannot be relativised in English. It is this non-extractability of subjects over a complementiser which has been termed the fixed subject constraint (FSC), or in the GB tradition the '*that*-trace effect'.

The data in (3.58) to (3.61) show that the FSC is also part of the grammar of French:

(3.58) la femme que Jean aime

the woman who.OBJ Jean love

the woman who Jean loves

(3.59) l'homme qui aime Marie
the man who.SBJ love Marie
the man who loves Marie

(3.60) la femme que Bill croit que Jean aime
the woman who.OBJ Bill believe that Jean love
the woman who Bill believes that Jean loves

(3.61) *l'homme qui Bill croit qu'aime Marie
the man who.SBJ Bill believe that love Marie
**the man who Bill thinks that loves Marie*

Steedman (1996) argues that the FSC is predicted by the CCG formalism for the grammars of English and French. In fact, CCG predicts that all configurational SVO languages will exhibit the FSC. The obvious lexical category for transitive verbs in a configurational SVO language is the following:

(3.62) $\mathsf{loves} \vdash (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$

Note that the subject argument carries a $\diamond$ modality. In other words, a transitive verb phrase will be a non-permutative expression and thus cannot participate in disharmonic composition operations. The following lexical assignments permit the formation of matrix subject and object relatives from such an SVO lexicon:

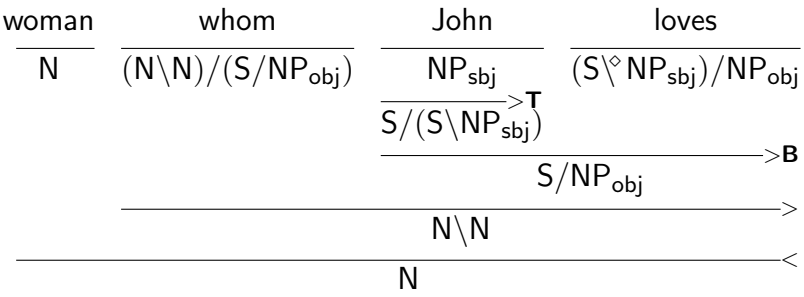
(3.63)
$$\begin{array}{rcl} \mathsf{whom} & \vdash & (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}) \\ \mathsf{who} & \vdash & (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}) \\ \mathsf{John}, \mathsf{Mary}, \mathsf{Bill} & \vdash & \mathsf{NP}_{\mathsf{sbj}},\ \mathsf{NP}_{\mathsf{obj}} \end{array}$$

Here is a derivation of a matrix subject relativisation in English:

(3.64)
$$\dfrac{\dfrac{\mathsf{man}}{\mathsf{N}} \quad \dfrac{\dfrac{\mathsf{who}}{(\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})} \quad \dfrac{\dfrac{\mathsf{loves}}{(\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}} \quad \dfrac{\mathsf{Mary}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}}}>}{\mathsf{N}\backslash\mathsf{N}}>}{\mathsf{N}}<$$

The following involves a matrix object relativisation:

(3.65)



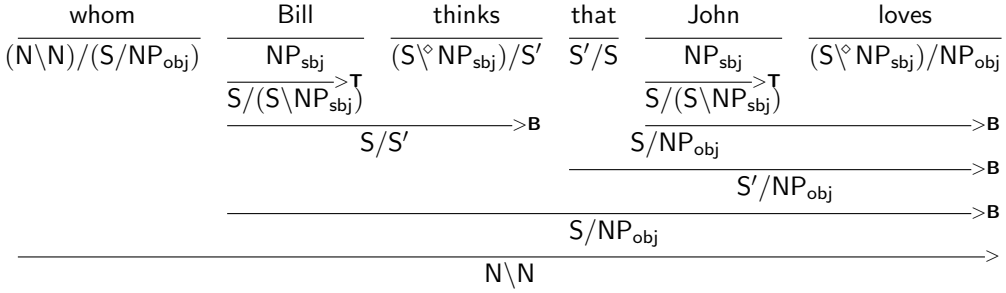
In addition, assume the following lexical assignments for sentential complement verbs and complementisers in configurational SVO languages:

(3.66) $\text{thinks} \vdash (S\backslash^{\diamond}NP_{sbj})/S'$

$\text{that} \vdash S'/S$

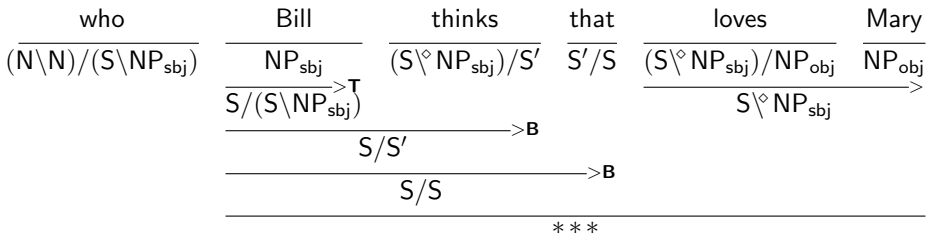
The configurational SVO lexicon in (3.62), (3.63) and (3.66) allows derivation of embedded object relativisations as follows:

(3.67)



Note here that the combinatory operations of forward raising and forward harmonic composition are used to create a constituent corresponding to the entire relativised clause without the extracted object i.e. 'Bill thinks that John loves'. However, the corresponding embedded subject relativisation is disallowed, since the necessary forward cross composition is blocked by the $\diamond$ modality:

(3.68)



Thus, as argued in Steedman (1996), CCG predicts that the FSC will be a feature of any language in which transitive verbs are assigned to the configurational SVO lexical category in (3.62), for example English and French. Note that simply altering the modality on the subject argument to the permutative $\times$ does not give rise to a language

which is exactly the same as English/French except that it licenses embedded subject extraction. In fact, the language generated by such a grammar would lack the property of configurationality i.e. word order would collapse, and non-sentences like **I John think that loves Mary* would be generated.

Embedded subject extraction in Italian

As first noted by Perlmutter (1971), the FSC is not a universal property of natural language. For example, the data in examples (3.69) to (3.72) show that the FSC is not part of the grammar of Italian:

(3.69) la donna che Gianni ama
the woman who Gianni love
the woman whom Gianni loves

(3.70) l'uomo che ama Maria
the man who love Maria
the man who loves Maria

(3.71) la donna che Bill creda che Gianni ama
the woman who Bill believe that Gianni love
the woman whom Bill believes that Gianni loves

(3.72) l'uomo che Bill creda che ama Maria
the man who Bill believe that love Maria
**the man who Bill believes that loves Maria*

Here, example (3.70) contains a matrix subject relativisation, and (3.69) and (3.71) illustrate matrix and embedded object relativisation respectively. These three examples are exactly parallel to the corresponding English examples in (3.54) to (3.56), and the French examples in (3.58) to (3.60). However, whereas the embedded subject relativisation examples in English (3.57) and French (3.61) are ungrammatical, the corresponding Italian relative clause in (3.72) *is* grammatical. In other words, the FSC does not apply in the grammar of Italian, as embedded subjects *can* be extracted across a complementiser

Now, Italian is generally considered to be a configurational SVO language, just like French and English, where transitive verbs are assigned to the lexical category in (3.62), repeated here:

(3.73) $\mathsf{ama} \vdash (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$

As discussed above, such a category blocks extraction of embedded subjects. How then might we derive the Italian embedded subject relativisation in (3.72)? Perlmutter (1971) notes that the languages which violate the FSC all appear to allow null pronominal subjects, and suggests that extractability of embedded subjects is a result of this so-called '*pro*-drop' property. Alternatively, Rizzi (1982) notes that Italian is not strictly speaking a 'pure' SVO language. In fact, Italian has the property of 'free subject inversion', where declarative main clauses allow both SVO and VOS order, with no change in truth-conditional meaning:

(3.74) Gianni ama Maria.

Gianni love Maria

Gianni loves Maria

(3.75) Ama Maria Gianni.

love Maria Gianni

Gianni loves Maria

These examples suggest that transitive verbs in Italian are actually assigned to the following lexical category, where the directionality of the subject argument is unspecified:

(3.76) $\mathsf{ama} \vdash (\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$

Both normal SVO order and the free subject inversion VOS order can thus be derived from a single transitive verb category:

(3.77)

$$\dfrac{\dfrac{\mathsf{Gianni}}{\mathsf{NP}_{\mathsf{sbj}}} \quad \dfrac{\dfrac{\mathsf{ama}}{(\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}} \quad \dfrac{\mathsf{Maria}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}}}\,{>}}{\mathsf{S}}\,{<} \qquad \dfrac{\dfrac{\dfrac{\mathsf{ama}}{(\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}} \quad \dfrac{\mathsf{Maria}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}}}\,{>} \quad \dfrac{\mathsf{Gianni}}{\mathsf{NP}_{\mathsf{sbj}}}}{\mathsf{S}}\,{>}$$

The lexicon for the Italian fragment can be completed as follows:

(3.78) $\begin{array}{rcl} \textsf{che} & \vdash & (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{sbj}}),\ (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}) \\ \textsf{creda} & \vdash & (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}' \\ \textsf{che} & \vdash & \mathsf{S}'/\mathsf{S} \end{array}$

Using this lexicon, we can derive the matrix object and subject relatives in examples (3.69) and (3.70), as well as the embedded object relative in (3.71). The derivation of the latter is as follows:

(3.79)

$$\dfrac{\dfrac{\textsf{che}}{(\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}})}\quad \dfrac{\dfrac{\dfrac{\textsf{Bill}}{\mathsf{NP}_{\mathsf{sbj}}}}{\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})}{>}\mathbf{T}\quad \dfrac{\textsf{creda}}{(\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}'}}{\mathsf{S}/\mathsf{S}'}{>}\mathbf{B}\quad \dfrac{\textsf{che}}{\mathsf{S}'/\mathsf{S}}\quad \dfrac{\dfrac{\dfrac{\textsf{Gianni}}{\mathsf{NP}_{\mathsf{sbj}}}}{\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})}{>}\mathbf{T}\quad \dfrac{\textsf{ama}}{(\mathsf{S}\mid^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}}{>}\mathbf{B}}{\mathsf{S}'/\mathsf{NP}_{\mathsf{obj}}}{>}\mathbf{B}}{\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}}{>}\mathbf{B}}{\mathsf{N}\backslash\mathsf{N}}{>}$$

Moreover, the neutral slash on the subject argument of transitive verbs means that the embedded subject relativisation in (3.72) can be derived as well:

(3.80)

$$\dfrac{\dfrac{\textsf{che}}{(\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{sbj}})}\quad \dfrac{\dfrac{\dfrac{\textsf{Bill}}{\mathsf{NP}_{\mathsf{sbj}}}}{\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})}{>}\mathbf{T}\quad \dfrac{\textsf{creda}}{(\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}'}}{\mathsf{S}/\mathsf{S}'}{>}\mathbf{B}\quad \dfrac{\textsf{che}}{\mathsf{S}'/\mathsf{S}}\quad \dfrac{\dfrac{\textsf{ama}}{(\mathsf{S}\mid^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}}\quad \dfrac{\textsf{Maria}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\mid^{\diamond}\mathsf{NP}_{\mathsf{sbj}}}{>}}{\mathsf{S}'/\mathsf{NP}_{\mathsf{sbj}}}{>}\mathbf{B}}{\mathsf{S}/\mathsf{NP}_{\mathsf{sbj}}}{>}\mathbf{B}}{\mathsf{N}\backslash\mathsf{N}}{>}$$

In sum, Steedman (1996) argues that the CCG formalism makes the following predictions: (a) configurational SVO languages *without* free subject inversion will disallow relativisation of embedded subjects; whereas (b) configurational SVO languages *with* free subject inversion will permit relativisation of embedded subjects. In other words, English and French are predicted to satisfy the FSC because verbs are assigned to categories of the form $(\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\$$, but Italian is predicted to violate the FSC because verbs are assigned to categories of the form $(\mathsf{S}\mid^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\$$. Thus, whether or not a language satisfies the FSC is determined by its lexicon, and no extralexical stipulation (for instance the 'empty category principle' of GB theory) is required.

The *pro*-drop parameter

The correlation in configurational SVO languages between extractability of embedded subjects and existence of free subject inversion is one aspect of an implicational universal which has become known in the generative grammar literature as the '*pro*-drop' or 'null subject' parameter. It will prove instructive to formalise this implicational universal, as I did with that discussed in subsection 3.2.1, in order to shed light on the claim that it is *predicted*, or *explained*, by the CCG formalism.

The domain of validity for this implicational universal is strictly speaking the class of *configurational SVO* languages, in particular the Romance languages and English. The first binary partition of the domain involves whether or not subjects are extractable from an embedded clause:

FSC subjects may *not* be extracted from embedded complement clauses

FSC′ subjects *may* be extracted from embedded complement clauses

The second binary partition involves the potential for free subject inversion in the language:

FI VOS ordering is possible in matrix declarative clauses

FI′ VOS ordering is *not* possible in matrix declarative clauses

Thus, the domain is partitioned into the following four mutually exclusive, exhaustive, complex classes:

1. $\text{FSC} \cap \text{FI} = \emptyset$
2. $\text{FSC} \cap \text{FI}' = \{\text{English, French}, \ldots\}$
3. $\text{FSC}' \cap \text{FI} = \{\text{Italian, Spanish, Romanian}, \ldots\}$
4. $\text{FSC}' \cap \text{FI}' = \emptyset$

In other words, there are clearly attested examples of configurational SVO languages which either lack free inversion and disallow extraction of embedded subjects, or possess free inversion and permit extraction of embedded subjects. However, there are no attested instances of configurational SVO languages which either allow free inversion but not embedded subject extraction, or disallow free inversion but permit embedded

subject extraction. Thus, the implicational universal that can be concluded is the following bidirectional one:

(3.81) $\text{FSC} = \text{FI}'$

CCG and the *pro*-drop parameter

We saw in subsection 3.2.1 that grammar formalism F can be argued to explain implicational universal $P \subseteq Q$ if: (a) there are F-grammars for $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages; but (b) there is *no* F-grammar for a $P \cap Q'$ language. As a special case, formalism F explains *bidirectional* implicational universal $P = Q$ if:

1. there are F-grammars for both $P \cap Q$ and $P' \cap Q'$ languages
2. there are *no* F-grammars for either $P \cap Q'$ or $P' \cap Q$ languages

Thus, turning to the universal in (3.81), we see that one way in which the CCG formalism might be argued to explain this would be if: (a) there are CCGs which generate $\text{FSC} \cap \text{FI}'$ languages; (b) there are CCGs which generate $\text{FSC}' \cap \text{FI}$ languages; (c) there is *no* CCG which generates an $\text{FSC} \cap \text{FI}$ language; and (d) there is *no* CCG which generates an $\text{FSC}' \cap \text{FI}'$ language. Let's deal with each of these four language-types in turn.

$\text{FSC} \cap \text{FI}'$: We have already seen that this is possible to formulate a CCG for this kind of language. The CCG whose lexicon is listed in (3.62), (3.63) and (3.66), repeated here, generates languages like French and English, which both satisfy the FSC and disallow free subject inversion:

(3.82)
$$\begin{aligned} \textsf{loves} &\vdash (\textsf{S}\backslash^{\diamond}\textsf{NP}_{\textsf{sbj}})/\textsf{NP}_{\textsf{obj}} \\ \textsf{whom} &\vdash (\textsf{N}\backslash\textsf{N})/(\textsf{S}/\textsf{NP}_{\textsf{obj}}) \\ \textsf{who} &\vdash (\textsf{N}\backslash\textsf{N})/(\textsf{S}\backslash\textsf{NP}_{\textsf{sbj}}) \\ \textsf{John} &\vdash \textsf{NP}_{\textsf{sbj}},\ \textsf{NP}_{\textsf{obj}} \\ \textsf{thinks} &\vdash (\textsf{S}\backslash^{\diamond}\textsf{NP}_{\textsf{sbj}})/\textsf{S}' \\ \textsf{that} &\vdash \textsf{S}'/\textsf{S} \end{aligned}$$

$\text{FSC}' \cap \text{FI}$: We have already seen a CCG for this kind of language as well. The Italian lexicon in (3.76) and (3.78), repeated here, generates a language which permits both

free subject inversion and embedded subject extraction:

(3.83)
$$\begin{aligned} \text{ama} &\vdash (\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}} \\ \text{che} &\vdash (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}) \\ \text{che} &\vdash (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{sbj}}) \\ \text{Gianni} &\vdash \mathsf{NP}_{\mathsf{sbj}},\ \mathsf{NP}_{\mathsf{obj}} \\ \text{creda} &\vdash (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}' \\ \text{che} &\vdash \mathsf{S}'/\mathsf{S} \end{aligned}$$

FSC $\cap$ **FI**: Can we write a CCG for a language of this type? In other words, can we find a grammar for a language which is like Italian, in that it allows VOS main clause ordering, but disallows extraction of embedded subjects? We can do this quite easily by taking the English-type lexicon in (3.82) and adding the following extra lexical category for proper nouns:

(3.84) $\text{Gianni} \vdash \mathsf{S}\backslash(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})$

This additional category allows VOS ordering to be derived as follows:

(3.85)
$$\dfrac{\dfrac{\dfrac{\text{ama}}{(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}}\quad\dfrac{\text{Maria}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}}{>}\quad\dfrac{\text{Gianni}}{\mathsf{S}\backslash(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})}}{\mathsf{S}}{<}$$

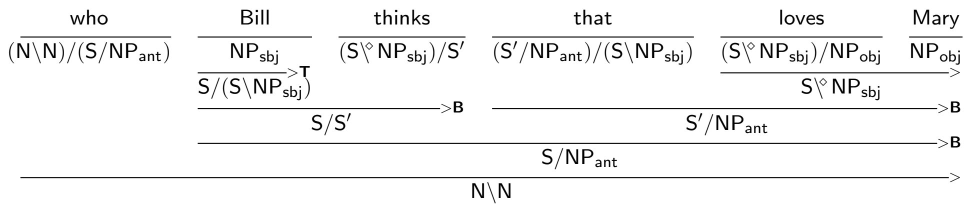
However, embedded subject extraction is still blocked as in derivation (3.68).

FSC$'$ $\cap$ **FI**$'$: Can we write a CCG for a language of this type? In other words, can we find a grammar for a language which is like English in that it disallows inverted subjects in declarative clauses, but violates the FSC by allowing extraction of embedded subjects? Again this turns out to be possible, if a little forced:

(3.86)
$$\begin{aligned} \text{loves} &\vdash (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}} \\ \text{whom} &\vdash (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{obj}}) \\ \text{who} &\vdash (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}) \\ \text{who} &\vdash (\mathsf{N}\backslash\mathsf{N})/(\mathsf{S}/\mathsf{NP}_{\mathsf{ant}}) \\ \text{thinks} &\vdash (\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}' \\ \text{that} &\vdash \mathsf{S}'/\mathsf{S} \\ \text{that} &\vdash (\mathsf{S}'/\mathsf{NP}_{\mathsf{ant}})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}) \\ \text{John} &\vdash \mathsf{NP}_{\mathsf{sbj}},\ \mathsf{NP}_{\mathsf{obj}} \end{aligned}$$

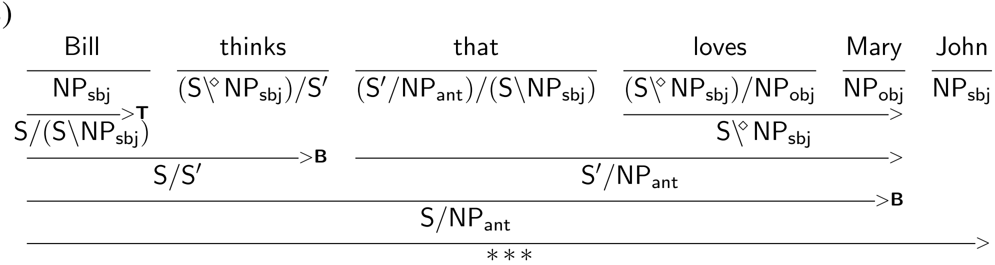
Essentially, this CCG lexicon is similar to the English-type one in (3.82) except that it makes a distinction between two kinds of subject arguments — the normal subject '$\mathsf{NP_{sbj}}$', and the 'antecedent-governed' subject '$\mathsf{NP_{ant}}$', which *must* be extracted. Complementisers are assigned to an additional, subject-extracting lexical category denoted by $(\mathsf{S'/NP_{ant}})/(\mathsf{S\backslash NP_{sbj}})$, and the subject relative pronoun is also assigned to category $(\mathsf{N\backslash N})/(\mathsf{S/NP_{ant}})$. Based on this lexicon, relativisation of embedded subjects is possible:

(3.87)



However, free subject inversion is blocked by the fact that lexical noun phrases do not belong to the antecedent-governed category:

(3.88)



Grammar ranking and the *pro*-drop parameter

Summing up the discussion so far, I have identified an implicational universal FSC $=$ FI$'$, stating a correlation between the existence of extractable embedded subjects and the potential for subject inversion in configurational SVO languages. It was argued that we can conclude that the CCG formalism predicts this universal if: (a) there exist CCGs for both FSC$\cap$FI$'$ and FSC$'\cap$FI languages; but (b) there are *no* CCGs for either FSC$\cap$FI or FSC$'\cap$FI$'$ languages. However, we have found that the CCG formalism provides grammars which generate *all four* kinds of language. Thus, the intuition that CCG predicts this implicational universal must rely upon a more subtle form of argumentation, since simple considerations of generative capacity are not enough in this particular case.

One possible way of explaining the 'unnaturalness' of the CCG lexicons for FSC$\cap$FI and FSC$'\cap$FI$'$ languages is by positing substantive constraints on natural language lexicons which these examples violate. Take the lexicon for an FSC$'\cap$FI$'$ language

presented in (3.86) above. One possible substantive constraint that this lexicon might be said to violate is a general ban on categories such as '$\mathsf{NP}_{\mathsf{ant}}$', which appear as arguments in lexical categories but not as lexical categories themselves. However, such antecedent-governed categories are independently motivated — both Steedman (2000) and Baldridge (2002) argue in considerable detail that the analysis of word order phenomena in Dutch and English crucially requires such a concept. Furthermore, there appears to be no obvious way in which the lexicon for the FSC $\cap$ FI language can be said to violate some such non-ad hoc substantive constraint.

One thing to note about the CCG lexicons for the two unattested language types is that they are *more complex* than the lexicons for the corresponding FSC $\cap$ FI$'$ and FSC$'$ $\cap$ FI languages. The lexicon for the FSC $\cap$ FI language involves taking the FSC $\cap$ FI$'$ lexicon and adding an additional, subject-postposing lexical entry for every lexical noun phrase. Similarly, the lexicon for the FSC$'$ $\cap$ FI$'$ language contains an additional, subject-extracting lexical category for complementisers and subject relative pronouns. These observations lead to the hypothesis that, when it comes to natural language competence, simpler grammars are preferred over complex ones. This notion has been formalised in CCG as the 'Principle of Head Categorial Uniqueness', defined in Steedman (2000) as follows (p.33):

> A single nondisjunctive lexical category for the head of a given construction specifies both the bounded dependencies that arise when its complements are in canonical position and the unbounded dependencies that arise when these complements are displaced under relativization, coordination, and the like.

This principle embodies a kind of ideal methodology for CCG design. The practical outcome is that, when we are extending a CCG for some fragment of a human language in order to account for some new unbounded syntactic construction, we should try to avoid the temptation to capture this by simply adding an additional lexical entry for a class of words already in the lexicon. This approach is justified in terms of lexical economy (ibid. p.32):

> The size of the lexicon involved is ... an important aspect of a grammar's complexity. Other things being equal, one lexical grammar is simpler than another if it captures the same pairings of strings and interpretations using a smaller lexicon.

This all suggests that the intuition that the CCG formalism predicts the correlation between extractability of embedded subjects and free subject inversion relies upon

something like the following generalised mode of argumentation:

Grammar formalism F can be argued to *explain* implicational universal $P \subseteq Q$ if the following conditions all hold:

1. there are equally simple F-grammars for equivalent $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages

2. the simplest F-grammar for any $P \cap Q'$ language, should one exist, is more complex than some F-grammar of one of its $P \cap Q$, $P' \cap Q$ or $P' \cap Q'$ equivalents

By extension, we can state that formalism F explains bidirectional universal $P = Q$ if there are equally simple F-grammars for $P \cap Q$ and $P' \cap Q'$ languages, but the simplest F-grammar for any $P \cap Q'$ or $P' \cap Q$ language is more complex than some F-grammar of one of its $P \cap Q$ or $P' \cap Q'$ equivalents.

There are a number of ways in which we might go about formulating a simplicity metric for comparing grammars. The most obvious approach, following Chomsky (1965), would be to define simplicity of a grammar *intrinsically*, as a function of its *length*. In this case, the second condition of the definition above would be reinterpreted as follows: the *shortest* F-grammar for any $P \cap Q'$ language, should one exist, is *longer* than some F-grammar of one of its $P \cap Q$, $P' \cap Q$ or $P' \cap Q'$ equivalents.

Introducing the notion of the 'length' of a formal grammar immediately raises the question of what we should be 'counting' when we 'measure' such a thing. Should we be counting the number of rules and lexical entries? The number of symbols? I contend that it is possible to give a rigourous definition of the length of an arbitrary grammar from a particular formalism, by making use of basic concepts from information theory. Recall that a CCG over alphabet Σ is defined formally as an ordered triple $\langle A, S, L \rangle$ where: (a) A is an alphabet of saturated category symbols; (b) $S \in A$; and (c) L is a finite mapping from Σ to CCG category descriptions over A. Based on this definition, there are at least three different ways we could measure the length of an arbitrary CCG.

Firstly, we could say that the length of CCG $\langle A, S, L \rangle$ over alphabet Σ is simply the *cardinality* of L i.e. we just count the number of lexical entries. This is the most obvious approach, and it is one which is adequate for the vast majority of comparisons. However, simply measuring the raw cardinality of the lexicon would fail to distinguish between two CCGs with exactly the same number of lexical assignments, but where one of them had, on average, significantly 'longer' lexical categories than the other.

Secondly, we could define the length of a CCG category description as the number of saturated category symbol-tokens which it contains, and then state that the length of CCG $\langle A,S,L\rangle$ is $\sum_{\langle\sigma,\phi\rangle\in L}(\phi')$, where ϕ' is the length of category description ϕ. In other words, the length of a CCG is the number of symbol-tokens in the lexicon. This approach would certainly give a more accurate measure than simply counting the number of lexical items, and for almost all purposes it would be sufficient. However, it fails to take into account the fact that two grammars with the same number of lexical items, each containing the same number of symbols, can still be of significantly different length, in the sense that the minimal bit encoding of one is much longer than the other. This situation would arise in the case where one grammar makes use of a much larger alphabet of saturated category symbols than the other.

What we need here is a means of scaling the measurement of grammar length to account for the number of distinct symbol-types which it contains. First of all note that the number of symbol-types in CCG $\langle A,S,L\rangle$ over alphabet Σ is $|\Sigma|+|A|+5$, assuming three slashes and two parenthesis symbols.[6] Thus, the number of bits required to encode each symbol will be $\log_2(|\Sigma|+|A|+5)$. The length of CCG $\langle A,S,L\rangle$ over alphabet Σ can then be ascertained by multiplying the total number of symbol-tokens in the lexicon by the number of bits required to encode each symbol-type:

$$\sum_{\langle\sigma,\phi\rangle\in L}(\phi'+1)\log_2(|\Sigma|+|A|+5) \tag{3.89}$$

Thus, it is possible to formulate a precise comparison of the length of two distinct CCGs, in terms of the minimal number of bits required to represent the grammar in the memory of some computing device. This method can be generalised to any properly defined grammar formalism. However, note that in the following discussion it will seldom be necessary to use such a precise measure, as a basic comparison of the number of lexical items will in general suffice to make a clear decision.

One way in which we might make use of such a length-based measure of grammar simplicity is in the case where we have two distinct CCGs of the *same* language, and we (or the language learner) want to choose the best. This leads to the acquisition strategy discussed in Chomsky (1965) and section 10.2 of Steedman (2000), whereby children learning a language select the shortest CCG which generates the set of sentences they have internalised at that point. The length-based grammar evaluation measure can also be used to compare grammars of *different* languages, and from this perspective

[6] I ignore the need for a separator symbol between lexical items.

can form the background against which claims that a grammar formalism predicts an implication universal can be considered.

Let's illustrate this point by returning to the example above. I identified a bidirectional language universal FSC = FI′, which Steedman (1996) claims is 'captured' in some way by the CCG formalism, but which cannot be explained in terms of raw generative capacity, since all four language types, including the two unattested ones, can be generated by CCGs. In terms of the generalised, simplicity based approach, this universal is captured by the CCG formalism if:

1. there are equally short CCGs for equivalent FSC ∩ FI′ and FSC′ ∩ FI languages
2. for every CCG of an FSC ∩ FI or FSC′ ∩ FI language, there is a *shorter* CCG of the equivalent FSC ∩ FI′ or FSC′ ∩ FI language

We saw above that the first condition holds — lexicon (3.82) generates an English-style FSC ∩ FI′ language, and lexicon (3.83) generates an Italian-style FSC′ ∩ FI language. In addition, both lexicons have the same cardinality and indeed contain the same number of symbol-tokens.

How about the second condition? We were able to provide a CCG for an FSC ∩ FI language by adding a second, subject-postposing lexical category for every proper noun in the FSC ∩ FI′ lexicon in (3.82). Is there a shorter CCG for an equivalent FSC ∩ FI′ or FSC′ ∩ FI language? We have already seen that there is. The CCG for the equivalent Italian-style FSC′ ∩ FI language in (3.83) is shorter, since it requires one fewer lexical entry for every proper noun. Note that the language generated by the latter is arguably *more* expressive than that generated by the former former, since it allows subject relativisation.

Similarly, we were able to provide a CCG for an FSC′ ∩ FI′ language by distinguishing between normal and antecedent-governed subjects, and adding another lexical entry for every verb in the lexicon. Again, this grammar proves to be substantially longer than the related FSC′ ∩ FI language, with again no corresponding gain in expressivity.

Competing factors

In sum, it appears that the generalised, simplicity-based approach to linguistic explanation, based on a straightforward notion of grammar-length, appears to provide the

underlying basis for the intuition that the CCG formalism captures the correlation between the possibility of embedded subject extraction and the existence of free inversion in configurational SVO languages. Before concluding this section, a couple of caveats are in order however.

Firstly, alert readers might have noted that the CCG presented above for the FSC$\cap$ FI language, involving an additional subject-postposing lexical entry for proper nouns, is arguably *not* the simplest possible. Simply taking the FSC$'\cap$FI lexicon in (3.83) and changing the modality on the subject argument of transitive verbs to a non-associative $\star$ (i.e. $(\mathsf{S}|\star\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$) will also generate a language which exhibits free subject inversion but not extraction of embedded subjects. However, there are independent reasons for not proceeding down this road, basically due to the fact that restricting the combinatory potential of subjects to such an extent would mean that, for example, right-node raising constructions would not be derivable. In other words, underlying the system of modalities in Baldridge (2002) is the assumption that $\star$ categories are 'marked' with respect to $\diamond$ ones.

Secondly, the above presentation of the simplicity-based approach to linguistic explanation is obviously a rather gross simplification. Recall that the basic methodology involved the claim that grammar G_1 is more likely to be a human language grammar than grammar G_2 if G_1 is both smaller than and no less expressive than G_2. However, this ignores one of the most crucial aspects of natural languages — the fact that they are *acquired* by children, and that language acquisition is a *dynamic* process, which proceeds in *stages*. In other words, linguistic competence should not be viewed as simply a static set of sentences generatable by the mature grammar, but rather as a sequence of snapshots over time, as the child develops an increasingly complex language system, building on what he/she has acquired before. It is *at each stage* of the process of internalising a grammar that the streamlining strategy which underlies the simplicity metric has its effects. Indeed, crucial aspects of this may be obscured in the adult grammar.

As an example of what I mean by this, let's go back to the issue of embedded subject extraction in English. It was noted above that English does not allow extraction of embedded subjects, as in example (3.57). However, this is only the case for subordinate clauses introduced by a complementiser. Many sentential complement verbs in English also allow a finite subordinate clause with *no* complementiser:

(3.90) Bill thinks John loves Mary.

In this case, relativisation of the embedded subject *is* possible:

(3.91) the man who Bill thinks loves Mary

The rather puzzling contrast between (3.57) and (3.91) has long been a topic of active research in generative grammar. One aspect which is generally ignored, but is pointed out by Steedman (1996), involves the fact that children acquiring English appear to internalise embedded object relatives like (3.56) at an earlier stage than embedded subject extractions like (3.91) (Stromswold, 1995). Steedman uses this as justification for giving the embedded subject extraction special treatment in the grammar of English. Verbs like 'think' are assigned to two extra lexical categories:

(3.92) $\mathsf{thinks} \vdash (\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{S}$

$((\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{ant}})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})$

The first of these licenses examples like (3.90). The second is a special embedded subject-extracting verb, which is invoked in examples like (3.91).

If we were to take a static approach to the simplicity-based approach to grammar markedness, then the resulting CCG for adult English would presumably score rather low, compared to say Italian. However, when one considers that acquisition of English is a dynamic process, and that later stages build on the grammar that has been formulated up to that point rather than starting afresh, i.e. the 'constructivist' assumption of Ingram (1989), then it makes more sense that English is the way it is. If children are motivated to acquire matrix relatives first, then there are two possible lexical category schemata for verbs in configurational SVO languages: (a) $(\mathsf{S}\backslash^{\diamond}\mathsf{NP}_{\mathsf{sbj}})/\$$ (i.e. English, French); and (b) $(\mathsf{S}\,|^{\diamond}\,\mathsf{NP}_{\mathsf{sbj}})/\$$ (i.e. Italian, Spanish). Assume that the child acquires whichever of these is most consistent with what he/she hears adults saying, and then moves on to the next stage of language acquisition, freezing those lexical generalisations he/she has acquired at this point.

At some subsequent stage of acquisition, the child is motivated to acquire embedded argument relativisations. The child who has hypothesised category (a) for verbs will thus be able to produce embedded object relativisations straight away, with no extensions to the grammar needed. Similarly, the child who has hypothesised category (b) will be able to produce both embedded objects and subjects straight away (as well as freely inverted subjects). Finally, at yet another subsequent stage of language acquisition, the first child will be motivated to enrich the expressiveness of the language he/she has internalised by adding embedded subject relativisation. Obviously,

the most efficient lexicon which does all this is the Italian-style one. However, the English-speaking child is not at liberty to simply delete all the knowledge he/she has internalised already, and start over with an $\text{FSC}' \cap \text{FI}$ lexicon. So what he/she does is make the fewest possible additions to the lexicon in order to generate the required constructions, which in this case involves adding the additional, subject-extracting lexical category for bare sentence-complement verbs in (3.92).

I hardly need to point out here that, from the perspective of the study of child language acquisition, this is highly speculative, and is by no means meant to constitute a fully-fledged theory of how children internalise the syntax of their first language. My intention is simply to point out that this kind of reasoning is implicit in many explanations of why human languages are the way they are. In particular, the claim that the CCG formalism captures the correlation between free subject inversion and relativisation of embedded subjects in configurational languages appears to rest on such an assumption of an acquisition-based, core-to-periphery continuum in the linguistic competence of an adult language-user.

Summary

In conclusion, this subsection started out by examining an implicational universal (the '*pro*-drop parameter') involving a bidirectional correlation between the existence of optional VOS ordering and the possibility of relativising embedded subjects in configurational SVO languages like English and Romance. It was discovered that the previously discussed approach to explaining such implicational universals in grammar formalisms like CCG, whereby the formalism can generate the attested language types but not the unattested ones, is inadequate in this case.

It was noted that the claim implicit in Steedman (1996) that the CCG formalism 'captures' these aspects of the *pro*-drop parameter appears to rest on the intuition that the grammars of the attested language types are in some way *simpler* than the simplest grammars of the unattested types, where simplicity of a grammar can be defined informally as the cardinality of the lexicon, and more formally in terms of the minimum number of bits required to encode the grammar in the memory of some computing device. This kind of explanation relies on a model of child language acquisition with the following features: (a) language acquisition proceeds in stages of increasing cognitive expressivity e.g. the child does not attempt to internalise all relative clause types at once, but rather acquires matrix argument relatives before embedded argument rela-

tives, or subject relatives before object relatives, and so on; (b) at each stage of language acquisition, the child formulates the smallest grammar for the fragment of the adult language he/she has internalised; however (c) during later stages of language acquisition, the child cannot simply wipe the slate clean and start over, but rather builds conservatively on the knowledge acquired in previous stages. The precise details of the learning model are not at issue here. What is important to note is that, when used sensitively and with full awareness of the dynamic, conservative nature of the language acquisition process, the following methodological approach to explaining implication universals is a valid pursuit:

Grammar formalism F can be argued to explain implicational universal $P \subseteq Q$ if the following conditions hold:

1. there are equally short F-grammars for equivalent $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages

2. the shortest F-grammar for any $P \cap Q'$ language, should one exist, is longer than some F-grammar of one of its $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ equivalents

3.2.3 Basic word order and gapping again

Subsection 3.2.1 discussed the following implicational universal, deriving from the work of Ross (1970):

(3.93) All languages with a backward-gapping construction are verb-final.

This universal was formalised set-theoretically as follows: B-GAP $\subseteq$ OV. It was then argued that the CCG formalism predicts this universal, since (a) there are CCGs for B-GAP $\cap$ OV, B-GAP$'$ $\cap$ OV and B-GAP$'$ $\cap$ VO languages; but (b) there is no CCG for a B-GAP $\cap$ VO language.

Now, at this point, it should be noted that CCG is not the only formalism which predicts that backward-gapping languages are verb-final. Indeed, it should be clear that even the basic CFG formalism makes the same prediction, under the assumption that coordination is of adjacent, like constituents i.e. for any non-terminal X:

(3.94) $X \rightarrow X$ CONJ X

In other words, it is not possible to formulate a CFG for a B-GAP $\cap$ VO language, although there are CFGs for B-GAP $\cap$ OV, B-GAP$'$ $\cap$ OV and B-GAP$'$ $\cap$ VO languages.

On the other hand, from the perspective of the generalised, simplicity-based approach to linguistic explanation discussed above, the CCG and CFG formalisms *do* make different predictions about the link between gapping constructions and basic word order in human language. Consider the simplest CFG for an B-GAP′ ∩ OV language, which will contain the following two rules, in addition to the basic constituent coordination schema above:

(3.95) $\quad \mathsf{S} \rightarrow \mathsf{NP_{sbj}\ VP}$

$\quad \mathsf{VP} \rightarrow \mathsf{NP_{obj}\ V}$

This grammar suffices to generate sentential and verb-phrase coordinations, but not coordination of argument clusters since the relevant constituent does not exist. The CFG for the equivalent B-GAP ∩ OV language requires both of the rules in (3.95), *as well as* the following two additions:

(3.96) $\quad \mathsf{S} \rightarrow \mathsf{AC\ V}$

$\quad \mathsf{AC} \rightarrow \mathsf{NP_{sbj}\ NP_{obj}}$

Note that the non-terminal symbol 'AC' is meant to stand for 'argument cluster'. Since the argument cluster is treated as a constituent in this expanded grammar, it is possible to generate backward-gapping constructions by simply coordinating the argument clusters in surface syntax.

Now, recall that grammar formalism F can be said to predict implicational universal OV ⊆ B-GAP′ if: (a) there are equally short F-grammars for equivalent OV ∩ B-GAP′, VO ∩ B-GAP′ and VO ∩ B-GAP languages; and (b) the shortest F-grammar for any OV ∩ B-GAP language is *longer* than some F-grammar of one of its OV ∩ B-GAP′, VO ∩ B-GAP′ and VO ∩ B-GAP equivalents. It is thus clear that the CFG formalism incorrectly predicts (issues of expressivity notwithstanding) that OV ⊆ B-GAP′ i.e. verb-final languages *lack* a backward-gapping construction. This is because the simplest CFG for an OV ∩ B-GAP language, i.e. that which consists of the four rules in (3.95) and (3.96), can be readily converted into a smaller CFG of an equivalent OV ∩ B-GAP′ language, simply by dropping the two extra rules in (3.96).

On the other hand, the CCG formalism does not make this incorrect prediction. Recall from the discussion in subsection 3.2.1 that the CCG for an OV language *is* the CCG for an VO ∩ B-GAP language. The argument cluster constituent necessary for deriving backward-gapping constructions as constituent coordination is made available

by the flexible constituency resulting from the inclusion of the operations of forward-type-raising and harmonic composition in the formalism, and does not require independent stipulation as it does in CFG.

In conclusion, although the generative capacities of the CFG formalism and a restricted version of CCG (i.e. CCG-lite) are identical, it is not the case that the two formalisms always make exactly the same predictions about which human languages are marked and which are unmarked. The example in this section makes it clear that, for the CCG-lite formalism, the existence of backward-gapping constructions in verb-final languages is the unmarked option, whereas in CFG this appears to be a marked alternative. To repeat the mantra of categorial grammarians: just because two grammar formalisms have the same weak generative capacity, this doesn't mean that they are notational variants of the one theory.

3.3 Summary

This chapter considered the application of the CCG formalism of Steedman (2000) and Baldridge (2002) as a theory of human linguistic competence, in the sense of Chomsky (1965).

Section 3.1 examined the claim that CCG is a *restrictive* grammar formalism, in the sense that its generative capacity with respect to alphabet Σ is a proper subset of the context-sensitive languages over Σ, identical to the generative capacities of other restrictive formalisms such as Tree Adjoining Grammar and Linear Indexed Grammar. I discussed some of the implications of this claim for the theory of natural language competence. In particular, I reviewed a number of arguments from the CCG literature concerning the relationship between the generative capacity of CCG and predictions it makes about the permitted complexity of unbounded dependency constructions in human languages. It was concluded that: (a) CCG predicts the existence of unbounded *nested* dependencies in linguistic competence, in common with all other formalisms whose generative capacity is context-free or higher; (b) CCG also predicts the existence of unbounded *cross-serial* dependencies in natural language competence, unlike context-free formalisms which predict that such constructions must be bounded at best; but (c) CCG predicts that human language competence *lacks* doubly unbounded long-distance scrambling constructions, although a certain degree of such scrambling is

permitted. In particular, it was argued that CCG makes clear, falsifiable predictions about what kinds of unbounded dependency construction are involved in human language competence. These predictions are theorems of the foundational principles of the CCG formalism, and crucially do not require independent stipulation in the form of constraints on structural representations.

Section 3.2 turned to the subject of implicational universals of human language i.e. statements about non-random correlations between two or more independent grammatical properties. I discussed two particular implicational universals which the CCG formalism has been argued to explain, and attempt to elucidate exactly what methodological approach underlies the intuition that a grammar formalism explains an implicational universal. The first universal, discussed in subsection 3.2.1, involved the observation of Ross (1970) that the potential for gapping constructions in a language is determined by basic word order. I argued that the mode of explanation followed here was based on the assumption that formalism F explains meaningful implicational universal $P \subseteq Q$ if:

1. there are F-grammars for $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages
2. there is *no* F-grammar for any $P \cap Q'$ language

The second implicational universal, discussed in subsection 3.2.2, involves the familiar 'null subject parameter' (or '*pro*-drop' parameter) of Government and Binding Theory (Chomsky, 1981), which identifies a correlation in the family of Romance languages between the possibility that the subject may appear optionally in a post-verbal position and the ability of an embedded subject to be extracted across an overt complementiser. I argued that this universal is captured by the CCG formalism in a subtly different way, based on the assumption that formalism F explains meaningful implicational universal $P \subseteq Q$ if:

1. there are equally short F-grammars for equivalent $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages
2. the shortest F-grammar for any $P \cap Q'$ language is longer than some F-grammar for one of its $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ equivalents.

I thus tentatively proposed a generalised, intrinsic approach to linguistic explanation based on grammatical simplicity, where this concept can be defined either informally

in terms of the cardinality of the CCG lexicon, or more formally as the minimum number of bits required to encode the grammar in the memory of some computing device. Many details of the approach remain to be worked out, in particular the underlying acquisition model, and the corresponding continuum from core aspects of linguistic competence, which are acquired early, to the more peripheral aspects acquired later. However, I contend that, when used sensitively and with full awareness of both the dynamic, conservative nature of the language acquisition process and the expressiveness of human language, the generalised, simplicity-based approach to explaining implicational universals can be of use in formulating an explanatory theory of linguistic competence.

I conclude this chapter with two observations. Firstly, I hope that this chapter has made it perfectly clear that one of the main aims, possibly even *the* aim, of the CCG programme is to provide a theory of human language competence — in other words, a theory of 'universal grammar'. I don't deny that CCG has great advantages as an engineering tool for natural language processing applications, as recent work has made amply clear (Hockenmaier and Steedman, 2002), (Hockenmaier, 2003), (Clark and Curran, 2004), (Bos et al., 2004). However, I prefer to think that this is *because* rather than *in spite* of its essentially linguistic focus.

Second, although much of the explanatory force of the CCG formalism has involved the fact that it is a restrictive, mildly context-sensitive formalism, this is not the end of the matter. Many of the linguistic phenomena that CCG has been claimed to capture can be seen to involve more than just simple, black-and-white judgments about generative capacity. Rather, in common with other grammar formalisms purporting to be theories of human language competence, CCG is also intimately concerned with providing a theory of language 'markedness', i.e. of accounting for the fact that, other things being equal, certain types of conceivable human language appear to be more 'natural' than others.

Chapter 4

Some problematic data

Chapter 2 introduced the 'baseline' CCG formalism underlying Steedman (2000) and others, including its category notation and combinatory operations. Chapter 3 commenced the process of evaluating this baseline CCG as a theory of human linguistic competence. It was concluded that the restrictiveness of the CCG formalism, in particular the fact that its generative capacity is a proper subset of the class of context-sensitive languages, means that it makes a number of clear, falsifiable, and apparently correct predictions about what kinds of unbounded dependency constructions may occur in human linguistic competence. In addition, it was argued that CCG formally predicts some well-known implicational universals of human language. I posited that grammar formalism F can be argued to predict implicational universal $P \subseteq Q$, where P and Q denote subclasses of human language, if the following two conditions hold:

1. there are equally short F-grammars for equivalent $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages

2. either there is *no* F-grammar for an $P \cap Q'$ language, or the *shortest* F-grammar for any $P \cap Q'$ language is *longer* than some F-grammar for one of its $P \cap Q$, $P' \cap Q$ or $P' \cap Q'$ equivalents

A similar means was proposed for bidirectional universals of the form $P = Q$. It was argued that, provided we remain sensitive to the dynamic, conservative nature of the language acquisition process, this general methodological approach both can be and has been highly serviceable. I demonstrated that this general methodology underlies claims that the baseline CCG formalism predicts the following two implicational universals:

1. languages with backward-gapping constructions are verb-final
2. those configurational SVO languages which allow subject extraction from embedded complement clauses also have free subject inversion, and vice versa

This chapter continues the process of evaluating the baseline CCG formalism as a theory of human linguistic competence. Here the focus is on implicational universals which are *not* straightforwardly captured by baseline CCG in terms of an acquisition-based preference for shorter grammars, even though we have some intuitive sense that the universals are based on some notion of grammatical economy. The existence of this problematic data will provide us with the motivation for generalising the baseline CCG category notation in the remainder of this thesis.

This chapter will proceed as follows:

Section 4.1 discusses a well-known implicational universal from Greenberg (1963), which relates the existence of prepositions or postpositions in the language with its basic clausal ordering, in particular whether direct objects tend to follow or precede a transitive verb. I argue that the baseline CCG formalism from Chapter 2 does not formally predict this universal in terms of a preference for shorter grammars, since the baseline lexicons for languages of the unattested types are not noticeably longer than those of their equivalent languages from the attested types. I then point out that the universal does appear to be motivated by considerations of grammar simplicity, and review a couple of proposals from the generative grammar literature which have tried to explain it in these terms.

Section 4.2 turns to a lesser-known implicational universal deriving from the work of Travis (1984) and Travis (1989). This concerns possible interactions between verb-object and verb-modifier serialisation in human languages. Again, I argue that the baseline CCG formalism does not formally predict this universal in terms of a preference for shorter grammars, although various proposals from the literature demonstrate that it is possible to augment the CFG formalism in such a way that this universal is explained by considerations of grammar-length.

4.1 Clausal order, prepositions and postpositions

In section 3.2, we already came across the following implicational universal from Greenberg (1963):

> Languages with dominant VSO order are always prepositional.

This is Greenberg's universal number 3. Compare it with number 4:

> With overwhelmingly greater than chance frequency, languages with normal SOV order are postpositional.

In subsequent work, for example Vennemann (1972) and Lehmann (1973), these universals were generalised into bidirectional universals. Lehmann argued that the position of the subject is not particularly relevant in discussions of cross-linguistic word order, and thus distinguished between 'VO' languages and 'OV' languages depending on whether or not the direct object tends to follow or precede its transitive verb. This distinction forms the basis of two bidirectional universals which can be stated as follows:

1. VO languages usually have prepositions but not postpositions, and languages with prepositions but not postpositions are usually VO

2. OV languages usually have postpositions but not prepositions, and languages with postpositions but not prepositions are usually OV

In section 3.2 I demonstrated how we can formalise such statements set-theoretically. The first step is the define the domain of languages which is under discussion. In this case, we are only interested in the class of languages where there is a basic ordering of direct objects with respect to their transitive verbs in *matrix declarative* clauses, i.e. either the object overwhelmingly precedes the verb or it overwhelmingly follows it. Thus, for example Turkish counts as part of our domain, since although VO ordering is *possible* in declarative clauses, it is a marked alternative. In the same way, Russian is deemed to be a VO language. Note also that according to this definition, German is a VO language — although OV orders are possible, they are either found in embedded clauses and are hence irrelevant, or are a marked, topicalised alternative.

A second restriction on the domain of languages we are interested in is that only languages which have a class of *adpositions* are to be included. Thus, most Australian

languages lie outwith the domain, lacking as they are in both prepositions and postpositions (Comrie, 1989). Third, only languages which have a demonstrable preference for either prepositions or postpositions will be admitted. Thus, we include Persian, which has lots of prepositions but just the one postposition, but exclude Estonian, which has significant numbers of both.

The next step in formalising an implicational universal is to formulate two independent binary partitions of the domain. In the current example, the first partition is as follows:

VO the direct object follows its head transitive verb in unmarked matrix declarative clauses

OV the direct object precedes its head transitive verb in unmarked matrix declarative clauses

Note that these two classes are mutually exclusive and exhaust the domain. Thus, for example:

(4.1) $\text{VO} = \{\text{English}, \text{Irish}, \text{German}, \text{Tagalog}, \text{French}, \text{Italian}, \text{Chinese}, \ldots\}$

$\text{OV} = \{\text{Japanese}, \text{Turkish}, \text{Tamil}, \text{Persian}, \text{Hindi}, \text{Korean}, \text{Kpelle}, \ldots\}$

Although both French and Italian require that pronominal direct objects be proclitics, they are still judged to be VO, since only full direct objects are relevant. The second binary partition of the domain is the following:

PO lots of prepositions but few if any postpositions

OP few if any prepositions but lots of postpositions

For example:

(4.2) $\text{PO} = \{\text{English}, \text{Irish}, \text{German}, \text{Tagalog}, \text{French}, \text{Italian}, \text{Chinese}, \text{Persian}, \ldots\}$

$\text{OP} = \{\text{Japanese}, \text{Turkish}, \text{Tamil}, \text{Hindi}, \text{Korean}, \text{Kpelle}, \ldots\}$

Once we have our two independent binary partitions of the domain, the next step is to distribute the languages into the four mutually exclusive, exhaustive complex subclasses:

1. $\text{VO} \cap \text{PO} = \{\text{English}, \text{Irish}, \text{German}, \text{Tagalog}, \text{French}, \text{Italian}, \text{Chinese}, \ldots\}$
2. $\text{VO} \cap \text{OP} = \emptyset$
3. $\text{OV} \cap \text{PO} = \{\text{Persian}\}$
4. $\text{OV} \cap \text{OP} = \{\text{Japanese}, \text{Turkish}, \text{Tamil}, \text{Hindi}, \text{Korean}, \text{Kpelle}, \ldots\}$

Since there are lots of attested languages in $\text{VO} \cap \text{PO}$ and $\text{OV} \cap \text{OP}$, none in $\text{VO} \cap \text{OP}$, and just a couple in $\text{OV} \cap \text{PO}$, it is reasonable to conclude Vennemann's and Lehmann's bidirectional implicational universal as follows:

(4.3) $\text{VO} \approx \text{PO}$

Having formulated the implicational universal, we now have to determine whether or not it is captured by the baseline CCG formalism from Chapter 2. To do this, we need to take each of the four subclasses of our domain in turn and try to formulate the simplest, linguistically intuitive CCG lexicons for the relevant parts of each language-type.

Let's start by making a couple of assumptions about these lexicons, in order to make sure we are always comparing like with like. First of all, transitive verbs will always be assigned to categories whose result is denoted by the category symbol 'VP' and whose argument is an 'NP', i.e. categories of the general form $\mathsf{VP}|\mathsf{NP}$. Secondly, adpositions will be assigned to categories whose result is denoted by the category symbol 'PP' and whose argument is an 'NP' i.e. categories of the general form $\mathsf{PP}|\mathsf{NP}$. Neither of these assumptions should be controversial, although the 'VP' symbol should probably be considered as a macro for something like $\mathsf{S}|\mathsf{NP}$.

Based on these assumptions, the simplest baseline CCG sublexicons for the relevant fragments of typical languages of each of the four language types are as presented in Table 4.1. Recall that we can argue that the CCG formalism formally predicts bidirectional implicational universal $\text{VO} \approx \text{PO}$ if the following two conditions hold:

1. there are equally short CCGs for equivalent $\text{VO} \cap \text{PO}$ and $\text{OV} \cap \text{OP}$ languages
2. either there are *no* CCGs for $\text{VO} \cap \text{OP}$ and $\text{OV} \cap \text{PO}$ languages, or the *shortest* CCG for them is *longer* than some CCG for one of their $\text{VO} \cap \text{PO}$ or $\text{OV} \cap \text{OP}$ equivalents

The first condition definitely holds, as is evident from the first and last rows in Table 4.1. How about the second condition? The first way in which a language-type can

VO ∩ PO	love, crucify, devour ⊢ VP/NP beyond, without, during ⊢ PP/NP
VO ∩ OP	love, crucify, devour ⊢ VP/NP beyond, without, during ⊢ PP\NP
OV ∩ PO	love, crucify, devour ⊢ VP\NP beyond, without, during ⊢ PP/NP
OV ∩ OP	love, crucify, devour ⊢ VP\NP beyond, without, during ⊢ PP\NP

Table 4.1: Baseline CCG lexical fragments for languages in VO ∩ PO, VO ∩ OP, etc.

be deemed 'unnatural' by a formalism is in the case that its language are simply ungeneratable by its grammars. This is not so here, since rows two and three of Table 4.1 contain CCGs for VO ∩ OP and OV ∩ PO languages.

A formalism can also predict that a given language-type is unnatural if grammars of languages of that type are significantly longer than grammars of equivalent languages from other types. Unfortunately, looking closely at the mini lexicons in Table 4.1, there is absolutely no sense in which we can state that those for the VO ∩ OP or OV ∩ PO languages are 'longer' than those of the VO ∩ PO and OV ∩ OP equivalents. First of all, note that each lexicon contains exactly the same number of relevant lexical items i.e. *one* for each transitive verb, and *one* for each adposition. Secondly, note that each lexicon contains exactly the same number of symbol-tokens — *four* for each transitive verb and adposition. Thirdly, each lexicon contains exactly the same number of category symbol-types i.e. VP, PP and NP. Thus, according to the definition given in (3.89) on page 113 for the 'bit-size' of a CCG, all four lexicons will have exactly the same number of bits in their smallest encoding.

Thus, it appears that we must conclude that the baseline CCG formalism from chapter 2 does *not* formally predict the implicational universal VO ≈ PO, in terms of an acquisition-based preference for shorter grammars over longer ones, since there is no sense in which CCG lexicons for languages of the unattested or rare types can be argued to be 'longer' than the equivalent language of the attested types. Note that this is also the case if we assume that noun phrases are unsaturated expressions taking

transitive verbs or adpositions as their argument.

There is however one caveat that must be considered here. I pointed out above that all four lexicons in Table 4.1 contain exactly the same number of saturated category symbol-types. However, the careful reader may have noted that the lexicons in the two middle rows actually contain one more general symbol-type than the top and bottom rows — the lexicons for the VO ∩ PO and the OV ∩ OP languages require just the one slash, respectively / and \; however, the lexicons for the VO ∩ OP and the OV ∩ PO languages require *both* slashes. This observation can form the basis of a more subtle evaluation measure for CCG lexicons, as follows.

Let us assume that a CCG over alphabet Σ is actually an ordered 4-tuple $\langle A, S, \Delta, L \rangle$, where: (a) as before A is an alphabet of saturated category symbols and $S \in A$; (b) Δ is a non-empty subset of $\{/, \backslash, |\}$; and (c) L is a finite mapping from Σ to category descriptions over A which only use the slashes in Δ. In other words, every CCG specifies a particular set of slashes and only these can be used in its lexicon. The previous definition of the number of symbol-types in a CCG in subsection 3.2.2 was $|\Sigma| + |A| + 5$, where the five miscellaneous symbols were the two brackets and the three slashes. The number of distinct symbol-tokens in a CCG grammar $\langle A, S, \Delta, L \rangle$ over alphabet Σ can now be redefined as $|\Sigma| + |A| + |\Delta| + 2$. In terms of this redefined notion of a CCG, it is easy to formulate a length measure such that lexicons which make use of one slash are, other things being equal, shorter than those which require two or three. In addition, if we weight this measure so that the cardinality of Δ has a disproportionately strong effect on the 'size' of a grammar, then grammars of VO ∩ OP and OV ∩ PO languages can be deemed to be much 'bigger' than those of VO ∩ PO and OV ∩ OP ones.

Unfortunately, there are a number of problems with this approach, all of which revolve around the fact that any serious CCG of a large fragment of any human language is going to have both / and \ categories at least somewhere in its lexicon. Firstly, many VO ∩ PO languages have at least one exceptional postposition, for example *ago* in English. Secondly, many VO ∩ PO languages have \ categories somewhere in the lexicon — for example, German non-finite and embedded finite transitive verbs are assigned to the category VP\NP. Thirdly, there is a tendency for modifiers in VO ∩ PO languages to *follow* their heads and in OV ∩ OP to *precede* them. In categorial grammars, modifiers are assumed to be unsaturated expressions taking the modified expressions as an argument.[1] In other words, premodifiers and postmodifiers are respectively assigned

[1] Indeed this is a forced assumption, since modifiers are optional and unrestricted in number.

to categories described as X/X and $X\backslash X$. Thus, it is clear that CCG lexicons for pretty much all the languages in our domain are going to contain large numbers of both / and \ expressions, and thus any approach which prefers grammars containing fewer slash-types will fail to capture the implicational universal VO $\approx$ PO.

Summing up the discussion so far, I started out with the bidirectional statistical universal from Vennemann (1972) and Lehmann (1973) equating VO languages with prepositional languages, and formalised it as VO $\approx$ PO, where VO is the set of all human languages where the direct object generally follows the verb in a matrix declarative clause and PO is the set of languages with lots of prepositions but few if any postpositions. I demonstrated that the baseline CCG formalism from chapter 2 does not formally predict this universal, in terms of a cross-linguistic tendency for shorter lexicons, at least assuming any reasonably simple kind of evaluation measure.

However, it does appear that there *are* grounds for believing that the VO $\approx$ PO universal is indeed motivated by considerations of grammatical economy. Traditionally, it is assumed that a transitive verb is the *head* of its clause and that the direct object is hence one of its *dependents*. In addition, the preposition is universally considered to be the head of a prepositional phrase, with its collocated noun phrase playing the role of dependent of the head. With this in mind, let's turn back to the four language-types listed in Table 4.1. Recall our assumptions that transitive verbs are universally assigned to a category denoted as $\mathsf{VP}|\mathsf{NP}$ and adpositions to one denoted as $\mathsf{PP}|\mathsf{NP}$. In this case, it is clear that the relevant part of the lexicon of a VO $\cap$ PO language can be captured using a *single* descriptive statement, i.e. *dependents follow their heads*. In the same way, the behaviour of transitive verbs and adpositions in OV $\cap$ OP languages can be handled by just the one descriptive statement: *dependents precede their heads*. However, the VO $\cap$ OP and OV $\cap$ PO languages cannot be so easily captured. Each of these requires *two*, slightly more complex descriptive statements to capture the behaviour of verbs and adpositions:

VO $\cap$ OP dependents follow verbal heads; dependents precede adpositional heads

OV $\cap$ PO dependents precede verbal heads; dependents follow adpositional heads

Thus, assuming a more flexible metalanguage for lexical description, which can for example distinguish between heads and dependents, arguments and modifiers, we might be able to come up with a CCG-based formalism which formally captures the VO $\approx$ PO

universal in terms of a basic preference for shorter grammars. Note that this kind of terminology is still sufficient to capture the behaviour of modifiers, since modifiers are still dependents — see Baldridge and Kruijff (2003) for a way of capturing the head-dependent distinction in terms of a secondary dimension on slash modalities.

This intuition, that the grammars of VO ∩ PO and OV ∩ OP languages are in some sense *simpler* than those of VO ∩ OP and OV ∩ PO ones, underlies many of the proposed generalisations of the CFG formalism familiar from the formal linguistics literature. Recall that the CFG formalism itself has little to say about the VO ≈ PO implicational universal. The relevant constructions in a VO ∩ PO language require *two* distinct CF-rules to be successfully described:

(4.4) $\mathsf{VP} \rightarrow \mathsf{V}\ \mathsf{NP}$

$\mathsf{PP} \rightarrow \mathsf{P}\ \mathsf{NP}$

However, no more rules are needed to capture a VO ∩ OP language:

(4.5) $\mathsf{VP} \rightarrow \mathsf{V}\ \mathsf{NP}$

$\mathsf{PP} \rightarrow \mathsf{NP}\ \mathsf{P}$

Thus it should be clear that the unadorned CFG formalism does *not* formally predict that VO ≈ PO. One way of generalising the basic CFG formalism so as to capture this universal in terms of number of rules involves what has become known as '$\bar{\text{X}}$-theory' (Chomsky, 1972), (Jackendoff, 1977). This assumes that non-terminal symbols are bipartite structures, consisting of a major category symbol like 'V', 'N', 'A' or 'P', together with a bar-level. More sophisticated versions assume that the major category symbols themselves are feature complexes based on the two boolean features ±V and ±N. In what follows I assume just two bar levels: '0' denotes a minimal projection and '1' a maximal projection.

A second assumption in $\bar{\text{X}}$-theory is that CF-rules can be 'schematised' using variables over major category symbols. Thus, in the grammar for a VO ∩ PO language, the two distinct rules in (4.4) can be replaced by just the one $\bar{\text{X}}$-theoretic rule:

(4.6) $X^1 \rightarrow X^0\ \mathsf{N}^1$

However, an $\bar{\text{X}}$-theoretic grammar for the corresponding VO ∩ OP language still requires two distinct rules, since no schematisation is possible:

(4.7) $\mathsf{V}^1 \rightarrow \mathsf{V}^0\ \mathsf{N}^1$

$\mathsf{P}^1 \rightarrow \mathsf{N}^1\ \mathsf{V}^0$

Thus, the effect of assuming an $\bar{X}$-theoretic perspective on the CFG formalism is that it allows implicational universals like VO $\approx$ PO to be captured formally in terms of an preference for grammars with fewer rules.

This basic $\bar{X}$-theoretic apparatus can be further elaborated by means of 'linear precedence' statements (Gazdar et al., 1985). The idea here is that all four languages in Table 4.1 will have an invariant rule component based on the following CF-style rule schema, where linear ordering of the symbols on the right hand-side is underspecified:

(4.8) $X^1 \rightarrow X^0, Y^1$

Information about subcategorisation can then be transfered to the lexicon, by means of selectional restrictions. In other words, the lexical entry of a transitive verb such as *devour* will state that its major category is V and that it subcategorises for just the one complement of major category N. From this perspective, a grammar consists of just two parts: (a) a lexicon; and (b) a set of linear precedence (LP) statements regulating the ordering of sister nodes in a phrase structure tree. For example, the grammar of a VO $\cap$ PO language will require just the one LP statement to capture the behaviour of transitive verbs and adpositions:

(4.9) $X^0 \prec Y^1$

This statement says that a minimal projection always precedes any sisters which are maximal projections, which is almost the same thing as saying heads precede dependents. On the other hand, the grammar of the corresponding VO $\cap$ OP language will need *two* distinct rules:

(4.10) $\mathsf{V}^0 \prec Y^1$
$Y^1 \prec \mathsf{P}^0$

In other words, transitive verbs precede their sisters but adpositions follow theirs. Thus it is clear that, just like with the $\bar{X}$-theoretic generalisation of CFG previously discussed, the effect of adding LP statements is that it allows implicational universals like VO $\approx$ PO to be captured formally in terms of an preference for grammars with fewer statements.

Summing up, this section introduced a bidirectional implicational universal VO $\approx$ PO deriving from the generalisation by Vennemann (1972) and Lehmann (1973) of some

original observations in Greenberg (1963). I argued that this universal is not formally predicted by the baseline CCG formalism defined in chapter 2 since baseline CCG lexicons of the two unattested/rare language-types do not appear to be any longer than those of the two frequently attested types. I then pointed out that the VO $\approx$ PO universal does appear to be motivated by considerations of grammatical economy, since given an appropriate metalanguage involving such notions as 'head' and 'dependent', grammars of the attested types require *fewer* descriptive statements than those of the unnatested types. I discussed how this insight has led to generalisations of the CFG formalism which predict that VO $\approx$ PO in terms of a preference for grammars with fewer rules/statements.

This is a good point at which to say something about the theory proposed in Flynn (1985), briefly introduced in subsection 2.5.9. Flynn's categorial framework is motivated by what he sees as the failure of $\bar{\text{X}}$-theory to provide a genuine explanatory account of word order univerals — when the alphabet of features is made rich enough to account for the full range of cross-categorial word order generalisations found in English, then the theory ends up making absolutely no predictions at all about word order in human language in general. Flynn's answer is to propose a categorial formalism based on *unordered* unsaturated categories, and where word order is governed by language-particular, cross-categorial 'word order conventions'. Provided that we can find a metalanguage which allows word order conventions to be formulated for VO $\cap$ PO and OV $\cap$ OP languages but not for VO $\cap$ OP or OV $\cap$ PO ones, then Flynn argues that the formalism successfully explains that VO $\approx$ PO.

One problem with Flynn's proposal is, of course, that it fails to account for the fact that many word order universals are tendencies rather than absolutes. However, this may be rectified by some kind of simplicity measure on word order conventions according to which rare language-types are governed by more complex word order conventions than common types. On the other hand, the reason I have chosen to reject Flynn's approach as the basis of a theory of the CCG lexicon is more fundamental — as mentioned in subsection 2.5.9, the word order conventions must be construed as constraints on derivations rather than on categories. This implies a parsing model which remembers and computes whole derivations rather than just form-category pairs, and as such is inconsistent with the underlying principles on which the CCG formalism is grounded.

4.2 Direct objects, indirect objects and modifiers

Travis (1989) contains a discussion of a range of data involving possible interactions between verb-object and verb-modifier ordering in human languages. This data can be easily reinterpreted as implicational universals in the usual way.

Let us start by isolating the domain of languages covered by one of these universals. Firstly, only those languages which have basic VO ordering in matrix declarative clauses are to be included. In other words, in our domain all languages are such that the verb overwhelmingly precedes its direct object, for example English, Irish, German, French, Italian and Chinese. Languages like Japanese, Korean, Tamil, Persian and Hindi are thus excluded. Secondly, we include only those languages which have lots of prepositions but few if any postpositions. From this perspective, it is clear that our domain is a subset of the VO $\cap$ PO languages discussed in section 4.1. Thirdly, our domain includes only those languages where prepositional phrases can act as both 'arguments' of verbs (as in English 'John depends *on Mary*') as well as 'modifiers' of verbs (e.g. 'John sees Mary *on Sundays*'). In the former case, the prepositional phrase can be said to act as the 'indirect object' of the verb, as opposed to the direct object which is universally a nominal argument. The first binary partition of this domain is as follows:

VI an indirect object *follows* its head verb in unmarked matrix declarative clauses

IV an indirect object *precedes* its head verb in unmarked matrix declarative clauses

Languages belonging to the class VI include English and Chinese. The following Chinese data, as with all the examples in this section, comes from Travis (1989):

(4.11) tā mài gěi wǒ chēzi le.
he sell to me car PAST
He sold a car to me.

This example shows that both direct and indirect objects follow the verb in Chinese. On the other hand, Travis cites 'Future Chinese' as an example of a language in IV i.e. where direct objects follow the verb but indirect objects precede.

The second independent binary partition of the domain concerns whether prepositional phrase adverbials are pre- or post-modifiers:

VX a prepositional adverbial *follows* its head verb in unmarked matrix declarative clauses

XV a prepositional adverbial *precedes* its head verb in unmarked matrix declarative clauses

Travis cites English as an exemplar VX language, whilst Chinese and 'Future Chinese' are XV ones. Here is an example from Chinese:

(4.12) tā gěi wǒ mài le chēzi le.
he for me sell PAST car PAST
He sold a car for me.

In this sentence, the prepositional phrase 'gěi wǒ' functions as a beneficiary adverbial rather than as an argument as in example (4.11). Based on these two independent binary partitions of the domain, we get the following four complex subclasses, along with Travis' claims about which types are attested and which are not:

1. $\text{VX} \cap \text{VI} = \{\text{English}, \ldots\}$
2. $\text{VX} \cap \text{IV} = \emptyset$
3. $\text{XV} \cap \text{VI} = \{\text{Chinese}, \ldots\}$
4. $\text{XV} \cap \text{IV} = \{\text{Future Chinese}, \ldots\}$

Travis thus claims that $\text{VX} \cap \text{IV}$ is an 'impossible' language-type, while the other three are all 'possible' language-types. Moreover, since there are many more attested languages in $\text{VX} \cap \text{VI}$ than in either $\text{XV} \cap \text{VI}$ or $\text{XV} \cap \text{IV}$, Travis concludes that the former is the 'unmarked' option.

Despite the obvious flaws in Travis' approach to data collection, I will follow Fodor and Crain (1990) in accepting her conclusions, in the hope that further data will clarify matters. The obvious implicational universal implied by Travis' analysis is the following:

(4.13) $\text{VX} \subseteq \text{VI}$

In other words, if prepositional adverbials follow the verb in VO language *L* then indirect objects also follow the verb in *L*. Having formulated this implicational universal,

we now have to determine whether or not is is captured by the baseline CCG formalism from Chapter 2. To do this, we again take each of the four subclasses of our domain in turn and try to formulate the simplest, linguistically intuitive CCG lexicons for the relevant parts of each language-type.

Let's start again by making a couple of assumptions about these lexicons, in order to make sure we are always comparing like with like. First of all transitive verbs will always be assigned to categories of the form VP/NP, since our domain is a subclass of the VO languages. Secondly, verbs which take indirect objects will be assigned to categories denoted by $\mathsf{VP}|\mathsf{PP}$, with the slash depending on whether they are in VI or IV. Thirdly, prepositions will be assigned to *two* distinct categories. The first of these is for when they head an indirect object and is denoted as PP/NP. The second is for when they head an adverbial, in which case the relevant category is denoted as $(\mathsf{VP}|\mathsf{VP})/\mathsf{NP}$, where again the choice of slash depends on whether the language is in VX or XV. Again, none of these assumptions should be controversial to a categorial grammarian.

Based on these assumptions, the simplest baseline CCG sublexicons for the relevant fragments of typical languages of each of the four language types are as presented in Table 4.2. Recall that we can argue that the CCG formalism formally predicts implicational universal VX $\subseteq$ VI if the following two conditions hold:

1. there are equally short CCGs for equivalent VX $\cap$ VI, XV $\cap$ VI and XV $\cap$ IV languages

2. the *shortest* CCG for a VX $\cap$ IV language is *longer* than some CCG for one of their VX $\cap$ VI, XV $\cap$ VI or XV $\cap$ IV equivalents

Since all four lexicons in Table 4.2 contain exactly the same number of symbol-tokens and symbol-types, it is clear that baseline CCG does not formally predict that VX $\subseteq$ VI, in terms of grammar-shortness. Since all four lexicons include both $/$ and $\backslash$ categories, evaluating grammars based on the number of slashes they use is of no use in this case. In addition, assuming that direct and indirect objects are unsaturated expressions taking verbs and prepositions as arguments makes no difference to the end result.

However, just like with the VO $\approx$ PO universal discussed in section 4.1, it does appear that there are grounds for believing that the VX $\subseteq$ VI universal is motivated by considerations of grammatical economy. Consider the explanation proposed by Travis (1989), which is couched within the 'Government and Binding'-era Principles

VX$\cap$VI	transitive verbs $\vdash \mathsf{VP}/\mathsf{NP}$ other verbs $\vdash \mathsf{VP}/\mathsf{PP}$ prepositions $\vdash \mathsf{PP}/\mathsf{NP}$ prepositions $\vdash (\mathsf{VP}\backslash\mathsf{VP})/\mathsf{NP}$
VX$\cap$IV	transitive verbs $\vdash \mathsf{VP}/\mathsf{NP}$ other verbs $\vdash \mathsf{VP}\backslash\mathsf{PP}$ prepositions $\vdash \mathsf{PP}/\mathsf{NP}$ prepositions $\vdash (\mathsf{VP}\backslash\mathsf{VP})/\mathsf{NP}$
XV$\cap$VI	transitive verbs $\vdash \mathsf{VP}/\mathsf{NP}$ other verbs $\vdash \mathsf{VP}/\mathsf{PP}$ prepositions $\vdash \mathsf{PP}/\mathsf{NP}$ prepositions $\vdash (\mathsf{VP}/\mathsf{VP})/\mathsf{NP}$
XV$\cap$IV	transitive verbs $\vdash \mathsf{VP}/\mathsf{NP}$ other verbs $\vdash \mathsf{VP}\backslash\mathsf{PP}$ prepositions $\vdash \mathsf{PP}/\mathsf{NP}$ prepositions $\vdash (\mathsf{VP}/\mathsf{VP})/\mathsf{NP}$

Table 4.2: Baseline CCG lexical fragments for languages in VX$\cap$VI, VX$\cap$IV, etc.

and Parameters Theory of Chomsky (1981). Travis proposes two binary word order parameters to supplement the familiar 'head parameter':

theta assignment parameter theta roles are assigned to the left/right

case assignment parameter case is assigned to the left/right

Note that: (a) direct and indirect objects are assigned theta roles by their head verb, but adverbials are not; (b) noun phrases need to be assigned case, but prepositional phrases do not. Travis also proposes that not every parameter need be set by a grammar. In other words, a grammar may contain a setting for both of these parameters, only one of them, or even neither. Travis proposes the following grammars for the four relevant language-types:

VX∩VI head-first

VX∩IV head-first, theta-left, case-right

XV∩VI head-last, theta-right

XV∩IV head-last, case-right

The grammar for a VX∩IV language is significantly *longer* than grammars for languages of the other three types, since it requires all three word order parameters to be set. Thus, Travis' parametric theory can be argued to successfully predict that VX ⊆ VI. Indeed, Travis' system also successfully captures the fact that VX∩VI languages are considered to be unmarked with respect to the other three types, since word order behaviour in these languages can be captured with just the one parameter setting. The argumentation in each case involves considerations of the *length* of a grammar, here defined as the number of parameter settings.

Fodor and Crain (1990) take Travis' analysis a step further, removing the parameters whilst retaining the overall economy flavour. They demonstrate that an augmented CFG formalism also can be argued to explain the VX ⊆ VI universal. Note first of all that the CFG formalism itself does not make the correct prediction. For example, here is the CFG for a simple VX∩VI language:

(4.14) $\mathsf{VP} \rightarrow \mathsf{V_{tr}}\ \mathsf{NP}$

$\mathsf{VP} \rightarrow \mathsf{V_{intr}}\ \mathsf{PP}$

$\mathsf{VP} \rightarrow \mathsf{VP}\ \mathsf{PP}$

The CFG for the equivalent VX∩IV language contains exactly the same number of similarly sized rules:

(4.15) $\mathsf{VP} \rightarrow \mathsf{V_{tr}}\ \mathsf{NP}$
$\mathsf{VP} \rightarrow \mathsf{PP}\ \mathsf{V_{intr}}$
$\mathsf{VP} \rightarrow \mathsf{VP}\ \mathsf{PP}$

Note that there is no sense in which the latter CFG is 'shorter' than the former. However, recall the augmented CFG formalism discussed in section 4.1, involving the following kind of unordered, $\bar{\text{X}}$-theoretic CF rule schemata:

(4.16) $X^1 \rightarrow X^0, Y^1$
$X^1 \rightarrow X^1, Y^1$

This schema is explicitly endocentric, in that the 'head' of a phrasal type is always represented by the variable X. Note also that 'arguments' and 'modifiers' are structurally distinct, being defined as sisters of X^0 and X^1 respectively. Assuming an appropriate lexicon, a grammar can thus be represented as a set of linear precedence statements. Fodor and Crain (1990) present essentially the following grammars of each of the four relevant language-types:

VX∩VI a head precedes its dependents

VX∩IV a head precedes its nominal arguments; prepositional arguments precede their heads; a head precedes its modifiers

XV∩VI a head precedes its arguments; modifiers precede their heads

XV∩IV a head precedes its nominal dependents; prepositional dependents precede their heads

Note that the grammar for a VX∩IV language involves *three* separate linear precedence statements, whereas grammars of the other three language-types require no more than *two*. Thus, it is clear that this particular augmented CFG formalism predicts formally that VX ⊆ VI since, other things being equal, grammars of the unattested language-type will be longer than those of the attested types. In addition, this formalism also predicts that VX∩VI languages represent the unmarked case, since grammars of this kind of language require just the one linear precedence statement to account for

word order in the verb phrase. Thus, both Travis (1989) and Fodor and Crain (1990) agree that an augmented CFG formalism can explain the VX $\subseteq$ VI universal in terms of grammar-length, although they disagree on the necessity or desirability of incorporating parameters in such a system.

Summing up, this section introduced an implicational universal deriving from work by Travis (1989). This universal can be stated as VX $\subseteq$ VI, where the domain is the class of VO languages, VX is the subset of the domain where prepositional adverbials *follow* their heads, and VI is the subset of the domain where indirect objects follow their heads. Note that Travis includes a parallel set of data for OV languages, which I have omitted for reasons of simplicity. I argued that this universal is not formally predicted by the baseline CCG formalism defined in chapter 2 since baseline CCG lexicons of the unattested language-type do not appear to be any longer than those of the three frequently attested types. I then pointed out that the VX $\subseteq$ VI universal does appear to be motivated by considerations of grammatical economy, since given an appropriate metalanguage involving either Travis' idea of optional parameters, or such notions as 'head', 'dependent', 'argument' and 'modifier', grammars of languages of the attested types require *fewer* statements than those of the unattested type.

4.3 Summary

This chapter has discussed the following two implicational universals:

1. VO languages usually have prepositions but not postpositions, and languages with prepositions but not postpositions are usually VO

2. in VO languages, if prepositional adverbials follow the verb, then indirect objects also follow the verb

I have argued that neither of these universals is predicted formally by the baseline CCG formalism defined in chapter 2, since there is no sense in which we can state that baseline CCG lexicons for languages of the unattested types are in any way 'longer' than those for equivalent languages of one of the attested types. However, for both these universals, I claimed that considerations of grammar-length can play a role in explaining them, assuming we have an expressive enough metalanguage capable of making statements such as the following:

- a head precedes its dependents
- a prepositional head follows its dependents
- a head precedes its nominal arguments
- a verbal head precedes its prepositional modifiers

Indeed, I reviewed proposals from the generative grammar literature involving attempts to augment the basic CFG formalism so as to come up with a theory which predicts these implicational universals in terms of an acquisition-based preference for shorter grammars.

The work reported in this thesis proceeds from a conviction that flexible categorial grammar formalisms such as CCG are *inherently better* theories of linguistic competence than phrase structure-based formalisms, at least in the current state-of-the-art. This is motivated by a number of considerations, all of which are a result of the derivational flexibility inherent in any (associative) CCG. Firstly, the fact that such unboundedly large strings as *John loves*, *John said that Bill loves*, *John said that Bill thinks that Tom loves*, etc. are just normal constituents, assigned to a normal category and receiving a semantic interpretation in a purely compositional way, means that a CCG account of unbounded extraction constructions such as relativisation and topicalisation is particularly simple, requiring only that the 'extracted' expression be assigned to a single additional lexical category. There is thus no need for additional mechanisms such as grammatical transformations, feature passing, or any of the other ways people have suggested supplementing CFGs to handle these constructions.

Secondly, the derivational flexibility of CCG means that the analysis of coordination constructions is considerably simplified. Pretty much any contiguous substring of a sentence can be considered to be a first-class constituent with its own category and semantic interpretation. One result of this is that, assuming a straightforward coordination rule which conjoins adjacent constituents of the same category, a CCG can directly account for such examples as *John loves and Bill hates Mary* or *John gave Mary a dog and Bill a cat*, which have customarily been dismissed in the generative grammar literature as 'non-constituent' coordination. Of course, this has implications for the theory of linguistic competence as well — in subsection 3.2.3 I argued that CCG makes more accurate predictions about the existence of gapping constructions

in human languages than does phrase structure-based formalisms, since in the latter, gapping constructions are necessarily marked alternatives.

Thirdly, another consequence of the derivational flexibility inherent in a CCG grammar involves the formulation of a model of how humans go about parsing utterances of their language. There is abundant evidence that semantic interpretations are built up incrementally, left-to-right, on word by word basis. The fact that almost all prefixes of a sentence are first-class constituents in CCG means that such an incremental processor can be implemented directly, without the need for any extragrammatical apparatus to construct interpretations for non-constituents.

Thus, all things considered, it is clear that abandoning the CCG formalism as a theory of linguistic competence, just because its baseline category notation leads to incorrect predictions about the implicational universals discussed in this chapter, would be rather premature. One alternative is to attempt to deconstruct the baseline category notation so as to come up with a CCG formalism which makes the *correct* predictions about this kind of universal in terms of an acquisition-based preference for shorter lexicons. This will be the subject of the remainder of this thesis.

Chapter 5

Type-hierarchical CCG

Chapter 2 introduced a baseline CCG formalism, founded on a model-theoretic conception of the theory of category notation, distinguishing between underlying category models and category descriptions. Chapters 3 and 4 then evaluated this baseline formalism as a theory of human linguistic competence. It turns out that, although there are some implicational universals which are predicted by baseline CCG either in terms of generative capacity or a simplicity-based grammar ranking, there are others which are not, even though we have an intuitive idea that they are economy-based in some way. It was concluded that it is necessary to add some extra descriptive machinery to the baseline CCG category notation, if we want it to explain these in terms of an acquisition-based preference for shorter lexicons. The next three chapters will attempt to do this.

In this chapter, I present a version of the CCG formalism where the alphabet of saturated category symbols is organised into a *type hierarchy*. I show that this formalism, which I call 'type-hierarchical' CCG (or T-CCG for short), allows for more efficient grammars to be formulated for human languages than the baseline CCG formalism from chapter 2, in the sense that the lexicon is closer to being a *function* from lexical forms to lexical category descriptions. I then demonstrate that T-CCG renders redundant a number of other proposed generalisations of the baseline CCG category notation, in particular: (a) the morphosyntactic CCG formalism of Bozsahin (2002) where saturated categories are prefixed by unary modalities representing syntactic features; and (b) most aspects of the extended CCGs discussed in Steedman (2000), Baldridge (2002) and Erkan (2003) which use typed feature structures to model saturated categories.

This chapter will proceed as follows:

Section 5.1 exemplifies the problem of lexical redundancy in baseline CCG grammars, with a simple example involving subject-verb agreement in a fragment of English. I introduce two ideals for human language lexicons — the criteria of functionality and atomicity, and show that it is simply not possible to formulate CCGs which even come close to satisfying these using only the baseline CCG category notation.

Section 5.2 formally defines the notion of a 'type hierarchy', based on the discussion in Carpenter (1992), as a particular kind of weak partial order.

Section 5.3 presents an explicit definition of the T-CCG formalism, as a minimal extension of baseline CCG. I specify a class of grammars, and a 'generates' relation defined through the combinatory projection of a T-CCG lexicon.

Section 5.4 then presents a proof system for the T-CCG formalism, allowing grammaticality to be determined in terms of operations on the T-CCG category descriptions themselves, rather than through the underlying models.

Section 5.5 shows that the T-CCG formalism allows more efficient grammars to be written for natural languages than does CCG, returning to the example of subject-verb agreement in English. In particular, it is possible to construct T-CCG lexicons which come closer to satisfying the criterion of functionality. I argue that the type hierarchy of saturated categories renders redundant many other proposed CCG categorial notations. For example, the morphosyntactic CCG formalism of Bozsahin (2002), where minor syntactic features are represented by unary modalities prefixed to saturated category symbols, is subsumed by the T-CCG category notation. In addition, the T-CCG formalism also subsumes versions of CCG where saturated categories are conceptualised as typed feature structures, unless it is assumed that coindexed variables can act as feature-values.

5.1 CCG and lexical redundancy

Consider the fragment of English schematised in Table 5.1. This language contains a finite number of sentences such as:

(5.1) John loves us.

He loves the girls.

John		John
he	love-s	me
the girl		you
girl-s		him
I		us
you	love	them
we		the girl
they		girl-s
girl-s		girl-s

Table 5.1: Schematisation of a fragment of English

Girls love you.

The following strings are *not* sentences of this language:

(5.2) *John love us.

* The girls loves John.

Note however that this fragment contains certain strings which are arguably *non-sentences* of English, for example **I love me* and **We love us*. The smallest, linguistically intuitive, baseline CCG lexicon of the fragment is presented in Table 5.2. The various atomic saturated category symbols utilised in this grammars are meant to be interpreted as follows:

S sentence

$\mathbf{NP_x}$ third person singular subject noun phrase

$\mathbf{NP_{obj}}$ object noun phrase

$\mathbf{NP_{sbj}}$ subject noun phrase which is *not* third person singular

$\mathbf{N_{sg}}$ singular common noun

$\mathbf{N_{pl}}$ plural common noun

Note in particular that the definition of the '$\mathsf{NP}_{\mathsf{sbj}}$' symbol is slightly counterintuitive.

Although Table 5.2 constitutes the simplest CCG lexicon which generates all and only the sentences of the fragment of English schematised in Table 5.1, even a superficial glance at its contents reveal that it is not particularly efficient:

$\mathsf{John} \vdash \mathsf{NP}_{\mathsf{x}}$	$\mathsf{John} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{girl} \vdash \mathsf{N}_{\mathsf{sg}}$	$\mathsf{s} \vdash \mathsf{N}_{\mathsf{pl}} \backslash \mathsf{N}_{\mathsf{sg}}$
$\mathsf{s} \vdash \mathsf{NP}_{\mathsf{sbj}} \backslash \mathsf{N}_{\mathsf{sg}}$	$\mathsf{s} \vdash \mathsf{NP}_{\mathsf{obj}} \backslash \mathsf{N}_{\mathsf{sg}}$
$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{x}} / \mathsf{N}_{\mathsf{sg}}$	$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{obj}} / \mathsf{N}_{\mathsf{sg}}$
$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{sbj}} / \mathsf{N}_{\mathsf{pl}}$	$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{obj}} / \mathsf{N}_{\mathsf{pl}}$
$\mathsf{I} \vdash \mathsf{NP}_{\mathsf{sbj}}$	$\mathsf{me} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{we} \vdash \mathsf{NP}_{\mathsf{sbj}}$	$\mathsf{us} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{you} \vdash \mathsf{NP}_{\mathsf{sbj}}$	$\mathsf{you} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{he} \vdash \mathsf{NP}_{\mathsf{x}}$	$\mathsf{him} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{they} \vdash \mathsf{NP}_{\mathsf{sbj}}$	$\mathsf{them} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{love} \vdash (\mathsf{S} \backslash \mathsf{NP}_{\mathsf{sbj}}) / \mathsf{NP}_{\mathsf{obj}}$	$\mathsf{s} \vdash ((\mathsf{S} \backslash \mathsf{NP}_{\mathsf{x}}) / \mathsf{NP}_{\mathsf{obj}}) \backslash ((\mathsf{S} \backslash \mathsf{NP}_{\mathsf{sbj}}) / \mathsf{NP}_{\mathsf{obj}})$

Table 5.2: A CCG lexicon for the language in Table 5.1

1. the proper noun ‘John’ and the second person pronoun ‘you’ are each mapped to *two* lexical category descriptions, for when they are used as subjects and objects respectively

2. the plural suffix ‘-s’ is mapped to *three* lexical category descriptions, depending on whether the result expression is used as a plural common noun, a bare subject or a bare object

3. the definite article ‘the’ is mapped to *four* lexical category descriptions, depending on whether the argument is a singular or plual common noun, and whether the result is a subject or object noun phrase

None of these multiple category assignments reflects any intuitive underlying idea of genuine semantic ambiguity.

It is generally accepted amongst grammar engineers and certain parts of the theoretical linguistics and psycholinguistics communities that positing multiple category assignments for a word or morpheme which is not pretheoretically ambiguous is not a desirable course to take. To this extent, it appears that there is a consensus that a human language lexicon, when conceptualised as a mapping from phonetic forms to lexical category descriptions, should be as close as possible to a ‘function’. In other words,

every form should be mapped to *exactly one* lexical category, unless it is genuinely ambiguous, in which case multiple category assignments are justified by the data. This ideal can be characterised as the following constraint on lexicons:

ideal of functionality a lexicon is ideally a *function* from morphemes to category descriptions

From the example above it is clear that the baseline CCG formalism is incapable of providing grammars for human languages whose lexicons satisfy the ideal of functionality. This is because the only way to capture bounded dependencies in a baseline CCG lexicon is by means of multiple category assignments. In the fragment of English in Table 5.1, the relevant bounded dependencies involve case, for example 'he' versus 'him', and subject-verb agreement, i.e. 'he loves' versus 'they love'. In the baseline CCG lexicon in Table 5.2, case is captured by assigning noun phrases which can be used either as subjects or objects to two distinct categories, one for each use. Also, subject-verb agreement is handled by assigning the definite article, which can head either third person singular or third person plural noun phrases, to two further categories. Note that there are other baseline CCG lexicons which capture this data in different ways, but none is globally simpler than the one chosen here.

The failure of the CCG lexicon in Table 5.2 to satisfy the ideal of functionality can be quantified — it contains *twenty-two* lexical items for *fourteen* distinct lexical forms, giving a 'functionality ratio' of $22/14 = 1.57$. The ideal is of course 1.

The reason why something like the ideal of functionality is of interest to grammar engineers is to do with the amount of *time* required to key the lexicon into a computer rather than the amount of *space* required to store it, given how large and fast online storage has become. Further engineering motivations include the reduction of ambiguity (and hence lower processing costs) and increased maintainability as the grammar increases in coverage. From this perspective, note that if every proper noun needs to have two distinct lexical assignments, then any wide coverage lexicon containing thousands of proper nouns is going to be much less efficient than one would want. Thus grammar engineers have a particularly pressing need for a grammar formalism which provides human language lexicons which satisfy the ideal of functionality.

These considerations suggest another way in which the baseline CCG lexicon in Table 5.2 can be argued to be inefficient. Note that whereas 'open class' lexical forms such as 'John' and 'girl' are mapped to lexical category descriptions consisting of a

single atomic symbol, the same is not true for the verb ‘love, whose lexical category is the decidedly non-atomic $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$. Since a wide coverage lexicon for English is going to contain thousands of such verbs, if each one is mapped to such a complex category description, then the lexicon will be much longer than might be desired.

The desire for open class lexical forms to be mapped to atomic lexical types can be codified as the following lexical principle:

ideal of atomicity a lexicon is a mapping from morphemes ideally to an alphabet of *atomic* lexical types

It should be clear that the baseline CCG formalism is incapable of providing lexicons for human languages which satisfy this ideal. Admittedly there are other baseline CCGs for the fragment of English in Table 5.1 in which the verb ‘love’ is mapped to a saturated lexical category like ‘V’. In this case, however, the non-atomicity will simply be transfered to other open class lexical items. For example, every proper noun will need to be assigned to lexical categories like $\mathsf{VP}\backslash\mathsf{V}$ and S/VP.

Again the non-atomicity of a CCG lexicon can be quantified — the lexicon in Table 5.2 contains *thirty-six* saturated category symbols for *fourteen* distinct lexical forms, giving an atomicity ratio of $36/14 = 2.57$. Again the ideal is 1.

Functionality and atomicity should be considered as ideals rather than necessities. The lexicon should satisfy them as closely as possible, other things being equal. This means that the ideal of functionality can be overridden by cases of genuine semantic or syntactic ambiguity. In addition, the dynamic nature of language acquisition can lead to non-functional lexicons, as discussed in subsection 3.2.2. Atomicity is only really an issue for open class lexical items, as the number of functional lexical expressions in a human language is tiny compared to the number of nouns, verbs and adjectives.

The importance of having compact non-redundant lexicons is particularly important for the methodology assumed in this thesis. Recall that the aim is to find a version of the CCG formalism which predicts implicational universals in terms of distinctions in lexical length. Unless the lexicons themselves are reasonable compact, the sheer massiveness of a typical human language lexicon will mask the subtle distinctions necessary for such a learning methodology to be successful. Thus although the argumentation above justified non-redundant lexicons from a grammar-engineering perspective, theoretical linguists might like to think about redundant lexicons as having missed significant linguistic generalisations about the behaviour of the words and morphemes of

the language.

Finally, the criteria of functionality and atomicity have played an important underlying role in the development of the CCG formalism. First of all, recall the principal methodology of CCG grammar design i.e. the Principle of Head Categorial Uniqueness from Steedman (2000):

> A single nondisjunctive lexical category for the head of a given construction specifies both the bounded dependencies that arise when its complements are in canonical position and the unbounded dependencies that arise when these complements are displaced under relativization, coordination, and the like.

In practice, this means that when we are extending a CCG for some fragment of a language, we should try to avoid the temptation of capturing some unbounded depenency construction by adding an additional lexical category for some word or morpheme already in the lexicon. This principle is obviously motivated by the ideal of functionality.

Secondly, the 'multiset' categories introduced in Hoffman (1995) are also motivated by a desire for lexicons in free word order languages like Turkish to satisfy the ideal of functionality. As Hoffman points out, an observationally adequate baseline CCG for Turkish needs to assign every transitive verb to *six* distinct lexical categories, one for each permutation of verb, subject and object. Using the multiset notation however, a transitive verb can be assigned to a category description like $\mathsf{S}\{\mathsf{NP}_{\mathsf{sbj}}, \mathsf{NP}_{\mathsf{obj}}\}$, where both the directionality and order of precedence of the two arguments are underspecified. Functionality also plays a role in the use of underspecified minor syntactic features and lexical rules in CCG analyses. For example, Steedman (1996) argues that the best CCG analysis of reflexives in English has the following features: (a) NPs have minor syntactic features for person, number and gender; (b) the subject of a third person singular present tense verb like 'eats' is a noun phrase whose gender feature is underspecified; (c) there are special 'reflexivised' lexical categories for transitive verbs, which select for a reflexive pronoun object referentially coindexed with the subject, and which are formed from the non-reflexive category by a lexical rule.

Finally, the ideal of atomicity can be argued to have had an effect on the development of CCG category notation, from the persectives of grammar implementation and the compact representation of derivations. In both cases it is customary to define a class of 'macros' for commonly used unsaturated category symbols. For example, we

can define the following macro:

(5.3) $\mathsf{VP} = \mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}$

And then use it as a lexical type:

(5.4) $\mathsf{dance} \vdash \mathsf{VP}$

In addition, we can define new macros in terms of already existing ones, for example transitive and ditransitive verbs:

(5.5) $$\begin{aligned} \mathsf{TV} &= \mathsf{VP}/\mathsf{NP}_{\mathsf{obj}} \\ \mathsf{DTV} &= \mathsf{TV}/\mathsf{NP}_{\mathsf{obj}} \end{aligned}$$

Again, these can be used as lexical types:

(5.6) $$\begin{aligned} \mathsf{love} &\vdash \mathsf{TV} \\ \mathsf{give} &\vdash \mathsf{DTV} \end{aligned}$$

Such category macros are commonly used in computational implementations of CCG.

I conclude this section with the observation that other grammar formalisms do allow the construction of lexicons for human languages which satisfy the ideals of functionality and atomicity. Take for example the CFG in Table 5.3. This CFG generates the fragment of English schematised in Table 5.1. Note that its lexicon satisfies the ideals of functionality (no open class terminal symbol occurs on the right-hand side of more than one production) and atomicity (every open class terminal symbol is on its own on the right-hand side of the rule which introduces it).

5.2 Type hierarchies

Section 5.1 argued that the baseline CCG category notation from section 2.1 is inadequate in one important sense — it is not possible to construct non-redundant lexicons for human language using baseline CCG notation. In this section I introduce the notion of a 'type hierarchy', which provides enough expressive power to resolve at least part of the problem.

The graph in Figure 5.1 represents an example of the kind of relational structure commonly known in the computational linguistics and computer programming literature as a 'type hierarchy'. As defined in Carpenter (1992), every type hierarchy is a

$S \rightarrow NP_{sg}\ VP_x$	$S \rightarrow NP_x\ VP_x$
$S \rightarrow NP_{pl}\ VP$	$S \rightarrow NP_s\ VP$
$S \rightarrow NP\ VP$	$VP_x \rightarrow V_x\ NP$
$VP_x \rightarrow V_x\ NP_{sg}$	$VP_x \rightarrow V_x\ NP_o$
$VP_x \rightarrow V_x\ NP_{pl}$	$VP \rightarrow V\ NP$
$VP \rightarrow V\ NP_{sg}$	$VP \rightarrow V\ NP_o$
$VP \rightarrow V\ NP_{pl}$	$NP_{sg} \rightarrow T\ N_s$
$NP_{pl} \rightarrow T\ N_p$	$NP_{pl} \rightarrow N_p$
$N_p \rightarrow N_s\ s$	$V_x \rightarrow V\ s$
$NP_{sg} \rightarrow John$	$N_s \rightarrow girl$
$T \rightarrow the$	$NP_s \rightarrow I$
$NP_o \rightarrow me$	$NP_s \rightarrow we$
$NP_o \rightarrow us$	$NP_x \rightarrow he$
$NP_o \rightarrow him$	$NP_s \rightarrow they$
$NP_o \rightarrow them$	$NP \rightarrow you$
$V \rightarrow love$	

Table 5.3: A CFG for the language schematised in Table 5.1

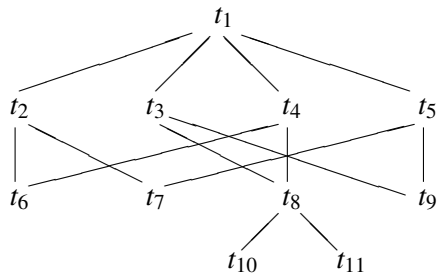
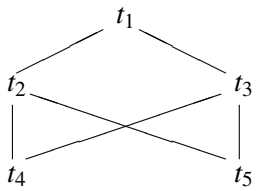


Figure 5.1: A type hierarchy

weak order, whose domain is a set of 'types' or 'sorts'. Types are ordered by a relation known as 'subsumption' — in Figure 5.1, type t_3 subsumes types t_8 and t_9. Subsumption is transitive — since t_3 subsumes t_8, and t_8 itself subsumes t_{10} and t_{11}, we deduce that t_3 also subsumes t_{10} and t_{11}.

Although every type hierarchy is a weak order, not every weak order counts as a type hierarchy, since the latter must pass the 'most general unifier' condition — if two types are consistent (i.e. there is at least one type which they both subsume), then exactly one of the types they subsume must itself subsume all the others. Take for example, the weak order represented in Figure 5.1. Types t_3 and t_4 are consistent, since they both subsume t_8, t_{10} and t_{11}. Of these three 'subtypes', t_8 subsumes both t_{10} and t_{11}, Thus, t_8 is the 'most general unifier' of t_3 and t_4. If every set of consistent types in a weak order has a unique most general unifier, then that weak order counts as a type hierarchy. An example of a weak order which does not count as a type hierarchy is the following:

(5.7)



Types t_2 and t_3 are consistent, since they both subsume t_4 and t_5. However, neither t_4 nor t_5 subsumes the other, so t_2 and t_3 lack a most general unifier, and hence this weak order is not a type hierarchy.

Formally, a type hierarchy is an ordered pair $\langle T, \sqsubseteq \rangle$ where:

1. T is an alphabet of types
2. $\sqsubseteq$ is a reflexive, antisymmetric, transitive relation on T, known as the 'subsumption' relation i.e. a type hierarchy is a weak order
3. $\langle T, \sqsubseteq \rangle$ has a first/least element i.e. there is one type which subsumes every other type, including itself
4. $\langle T, \sqsubseteq \rangle$ is such that every subset of T which has an upper bound has a least upper bound

Where $\mathcal{H} = \langle T, \sqsubseteq \rangle$ is a type hierarchy, the set of 'maximal' types in $\mathcal{H}$ is defined as the set of types in T which do not subsume any type other than themselves. In other

words, a maximal type is one which has no subtypes. For example, the maximal types in the type hierarchy represented in Figure 5.1 are t_6, t_7, t_{10}, t_{11} and t_9.

Note finally that a type hierarchy can be a tree, in which case the only way in which two distinct types can be consistent is if one subsumes the other, and the most general unifier of two consistent types is the more specific of the two. A type hierarchy which can be represented as a tree is known as a 'single inheritance' type hierarchy. A type hierarchy which is not a tree, such as that in Figure 5.1 is known as a 'multiple inheritance' type hierarchy. That this is a multiple inheritance hierarchy is evident from the fact that, for example, type t_7 has two parent types, t_2 and t_5.

5.3 The T-CCG formalism

The T-CCG formalism is a version of CCG where the alphabet of saturated categories is organised into a type hierarchy. In this section, I present an explicit definition of the T-CCG formalism, making careful reference to the corresponding definition of CCG in chapter 2.

Subsection 5.3.1 formalises the class of model-theoretic structures that underly the category notation of T-CCG, noting that they are simply baseline CCG category models which are sort-resolved relative to some type hierarchy of saturated category symbols.

Subsection 5.3.2 defines the category description language assumed by T-CCG, which is exactly the same as that for baseline CCG.

Subsection 5.3.3 presents the satisfaction relation linking T-CCG category models and T-CCG category descriptions.

Subsection 5.3.4 specifies the class of T-CCG grammars.

Subsection 5.3.5 defines the combinatory projection of a T-CCG lexicon.

Subsection 5.3.6 formulates the T-CCG 'generates' relation, defining which strings are generated by which T-CCGs.

5.3.1 T-CCG category models

Recall the definition of baseline CCG category models from subsection 2.1.4 — a category model over alphabet A of saturated categories takes the form of an ordered 5-tuple $\langle Q, Res, Arg, V_S, V_A \rangle$, where: (a) Q is a set of points; (b) Res and Arg are the

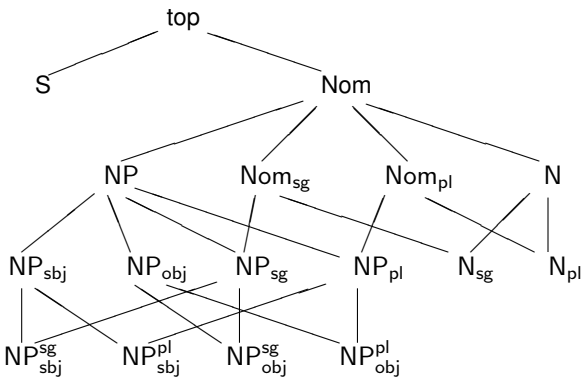
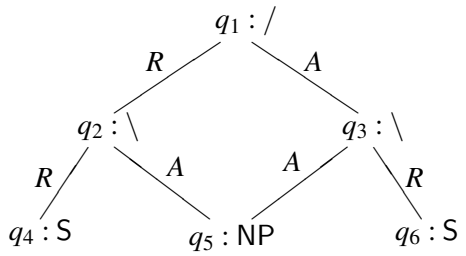


Figure 5.2: A type hierarchy of saturated category symbols for the language in Table 5.1

'result' and 'argument' relations on Q; (c) V_S assigns a slash to every non-end point; and (d) V_A assigns a saturated category symbol from A to every end point. An example of such a category model over alphabet $\{\mathsf{S}, \mathsf{NP}\}$ is as follows:

(5.8)



This represents the category traditionally denoted as something like $(\mathsf{S}\backslash\mathsf{NP}_i)/(\mathsf{S}\backslash\mathsf{NP}_i)$, where the coindexation enforces token identity.

From now on, I will be assuming that the alphabet of saturated category symbols is organised into a type hierarchy. Figure 5.2 contains an example type hierarchy of saturated category symbols, intended as part of a grammar of subject-verb agreement in English. Of the seventeen saturated categories in this hierarchy, only seven are maximal. These are to be interpreted as follows:

S sentence

NP$^{sg}_{sbj}$ third person singular subject noun phrase

NP$^{pl}_{sbj}$ subject noun phrase which is *not* third person singular

$\mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$ third person singular object noun phrase

$\mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$ object noun phrase which is not third person singular

N_{sg} singular common noun

N_{pl} plural common noun

Note in particular that those symbols containing a 'pl' superscript are slightly counterintuitive, in that they are intended to denote not just plurality but the more general property of not being third person singular. Note also that these are all to be understood as *atomic* symbols — the subscripts and superscripts have mnemonic significance only.

Of the non-maximal types, the most general type is by convention named 'top', and includes both sentences and nominals. The most general nominal type 'Nom' subsumes all common nouns, proper nouns and noun phrases. It is partitioned along two dimensions: (a) 'NP' (noun phrase) vs. 'N' (common noun);[1] and (b) '$\mathsf{Nom}_{\mathsf{sg}}$' (third person singular) vs. '$\mathsf{Nom}_{\mathsf{pl}}$' (other). Based on these two binary partitions we get four subtypes: (a) the maximal types 'N_{sg}' and 'N_{pl}'; and (b) the non-maximal types '$\mathsf{NP}_{\mathsf{sg}}$' (third person singular noun phrase) and '$\mathsf{NP}_{\mathsf{pl}}$' (noun phrase which is not third person singular). In addition, the non-maximal 'NP' type is partitioned independently along the dimension '$\mathsf{NP}_{\mathsf{sbj}}$' (subject) vs. '$\mathsf{NP}_{\mathsf{obj}}$' (object). Since 'NP' has two binary partitions, there are again four subtypes, the maximal noun phrase types discussed above. For those who prefer HPSG-style partition diagrams, the type hierarchy in Figure 5.2 can be represented as in Figure 5.3. The former can be compiled out from the latter by means of 'conjunctive type construction' (Carpenter, 1992).

Just like baseline CCG category models over alphabet A, a T-CCG category model over type hierarchy $\langle A, \sqsubseteq \rangle$ of saturated category symbols is a category *structure* (see subsection 2.1.3) where every end point is labelled by exactly one symbol in A. However, following standard practice in model-theoretic formalisations of unification-based grammars, I assume that T-CCG category models are 'sort-resolved' — every end point is labelled with exactly one *maximal* type in $\langle A, \sqsubseteq \rangle$. For example, Figure 5.4 contains two category structures whose end points are labelled with types from the hierarchy in Figure 5.2. Note that only the one on the *left* counts as a T-CCG category model over the type hierarchy in Figure 5.2. The category structure on the right has one end point,

[1] Think of this distinction in terms of the 'bar' levels of GB, GPSG and HPSG.

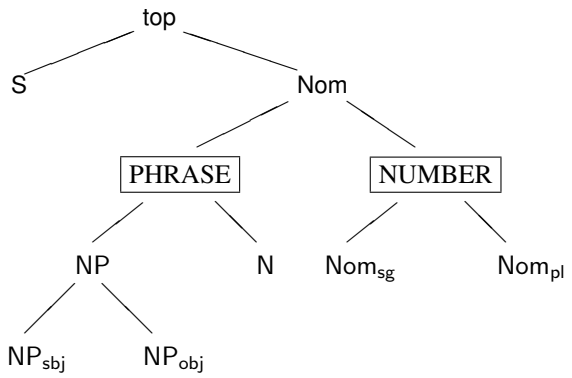


Figure 5.3: A partition diagram representing the type hierarchy in Figure 5.2

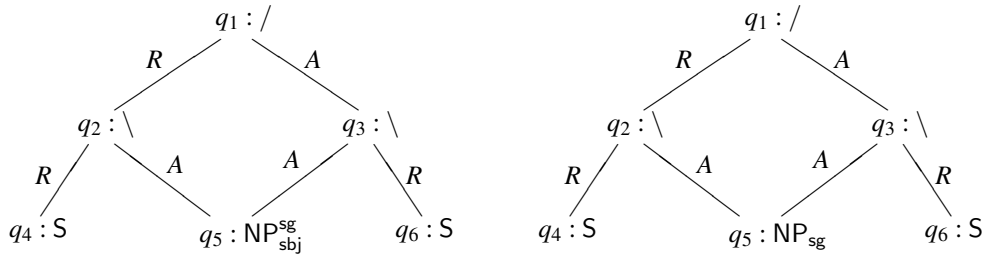


Figure 5.4: A T-CCG category model and a non-T-CCG category model over the type hierarchy in Figure 5.2

q_5 which is labelled by a non-maximal type, $\mathsf{NP}_{\mathsf{sg}}$, and thus does not count as a T-CCG category model over the type hierarchy in Figure 5.2.

Formally, a T-CCG category model over type hierarchy $\langle A, \sqsubseteq \rangle$ of saturated category symbols is an ordered 5-tuple $\langle Q, Res, Arg, V_S, V_A \rangle$, where:

1. $\langle Q, Res, Arg, V_S \rangle$ is a category structure, as defined in subsection 2.1.3
2. V_A is a function from the end points of $\langle Q, Res, Arg \rangle$ to the set of *maximal* types in $\langle A, \sqsubseteq \rangle$ i.e. every end point in the structure is labelled with exactly one saturated category symbol which has no subtypes

5.3.2 T-CCG category descriptions

Recall from the definition of baseline CCG category notation in section 2.1 that, in addition to defining a class of category *models* (subsection 2.1.4), I also defined in subsection 2.1.5 a set of category *descriptions* for talking about category models, the two concepts being related by the standard logical notion of satisfaction. The set of T-CCG category descriptions over alphabet A of saturated category symbols is exactly the same as the set of baseline CCG category descriptions over A, i.e. the smallest set Φ such that:

1. $A \subseteq \Phi$
2. for all $\phi, \psi \in \Phi$, (ϕ/ψ), $(\phi\backslash\psi)$ and $(\phi \mid \psi)$ are all in Φ

Note that T-CCG category descriptions, unlike T-CCG category models, can contain *non-maximal* types from the hierarchy of saturated category symbols. So, for example, although the rightmost structure in Figure 5.4 is *not* a T-CCG category model over the type hierarchy in Figure 5.2, the following do count as category descriptions over the set of types, and indeed both are satisfied by the left category model in Figure 5.4:

(5.9) $(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sg}})/(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sg}})$

$(\mathsf{top} \mid \mathsf{Nom}_{\mathsf{sg}})/(\mathsf{S}\backslash\mathsf{Nom})$

5.3.3 T-CCG satisfaction

The T-CCG satisfaction relation links T-CCG category models and T-CCG category descriptions — it tells us which category models satisfy which descriptions, or alternatively which sets of category models are 'picked out' by which category descriptions.

The global satisfaction relation for T-CCG is exactly the same as that for baseline CCG, as defined in subsection 2.1.6, i.e. T-CCG category model $\mathcal{M}$ over type hierarchy $\langle A, \sqsubseteq \rangle$ satisfies T-CCG category description ϕ over A, written $\mathcal{M} \models \phi$, if and only if $\mathcal{M}, q \models \phi$, where q is the root of $\mathcal{M}$. In other words, a description is globally satisfied by a model just in case it is *locally* satisfied from the root point in the model.

The baseline CCG local satisfaction relation, however, must be rewritten slightly to take account of the subsumption ordering on saturated category symbols. Previously, a saturated category symbol was satisfied at an end point just in case the point was labelled with exactly that symbol. Since T-CCG saturated category symbols are related by subsumption, a saturated category is satisfied at a point if and only if the point is labelled either by that symbol itself, *or by one which it subsumes*. For example, where $\mathcal{M}$ is the T-CCG category model on the left in Figure 5.4, not only is it the case that $\mathcal{M}, q_5 \models \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$, but also $\mathcal{M}, q_5 \models \mathsf{NP}_{\mathsf{sbj}}$, $\mathcal{M}, q_5 \models \mathsf{NP}$, and so on. However, it is still the case that $\mathcal{M}, q_5 \not\models \mathsf{NP}_{\mathsf{pl}}$ since $\mathsf{NP}_{\mathsf{pl}} \not\sqsubseteq \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$ in the type hierarchy in Figure 5.2.

Formally, T-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over a given type hierarchy $\langle A, \sqsubseteq \rangle$ locally satisfies T-CCG category description ϕ over A, from point $q \in Q$, written $\mathcal{M}, q \models \phi$, if and only if:

1. where $\phi \in A$: for some $\alpha \in A$ such that $\phi \sqsubseteq \alpha$, $V_A(q) = \alpha$
2. where $\phi = (\psi_1 / \psi_2)$: $V_S(q) = /$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
3. where $\phi = (\psi_1 \backslash \psi_2)$: $V_S(q) = \backslash$, $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$
4. where $\phi = (\psi_1 \,|\, \psi_2)$: $\mathcal{M}, Res(q) \models \psi_1$ and $\mathcal{M}, Arg(q) \models \psi_2$

The last three clauses of the local satisfaction definition are the same as those for the baseline CCG formalism in subsection 2.1.6.

5.3.4 Type-hierarchical CCGs

In subsection 2.3.1, I defined a CCG over alphabet Σ as an ordered triple $\langle A, S, L \rangle$, where A is an alphabet of saturated category symbols, S is a distinguished element of

$\mathsf{John} \vdash \mathsf{NP}_{\mathsf{sg}}$	$\mathsf{girl} \vdash \mathsf{N}_{\mathsf{sg}}$
$\mathsf{I} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{me} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{we} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{us} \vdash \mathsf{NP}_{\mathsf{obj}}$
$\mathsf{you} \vdash \mathsf{NP}_{\mathsf{pl}}$	$\mathsf{he} \vdash \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$
$\mathsf{him} \vdash \mathsf{NP}_{\mathsf{obj}}$	$\mathsf{they} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$
$\mathsf{them} \vdash \mathsf{NP}_{\mathsf{obj}}$	$\mathsf{love} \vdash (\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$
$\mathsf{s} \vdash \mathsf{Nom}_{\mathsf{pl}}\backslash\mathsf{N}_{\mathsf{sg}}$	$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{sg}}/\mathsf{N}_{\mathsf{sg}}$
$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{pl}}/\mathsf{N}_{\mathsf{pl}}$	$\mathsf{s} \vdash ((\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}})\backslash((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{pl}})/\mathsf{NP})$

Table 5.4: A T-CCG lexicon for the language in Table 5.1

A, and L is a lexicon i.e. a finite mapping from Σ to baseline CCG category descriptions over A. A T-CCG is a rather minimal extension of such a baseline CCG, in that there is a subsumption relation on the alphabet A of saturated category symbols.

Formally, a T-CCG over alphabet Σ is an ordered 4-tuple $\langle A, \sqsubseteq, S, L\rangle$ where:

1. $\langle A, \sqsubseteq\rangle$ is a type hierarchy of saturated category symbols
2. S is a distinguished element of A
3. L is a finite mapping from Σ to T-CCG category descriptions over A

For example, where $\langle A, \sqsubseteq\rangle$ is the type hierarchy represented in Figure 5.2, and L is the set of sixteen ordered pairs in Table 5.4, then $\langle A, \sqsubseteq, \mathsf{S}, L\rangle$ is a T-CCG of the fragment of English schematised in Table 5.1.

5.3.5 Combinatory projection of a T-CCG lexicon

Recall from subsection 2.3.2 that, although a CCG lexicon was defined as a finite mapping from symbols to category descriptions, its *combinatory projection* was a mapping from strings to category models. The base of the combinatory projection of a CCG lexicon is simply the result of 'compiling out' the underspecified lexical entries — if entry $\langle s, \phi\rangle$ is in lexicon L and category model $\mathcal{M}$ satisfies description ϕ, then the pair $\langle s, \mathcal{M}\rangle$ is in the base of the combinatory projection of L. The combinatory projection

of a lexicon is then defined as the closure of this base under the CCG combinatory operations i.e. application, raising, composition, cross composition, and substitution, as defined in section 2.2. The combinatory projection of a T-CCG lexicon is defined in exactly the same way:

Given T-CCG $\langle A, \sqsubseteq, S, L\rangle$ over alphabet Σ, the combinatory projection of lexicon L is defined as the smallest set L', such that:

1. for all $\langle s, \phi\rangle \in L$, and all category models $\mathcal{M}$ over $\langle A, \sqsubseteq\rangle$ such that $\mathcal{M} \models \phi$, $\langle s, \mathcal{M}\rangle \in L'$

2. for all $\langle s, \mathcal{M}_1\rangle \in L'$ and all category models $\mathcal{M}_2$ over $\langle A \sqsubseteq\rangle$, where $\mathcal{M}_3$ is the result of either forward or backward raising $\mathcal{M}_1$ over $\mathcal{M}_2$, then $\langle s, \mathcal{M}_3\rangle \in L'$

3. for all $\langle s_1, \mathcal{M}_1\rangle, \langle s_2, \mathcal{M}_2\rangle \in L'$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to $\mathcal{M}_3$ by means of one of the binary CCG combinatory operations defined in section 2.2, $\langle s_1 s_2, \mathcal{M}_3\rangle \in L'$

The T-CCG combinatory operations are defined exactly as their CCG equivalents in section 2.2. Since they operate on category models rather than category descriptions, and since category models are by definition sort-resolved, there is no need to redefine the operations to take account of the T-CCG subsumption ordering.

5.3.6 T-CCG generation

Finally we come to the 'generates' relation for T-CCG. The equivalent definition for the baseline CCG formalism, presented in subsection 2.3.3, said that CCG $\langle A, S, L\rangle$ over alphabet Σ generates string s over Σ just in case the ordered pair $\langle s, S\rangle$ is in the combinatory projection of L — in other words, the string must be mapped to a simple category in the combinatory projection, whose root node is labelled with the distinguished symbol.

The equivalent definition for the T-CCG formalism must take into account the fact that the distinguished symbol may be a non-maximal type in the relevant subsumption hierarchy. In other words, T-CCG $\langle A, \sqsubseteq, S, L\rangle$ over alphabet Σ generates string $s \in \Sigma^*$ if and only if for some X such that $S \sqsubseteq X$, the ordered pair $\langle s, X\rangle$[2] is an element of the combinatory projection of lexicon L.

[2]More precisely, the ordered pair $\langle s, \langle \{q\}, \emptyset, \emptyset, \emptyset, \{\langle q, X\rangle\}\rangle\rangle$ for some point q.

5.4 A proof system for T-CCG

In section 5.3 I presented a partially model-theoretic formulation of the T-CCG formalism, where the lexicon maps lexical strings to category *descriptions*, these category descriptions are satisfied by underlying, sort-resolved category *models*, and the various CCG combinatory operations apply to the underlying models rather than to the descriptions. This section present a proof system for the T-CCG formalism, which allows us to test the grammatiality of a string relative to a grammar using a set of inference rules operating on category *descriptions*. These inference rules will form the basis of T-CCG derivations.

5.4.1 Lexical insertion

The T-CCG proof system includes an inference rule which can usefully be called 'lexical insertion'. Given T-CCG $G = \langle A, \sqsubseteq, S, L\rangle$ over alphabet Σ, for all $\langle s, \phi\rangle \in L$, the following is a valid inference according to G:

(5.10)
$$\frac{s}{\phi}$$

5.4.2 Compatibility of T-CCG category descriptions

T-CCG category descriptions ϕ and ψ over alphabet A are $\sqsubseteq$-compatible if and only if one of the following holds:

1. $\phi, \psi \in A$ and $\{\phi, \psi\}$ has an upper bound in $\langle A, \sqsubseteq\rangle$
2. $\phi = (\phi_1/\phi_2)$, ψ is either (ψ_1/ψ_2) or $(\psi_1 \,|\, \psi_2)$, ϕ_1 and ψ_1 are $\sqsubseteq$-compatible, and ϕ_2 and ψ_2 are $\sqsubseteq$-compatible
3. $\phi = (\phi_1 \backslash \phi_2)$, ψ is either $(\psi_1 \backslash \psi_2)$ or $(\psi_1 \,|\, \psi_2)$, ϕ_1 and ψ_1 are $\sqsubseteq$-compatible, and ϕ_2 and ψ_2 are $\sqsubseteq$-compatible
4. $\phi = (\phi_1 \,|\, \phi_2)$, ψ is either (ψ_1/ψ_2), $(\psi_1 \backslash \psi_2)$ or $(\psi_1 \,|\, \psi_2)$, ϕ_1 and ψ_1 are $\sqsubseteq$-compatible, and ϕ_2 and ψ_2 are $\sqsubseteq$-compatible

For example, where $\sqsubseteq$ is the subsumption relation in Figure 5.2 on page 154, $\mathsf{Nom}_{\mathsf{sg}}$ and $\mathsf{NP}_{\mathsf{sbj}}$ are $\sqsubseteq$-compatible, because they have the (least) upper bound $\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$. However, N and $\mathsf{NP}_{\mathsf{obj}}$ are $\sqsubseteq$-incompatible, having no subtypes in common. By exten-

sion, $(\mathsf{S}|\mathsf{NP}_{\mathsf{sbj}})$ and $(\mathsf{top}/\mathsf{Nom}_{\mathsf{sg}})$ are $\sqsubseteq$-compatible, since both the result and argument types are $\sqsubseteq$-compatible and the slashes are consistent. However, $(\mathsf{S}/\mathsf{NP}_{\mathsf{sbj}})$ and $(\mathsf{top}/\mathsf{N})$ are $\sqsubseteq$-incompatible, since the argument types are $\sqsubseteq$-incompatible.

5.4.3 Application

The following T-CCG inference rule corresponds to the combinatory operations of forward and backward application, defined in subsections 2.2.2 and 2.2.3:

Given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, for all T-CCG category descriptions ϕ, ψ and ψ' over A such that ψ and ψ' are $\sqsubseteq$-compatible, the following are all valid inferences according to G:

(5.11) $$\frac{\phi/\psi \quad \psi'}{\phi}{>} \qquad \frac{\phi|\psi \quad \psi'}{\phi}{>} \qquad \frac{\psi' \quad \phi\backslash\psi}{\phi}{<} \qquad \frac{\psi' \quad \phi|\psi}{\phi}{<}$$

5.4.4 Raising

The following T-CCG inference rule corresponds to the combinatory operations of forward and backward raising, defined in subsections 2.2.6 and 2.2.7:

Given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, for all category descriptions ϕ and ψ over A (the latter containing only *maximal* types), the following are both valid inferences according to G:

(5.12) $$\frac{\phi}{\psi/(\psi\backslash\phi)}{>\mathbf{T}} \qquad \frac{\phi}{\psi\backslash(\psi/\phi)}{<\mathbf{T}}$$

5.4.5 Composition

The following T-CCG inference rule corresponds to the combinatory operations of forward (harmonic) composition defined in subsections 2.2.4:

Given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, for all T-CCG category descriptions ϕ, ψ, ϕ' and χ over A such that ψ and ψ' are $\sqsubseteq$-compatible, the following are all valid inferences according to G:

(5.13) $$\frac{\phi/\psi \quad \psi'/\chi}{\phi/\chi}{>\mathbf{B}1} \qquad \frac{\phi/\psi \quad \psi'|\chi}{\phi/\chi}{>\mathbf{B}1} \qquad \frac{\phi|\psi \quad \psi'/\chi}{\phi/\chi}{>\mathbf{B}1} \qquad \frac{\phi|\psi \quad \psi'|\chi}{\phi/\chi}{>\mathbf{B}1}$$

In addition, given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$ over alphabet Σ, for all T-CCG category descriptions ϕ, ϕ', ψ, ψ' over A, and all $n \geq 1$, if the following is a valid inference according to G:

(5.14) $$\frac{\phi/\psi \quad \psi'}{\phi'} {>}\mathbf{B}n$$

Then, for all category descriptions χ over A, the following are also valid inferences according to G:

(5.15) $$\frac{\phi/\psi \quad \psi'/\chi}{\phi'/\chi} {>}\mathbf{B}n{+}1 \qquad \frac{\phi/\psi \quad \psi'\backslash\chi}{\phi'/\chi} {>}\mathbf{B}n{+}1 \qquad \frac{\phi\backslash\psi \quad \psi'/\chi}{\phi'/\chi} {>}\mathbf{B}n{+}1 \qquad \frac{\phi\backslash\psi \quad \psi'\backslash\chi}{\phi'/\chi} {>}\mathbf{B}n{+}1$$

Similar definitions can be given for proof rules corresponding to backward composition, the cross composition rules, and the substitution ones too.

5.4.6 Properties of the proof system

The proof system presented above needs to be evaluated according to the following criteria:

soundness Is it the case that, given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, for every valid inference $s \vdash S$ according to G, G generates string s?

completeness Is it the case that, given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, for every string s generated by G, $s \vdash S$ is a valid inference according to G?

decidability Is it the case that, given T-CCG $G = \langle A, \sqsubseteq, S, L \rangle$, every valid inference $s \vdash S$ according to G will be found?

I have no formal results about these properties to present. However, a couple of points do stand out. Firstly, the fact that the inference rules contain a version of composition which is unbounded, whereas the CCG formalism assumes that composition must be *lexically bounded*, means that the proof system is arguably non-sound. This can be easily remedied by replacing the unbounded composition inference rule with a finite set of bounded versions, at a certain loss of elegance. Secondly, the way the inference rules corresponding to forward and backward raising have been defined, means that there are some proofs that will never terminate, since raising can occur recursively. This can be

rectified by restricting the raising rules to operate only on string-category pairs which are the direct result of lexical insertion, although this results in non-completeness. The following derivation demonstrates the forward application inference rule at work, based on the lexicon in Table 5.4 and the type hierarchy in Figure 5.2:

(5.16)
$$\dfrac{\dfrac{\text{loves}}{(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}} \quad \dfrac{\text{girls}}{\mathsf{Nom}_{\mathsf{pl}}}}{\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}}\,{>}$$

Note that, even though the argument types $\mathsf{NP}_{\mathsf{obj}}$ and $\mathsf{Nom}_{\mathsf{pl}}$ are non-identical, they are still compatible according to the type hierarchy, since they have an upper bound $\mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$, and hence the operation is able to proceed. Another example is as follows:

(5.17)
$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\text{John}}{\mathsf{NP}_{\mathsf{sg}}} \quad \dfrac{\dfrac{\text{love}}{(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}} \quad \dfrac{\text{-s}}{((\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}})\backslash((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{pl}})/\mathsf{NP})}}{(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}}\,{<}}{\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}}\,{>}}{\mathsf{S}}\,{<}$$

5.5 T-CCG and the functionality condition on lexicons

Section 5.3 defined the T-CCG formalism as a partially model-theoretic system, incorporating a set of grammars and a 'generates' relation. The key feature of this formalism is that the alphabet of saturated category symbols is organised into a type hierarchy. Section 5.4 provided a proof system for the T-CCG formalism, thus permitting derivations to be carried out using category descriptions rather having to go through the underlying category models.

One advantage of organising the alphabet of saturated category symbols of a CCG into a type hierarchy is that this allows us to formulate grammars of human languages whose lexicons store information more efficiently. Recall the criterion of functionality discussed in section 5.1 according to which human language lexicons should be as close as possible to *functions* from phonetic forms to category descriptions. It was demonstrated that the baseline CCG category notation does not allow lexicons with this property to be constructed, since multiple lexical assignments are necessary to capture bounded dependency constructions in a CCG.

In particular, recall the fragment of English schematised in Table 5.1 on page 145, involving some simple case and agreement phenomena. The simplest baseline CCG

lexicon for this fragment was that in Table 5.2 on page 146, which involved 22 lexical assignments for 14 distinct lexical forms — a functionality ratio of 1.57. On the other hand, the T-CCG category notation allows us to formulate lexicons with better functionality ratios, thanks to the underspecification inherent in the type hierarchy of saturated category symbols. Recall the T-CCG lexicon in Table 5.4 on page 159, which also generates the fragment of English in Table 5.1. Note the following:

1. in the baseline CCG lexicon the proper noun 'John' and the second person pronoun 'you' are each mapped to *two* lexical category descriptions, for when they are used as subjects and objects respectively. In the T-CCG lexicon they are each mapped to just *one* underspecified category description — respectively $\mathsf{NP}_{\mathsf{sg}}$ and $\mathsf{NP}_{\mathsf{pl}}$

2. in the baseline CCG lexicon, the plural suffix '-s' is mapped to *three* lexical category descriptions, depending on whether the result expression is used as a plural common noun, a bare subject or a bare object. In the T-CCG lexicon it is mapped to just the *one* i.e. $\mathsf{Nom}_{\mathsf{pl}}\backslash\mathsf{N}_{\mathsf{sg}}$, since the $\mathsf{Nom}_{\mathsf{sg}}$ type subsumes the three distinct uses

3. in the baseline CCG lexicon the definite article 'the' is mapped to *four* lexical category descriptions, depending on whether the argument is a singular or plual common noun, and whether the result is a subject or object noun phrase. In the T-CCG version, underspecified types again allow it to be mapped to just *two*

Indeed, the T-CCG lexicon contains just *sixteen* lexical assignments for the *fourteen* distinct lexical forms. Its functionality ratio is thus $16/14 = 1.07$, which is very close to the ideal. Thus, even for the simple fragment of English schematised in Table 5.1, it is possible to write a T-CCG which is substantially smaller than the smallest CCG, in the sense that its lexicon is closer to being a function mapping lexical forms to lexical category descriptions.

The remainder of this section argues that the T-CCG formalism in effect renders redundant several other proposed generalisations of the baseline CCG category notation.

Subsection 5.5.1 discusses the morphosyntactic CCG formalism of Bozsahin (2002) where saturated category symbols are prefixed by a range of unary modality symbols.

Subsection 5.5.2 investigates the idea that saturated category symbols in CCG must be modelled as typed feature structures.

noun	gloss	meaning
adam-ı	man-ACC	*the man*
adam-lar-ı	man-PLUR-ACC	*the men*
adam-ım-ı	man-1SG.POSS-ACC	*my man*
adam-lar-ım-ı	man-PLUR-1SG.POSS-ACC	*my men*

Table 5.5: Some accusative case noun-forms in Turkish

In both cases, I demonstrate that for every such generalised CCG, there is a T-CCG which both generates the same language and has a lexicon with just as good a functionality ratio.

5.5.1 T-CCG and morphosyntactic CCG

In this subsection I examine the problem of the morphological structure of nouns in Turkish and show how the baseline CCG formalism is ill-equipped to provide an elegant account of this highly intricate phenomenon. I then present the morphosyntactic CCG formalism of Bozsahin (2002), which was designed, at least in part, to provide a non-redundant grammar of Turkish nouns and verbs. In morphosyntactic CCG saturated category symbols are prefixed by a range of unary modality symbols, themselves organised into a kind of hierarchy. I argue that this category notation is subsumed by that of the T-CCG formalism, where saturated category symbols are themselves ordered by subsumption.

Table 5.5 exemplifies a range of accusative case noun-forms in Turkish. Note that the glosses are to be interpreted as follows: (a) ACC denotes the accusative case suffix; (b) PLUR denotes the plural suffix; and (c) 1SG.POSS denotes the first person singular possessive suffix. Note that the suffixes must occur in the fixed order PLUR $\prec$ 1SG.POSS $\prec$ ACC. Thus, the following strings do *not* represent well-formed Turkish nouns.

(5.18) * adam-ı-lar, * adam-ı-ım, * adam-ı-ım-lar, * adam-ım-lar-ı

The simplest baseline CCG lexicon which will generate all and only the above well-

formed Turkish accusative nouns is the following:

(5.19) adam $\vdash$ N_{base}

-lar $\vdash$ $N_{num}\backslash N_{base}$

-ım $\vdash$ $N_{poss}\backslash N_{base}$, $N_{poss}\backslash N_{num}$

-ı $\vdash$ $N_{acc}\backslash N_{base}$, $N_{acc}\backslash N_{num}$, $N_{acc}\backslash N_{poss}$

Those familiar with Turkish morphophonology will note that I am ignoring the range of phonologically sensitive allomorphs of each of these suffixes (so-called 'vowel harmony'). The saturated category symbols are to be interpreted as follows:

N_{base} a noun-form which has no suffixes e.g. the base stem *adam*

N_{num} a noun-form which has a plural suffix but neither a possessive nor a case suffix e.g. *adam-lar*

N_{poss} a noun-form which has a possessive suffix and possibly a plural suffix too, but not a case suffix e.g. *adam-ım* and *adam-lar-ım*

N_{acc} any noun-form with an accusative suffix e.g.*adam-ı*, *adam-lar-ı*, *adam-ım-ı* and *adam-lar-ım-ı*

The baseline CCG lexicon in (5.19) is used in derivations like the following:

(5.20)

$$\dfrac{\dfrac{\dfrac{\text{adam} \quad \text{lar}}{N_{base} \quad N_{num}\backslash N_{base}}<}{N_{num}} \quad \text{ım} \quad \text{ı}}{N_{poss}} \quad \dfrac{}{N_{poss}\backslash N_{num}} \quad \dfrac{}{N_{acc}\backslash N_{poss}}$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\text{adam} \quad \text{lar}}{N_{base} \quad N_{num}\backslash N_{base}}<}{N_{num}} \quad \text{ım} \quad \text{ı}}{N_{poss}} \quad \dfrac{}{N_{poss}\backslash N_{num}} \quad \dfrac{}{N_{acc}\backslash N_{poss}}}{N_{acc}}$$

It should be noted that the CCG lexicon in (5.19) is not particularly economical, since it contains *two* distinct lexical assignments for the possessive suffix '-ım' and *three* for the accusative case suffix '-ı'. The two category descriptions for the possessive suffix are necessary since it can combine with either a base noun or a plural marked noun, and having three for the accusative suffix licenses its combination with either a base noun, a plural marked noun, or a possessive marked noun form. These multiple category assignments ensure that the lexicon in (5.19) fails to satisfy the criterion of functionality. This failure might be argued to be mitigated by the fact that in both cases we are talking about closed class, functional morphemes. However, it is still clear from

the perspective of theoretical syntax that some kind of significant generalisation is being missed. Moreover, since Turkish transitive verbs take a non-case marked subject and an optionally accusative-case marked object, this analysis implies that a verb such as *sever* ('like') has at least *twelve* lexical categories including the following (ignoring non-SOV orders to keep things simple):

$$\begin{aligned}\textsf{sev-er} \;\vdash\; & (\textsf{S}\backslash\textsf{N}_{\textsf{base}})\backslash\textsf{N}_{\textsf{acc}},\ (\textsf{S}\backslash\textsf{N}_{\textsf{base}})\backslash\textsf{N}_{\textsf{base}},\ (\textsf{S}\backslash\textsf{N}_{\textsf{base}})\backslash\textsf{N}_{\textsf{num}},\\ & (\textsf{S}\backslash\textsf{N}_{\textsf{base}})\backslash\textsf{N}_{\textsf{poss}},\ (\textsf{S}\backslash\textsf{N}_{\textsf{num}})\backslash\textsf{N}_{\textsf{acc}},\ \ldots\end{aligned}$$

Since transitive verbs are open class lexical items and since the Turkish lexicon contains thousands of them, it is clear that the functionality ratio of any wide coverage baseline CCG lexicon for Turkish is going to be unacceptably high.

Bozsahin (2002) argues that the lexicon of a morphologically rich language like Turkish needs to be viewed as a mapping from *morphemes* to categories rather than from words to categories. With this in mind, he presents a variant CCG formalism, which can usefully be called 'morphosyntactic' CCG, which allows us to formulate more economical CCG lexicons for Turkish nouns and verbs than the baseline CCG lexicon in (5.19) and (5.21). A morphosyntactic CCG over alphabet Σ is something like an ordered 5-tuple $\langle A, F, \preceq, S, L\rangle$ where:

1. A is an alphabet of saturated category symbols
2. F is an alphabet of morphosyntactic features
3. $\preceq$ is a weak ordering relation[3] on F
4. S is a distinguished element of A
5. L is a finite mapping from Σ to morphosyntactic CCG categories over A and F

The set of morphosyntactic CCG categories over alphabets A of saturated category symbols and F of morphosyntactic features is the smallest set A' such that:

1. for all $\alpha \in A$ and all $f \in F$, $\overset{f}{\bowtie}\alpha \in A'$
2. for all $\alpha \in A$ and all $f \in F$, $\overset{f}{\lhd}\alpha \in A'$
3. for all $X, Y \in A'$, $X/Y \in A'$

[3]Strictly speaking, $\langle A, \preceq\rangle$ is defined as a join semi-lattice.

4. for all $X, Y \in A'$, $X \backslash Y \in A'$

Given morphosyntactic CCG $\langle A, F, \preceq, S, L \rangle$ over alphabet Σ, the combinatory projection of lexicon L is the closure of L under the combinatory operations of morphosyntactic CCG. For example, backward application is partially defined as follows, where $g \preceq f$:

$$\begin{array}{lll} \overset{f}{\bowtie} Y & X \backslash \overset{f}{\bowtie} Y & \Rightarrow \; X \\ \overset{f}{\lhd} Y & X \backslash \overset{f}{\bowtie} Y & \Rightarrow \; X \\ \overset{g}{\bowtie} Y & X \backslash \overset{f}{\lhd} Y & \Rightarrow \; X \\ \overset{g}{\lhd} Y & X \backslash \overset{f}{\lhd} Y & \Rightarrow \; X \end{array} \tag{5.21}$$

In addition, Bozsahin includes different modes of concatenation in his formalism, distinguishing cliticisation, affixation and normal syntactic concatenation. I am ignoring all but the last of these here, since I am only concerned with the ordering of morphemes, not with their morphosyntactic nature.

Using Bozsahin's morphosyntactic CCG formalism, it is possible to write an efficient grammar of the language of Turkish accusative nouns from Tables 5.5. Let's assume the alphabet $\{\mathsf{S}, \mathsf{N}\}$ of saturated category symbols, and the alphabet $\{\mathsf{base}, \mathsf{num}, \mathsf{poss}, \mathsf{acc}\}$ of morphosyntactic features for nouns. The meanings of the four morphosyntactic features are as before, except that now they are true independent features whereas before they only had mnemonic value. Let's also assume that the morphosyntactic features are ordered as follows:

$$\mathsf{base} \preceq \mathsf{num} \preceq \mathsf{poss} \preceq \mathsf{acc} \tag{5.22}$$

Using these morphosyntactic features, the nominal saturated categories from the baseline CCG lexicon in (5.19) can be simulated using the $\bowtie$ modality i.e. $\overset{\mathsf{base}}{\bowtie} \mathsf{N}$ is equivalent to $\mathsf{N}_{\mathsf{base}}$, $\overset{\mathsf{num}}{\bowtie} \mathsf{N}$ is equivalent to $\mathsf{N}_{\mathsf{num}}$, and so on. Furthermore, the $\lhd$ modality can be used to simulate *sets* of these saturated categories, when taken together with the ordering on morphosyntactic features in (5.22). For example, $\overset{\mathsf{num}}{\lhd} \mathsf{N}$ is equivalent to the set $\{\mathsf{N}_{\mathsf{base}}, \mathsf{N}_{\mathsf{num}}\}$ i.e. those noun-forms which are marked up to and including number, but not beyond. To take another example, $\overset{\mathsf{poss}}{\lhd} \mathsf{N}$ is equivalent to the set $\{\mathsf{N}_{\mathsf{base}}, \mathsf{N}_{\mathsf{num}}, \mathsf{N}_{\mathsf{poss}}\}$ i.e. those noun-forms which are marked up to and including possessive, but without a case marker.

Thus, the following morphosyntactic CCG lexicon generates all and only the well-formed accusative-case Turkish nouns from Table 5.5:

(5.23) $\textsf{adam} \vdash {}^{\textsf{base}}\bowtie \textsf{N}$

$\textsf{-lar} \vdash {}^{\textsf{num}}\bowtie \textsf{N} \backslash {}^{\textsf{base}}\lhd \textsf{N}$

$\textsf{-ım} \vdash {}^{\textsf{poss}}\bowtie \textsf{N} \backslash {}^{\textsf{num}}\lhd \textsf{N}$

$\textsf{-ı} \vdash {}^{\textsf{acc}}\bowtie \textsf{N} \backslash {}^{\textsf{poss}}\lhd \textsf{N}$

This morphosyntactic CCG is used in derivations like the following:

(5.24)

$$\dfrac{\dfrac{\dfrac{\textsf{adam} \quad \textsf{lar}}{{}^{\textsf{num}}\bowtie \textsf{N}}<}{\textsf{adam-lar}} \quad \textsf{ım}}{{}^{\textsf{poss}}\bowtie \textsf{N}}< \quad \textsf{ı}}{{}^{\textsf{acc}}\bowtie \textsf{N}}<$$

Note that the saturated categories ${}^{\textsf{base}}\bowtie \textsf{N}$ and ${}^{\textsf{base}}\lhd \textsf{N}$ are compatible, since the latter includes the former. A further example illustrating how the modalities interact is:

(5.25)

$$\dfrac{\dfrac{\textsf{adam} \quad \textsf{ım}}{{}^{\textsf{poss}}\bowtie \textsf{N}}< \quad \textsf{ı}}{{}^{\textsf{acc}}\bowtie \textsf{N}}<$$

Note here the first backward application — the categories ${}^{\textsf{base}}\bowtie \textsf{N}$ and ${}^{\textsf{num}}\lhd \textsf{N}$ are compatible because the former (unmarked nouns) is included within the latter (nouns marked up to and including possessive).

In addition, all twelve of the baseline CCG categories in (5.21) needed for a typical transitive verb in Turkish can be reduced to a single morphosyntactic CCG category:[4]

(5.26) $\textsf{sev-er} \vdash (\textsf{S} \backslash {}^{\textsf{poss}}\lhd \textsf{N}) \backslash {}^{\textsf{acc}}\lhd \textsf{N}$

It is clear that this morphosyntactic CCG for Turkish accusative noun and transitive verbs in (5.23) and (5.26) is much more economical than the baseline CCG equivalent in (5.19) and (5.21). This is because its functionality ratio will be much smaller since:

[4] I am ignoring the issue of morphosyntactic features for the saturated category symbol S.

1. the baseline CCG lexicon mapped the possessive suffix to *two* lexical categories, but the morphosyntactic CCG lexicon maps it to just *one*

2. the baseline CCG lexicon mapped the accusative suffix to *three* lexical categories, but the morphosyntactic CCG lexicon again maps it to just *one*

3. the baseline CCG lexicon mapped transitive verbs to *twelve* lexical categories, but yet again the morphosyntactic CCG lexicon maps it to just the *one* (issues of local scrambling notwithstanding)

Thus, the morphosyntactic CCG formalism of Bozsahin (2002) has one key advantage over baseline CCG — it allows for an economical treatment of the word-internal grammar of nouns and verbs in Turkish, and other morphologically rich languages, where functional affixes are optional yet typically occur in fixed orders with respect to each other. However, this advantage is shared by the T-CCG formalism defined in section 5.3, where the alphabet of saturated category symbols is organised into a type hierarchy.

Note first of all that it is possible to construct a T-CCG lexicon for the language of Turkish nominative and accusative noun-forms, which satisfies the criterion of functionality. I start by assuming the following alphabet of saturated category symbols:

N any noun-form, including basic forms and all kinds of legal inflected forms

$\mathbf{N_{acc}}$ accusative-marked noun-forms e.g. *adam-ı*, *adam-lar-ı*, *adam-lar-ım-ı*

$\mathbf{N_{nacc}}$ noun-froms without accusative marking e.g. *adam*, *adam-lar*, *adam-ım*

$\mathbf{N_{poss}}$ noun-forms which are marked for possessive but not accusative case e.g. *adam-ım*, *adam-lar-ım*

$\mathbf{N_{nmbs}}$ noun-forms which are *not* inflected for either case or possessor, but possibly for number e.g. *adam* and *adam-lar*

$\mathbf{N_{num}}$ noun-forms which are plural-marked but not possessive or accusative marked e.g. *adam-lar*

$\mathbf{N_{base}}$ noun-formss which have no inflections at all (i.e. *adam*).

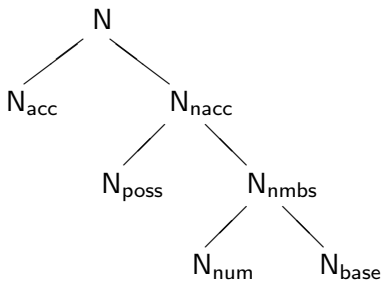


Figure 5.5: A type hierarchy for Turkish nouns

I also assume that this alphabet of saturated category symbols is organised into the type hierarchy in Figure 5.5, and that the T-CCG lexicon for Turkish includes the following fragment:

(5.27) $\textsf{adam} \vdash \mathsf{N}_{\mathsf{base}}$

$\textsf{-lar} \vdash \mathsf{N}_{\mathsf{num}} \backslash \mathsf{N}_{\mathsf{base}}$

$\textsf{-ım} \vdash \mathsf{N}_{\mathsf{poss}} \backslash \mathsf{N}_{\mathsf{nmbs}}$

$\textsf{-ı} \vdash \mathsf{N}_{\mathsf{acc}} \backslash \mathsf{N}_{\mathsf{nacc}}$

This T-CCG lexicon suffices to generate all and only the legal Turkish noun forms in Table 5.5. For example, here is a T-CCG derivation of the accusative form *adam-lar-ım-ı*:

(5.28)

$$\dfrac{\dfrac{\dfrac{\textsf{adam} \quad \textsf{lar}}{\mathsf{N}_{\mathsf{base}} \quad \mathsf{N}_{\mathsf{num}} \backslash \mathsf{N}_{\mathsf{base}}} < \quad \textsf{ım}}{\mathsf{N}_{\mathsf{num}} \quad \mathsf{N}_{\mathsf{poss}} \backslash \mathsf{N}_{\mathsf{nmbs}}} < \quad \textsf{ı}}{\mathsf{N}_{\mathsf{poss}} \quad \mathsf{N}_{\mathsf{acc}} \backslash \mathsf{N}_{\mathsf{nacc}}} <$$
$$\mathsf{N}_{\mathsf{acc}}$$

Note how type subsumption does the work here of Bozshahin's ordered morphosyntactic features. For example the second application operation can precede because saturated categories $\mathsf{N}_{\mathsf{nmbs}}$ and $\mathsf{N}_{\mathsf{num}}$ are compatible in the type hierarchy in Figure 5.5, the former being a supertype of the latter. A further example derivation showing subsumption at work is:

(5.29)

$$\dfrac{\dfrac{\text{adam}}{\mathsf{N_{base}}} \quad \dfrac{\dfrac{\text{ım}}{\mathsf{N_{poss}\backslash N_{nmbs}}}}{\mathsf{N_{poss}}}< \quad \dfrac{\dfrac{\text{ı}}{\mathsf{N_{acc}\backslash N_{nacc}}}}{}}{\mathsf{N_{acc}}}<$$

The T-CCG lexicon in (5.27) satisfies the criterion of functionality, since no lexical form has more than one category assignment. As such it is equivalent to the morphosyntactic CCG lexicon in (5.23) and better than the baseline CCG lexicon in (5.19). Transitive verbs can also be assigned to a single lexical category in T-CCG:

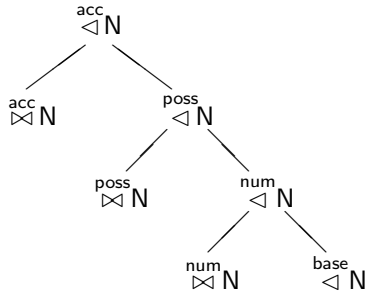
(5.30) $\mathsf{sev\text{-}er} \vdash (\mathsf{S}\backslash\mathsf{N_{nacc}})\backslash\mathsf{N}$

The T-CCG lexicon of this mini-language of Turkish accusative nouns is thus equally as efficient as the morphosyntactic CCG from the previous section, in that it has an identical, perfect functionality ratio. This fact is not coincidental. Indeed, there is a very close relationship between the morphosyntactic CCG in (5.22) and (5.23) and the T-CCG in Figure 5.5 and (5.27). Recall that the morphosyntactic CCG included the following linear ordering on morphosyntactic features:

(5.31) $\mathsf{base} \preceq \mathsf{num} \preceq \mathsf{poss} \preceq \mathsf{acc}$

As noted by Erkan (2003), this ordering can be converted directly into the following type hierarchy, where the composite morphosyntactic categories are reinterpreted as names of atomic types:

(5.32)



This type hierarchy is clearly homomorphic to that in Figure 5.5. Any linear ordering of morphosyntactic features can be turned into such a binary type hierarchy, and non-linear orderings can also be converted into more complex kinds of hierarchy.

In conclusion, this section has argued that the baseline CCG category notation defined in section 2.1 is unable to capture the morphological structure of Turkish noun-forms,

since there is no baseline CCG lexicon which can geneate all and only the well-formed instances, whilst satisfying the criterion of functionality. Thus the baseline notation appears to be missing some kind of significant generalisation. I then discussed the morphosyntactic CCG formalism of Bozsahin (2002), which was designed, at least in part, with this problem in mind. Morphosyntactic CCG assumes a category notation where every saturated category symbol is prefixed by unary modalities encoding minor syntactic features such as number and case, these features themselves being organised into a kind of hierarchy. I then showed that, since a morphosyntactic CCG lexicon can be easily converted into a T-CCG lexicon which captures exactly the same generalisations in its type hierarchy, it is reasonable to conclude that the morphosyntactic CCG notation, with its unary modalities $\bowtie$ and $\lhd$ and its ordered syntactic features, is just another way of representing a T-CCG. It is unclear whether the reverse is true as well — there is no obvious way in which the more complex type hierarchy in Figure 5.2 can be converted into an ordering of syntactic features.

5.5.2 T-CCG and typed feature structures

Having argued that T-CCG subsumes the morphosyntactic CCG notation of Bozsahin (2002), where saturated category symbols are prefixed by unary modalities encoding ordered minor syntactic features, I turn to those extensions of CCG which handle saturated categories not as atomic symbols, but rather as 'typed feature structures'.

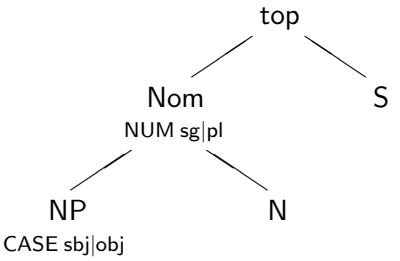
In both Steedman (1996) and Steedman (2000), it is clear that saturated CCG categories cannot simply be viewed as atomic symbols, but must involve some kind of internal structure. As Steedman (1996) puts it (p.10):

> Although for many purposes this notation will continue to be sufficient, symbols like *S*, *NP*, and $S\backslash NP$ can, and in fact must, be regarded as complex objects that include both major syntactic features, of the kind used in $\bar{\text{X}}$ theory, and minor syntactic features like number, gender and person agreement. Such syntactic feature-bundles will for present purposes be abbreviated as *S*, NP_{3S}, $S\backslash NP_{3S}$, and so on, since the particular set is not at issue here and the precise implementation of minor features of feature bundles like agreement is of no immediate relevance.

Baldridge (2002), inspired by Villavicencio (2002), takes this idea to its obvious conclusion, proposing that CCG saturated categories be implemented as *typed feature structures*, in the sense of Carpenter (1992) and Copestake (2002). This idea was worked out in detail in Erkan (2003).

Category notations based on feature structures almost inevitably take us back into the realm of the model-theoretic, or constraint-based, paradigm, discussed in subsection 1.1.3.[5] The basis of any set of typed feature structures is a *type signature*, essentially a type hierarchy of saturated category symbols, each of which is associated with appropriate features and sets of values. A simple example is as follows:

(5.33)



This signature is to be interpreted in the following manner:

1. every saturated category is either of type Nom or of type S, but not both
2. every saturated category of type Nom has exactly one NUM feature, whose value must be either sg or pl
3. every saturated category of type Nom is either of type NP or of type N, but not both
4. every saturated category of type NP has exactly one CASE feature, whose value must be either sbj or obj
5. no saturated category of type N has any other syntactic feature apart from NUM
6. no saturated category of type NP has any other syntactic feature apart from NUM and CASE
7. no saturated category of type S has a syntactic feature

Note that types *inherit* the appropriate features of their supertypes. For example, in the signature in (5.33) the feature NUM is appropriate for both the NP and N types, but not the S one.

When conceptualised as a typed feature structure, a saturated category over signature S is a kind of directed, acyclic graph. In the case of CCG saturated categories,

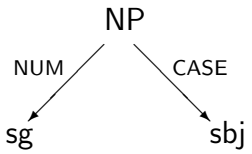
[5] See for example Kasper and Rounds (1986), Gazdar et al. (1988), Johnson (1988), Blackburn (1993) for discussion of this point.

which are by definition non-recursive, 'flat' structures, this means that a saturated category symbol over a given type signature has the following features:

1. there is a root point labelled with one of the *maximal* types in the signature
2. there are arcs departing from the root point, one for each appropriate feature for its type
3. each non-root point carries an appropriate value, depending on the feature linking it to the root

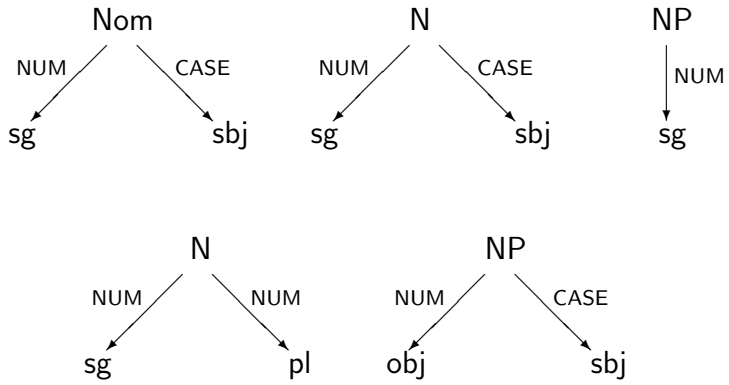
Note that according to this definition, the feature structures underlying saturated CCG categories are *sort-resolved* (every root carries a maximal type), *well-typed* (every feature is appropriate to the type), and indeed *totally well-typed* (every appropriate feature must be present). An example of a typed feature structure based on the type signature in (5.33) is as follows:

(5.34)



Note however that none of the following feature structures are legal according to the type signature in (5.33):

(5.35)



The first is not sort-resolved, since the root type Nom is not a maximal type in signature (5.33). The second is sort-resolved but not well-typed, since CASE is not an appropriate feature for N. The third is sort-resolved and well-typed but not totally well-typed,

since NP requires a CASE feature. The fourth is not a feature structure, as two distinct arcs have the same feature. Finally, the fifth graph has an inappropriate value for the NUM feature.

In the theory of typed feature structures, a distinction is drawn between the structures themselves and formulas which describe them. A satisfaction relation determines which formulas are satisfied by which feature structures. The usual way of representing a feature structure description is as an 'attribute-value matrix' (AVM), for example:

(5.36)
$$\mathsf{NP}\left[\begin{array}{ll}\mathsf{NUM} & \mathsf{sg}\\ \mathsf{CASE} & \mathsf{sbj}\end{array}\right]$$

According to the satisfaction relation, this AVM is satisfied by the feature structure in (5.34). Since AVMs can be underspecified, the following are also all satisfied by this structure:

(5.37)
$$\mathsf{Nom}\left[\begin{array}{ll}\mathsf{NUM} & \mathsf{sg}\\ \mathsf{CASE} & \mathsf{sbj}\end{array}\right] \qquad \mathsf{top}\left[\begin{array}{ll}\mathsf{NUM} & \mathsf{sg}\end{array}\right] \qquad \mathsf{NP}\left[\begin{array}{ll}\mathsf{CASE} & \mathsf{sbj}\end{array}\right] \qquad \mathsf{NP}\left[\ \right]$$

Although such AVMs constitute the standard notation for feature structure descriptions, it is customary to use a one-dimensional abbreviation in CCG derivations. For example, the AVM in (5.36) can also be written as $\mathsf{NP}_{[\mathsf{NUM}=\mathsf{sg},\mathsf{CASE}=\mathsf{sbj}]}$, or even as $\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$, where the feature labels are left implicit.

When saturated categories are modelled as typed feature structures described by AVMs, a CCG grammar over alphabet Σ has at least the following parts:

1. an alphabet A of saturated category symbols organised into a type signature over features F and values V
2. a distinguished symbol from A
3. a lexical mapping L from Σ to CCG categories over the set of AVMs over A, F and V

The combinatory projection of this kind of lexicon is defined similarly to the baseline CCG in subsection 2.3.2. In particular, the base of the projection is defined as:

(5.38) $\{\langle s, \mathcal{M}\rangle \mid \exists \langle s, \phi\rangle \in L, \mathcal{M} \models \phi\}$

$\mathsf{John} \vdash \mathsf{NP}_{[\mathsf{NUM=sg}]}$	$\mathsf{girl} \vdash \mathsf{N}_{[\mathsf{NUM=sg}]}$
$\mathsf{I} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=sbj}]}$	$\mathsf{me} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=obj}]}$
$\mathsf{we} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=sbj}]}$	$\mathsf{us} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=obj}]}$
$\mathsf{you} \vdash \mathsf{NP}_{[\mathsf{NUM=pl}]}$	$\mathsf{he} \vdash \mathsf{NP}_{[\mathsf{NUM=sg,CASE=sbj}]}$
$\mathsf{him} \vdash \mathsf{NP}_{[\mathsf{NUM=sg,CASE=obj}]}$	$\mathsf{they} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=sbj}]}$
$\mathsf{them} \vdash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=obj}]}$	$\mathsf{s} \vdash \mathsf{Nom}_{[\mathsf{NUM=pl}]} \backslash \mathsf{N}_{[\mathsf{NUM=sg}]}$
$\mathsf{the} \vdash \mathsf{NP}_{[\mathsf{NUM=sg}]} / \mathsf{N}_{[\mathsf{NUM=sg}]}$	$\mathsf{the} \vdash \mathsf{NP}_{[\mathsf{NUM=pl}]} / \mathsf{N}_{[\mathsf{NUM=pl}]}$
$\mathsf{love} \vdash (\mathsf{S} \backslash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=obj}]}) / \mathsf{NP}_{[\mathsf{CASE=obj}]}$	
$\mathsf{s} \vdash ((\mathsf{S} \backslash \mathsf{NP}_{[\mathsf{NUM=sg,CASE=sbj}]}) / \mathsf{NP}_{[\mathsf{CASE=obj}]}) \backslash ((\mathsf{S} \backslash \mathsf{NP}_{[\mathsf{NUM=pl,CASE=sbj}]}) / \mathsf{NP}_{[\mathsf{CASE=obj}]})$	

Table 5.6: A typed feature structure-based CCG lexicon for the language schematised in Table 5.1

In other words, the base is the projection of the lexicon under the satisfaction definition. The combinatory projection itself is the closure of the lexicon under the CCG combinatory operations, as before.

Table 5.6 contains a CCG lexicon for the fragment of English schematised in Table 5.1. The underlying type signature is of course that in (5.33). The following derivation is based on this lexicon:

(5.39)

you	love	the	girl
$\mathsf{NP}_{[\mathsf{NUM=pl}]}$	$(\mathsf{S} \backslash \mathsf{NP}_{[\mathsf{CASE=sbj,NUM=pl}]}) / \mathsf{NP}_{[\mathsf{CASE=obj}]}$	$\mathsf{NP}_{[\mathsf{NUM=sg}]} / \mathsf{N}_{[\mathsf{NUM=sg}]}$	$\mathsf{N}_{[\mathsf{NUM=sg}]}$
		$\mathsf{NP}_{[\mathsf{NUM=sg}]}$ (>)	
	$\mathsf{S} \backslash \mathsf{NP}_{[\mathsf{CASE=sbj,NUM=pl}]}$ (>)		
S (<)			

Note that AVM compatibility is determined by the familiar notion of 'unification', according to which two AVMs are unifiable just in case there is some set of feature structures which satisfy both of them, and their unification is the most specific AVM which is satisfied by the whole set.

Obviously the major benefit of implementing saturated categories as typed feature structures satisfied by AVMs is that it is possible to construct human language lexicons which come closer to satisfying the criterion of functionality, and thus capture a greater range of significant linguistic generalisations. To this extent, this kind of generalised CCG (let's call it AVM-CCG for now) is equivalent to the T-CCG formalism.

Indeed, there is a very close relationship between AVM-CCG and T-CCG — every AVM-CCG can be converted into a T-CCG whose lexicon has exactly the same functionality ratio. This is basically a result of the fact that every type signature, with its vocabulary of types, features and values, can be converted into a straightforward type hierarchy of atomic category symbols. For example, the type signature in (5.33) can be converted into the type hierarchy in Figure 5.2 on page 154 by simply compiling out the feature-value pairs as independent partitions and then applying the conjunctive type construction algorithm. Thus, there is a significant redundancy in such type signatures since they contain two distinct mechanisms, type subsumption and feature appropriateness, each of which can do the job of the other on its own.

Therefore, I contend that the extra vocabulary of features and values is essentially redundant, since in each AVM-CCG there is a *finite* number of features, each taking one of a *finite* number of *atomic* values. There is however one sense in which the AVM machinary can be construed as providing greater descriptive power — if we assume that the value of a feature in an AVM can be a variable, coindexed with another feature value in another AVM. To illustrate this, recall that the definite article was assigned to two distinct lexical categories in the AVM-CCG lexicon in Table 5.6, one for each number distinction:

(5.40) $\mathsf{the} \vdash \mathsf{NP}_{[\mathsf{NUM}=\mathsf{sg}]}/\mathsf{N}_{[\mathsf{NUM}=\mathsf{sg}]}$
$\mathsf{the} \vdash \mathsf{NP}_{[\mathsf{NUM}=\mathsf{pl}]}/\mathsf{N}_{[\mathsf{NUM}=\mathsf{pl}]}$

If we enrich the feature structure description language to include variables $x, y, z, \ldots$, along with an appropriate satisfaction definition and proof theory, these could be subsumed into just one lexical item:

(5.41) $\mathsf{the} \vdash \mathsf{NP}_{[\mathsf{NUM}=x]}/\mathsf{N}_{[\mathsf{NUM}=x]}$

In this case, the AVM-CCG lexicon would have an even lower functionality ratio than the corresponding T-CCG lexicon, and hence there would be linguistically significant generalisations which can be captured by the former but not the latter.

However, it should be noted at this point that adding AVMs with variables as values to the formalism is not the only way to capture these generalisations. This will become clear in the next chapter, where I discuss the incorporation of lexical inheritance hierarchies into the CCG formalism, but for now I simply point out that this problem can also be solved at the level of semantic representation. Recall from subsection 2.4.2

that dependency-based semantic representations can be incorporated into CCG lexicons as follows, where the subscripts are to be understood as referential indices and each lexical item is associated with a set of constraints:

(5.42) $\begin{array}{lcl} \textsf{John} & \vdash & \textsf{NP}_x : \mathit{john}'(x) \\ \textsf{Mary} & \vdash & \textsf{NP}_x : \mathit{mary}'(x) \\ \textsf{loves} & \vdash & (\textsf{S}_x\backslash\textsf{NP}_y)/\textsf{NP}_z : \mathit{loving}'(x),\ \mathit{sbj}'(x,y),\ \mathit{obj}'(x,z) \end{array}$

This kind of lexicon can be used in derivations like the following:

(5.43)
$$\dfrac{\dfrac{\dfrac{\textsf{John}}{\textsf{NP}_x : \mathit{john}'(x)}\quad \dfrac{\textsf{loves}}{(\textsf{S}_y\backslash\textsf{NP}_x)/\textsf{NP}_z : \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z)}}{\textsf{S}_y/(\textsf{S}_y\backslash\textsf{NP}_x) : \mathit{john}'(x)}\,{>}\mathbf{T}\quad \dfrac{\textsf{Mary}}{\textsf{NP}_z : \mathit{mary}'(z)}}{\textsf{S}_y/\textsf{NP}_z : \mathit{john}'(x),\ \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z)}\,{>}\mathbf{B}1}{\textsf{S} : \mathit{john}'(x),\ \mathit{loving}'(y),\ \mathit{sbj}'(y,x),\ \mathit{obj}'(y,z),\ \mathit{mary}'(z)}\,{>}$$

Recall that such a notion of semantic construction used here is explicitly *monotonic*, since the semantic operation associated with each of the CCG combinatory operations is essentially set union, and thus no information gets deleted during a derivation.

The key to capturing the generalisation in (5.41) is to assume that agreement features (person, number and gender) are actually properties of the referential indices themselves rather than of the saturated category symbols. Let's simplify things for the moment by assuming that proposition $\mathit{sg}'(x)$ means 'x is third person singuar' whilst proposition $\mathit{pl}'(x)$ means 'x is not third person singular'. Obviously, these are mutually exclusive:

(5.44) $\forall x.\mathit{sg}'(x) \leftrightarrow \neg \mathit{pl}'(x)$

Using this technique, we can formulate the T-CCG lexicon in Table 5.7 for the fragment of English schematised in Table 5.1. This lexicon assumes a type hierarchy of saturated category symbols where there is a general type Nom, subsuming both NP and N; in addition the NP type subsumes both $\textsf{NP}_{\textsf{sbj}}$ and $\textsf{NP}_{\textsf{obj}}$. The important lexical entry to note here is the one for the definite article:

(5.45) $\textsf{the} \vdash \textsf{NP}_x/\textsf{N}_x$

Note that the argument and result are referentially coindexed. Definite noun phrases are constructed as follows:

John $\vdash \mathsf{NP}_x : sg'(x)$	girl $\vdash \mathsf{N}_x : sg'(x)$
I $\vdash \mathsf{NP}_{\mathsf{sbj}_x} : pl'(x)$	me $\vdash \mathsf{NP}_{\mathsf{obj}_x} : pl'(x)$
we $\vdash \mathsf{NP}_{\mathsf{sbj}_x} : pl'(x)$	us $\vdash \mathsf{NP}_{\mathsf{obj}_x} : pl'(x)$
you $\vdash \mathsf{NP}_x : pl'(x)$	he $\vdash \mathsf{NP}_{\mathsf{sbj}_x} : sg'(x)$
him $\vdash \mathsf{NP}_{\mathsf{obj}_x} : sg'(x)$	they $\vdash \mathsf{NP}_{\mathsf{sbj}_x} : pl'(x)$
them $\vdash \mathsf{NP}_{\mathsf{obj}_x} : pl'(x)$	s $\vdash \mathsf{Nom}_x\backslash\mathsf{N}_y : sg'(y), pl'(x)$
the $\vdash \mathsf{NP}_x/\mathsf{N}_x$	
love $\vdash (\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}_x})/\mathsf{NP}_{\mathsf{obj}_y} : pl'(x)$	
s $\vdash ((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}_x})/\mathsf{NP}_{\mathsf{obj}})\backslash((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}_y})/\mathsf{NP}_{\mathsf{obj}}) : pl'(y), sg'(x)$	

Table 5.7: A T-CCG lexicon for the language schematised in Table 5.1, with dependency semantics

(5.46)
$$\dfrac{\dfrac{\dfrac{\text{the}}{\mathsf{NP}_x/\mathsf{N}_x}\quad\dfrac{girl}{\mathsf{N}_x : sg'(x)}}{\mathsf{NP}_x : sg'(x)}>}{\mathsf{NP}_x : sg'(y), pl'(x)}> \qquad \dfrac{\dfrac{\dfrac{\text{the}}{\mathsf{NP}_x/\mathsf{N}_x}\quad\dfrac{girl}{\mathsf{N}_y : sg'(y)}\quad\dfrac{-\text{s}}{\mathsf{Nom}_x\backslash\mathsf{N}_y : sg'(y), pl'(x)}}{\mathsf{Nom}_x : sg'(y), pl'(x)}<}{\mathsf{NP}_x : sg'(y), pl'(x)}>$$

If a plural noun phrase then attempts to combine with a verb requiring a singular argument the following happens:

(5.47)
$$\dfrac{\dfrac{\dfrac{\text{the girls}}{\mathsf{NP}_x : pl'(x)}\quad\dfrac{\text{loves}}{(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}_x})/\mathsf{NP}_{\mathsf{obj}_z} : sg'x}}{\mathsf{S}/(\mathsf{S}\backslash\mathsf{NP}_x) : pl'(x)}>\mathbf{T}}{\mathsf{S}/\mathsf{NP}_{\mathsf{obj}} : pl'(x), sg'(x)}>\mathbf{B}$$

The resulting semantic representation is obviously inconsistent with the requirement in (5.44) that no referent be both singular and plural.

Thus, the dependency semantic representations can be used to allow coindexation of referential indices, and thus provide a T-CCG which assigns the definite article to a single lexical category. This crucially relies on agreement features being properties of the indices themselves, an assumption which has independent motivation. Indeed formalisms like Head-Driven Phrase Structure Grammar (Pollard and Sag, 1987) and Dicourse Representation Theory (Kamp and Reyle, 1993) also make this assumption.

The reason for this is that such features are typically implicated in discourse dependencies such as the relation between a pronoun and its antecedent, and thus the information that a given entity is grammatically singular or masculine or whatever must be available beyond the derivation of a sentence. In other words, it must be available in semantic representations.

A nice example of this comes from the analysis of Chicheŵa in Bresnan and Mchombo (1987). Consider the following two-sentence discourse:

(5.48) Fîsi anagúlá chipéwá ku San Francíscó dzulo.
hyena bought hat(7) in San Francisco yesterday
The hyena bought a hat in San Francisco yesterday.

(5.49) Madzŭlo anapítá ku San Jose kuméné á-ná-ká-chí-gulítsá kw'á ḿlóndá.
evening he-went to San Jose where he-PAST-go-it(7)-sell to guard.
In the evening he went to San Jose, where he went to sell it to the guard.

Note that the noun *chipéwá* ('hat') in the first sentence belongs to gender class 7 in Chicheŵa, and that the gender class of a noun is not in general predictable by its natural gender. The second sentence contains an incorporated pronoun *chí* whose antecedent must be of gender class 7. Thus, in order to determine the referent of this pronoun, the parser must have information about the gender class of every discourse referent created thus far, including those from previous sentences. Similar arguments can be found for including person and number information in semantic structure, rather than viewing them as purely syntactic features.

5.6 Conclusion

This chapter presented T-CCG, the 'type-hierarchical' CCG formalism, a version of CCG where the alphabet of saturated category symbols is organised into a *type hierarchy*, thus allowing more efficient CCG lexicons to be constructed for human languages.

Section 5.1 exemplified the problem of lexical redundancy in baseline CCG grammars, with a simple example involving subject-verb agreement in a fragment of English. I introduced two ideals for human language lexicons — the criteria of functionality (lexicons should be functions) and atomicity (lexical types should be atomic), and show

that it is simply not possible to formulate CCGs which even come close to satisfying these using only the baseline CCG category notation defined in chapter 2.

Section 5.2 formally defined the notion of a 'type hierarchy', based on the discussion in Carpenter (1992), as a particular kind of weak partial order.

Section 5.3 presented an explicit definition of the T-CCG formalism, as a minimal extension of baseline CCG. I specified a class of grammars, and a 'generates' relation defined through the combinatory projection of a T-CCG lexicon.

Section 5.4 then presented a proof system for the T-CCG formalism, allowing grammaticality to be determined in terms of operations on the T-CCG category descriptions themselves, rather than through the underlying models.

Section 5.5 demonstrated that the T-CCG formalism allows more efficient grammars to be written for natural languages than does CCG, returning to the example of subject-verb agreement in English. In particular, it is possible to construct T-CCG lexicons which come closer to satisfying the criterion of functionality. I argued that the type hierarchy of saturated categories renders redundant many other proposed CCG categorial notations. For example, the morphosyntactic CCG formalism of Bozsahin (2002), where minor syntactic features are represented by unary modalities prefixed to saturated category symbols, is subsumed by the T-CCG category notation. In addition, the T-CCG formalism also subsumes versions of CCG where saturated categories are conceptualised as typed feature structures, unless it is assumed that coindexed variables can act as feature-values — even in the latter case we can make use of the referential indices in dependency-type semantic representations to capture this kind of coindexation.

The moral of this chapter is that we can and should be discriminating in our choice of which aspects of unification-based formalisms like the typed-default feature structure (TDFS) system of Copestake (2002) to incorporate into a theory of the CCG lexicon. I contend that, although multiple inheritance type hierarchies are undoubtedly a necessary part of such a theory, it is debatable whether the expressivity benefits gained from conceptualising saturated categories as typed feature structures outweigh the cost of implementing them.

Chapter 6

Inheritance-driven CCG

Chapter 2 of this thesis defined a so-called 'baseline' CCG formalism, founded on a model-theoretic conception of the theory of category notation, distinguishing between underlying category models and category descriptions. Chapters 3 and 4 then evaluated this baseline formalism as a theory of human linguistic competence. It was argued that, although there are some implicational universals which are predicted by baseline CCG either in terms of generative capacity or a simplicity-based grammar ranking, there are others which are not, though we have an intuitive idea that they are 'economy-based' in some way. I concluded that we need to add some extra descriptive machinery to the baseline CCG formalism, if we want it to explain these universals in terms of an acquisition-based preference for shorter grammars. The procedure to be followed is as follows: to keep the underlying category models and combinatory operations from chapter 2 pretty much as is, and to experiment with different systems of category description.

Chapter 5 began the process of generalising the CCG category notation to this end. I proposed the T-CCG formalism, where the alphabet of saturated category symbols is organised into a type hierarchy based on a subsumption ordering. It was shown that this version of CCG allows us to construct human language lexicons which encode linguistic information more efficiently. To be precise, such lexicons come closer to satisfying the 'ideal of functionality' which states that an ideal human language lexicon is ideally a *function* from forms to category descriptions i.e. words and morphemes are mapped to exactly one lexical category description in the lexical encoding.

This chapter continues the process of constructing a more flexible category notation, so as to provide more efficient lexicons and a more subtle kind of lexical descrip-

tive power. I present a version of the CCG formalism incorporating an inheritance hierarchy of lexical types over a simple but flexible, feature-based constraint language. I call this formalism 'inheritance-driven' CCG (or I-CCG for short), and demonstrate that it allows for the construction of maximally efficient human language lexicons.

This chapter will proceed as follows:

Section 6.1 presents the new category description language underlying the I-CCG formalism. Whereas the original class of category descriptions from Chapter 2 was just the traditional CCG category notation, the new language represents a radical departure, inspired by the unification-based categorial grammar implementations of, for example, Uszkoreit (1986) and Zeevat et al. (1987). This new constraint language is based on a simple notion of embeddable attribute-value pairs, and allows a much more flexible approach to specifying sets of category models.

Section 6.2 illustrates the use of the new category description language in a redefinition of the T-CCG formalism of Chapter 5. I include a specification of the class of grammars, and a 'generates' relation, as well as a discussion of the corresponding proof system.

Section 6.3 introduces the notion of an 'inheritance hierarchy' over a given constraint language. Essentially, an inheritance hierarchy is simply a type hierarchy where the types carry formulas from the constraint language. The underlying intuition is that objects of some type satisfy the constraints attached to the type itself, as well as all those they 'inherit' from the supertypes.

Section 6.4 then defines the I-CCG formalism itself, including a class of grammars and a 'generates' relation. Essentially, an I-CCG is a T-CCG which incorporates an inheritance hierarchy of lexical types over the flexible category description language.

Section 6.5 demonstrates that the I-CCG formalism allows highly efficient grammars to be constructed for human languages, returning to the example involving subject-verb agreement in English. In other words, it is possible to formulate I-CCG lexicons which satisfy the ideals of functionality and atomicity. I then argue that the incorporation of a lexical inheritance hierarchy in this way renders the 'multiset' category notation of Hoffman (1995) redundant.

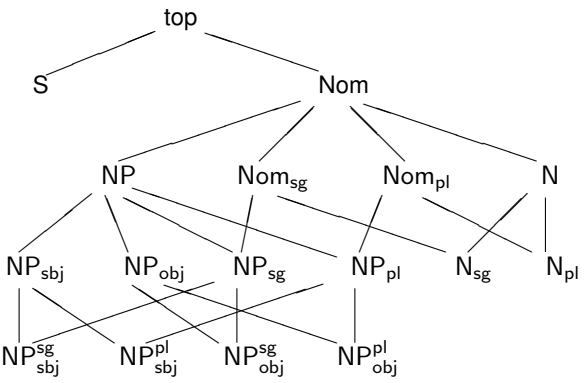
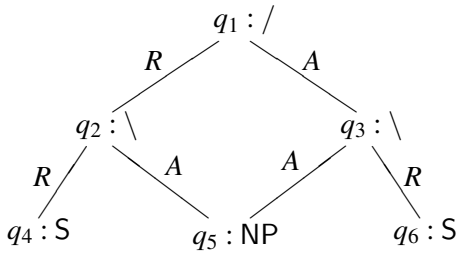


Figure 6.1: The type hierarchy from Figure 5.2 repeated

6.1 A new description language for CCG categories

In section 2.1, I proposed a formal distinction between CCG category 'models' and CCG category 'descriptions'. Recall that a category model over alphabet A of saturated categories was defined as an ordered 5-tuple $\langle Q, Res, Arg, V_S, V_A \rangle$ where: (a) Q is a set of points; (b) Res and Arg are the 'result' and 'argument' relations on Q; (c) V_S assigns a slash to every non-end point; and (d) V_A assigns a saturated category symbol from A to every end point. An example of such a category model over alphabet $\{\mathsf{S}, \mathsf{NP}\}$ is as follows:

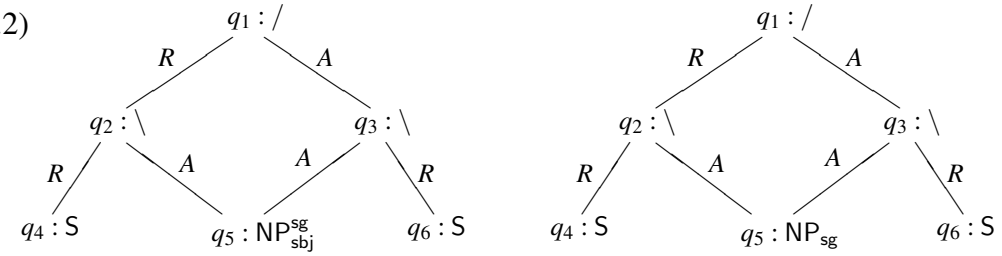
(6.1)



In subsection 5.3.1, I revised this slightly to ensure that category models are 'sort-resolved' relative to some type hierarchy: T-CCG category models over type hierarchy $\langle A, \sqsubseteq \rangle$ of saturated category symbols are ordered 5-tuples $\langle Q, Res, Arg, V_S, V_A \rangle$, subject to all the restrictions on baseline CCG category models over A, but also where V_A is limited to assigning a *maximal* type in $\langle A, \sqsubseteq \rangle$ to every end point. Take for example the type hierarchy of saturated category symbols in Figure 6.1, which we already came

across in Chapter 5. Here are two category structures whose end-points are labelled with saturated category symbols from this type hierarchy:

(6.2)



Recall that only the one on the left is actually a legal category model over the type hierarchy; the structure on the right has at least one end-point q_5 which is labelled with a non-maximal symbol and hence it does not constitute a T-CCG category model over the hierarchy.

The counterpart to category models is of course category *descriptions*. These are to be understood as logical expressions, like the formulas of predicate logic. In subsection 2.1.5, the set of category descriptions over alphabet A of saturated category symbols was defined as the smallest set Φ such that:

1. $A \subseteq \Phi$
2. for all $\phi, \psi \in \Phi$: (ϕ/ψ), $(\phi\backslash\psi)$ and $(\phi \mid \psi)$ are all in Φ

In other words, the baseline category description language is simply the traditional CCG category notation, along with the neutral slash symbol $|$. Example baseline category descriptions over the alphabet underlying the type hierarchy of saturated category symbols in Figure 6.1 are:

(6.3) $(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})$

$(\mathsf{S} \mid \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/(\mathsf{S} \mid \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})$

$(\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}}) \mid (\mathsf{top}\backslash\mathsf{Nom}_{\mathsf{sg}})$

Category models and category descriptions were linked by a relation of satisfaction, determining for every model and every description over a common alphabet, whether the model satisfies the description or not. So, for example, each of the three category descriptions in (6.3) is satisfied by the left-hand category model in (6.2), assuming the subsumption ordering in Figure 6.1.

Note that this logic of category notation does allow a limited amount of flexibility, in the sense that the mapping between category models and descriptions is not biunique. We have just seen in (6.3) that a single category model can satisfy more than one category description, thanks to the inclusion of the neutral slash | in the definition of baseline CCG category descriptions and the subsumption ordering on saturated category symbols in the T-CCG formalism.

However, if we are to progress more in our search for a theory of the CCG lexicon, we need more flexibility in the relationship between underlying category models and their descriptions. In short, we need to allow for a far greater degree of underspecification in descriptions than the T-CCG category logic allows — underspecifying argument directionality and saturated category symbols is not enough. We want to be able to pick out, for example, all category models whose argument is a noun phrase, or all category models which ultimately result in a finite sentence. This section presents a more radically flexible description language for CCG categories, based on the attribute-value notation familiar from the unification-based literature, in particular the categorial grammar implementations of Uszkoreit (1986) and Zeevat et al. (1987).

Subsection 6.1.1 specifies the class of expressions of this new constraint language.

Subsection 6.1.2 then presents the satisfaction relation between the T-CCG category models defined in subsection 5.3.1 and these flexible category descriptions.

6.1.1 Flexible category descriptions

The set of flexible category descriptions over alphabet A of saturated category symbols is defined as the smallest set Φ such that:

1. $A \subseteq \Phi$
2. for all $\delta \in \{/, \backslash\}$, $(\mathsf{SLASH}\ \delta) \in \Phi$
3. for all $\phi \in \Phi$, $(\mathsf{ARG}\ \phi) \in \Phi$
4. for all $\phi \in \Phi$, $(\mathsf{RES}\ \phi) \in \Phi$

I will briefly elaborate on the intuitive meanings of each of these four kinds of statement. Firstly, a statement consisting of a single saturated category symbol, for example $\mathsf{NP}_{\mathsf{sg}}$, picks out all simple category models whose sole point is labelled by a symbol

which it subsumes. Secondly, a statement of the form (SLASH δ), where δ is one of the two directional slashes / and \, picks out all those category models which seek their argument from the required direction. Thus, (SLASH /) picks out all those category models which need a *following* first argument, for example S/NP, (S\NP)/PP and (S\NP)/(PP\NP) but not S\NP. In a more traditional notation, an equivalent expression to (SLASH /) would be something like X/Y.

Thirdly, a statement of the form (ARG ϕ) specifies all and only the category models whose argument can be described by statement ϕ. For example, (ARG NP) picks out the category models whose argument can be described as NP e.g. S/NP and S\NP but not (S/NP)/PP. A more traditional equivalent notation would be $X|\mathsf{NP}$. Finally, a statement of the form (RES ϕ) specifies all and only category models whose result can be described by statement ϕ. For example, (RES S) picks out the category models whose result can be described as S e.g. S/NP and S\PP but not (S\NP)/PP. Again a more traditional representation of (RES S) would be $\mathsf{S}|X$.

Note that these expressions can also be embedded within one another. For example, the expression (RES(SLASH \)) denotes all categories whose result itself requires a *preceding* argument i.e. $(X\backslash Y)|Z$. In general, when writing such embedded constraints, I will drop the brackets where no confusion can arise. Thus (RES (SLASH \)) will be written simply as RES SLASH \.

6.1.2 T-CCG satisfaction

The 'global' satisfaction relation between category models and category descriptions remains unchanged from the corresponding definition for the baseline category notation in subsection 2.1.6 and subsection 5.3.3: T-CCG category model $\mathcal{M}$ over type hierarchy $\langle A, \sqsubseteq\rangle$ of saturated category symbols, as defined in subsection 5.3.1, satisfies flexible category description ϕ over A, written $\mathcal{M} \models \phi$, if and only if $\mathcal{M}, q \models \phi$, where q is the root of $\mathcal{M}$. In other words, global satisfaction remains local satisfaction from the *root point* of a model.

On the other hand, the 'local' satisfaction definition, telling us which descriptions are satisfied *from which points* in a category model, must be completely rewritten to take into account the new kinds of category description:

T-CCG category $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A\rangle$ over type hierarchy $\langle A, \sqsubseteq\rangle$ of saturated category symbols locally satisfies flexible category description ϕ over A from point $q \in Q$,

written $\mathcal{M}, q \models \phi$, if and only if:

1. where $\phi \in A$: for some $\psi \in A$ such that $\phi \sqsubseteq \psi$, $V_A(q) = \psi$
2. where $\phi = (\mathsf{SLASH}\ \delta)$: $V_S(q) = \delta$
3. where $\phi = (\mathsf{ARG}\ \psi)$: $\mathcal{M}, Arg(q) \models \psi$
4. where $\phi = (\mathsf{RES}\ \psi)$: $\mathcal{M}, Res(q) \models \psi$

The first clause in this definition states that, if we have a flexible category description consisting of a single saturated category symbol, then that description is satisfied at all and only the end points which are labelled with either the symbol itself or by one which it subsumes. Take for example the T-CCG category model $\mathcal{M}$ on the left in (6.2), defined over the type hierarchy of saturated category symbols in Figure 6.1. Note that:

(6.4)
$$\begin{aligned} \mathcal{M}, q_5 &\models \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}} \\ \mathcal{M}, q_5 &\models \mathsf{NP}_{\mathsf{sbj}} \text{ since } \mathsf{NP}_{\mathsf{sbj}} \sqsubseteq \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}} \\ \mathcal{M}, q_5 &\models \mathsf{Nom}_{\mathsf{sg}} \text{ since } \mathsf{Nom}_{\mathsf{sg}} \sqsubseteq \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}} \\ \mathcal{M}, q_5 &\not\models \mathsf{NP}_{\mathsf{obj}} \text{ since } \mathsf{NP}_{\mathsf{obj}} \not\sqsubseteq \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}} \end{aligned}$$

The second clause in the definition states that, if we have a flexible category description of the form $\mathsf{SLASH}\ \delta$, where δ is one of the two directional symbols $/$ and $\backslash$, then this description is satisfied from all and only the non-end points which carry the appropriate label δ. Thus, returning to the same category model $\mathcal{M}$ in (6.2):

(6.5)
$$\begin{aligned} \mathcal{M}, q_2 &\models \mathsf{SLASH}\ \backslash \\ \mathcal{M}, q_1 &\models \mathsf{SLASH}\ / \\ \mathcal{M}, q_3 &\not\models \mathsf{SLASH}\ / \text{ since } V_S(q_3) \neq / \end{aligned}$$

The third clause in the definition of local satisfaction states that, if we have a flexible category description of the form $\mathsf{ARG}\ \phi$, then this description is satisfied from all and only the non-end points whose 'argument' point satisfies description ϕ. So for example:

(6.6)
$$\begin{aligned} \mathcal{M}, q_2 &\models \mathsf{ARG}\ \mathsf{NP} \text{ since } Arg(q_2) = q_5 \text{ and } \mathcal{M}, q_5 \models \mathsf{NP} \\ \mathcal{M}, q_3 &\not\models \mathsf{ARG}\ \mathsf{SLASH}\ \backslash \text{ since } Arg(q_3) = q_5 \text{ and } \mathcal{M}, q_5 \not\models \mathsf{SLASH}\ \backslash \\ \mathcal{M}, q_1 &\models \mathsf{ARG}\ \mathsf{ARG}\ \mathsf{Nom}_{\mathsf{sg}} \text{ since } Arg(Arg(q_1)) = q_5 \text{ and } \mathcal{M}, q_5 \models \mathsf{Nom}_{\mathsf{sg}} \end{aligned}$$

The fourth and final clause in the definition of local satisfaction above states that, if we have a flexible category description of the form RES ϕ, then this description is satisfied from all and only the non-end points whose 'result' point satisfies description ϕ. So for example, where $\mathcal{M}$ is again the T-CCG category model in (6.2):

(6.7) $\mathcal{M}, q_2 \models$ RES S since $Res(q_2) = q_4$ and $\mathcal{M}, q_4 \models$ S

$\mathcal{M}, q_3 \not\models$ RES ARG top since $Res(q_3) = q_6$ and $\mathcal{M}, q_6 \not\models$ ARG top

$\mathcal{M}, q_1 \models$ RES ARG $\mathsf{Nom_{sg}}$ since $Arg(Res(q_1)) = q_5$ and $\mathcal{M}, q_5 \models \mathsf{Nom_{sg}}$

Finally, note that the following global satisfaction judgements hold for the T-CCG category model in (6.2), since each category description is or is not locally satisfied from the root point q_1:

(6.8) $\mathcal{M} \models$ SLASH /

$\mathcal{M} \models$ RES SLASH \

$\mathcal{M} \models$ ARG RES S

$\mathcal{M} \not\models$ ARG ARG SLASH \

In conclusion, this section started by noting the need for a more flexible category description language than the baseline CCG notation from chapter 2 provided. Subsection 6.1.1 provided such a notation, and subsection 6.1.2 showed how it is to be interpreted against the class of T-CCG category models over some type hierarchy of saturated category symbols.

6.2 T-CCG with flexible category descriptions

In order to illustrate the use of the flexible category description language defined in the previous section, this section will provide a brief discussion of how the T-CCG formalism from chapter 5 can be redefined using the new flexible notation.

Subsection 6.2.1 characterises and exemplifies the class of T-CCG grammars.

Subsection 6.2.2 defines the combinatory projection of a T-CCG lexicon.

Subsection 6.2.3 specifies the 'generates' relation between arbitrary T-CCGs over alphabet Σ and arbitrary strings over Σ.

Subsection 6.2.4 discusses the corresponding proof system.

6.2.1 T-CCGs

A T-CCG over alphabet Σ is an ordered 4-tuple $\langle A, \sqsubseteq, S, L \rangle$ where:

1. $\langle A, \sqsubseteq \rangle$ is a type hierarchy of saturated category symbols
2. S is a distinguished element of A
3. L is a finite mapping from Σ to sets of flexible category descriptions over A

This definition is the same as the previous one in subsection 5.3.4 with the following notable exception — previously a lexical item paired a phonetic form with a baseline category description; now forms are mapped to *sets* of our new *flexible* category descriptions.

Take for example the old-style T-CCG from section 5.5, which generated the simple fragment of English schematised in Table 5.1 on page 145. The type hierarchy of saturated category symbols was of course that in Figure 6.1 on page 187. The old-style T-CCG lexicon itself was the following:

(6.9)

$\mathsf{John} \vdash \mathsf{NP}_{\mathsf{sg}}$	$\mathsf{girl} \vdash \mathsf{N}_{\mathsf{sg}}$
$\mathsf{I} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{me} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$
$\mathsf{we} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{us} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$
$\mathsf{you} \vdash \mathsf{NP}_{\mathsf{pl}}$	$\mathsf{he} \vdash \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$
$\mathsf{him} \vdash \mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$	$\mathsf{they} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$
$\mathsf{them} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$	$\mathsf{love} \vdash (\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}}$
$\mathsf{s} \vdash \mathsf{Nom}_{\mathsf{pl}}\backslash\mathsf{N}_{\mathsf{sg}}$	$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{sg}}/\mathsf{N}_{\mathsf{sg}}$
$\mathsf{the} \vdash \mathsf{NP}_{\mathsf{pl}}/\mathsf{N}_{\mathsf{pl}}$	$\mathsf{s} \vdash ((\mathsf{S}\backslash\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}})\backslash((\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}})$

In order to convert this into a new-style T-CCG lexicon mapping forms to sets of flexible category descriptions, we start by taking those lexical forms which are mapped to saturated category symbols and translating them directly:

(6.10)

$\mathsf{John} \vdash \{\mathsf{NP}_{\mathsf{sg}}\}$	$\mathsf{girl} \vdash \{\mathsf{N}_{\mathsf{sg}}\}$
$\mathsf{I} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}\}$	$\mathsf{me} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}\}$
$\mathsf{we} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}\}$	$\mathsf{us} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}\}$
$\mathsf{you} \vdash \{\mathsf{NP}_{\mathsf{pl}}\}$	$\mathsf{he} \vdash \{\mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}\}$
$\mathsf{him} \vdash \{\mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}\}$	$\mathsf{they} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}\}$
$\mathsf{them} \vdash \{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}\}$	

The old-style T-CCG lexicon in (6.9) also contains five other lexical items, all of which involve pairings of lexical forms with more complex lexical category descriptions. Each of these must be translated into a set of our new-style flexible category descriptions:

(6.11)
$$\begin{aligned}
\textsf{love} &\vdash \{\textsf{SLASH}\ /,\ \textsf{ARG}\ \textsf{NP}_{\textsf{obj}},\ \textsf{RES}\ \textsf{SLASH}\ \backslash,\ \textsf{RES}\ \textsf{ARG}\ \textsf{NP}^{\textsf{pl}}_{\textsf{sbj}},\ \textsf{RES}\ \textsf{RES}\ \textsf{S}\}\\
\textsf{-s} &\vdash \{\textsf{SLASH}\ \backslash,\ \textsf{ARG}\ \textsf{N}_{\textsf{sg}},\ \textsf{RES}\ \textsf{Nom}_{\textsf{pl}}\}\\
\textsf{the} &\vdash \{\textsf{SLASH}\ /,\ \textsf{ARG}\ \textsf{N}_{\textsf{sg}},\ \textsf{RES}\ \textsf{NP}_{\textsf{sg}}\}\\
\textsf{the} &\vdash \{\textsf{SLASH}\ /,\ \textsf{ARG}\ \textsf{N}_{\textsf{pl}},\ \textsf{RES}\ \textsf{NP}_{\textsf{pl}}\}\\
\textsf{-s} &\vdash \{\textsf{SLASH}\ \backslash,\ \textsf{ARG}\ \textsf{SLASH}\ /,\ \textsf{ARG}\ \textsf{ARG}\ \textsf{NP}_{\textsf{obj}},\ \textsf{ARG}\ \textsf{RES}\ \textsf{SLASH}\ \backslash,\\
&\quad \textsf{ARG}\ \textsf{RES}\ \textsf{ARG}\ \textsf{NP}^{\textsf{pl}}_{\textsf{sbj}},\ \textsf{ARG}\ \textsf{RES}\ \textsf{RES}\ \textsf{S},\ \textsf{RES}\ \textsf{SLASH}\ /,\\
&\quad \textsf{RES}\ \textsf{ARG}\ \textsf{NP}_{\textsf{obj}},\ \textsf{RES}\ \textsf{RES}\ \textsf{SLASH}\ \backslash,\ \textsf{RES}\ \textsf{RES}\ \textsf{ARG}\ \textsf{NP}^{\textsf{sg}}_{\textsf{sbj}},\\
&\quad \textsf{RES}\ \textsf{RES}\ \textsf{RES}\ \textsf{S}\}
\end{aligned}$$

Thus, the complete new-style T-CCG lexicon for the fragment of English schematised in Table 5.1 is the set union of the lexical entries in (6.10) and (6.11).

Now, I am the first to admit that large sets of flexible category descriptions, such as that which constituted the lexical assignment for the third-person singular verbal suffix '-s', are not straightforwardly human-readable on the printed page. Thus, to make things easier on the reader I will sometimes present such sets as embedded attribute-value matrices. So, the lexical entry for the third-person singular verbal suffix can also look like the following:

(6.12)
$$\textsf{-s} \vdash \begin{bmatrix}
\textsf{SLASH} & \backslash & & \\
\textsf{ARG} & \begin{bmatrix} \textsf{SLASH} & / \\ \textsf{ARG} & \textsf{NP}_{\textsf{obj}} \\ \textsf{RES} & \begin{bmatrix} \textsf{SLASH} & \backslash \\ \textsf{ARG} & \textsf{NP}^{\textsf{pl}}_{\textsf{sbj}} \\ \textsf{RES} & \textsf{S} \end{bmatrix} \end{bmatrix} \\
\textsf{RES} & \begin{bmatrix} \textsf{SLASH} & / \\ \textsf{ARG} & \textsf{NP}_{\textsf{obj}} \\ \textsf{RES} & \begin{bmatrix} \textsf{SLASH} & \backslash \\ \textsf{ARG} & \textsf{NP}^{\textsf{sg}}_{\textsf{sbj}} \\ \textsf{RES} & \textsf{S} \end{bmatrix} \end{bmatrix}
\end{bmatrix}$$

This kind of representation could be implemented directly in the language of flexible category descriptions simply by incorporating boolean conjunction into the definition

in subsection 6.1.1. For the moment however, I will assume that the AVMs are simply a notational convenience for readers, rather than being an integral part of the description language. This will make definition of a proof system for T-CCG easier.

6.2.2 Combinatory projection of a T-CCG lexicon

The definition of the combinatory projection of a new-style T-CCG lexicon is similar to the previous old-style definition in subsection 5.3.5. The only alteration is that we need to remember that lexical forms are now mapped to *sets* of category descriptions, each element of which must be satisfied:

Given T-CCG $\langle A, \sqsubseteq, S, L\rangle$ over alphabet Σ, the combinatory projection of lexicon L is defined as the smallest set L', such that:

1. for all $\langle s, \Phi\rangle \in L$, and all category models $\mathcal{M}$ over $\langle A, \sqsubseteq\rangle$ such that for all $\phi \in \Phi$, $\mathcal{M} \models \phi$, $\langle s, \mathcal{M}\rangle \in L'$ i.e. the base of the projection is the closure of the lexicon under T-CCG satisfaction — a lexical form is mapped to all the category models which satisfy *all* of its lexical category descriptions

2. for all $\langle s, \mathcal{M}_1\rangle \in L'$ and all category models $\mathcal{M}_2$ over $\langle A \sqsubseteq\rangle$, where $\mathcal{M}_3$ is the result of either forward or backward raising $\mathcal{M}_1$ over $\mathcal{M}_2$, then $\langle s, \mathcal{M}_3\rangle \in L'$ i.e. the combinatory projection is closed under raising

3. for all $\langle s_1, \mathcal{M}_1\rangle, \langle s_2, \mathcal{M}_2\rangle \in L'$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to $\mathcal{M}_3$ by means of one of the binary CCG combinatory operations defined in section 2.2, $\langle s_1 s_2, \mathcal{M}_3\rangle \in L'$ i.e. the combinatory projection is closed under application, composition etc.

Again, the T-CCG combinatory operations are defined exactly as their CCG equivalents in section 2.2. Since they operate on category models rather than category descriptions, and since category models are by definition sort-resolved, there is still no need to redefine the operations to take account of the T-CCG subsumption ordering.

6.2.3 T-CCG generation

Finally, the 'generates' relation is exactly the same as that previously defined in subsection 5.3.6:

T-CCG $\langle A, \sqsubseteq, S, L\rangle$ over alphabet Σ generates string $s \in \Sigma^*$ if and only if for some X such that $S \sqsubseteq X$, the ordered pair $\langle s, X\rangle$ is an element of the combinatory projection of lexicon L.

6.2.4 The T-CCG proof system

Recall from section 5.4 that, after defining the T-CCG formalism in terms of category models, I also provided the beginnings of a proof system. This allowed derivations to be presented in terms of category descriptions and was intended to form the basis for a computational implementation of T-CCG. Obviously since the language of category descriptions has now changed, we will need a new set of inference rules. This subsection contains a brief discussion of what these might look like.

First of all, the lexical insertion inference rule from subsection 5.4.1 can be incorporated as is: Given T-CCG $G = \langle A, \sqsubseteq, S, L\rangle$, for all $\langle s, \Phi\rangle \in L$, $\frac{s}{\Phi}$ is a valid inference according to G. This permits us to perform the initial stage of a derivation where we line up words and morphemes with their lexical category descriptions. For example, assuming the T-CCG lexicon in (6.10) and (6.11), we get:

(6.13)

$$\frac{\textsf{they}}{\textsf{NP}^{\textsf{pl}}_{\textsf{sbj}}} \qquad \frac{\textsf{love}}{\textsf{SLASH}\ /, \textsf{ARG}\ \textsf{NP}_{\textsf{obj}}, \textsf{RES}\ \textsf{SLASH}\ \backslash,\ \textsf{RES}\ \textsf{ARG}\ \textsf{NP}^{\textsf{pl}}_{\textsf{sbj}}, \textsf{RES}\ \textsf{RES}\ \textsf{S}} \qquad \frac{\textsf{you}}{\textsf{NP}_{\textsf{pl}}}$$

Before going on to define inference rules corresponding to the CCG combinatory operations like forward and backward application, we need to redefine the notion of *compatibility* of sets of category descriptions with respect to a type hierarchy:

Set Φ of flexible category descriptions over alphabet A of saturated category symbols is $\sqsubseteq$-compatible, where $\sqsubseteq$ is a subsumption relation on A, if and only if one of the following holds:

1. $\Phi \subseteq A$ and Φ has an upper bound in $\langle A, \sqsubseteq\rangle$ i.e. this accounts for saturated category symbols
2. Φ and A are disjoint and all the following hold:
   (a) $\{/, \backslash\} \not\subseteq \{\delta \mid (\textsf{SLASH}\ \delta) \in \Phi\}$ i.e. any slashes are consistent
   (b) $\{\phi \mid (\textsf{ARG}\ \phi) \in \Phi\}$ is $\sqsubseteq$-compatible i.e. information about the argument is consistent

(c) $\{\phi \mid (\mathsf{RES}\ \phi) \in \Phi\}$ is $\sqsubseteq$-compatible i.e. information about the result is consistent

For example, where $\sqsubseteq$ is the subsumption relation over saturated category symbols in Figure 6.1 on page 187, we can state that the following set of flexible category descriptions is $\sqsubseteq$-compatible, since all three symbols have an upper bound in common i.e. $\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$:

(6.14) $\{\mathsf{NP}_{\mathsf{sbj}}, \mathsf{Nom}_{\mathsf{pl}}, \mathsf{top}\}$

However, the following set is not $\sqsubseteq$-compatible:

(6.15) $\{\mathsf{NP}_{\mathsf{sbj}}, \mathsf{Nom}_{\mathsf{pl}}, \mathsf{N}\}$

This is because there is no type in the hierarchy which is subsumed by all three of these types. Now consider following set of more complex descriptions:

(6.16) $\{\mathsf{SLASH}\ /, \mathsf{ARG}\ \mathsf{NP}_{\mathsf{obj}}, \mathsf{RES}\ \mathsf{SLASH}\ \backslash, \mathsf{RES}\ \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{RES}\ \mathsf{S}, \mathsf{ARG}\ \mathsf{NP}_{\mathsf{pl}}\}$

To find out whether this set is $\sqsubseteq$-compatible, we need to answer three simpler questions. First, does $\{\delta \mid (\mathsf{SLASH}\ \delta) \in (6.16)\}$ contain inconsistent slashes? The relevant set is simply $\{/\}$, so the answer is no. Second, is $\{\phi \mid (\mathsf{ARG}\ \phi) \in (6.16)\}$ $\sqsubseteq$-compatible? Here the relevant set is $\{\mathsf{NP}_{\mathsf{obj}}, \mathsf{NP}_{\mathsf{pl}}\}$ and a quick glance at the type hierarchy shows that these symbols are $\sqsubseteq$-compatible. So the answer to the second question is yes. Third, is $\{\phi \mid (\mathsf{RES}\ \phi) \in (6.16)\}$ $\sqsubseteq$-compatible? The relevant set is $\{\mathsf{SLASH}\ \backslash, \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{S}\}$. In order to find out whether this set of result information is $\sqsubseteq$-compatible, we need to ask the same three questions. The reader may like to verify that this set is in fact $\sqsubseteq$-compatible, and thus so is that in (6.16).

Now that we have a well-defined notion of the compatibility of sets of flexible category descriptions relative to some subsumption relation on saturated category symbols, we are in a position to define inference rules based on the combinatory operations of CCG. For example, that corresponding to forward application is as follows:

Given T-CCG $G = \langle A, \sqsubseteq, S, L\rangle$, for all sets Φ and Ψ of flexible category descriptions over A such that:

1. $(\mathsf{SLASH}\ \backslash) \notin \Phi$
2. $\{\phi \mid (\mathsf{ARG}\ \phi) \in \Phi\} \cup \Psi$ is $\sqsubseteq$-compatible

The following is a valid inference according to G:

(6.17)
$$\frac{\Phi \qquad \Psi}{\{\phi \mid (\mathsf{RES}\ \phi) \in \Phi\}}>$$

Let's illustrate this by walking through an example. Recall the derivation we started previously based on the T-CCG lexicon in (6.10) and (6.11):

(6.18)
$$\frac{\mathsf{they}}{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}} \quad \frac{\mathsf{love}}{\begin{array}{c}\mathsf{SLASH}\ /, \mathsf{ARG}\ \mathsf{NP}_{\mathsf{obj}}, \mathsf{RES}\ \mathsf{SLASH}\ \backslash, \\ \mathsf{RES}\ \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{RES}\ \mathsf{S}\end{array}} \quad \frac{\mathsf{you}}{\mathsf{NP}_{\mathsf{pl}}}$$

Can we use the forward application inference rule to combine *love* and *you* into a phrasal constituent *love you*? We need to answer two simpler questions. First, is $(\mathsf{SLASH}\ \backslash)$ in the category description of *love*? No, so forward application is not totally ruled out. Second, is $\{\mathsf{NP}_{\mathsf{obj}}, \mathsf{NP}_{\mathsf{pl}}\}$ $\sqsubseteq$-compatible? Yes, so *you* is a possible argument for *love*. Thus, the operation can go ahead. The category description of the resulting phrase is simply the following, where Φ is the set of lexical category descriptions for *love*:

(6.19) $\quad \{\phi \mid (\mathsf{RES}\ \phi) \in \Phi\} = \{\mathsf{SLASH}\ \backslash, \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{S}\}$

So the derivation can proceed as follows:

(6.20)
$$\frac{\mathsf{they}}{\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}} \quad \frac{\begin{array}{c}\mathsf{love} \\ \hline \mathsf{SLASH}\ /, \mathsf{ARG}\ \mathsf{NP}_{\mathsf{obj}}, \mathsf{RES}\ \mathsf{SLASH}\ \backslash \\ \mathsf{RES}\ \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{RES}\ \mathsf{S}\end{array}}{\mathsf{SLASH}\ \backslash, \mathsf{ARG}\ \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}, \mathsf{RES}\ \mathsf{S}}> \quad \frac{\mathsf{you}}{\mathsf{NP}_{\mathsf{pl}}}$$

Inference rules corresponding to most of the other combinatory rules of CCG can be constructed quite easily based on this model. The only complication comes with the forward and backward raising operations, where we need some additional means of keeping track of the token identities.

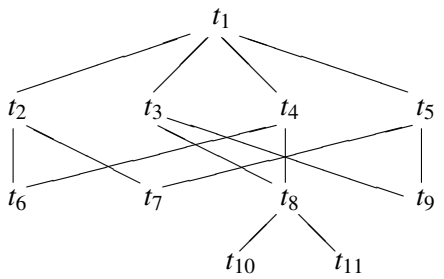
6.3 Inheritance hierarchies

The point of section 6.2 was simply to exemplify how the flexible category description language presented in section 6.1 can be used in a CCG-based grammar formalism,

in particular what the proof system for such a formalism might look like. This was accomplished using the example of the T-CCG formalism from chapter 5. Now I turn my attention to developing the formalism promised at the start of this chapter — the 'inheritance-driven' I-CCG which incorporates an inheritance hierarchy of lexical types into the definition of a grammar. This section discusses the notion of an 'inheritance hierarchy' over a given constraint language, before section 6.4 takes up the challenge of defining the I-CCG formalism itself.

The inheritance hierarchies I assume here are based closely on the type hierarchies defined in section 5.2. Recall that a type hierarchy is formally an ordered pair $\langle T, \sqsubseteq \rangle$, where T is an alphabet and $\sqsubseteq$ is a special kind of partial weak ordering relation on T. Here is an example type hierarchy, repeated from the previous chapter:

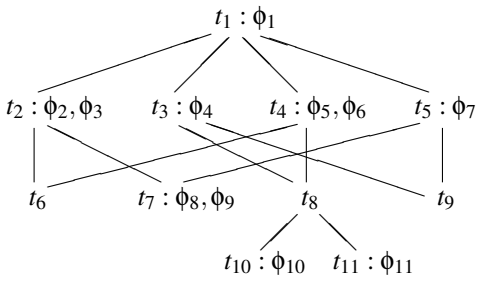
(6.21)



Recall that if type t_i can be reached from type t_j by travelling exclusively downwards, then t_i is subsumed by t_j. In addition, the *maximal* types were defined as those which have no subtypes themselves, for example t_6.

An inheritance hierarchy is simply a type hierarchy where types are potentially associated with formulas of some constraint language. Thus, assuming the schematised language $\Phi = \{\phi_1, \phi_2, \ldots\}$, an example inheritance hierarchy over Φ, based on the above type hierarchy is the following:

(6.22)



The intuition underlying the annotation of types with formulas is that objects of a type are characterised by the fact that they satisfy the relevant formulas. Thus, the

inheritance hierarchy in (6.22) tells us that all objects of type t_4 have both properties ϕ_5 and ϕ_6. Properties are also inherited from supertypes — objects of type t_4 also have property ϕ_1 which they 'inherit' from supertype t_1.

Formally, an inheritance hierarchy over constraint language Φ can be defined as an ordered triple $\langle T, \sqsubseteq, f \rangle$ where:

1. $\langle T, \sqsubseteq \rangle$ is a type hierarchy
2. f is a function from T to $\wp(\Phi)$ i.e. every type in the hierarchy is annotated with a (possibly empty) set of constraints

The 'constraint set' of some type in an inheritance hierarchy is simply the set of all its constraints/properties, including those which it inherits. For example, the constraint set of maximal type t_7 in (6.22) is $\{\phi_1, \phi_2, \phi_3, \phi_7, \phi_8, \phi_9\}$. Formally, given some inheritance hierarchy $\mathcal{H} = \langle T, \sqsubseteq, f \rangle$ over constraint language Φ, the constraint set of type $t \in T$ in $\mathcal{H}$ is defined as:

$$\bigcup_{t' \sqsubseteq t} f(t') \tag{6.23}$$

Note that in this thesis I will assume that inheritance is *monotonic* i.e. for all inheritance hierarchies $\mathcal{H} = \langle T, \sqsubseteq, f \rangle$ over constraint language Φ, and all $t \in T$, the constraint set of t in $\mathcal{H}$ is *consistent*. An alternative approach would be to allow for the possibility of some subtype *overriding* one of its inherited constraints.

My reasons for not allowing this kind of 'default' inheritance are threefold. Firstly, using default inheritance in a multiple inheritance system is inherently unstable, since there is the potential for inheritance conflicts (cf. the well-known 'Nixon diamond' problem). Secondly, any linguistic generalisation which can be expressed using defaults can also be captured by means of multiple inheritance. Finally, systems combining multiple inheritance and constraint overriding are notoriously difficult to implement. For example, the LKB system of Copestake (2002) has to automatically transform a non-monotonic inheritance hierarchy into a larger monotonic one, before parsing can take place. Since one of the aims of the CCG programme is to develop a formalism where the competence grammar and the processing grammar are identical (i.e. 'strong competence'), this option is not to be encouraged.

6.4 The I-CCG formalism

Recall that the T-CCG formalism from chapter 5 is a version of CCG where the alphabet of saturated category symbols is organised into a type hierarchy. The I-CCG formalism goes a step further — not only is there a type hierarchy of saturated category symbols, but also incorporated is an inheritance hierarchy of lexical types, based on the flexible category description language defined in section 6.1. This section presents an explicit definition of the I-CCG formalism.

Subsection 6.4.1 specifies the class of I-CCG grammars.

Subsection 6.4.2 defines the set of I-CCG category models.

Subsection 6.4.3 formulates the satisfaction relation between I-CCG category models and the flexible category description language defined in section 6.1.1.

Subsection 6.4.4 defines the combinatory projection of an I-CCG lexicon.

Subsection 6.4.5 formulates the I-CCG 'generates' relation, determining which strings are generated by which I-CCGs.

Subsection 6.4.6 discusses a proof system for I-CCG grammars.

6.4.1 I-CCGs

Subsection 6.2.1 defined a T-CCG grammar over alphabet Σ as an ordered 4-tuple $\langle A, \sqsubseteq, S, L\rangle$ where: (a) $\langle A, \sqsubseteq\rangle$ is a type hierarchy of saturated category symbols; (b) $S \in A$; and (c) L is a lexical mapping from Σ to sets of flexible category descriptions over A. An I-CCG is an extension of a T-CCG which incorporates an inheritance hierarchy of lexical types over the flexible category description language defined in section 6.1.

Formally, an I-CCG over alphabet Σ is an ordered 7-tuple $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, S, L\rangle$, where:

1. $\langle A, \sqsubseteq_A\rangle$ is a type hierarchy of saturated category symbols
2. $\langle B, \sqsubseteq_B, b\rangle$ is an inheritance hierarchy of lexical types over the set of flexible category descriptions over $A \cup B$
3. A and B are disjoint
4. S is a distinguished element of A

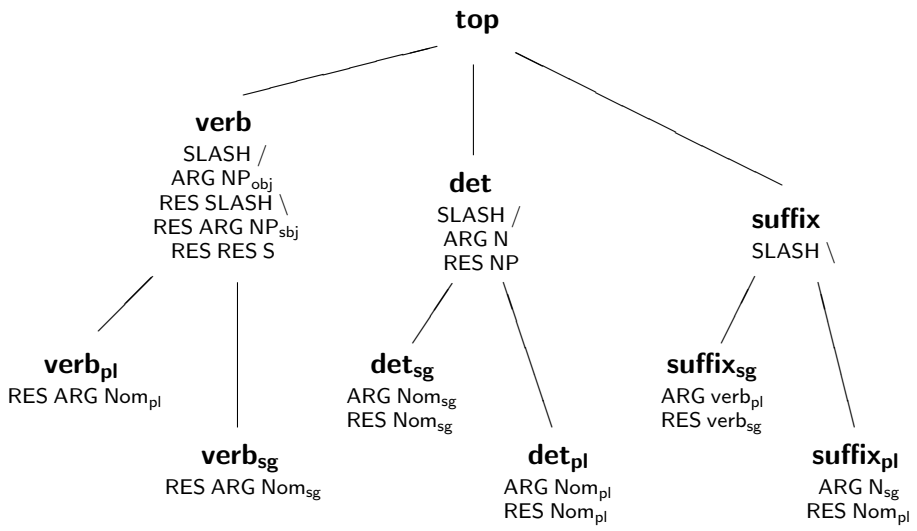


Figure 6.2: The lexical inheritance hierarchy for the language in Table 5.1

5. L is a function from Σ to $A \cup B$

Note that the lexicon is itself defined as a *function* from the alphabet of lexical forms. Thus, an I-CCG lexicon will by definition satisfy the criterion of *functionality* introduced in section 5.1, although this rather strict requirement can be loosened somewhat in cases of genuine ambiguity or for closed class lexical items. Note also that the *range* of the lexicon is defined as the set union of the alphabets of saturated category symbols and lexical types. In other words, lexical forms are always mapped to *atomic* category symbols in an I-CCG lexicon. This means that I-CCG lexicons also satisfy the criterion of *atomicity* from section 5.1. Again this can be mitigated for closed class lexical items where the issue of space is not so important. What this entails is that the I-CCG formalism is capable of providing maximally efficient lexicons for human languages, where all linguistically significant generalisations about the behaviour of words and morphemes are captured in the lexical type. I return to this issue in section 6.5.

I conclude this section with an example of an I-CCG grammar. The type hierarchy of saturated category symbols for this grammar will be that in Figure 6.1 on page 187, which has been discussed before. The lexical inheritance hierarchy for the grammar is presented in Figure 6.2, intended as part of a grammar of subject-verb agreement in English. In addition to the root type, this hierarchy contains the following lexical

types:

verb transitive verb

$\mathbf{verb_{pl}}$ plural transitive verb, subtype of 'verb' (e.g. *love*)

$\mathbf{verb_{sg}}$ singular transitive verb, subtype of 'verb' (e.g. *loves*)

det determiner

$\mathbf{det_{sg}}$ singular determiner, subtype of 'det' (e.g. *the* in *the girl*)

$\mathbf{det_{pl}}$ plural determiner, subtype of 'det' (e.g. *the* in *the girls*)

suffix suffix

$\mathbf{suffix_{sg}}$ third person singular verbal suffix, subtype of 'suffix' (e.g. *-s* in *loves*)

$\mathbf{suffix_{pl}}$ plural nominal suffix, subtype of 'suffix' (e.g. *-s* in *girls*)

Note in addition that the constraints listed on the nodes in this lexical inheritance hierarchy are all taken from the flexible category description language over the type hierarchy of saturated category symbols in Figure 6.1 on page 187.

The lexicon for our example I-CCG grammar is presented in Table 6.1. Note that

$\mathsf{John} \vdash \mathsf{NP}_{\mathsf{sg}}$	$\mathsf{girl} \vdash \mathsf{N}_{\mathsf{sg}}$
$\mathsf{I} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{me} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$
$\mathsf{we} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$	$\mathsf{us} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$
$\mathsf{you} \vdash \mathsf{NP}_{\mathsf{pl}}$	$\mathsf{he} \vdash \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$
$\mathsf{him} \vdash \mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$	$\mathsf{they} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}}$
$\mathsf{them} \vdash \mathsf{NP}^{\mathsf{pl}}_{\mathsf{obj}}$	$\mathsf{love} \vdash \mathsf{verb}_{\mathsf{pl}}$
$\mathsf{s} \vdash \mathsf{suffix}$	$\mathsf{the} \vdash \mathsf{det}$

Table 6.1: An I-CCG lexicon for the language in Table 5.1 assuming the hierarchies in Figures 6.1 and 6.2

this lexicon satisfies both the ideals of functionality and atomicity. I will discuss the significance of this in more detail below.

In conclusion, where $\langle A, \sqsubseteq_A \rangle$ is the type hierarchy of saturated category symbols in Figure 6.1 on page 187, $\langle B, \sqsubseteq_B, b \rangle$ is the inheritance hierarchy of lexical types in Figure

6.2, and L is the I-CCG lexicon in Table 6.1, the ordered 7-tuple $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, \mathsf{S}, L\rangle$ is thus an ICCG, generating the fragment of English schematised in Table 5.1 on page 145.

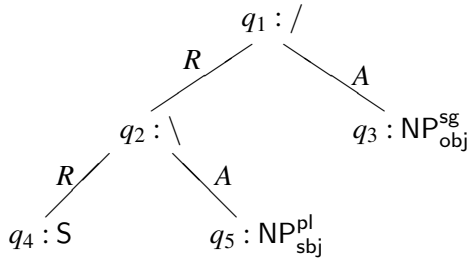
One final point concerns the relation between the *nodes* in a lexical inheritance hierarchy, such as that in Figure 6.2, and the lexical type *labels* that identify them. It is important to note that the labels themselves are not the fundamental constituents in the system. Rather they are merely a convenient means of representing *pointers* from the lexical entries to nodes in the hierarchy. This point is underscored by the fact that the type labels are absent from the underlying category models, as defined in the next section.

6.4.2 I-CCG category models

The category models for the I-CCG formalism are exactly the same as those for the T-CCG formalism, defined in section 5.3.1. In other words, I-CCG category models over type hierarchy $\langle A, \sqsubseteq\rangle$ of saturated category symbols are ordered 5-tuples $\langle Q, Res, Arg, V_S, V_A\rangle$, where: (a) Q is a set of points; (b) Res and Arg are the 'result' and 'argument' relations on Q; (c) V_S assigns a slash to every non-end point in Q; and (d) V_A assigns a *maximal* saturated category symbol from the type hierarchy $\langle A, \sqsubseteq\rangle$ to each end point in Q.

An example of an I-CCG category model over the type hierarchy of saturated category symbols in Figure 6.1 on page 187 is as follows:

(6.24)



This category model can be denoted as $(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$ in traditional CCG notation.

Note once again that I-CCG category models are 'sort-resolved', in that no end point carries a non-maximal symbol from the type hierarchy of saturated category symbols.

6.4.3 I-CCG satisfaction

As before, I-CCG global satisfaction is local satisfaction relative to the root point of a category model. In other words, I-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over type hierarchy $\langle A, \sqsubseteq_A \rangle$ of saturated category symbols satisfies flexible category description ϕ over $A \cup B$, with respect to lexical inheritance hierarchy $\mathcal{H} = \langle B, \sqsubseteq_B, b \rangle$, written $\mathcal{M} \models_{\mathcal{H}} \phi$, if and only if $\mathcal{M}$ locally satisfies ϕ, with respect to $\mathcal{H}$, from its root point, i.e. $\mathcal{M}, q \models_{\mathcal{H}} \phi$.

The local satisfaction definition from section 6.1.2 must be supplemented in order to account for flexible category descriptions which consist of atomic lexical types (rather than atomic saturated category symbols). From this perspective, I-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over type hierarchy $\langle A, \sqsubseteq_A \rangle$ of saturated category symbols locally satisfies flexible category description ϕ over $A \cup B$, with respect to lexical inheritance hierarchy $\mathcal{H} = \langle B, \sqsubseteq_B, b \rangle$, from point $q \in Q$, written $\mathcal{M}, q \models_{\mathcal{H}} \phi$, if and only if:

1. where $\phi \in A$: for some $\psi \in A$ such that $\phi \sqsubseteq_A \psi$, $V_A(q) = \psi$, i.e. point q is labelled with some saturated category symbol subsumed by ϕ

2. where $\phi \in B$: for some *maximal subtype* χ of ϕ in $\mathcal{H}$, and for all constraints ψ in the constraint set of χ, $\mathcal{M}, q \models_{\mathcal{H}} \psi$, i.e. every category description inherited by some maximally specific subtype of lexical type ϕ is satisfied from point q

3. where $\phi = (\mathsf{SLASH}\ \delta)$: $V_S(q) = \delta$, i.e. point q carries the correct directionality symbol

4. where $\phi = (\mathsf{ARG}\ \psi)$: $\mathcal{M}, Arg(q) \models_{\mathcal{H}} \psi$, i.e. category description ψ is satisfied from the argument point of q

5. where $\phi = (\mathsf{RES}\ \psi)$: $\mathcal{M}, Res(q) \models_{\mathcal{H}} \psi$, i.e. category description ψ is satisfied from the result point of q

The new clause is number 2 — a description consisting of a single lexical type symbol is satisfied from a particular point just in case every one of the constraints that are inherited by one of its maximal subtypes is also satisfied from that point.

To run through a simple example of global satisfaction in I-CCG, note that the I-CCG category model in Figure 6.24 satisfies the simple category description '$\mathsf{verb_{pl}}$', with respect to the type hierarchy of saturated category symbols in Figure 6.1 on page

187 and the lexical inheritance hierarchy in Figure 6.2 on page 202. Where the model is denoted $\mathcal{M}$ and the lexical inheritance hierarchy $\mathcal{H}$, the definition of global satisfaction tells us that $\mathcal{M} \models_{\mathcal{H}} \mathsf{verb_{pl}}$ just in case $\mathcal{M}, q_1 \models_{\mathcal{H}} \mathsf{verb_{pl}}$, since q_1 is the root point of $\mathcal{M}$. The local satisfaction definition then tells us that $\mathcal{M}, q_1 \models_{\mathcal{H}} \mathsf{verb_{pl}}$ if and only if for every simple category description ϕ in the constraint set of $\mathsf{verb_{pl}}$ in $\mathcal{H}$, $\mathcal{M}, q_1 \models_{\mathcal{H}} \phi$, i.e.

(6.25) $\mathcal{M}, q_1 \models_{\mathcal{H}}$ $\mathsf{RES\ ARG\ Nom_{pl}}$, $\mathsf{SLASH}\ /$, $\mathsf{ARG\ NP_{obj}}$, $\mathsf{RES\ SLASH}\ \backslash$, $\mathsf{RES\ ARG\ NP_{sbj}}$, $\mathsf{RES\ RES\ S}$

With regard to the first of these, $\mathcal{M}, q_1 \models_{\mathcal{H}} \mathsf{RES\ ARG\ Nom_{pl}}$, the following process of deduction shows us that it is true:

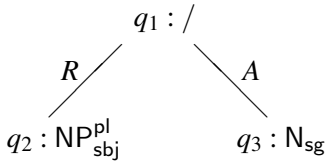
(6.26)
$\mathcal{M}, q_1 \models_{\mathcal{H}} \mathsf{RES\ ARG\ Nom_{pl}}$
iff. $\mathcal{M}, q_2 \models_{\mathcal{H}} \mathsf{ARG\ Nom_{pl}}$ [since $Res(q_1) = q_2$]
iff. $\mathcal{M}, q_5 \models_{\mathcal{H}} \mathsf{Nom_{pl}}$ [since $Arg(q_2) = q_5$]

Finally, it is clear that $\mathcal{M}, q_5 \models_{\mathcal{H}} \mathsf{Nom_{pl}}$ since point q_5 is labelled with a saturated category symbol $\mathsf{NP^{pl}_{sbj}}$ which is subsumed by $\mathsf{Nom_{pl}}$ in the type hierarchy in Figure 6.24.

Proving that the other simple category descriptions listed in (6.25) are satisfied from the root point in Figure 6.24 is left as an exercise for the interested reader.

Another example will illustrate the importance of ensuring that satisfaction of lexical types be defined in terms of maximally specific subtypes in the lexical inheritance hierarchy. Take the following category model over the type hierarchy of saturated category symbols in Figure 6.1 on page 187:

(6.27)



This category model can be denoted as $\mathsf{NP^{pl}_{sbj}}/\mathsf{N_{sg}}$ in traditional CCG notation. Note that this model fails to satisfy the underspecified determiner lexical type det in the lexical inheritance hierarchy — even though the model satisfies all the constraints inherited by det itself, it does not satisfy all constraints inherited by either of the two maximal subtypes of det, i.e. $\mathsf{det_{sg}}$ or $\mathsf{det_{pl}}$.

6.4.4 Combinatory projection of an I-CCG lexicon

Recall from subsection 6.2.2 that, although a T-CCG lexicon was defined as a finite mapping from symbols to sets of category descriptions, its combinatory projection was a mapping from strings to category *models*. The base of the combinatory projection of a T-CCG lexicon is simply the result of 'compiling out' the underspecified lexical entries — if entry $\langle s, \Phi \rangle$ is in lexicon L and T-CCG category model $\mathcal{M}$ satisfies every description in Φ, then the pair $\langle s, \mathcal{M} \rangle$ is in the base of the combinatory projection of L. The combinatory projection of a lexicon is then defined as the closure of this base under the CCG combinatory operations i.e. application, raising, composition, cross composition, and substitution, as defined in section 2.2. The combinatory projection of an I-CCG lexicon is defined in much the same way:

Given I-CCG $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, S, L \rangle$ over alphabet Σ, the combinatory projection of lexicon L is defined as the smallest set L', such that:

1. for all $\langle s, \phi \rangle \in L$ and all I-CCG category models $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A, V_B \rangle$ over $\langle A, \sqsubseteq_A \rangle$ such that $\mathcal{M} \models_{\langle B, \sqsubseteq_B, b \rangle} \phi$, $\langle s, \mathcal{M} \rangle \in L'$

2. for all $\langle s, \mathcal{M}_1 \rangle \in L'$ and all I-CCG category models $\mathcal{M}_2$ over hierarchies $\langle A, \sqsubseteq_A \rangle$, where $\mathcal{M}_3$ is the result of either forward or backward raising $\mathcal{M}_1$ over $\mathcal{M}_2$, then $\langle s, \mathcal{M}_3 \rangle \in L'$

3. for all $\langle s_1, \mathcal{M}_1 \rangle, \langle s_2, \mathcal{M}_2 \rangle \in L'$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to $\mathcal{M}_3$ by means of one of the binary CCG combinatory operations defined in section 2.2, then $\langle s_1 s_2, \mathcal{M}_3 \rangle \in L'$

The second and third clauses here simply say, as before, that the combinatory projection of an I-CCG lexicon is closed under the CCG combinatory operations defined in section 2.2. In addition, the first clause is very similar to what came before — the 'base' of the combinatory projection of an I-CCG lexicon is essentially the result of compiling out the underspecification inherent in the satisfaction relation between I-CCG category models and flexible category descriptions.

As a brief exemplification of this, consider the I-CCG lexicon in Table 6.1 on page 203, over the type hierarchy of saturated category symbols in Figure 6.1 and the inheritance hierarchy of lexical types in Figure 6.2 on page 202. Note that this lexicon contains the following lexical entry:

(6.28) $\mathsf{John} \vdash \mathsf{NP}_{\mathsf{sg}}$

The symbol '$\mathsf{NP}_{\mathsf{sg}}$' is a saturated category symbol from the type hierarchy in Figure 6.1. Furthermore it is a *non-maximal* type in this hierarchy. The following two single-point I-CCG category models carry a symbol on their root point which is subsumed by '$\mathsf{NP}_{\mathsf{sg}}$':

(6.29) $q_1 : \mathsf{NP}^{\mathsf{sg}}_{\mathsf{sbj}}$ $q_1 : \mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$

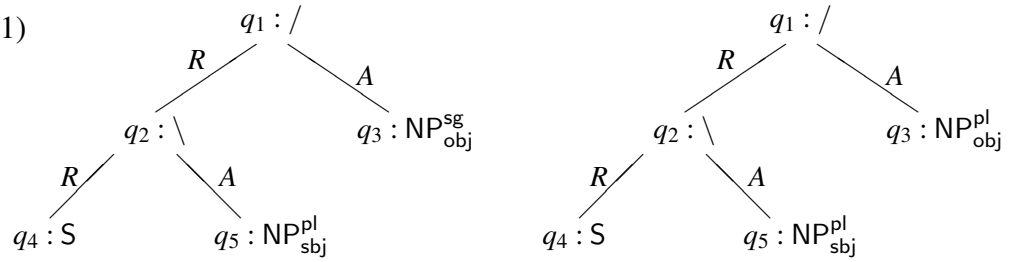
Both these models satisfy the saturated category symbol assigned to *John* in the lexicon. Thus, the lexical form *John* is mapped to both of these I-CCG category models in the base of the combinatory projection of the I-CCG lexicon in Table 6.1.

The following lexical entry is also in this lexicon:

(6.30) $\mathsf{love} \vdash \mathsf{verb}_{\mathsf{pl}}$

Note that '$\mathsf{verb}_{\mathsf{pl}}$' is not a saturated category symbol but rather one of the lexical types in the inheritance hierarchy in Figure 6.2. Indeed it is a *maximal* type in this hierarchy. Consider the following two I-CCG category models over the relevant type hierarchy:

(6.31)



The only difference between these two models is visible at point q_3. Note that these models both satisfy the lexical type $\mathsf{verb}_{\mathsf{pl}}$ with respect to the inheritance hierarchy in Figure 6.2 on page 202. Thus, we can conclude that the lexical form *love* is mapped to both of these I-CCG category models in the base of the combinatory projection of the lexicon in Table 6.1 on page 203.

Finally, recall that the combinatory projection of an I-CCG lexicon is closed under the CCG combinatory operations discussed in section 2.2. Note that these operations do not need to be redefined to account for I-CCG category models — we just need to assume that the lexical types in a category model are ignored by the operations. For example, it should be clear that the leftmost category model in (6.31) and the rightmost category model in (6.29) can reduce by means of forward application. In other words, *loves John* is also in the combinatory projection of the I-CCG lexicon in Table 6.1, mapped to the category model rooted at point q_2 in the leftmost model in (6.31).

6.4.5 I-CCG generation

Finally we come to the 'generates' relation for the I-CCG formalism. This can be expressed in exactly the same terms as the equivalent definition for T-CCG in subsection 6.2.3:

I-CCG $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, S, L\rangle$ over alphabet Σ generates string $s \in \Sigma^*$ if and only if for some X such that $S \sqsubseteq_A X$, the ordered pair $\langle s, X\rangle$ is an element of the combinatory projection of lexicon L.

6.4.6 An I-CCG proof system

Recall that subsection 6.2.4 presented the beginnings of a proof system for the T-CCG formalism based on the flexible category description language discussed in section 6.1. This allowed T-CCG derivations to be presented in terms of category descriptions and was intended to form the basis for a computational implementation of the formalism. The T-CCG proof system involved:

1. a lexical insertion rule
2. a $\sqsubseteq$-compatibility relation on sets of T-CCG category descriptions
3. inference rules corresponding to the CCG combinatory operations

All three of these are also incorporated into the proof system for the I-CCG formalism, exactly as defined in subsection 6.2.4. This is possible since the category description language underlying both formalisms is the same.

However, we also need a new inference rule to 'cache out' the constraints satisfied by a lexical type. The following is a provisional attempt at such a rule:

Given I-CCG $G = \langle A, \sqsubseteq_A, B \sqsubseteq_B, b, S, L\rangle$, for all $\beta \in B$, the following is a valid inference according to G:

(6.32)
$$\frac{\beta}{\Phi}$$

where Φ is the constraint set of some maximal subtype of β in $\langle B \sqsubseteq_B, b\rangle$.

For example, let's run through the first few steps in the I-CCG derivation of string *they love you* based on the I-CCG lexicon in Table 6.1, the type hierarchy of saturated category symbols in Figure 6.1, and the inheritance hierarchy of lexical types in Figure 6.2. The first step involves lining up the words themselves:

(6.33) they love you

Then we perform three instances of the lexical insertion inference rule:

(6.34)
$$\frac{\text{they}}{\text{NP}^{\text{pl}}_{\text{sbj}}} \quad \frac{\text{love}}{\text{verb}_{\text{pl}}} \quad \frac{\text{you}}{\text{NP}_{\text{pl}}}$$

Next we need to cache out the lexical type according to the lexical inheritance hierarchy in Figure 6.2:

(6.35)
$$\frac{\text{they}}{\text{NP}^{\text{pl}}_{\text{sbj}}} \quad \frac{\dfrac{\text{love}}{\text{verb}_{\text{pl}}}}{\begin{array}{c}\text{SLASH } /, \text{ARG NP}_{\text{obj}}, \text{RES SLASH } \backslash, \\ \text{RES ARG NP}^{\text{pl}}_{\text{sbj}}, \text{RES RES S}\end{array}} \quad \frac{\text{you}}{\text{NP}_{\text{pl}}}$$

And then we are ready to do forward application:

(6.36)
$$\frac{\text{they}}{\text{NP}^{\text{pl}}_{\text{sbj}}} \quad \frac{\dfrac{\text{love}}{\text{verb}_{\text{pl}}}}{\begin{array}{c}\text{SLASH } /, \text{ARG NP}_{\text{obj}}, \text{RES SLASH } \backslash, \\ \text{RES ARG NP}^{\text{pl}}_{\text{sbj}}, \text{RES RES S}\end{array}} \quad \frac{\text{you}}{\text{NP}_{\text{pl}}}$$
$$\frac{\qquad\qquad\qquad\qquad\qquad\qquad}{\text{SLASH } \backslash, \text{ARG NP}^{\text{pl}}_{\text{sbj}}, \text{RES S}}\ >$$

Obviously, the 'cache out' inference rule will need to be generalised for cases where lexical types are embedded within lexical types. This should however not prove too complicated.

6.5 I-CCG and non-redundant lexicons

I mentioned above that one of the advantages of the I-CCG formalism is that it allows us to construct maximally efficient CCG lexicons for human languages. In this section, I demonstrate this point in detail.

I start by recapping the discussion about lexical redundancy from chapter 5. In section 5.1, I introduced a tiny fragment of English, schematised in Table 5.1, which nevertheless encapsulates some non-trivial facts about subject-verb number agreement and subject-object case distinctions. I argued that the shortest baseline CCG lexicon for this fragment, that in Table 5.2 on page 146, was unacceptably inefficient. In the first place, it failed to satisfy what I called the 'ideal of functionality', a formalisation of the

generally accepted requirement that lexicons have as few entries as possible. Recall that the baseline CCG lexicon had *twenty-two* lexical entries for just *fourteen* distinct lexical forms, giving a functionality ratio of $22/14 = 1.57$. In addition, the baseline CCG lexicon failed to satisfy the so-called 'ideal of atomicity', which encapsulates the ideal that lexical category labels should be atomic symbols. The baseline lexicon contained *thirty-six* saturated category symbols for the *twenty-two* lexical entries, giving an atomicity ratio of $36/22 = 1.64$.

Section 5.5 then examined the shortest T-CCG of the same fragment of English. Recall that this consisted of the type hierarchy of saturated category symbols in Figure 5.2 on page 154 and the lexicon in Table 5.4 on page 159. I observed that this T-CCG came close to satisfying the criterion of functionality, since its functionality ratio was $16/14 = 1.07$. However, it was only marginally more efficient than the baseline CCG with respect to the criterion of atomicity, since its atomicity ratio was $26/16 = 1.63$.

Now let's consider the I-CCG for the same fragment of English, which consists of the type hierarchy of saturated category symbols in Figure 6.1 on page 187, the inheritance hierarchy of lexical types in Figure 6.2 on page 202, and the lexicon in Table 6.1 on page 203. Note first of all that this lexicon contains exactly one lexical item for each distinct lexical form. Thus, it satisfies the ideal of functionality perfectly. Whereas the T-CCG lexicon needed to assign the definite article to *two* distinct categories, in the I-CCG lexicon it is assigned to just the one lexical type 'det', which is itself a supertype of the two more specific subtypes '$\mathsf{det}_{\mathsf{sg}}$' and '$\mathsf{det}_{\mathsf{pl}}$'. Secondly, note that the I-CCG lexicon in Table 6.1 satisfies the ideal of atomicity as well, since every lexical form is mapped to an *atomic* lexical type. Recall that the T-CCG lexicon was unable to assign every form to an atomic type. In particular determiners, verbs and suffixes needed to be assigned to categories denoted by complex labels. In the I-CCG lexicon it is again the lexical inheritance hierarchy which allows a more efficient lexicon to be formulated — determiners, verbs and suffixes are identified as lexical types and their associated constraints encode the combinatory behaviour of these types.

In conclusion, even for such a simple fragment of English as that schematised in Table 5.1, it is possible to construct an I-CCG lexicon which is shorter than the shortest baseline CCG or even T-CCG lexicons. Indeed, the lexical apparatus of the I-CCG formalism is powerful enough to formulate a functional, atomic lexicon for any human language.

Section 5.5 argued that the incorporation of a type hierarchy of saturated category symbols into the CCG formalism rendered redundant a number of other proposed generalised category notations, for example the 'morphosyntactic' CCG notation of Bozsahin (2002) and the conceptualisation of saturated category symbols as typed feature structures from Erkan (2003). This argument was based on the observation that every lexicon using these other notations could be converted into a T-CCG lexicon with an identical functionality ratio.

Here I turn my attention to the 'multiset' CCG notation first used in Hoffman (1995). Baldridge (2002) uses the following examples from English to illustrate the benefits of the multiset category notation:

(6.37) Marcos picked up the ball.

Marcos picked the ball up.

In other words, with English phrasal verbs like *pick up*, the direct object can either follow or precede the particle *up*. This implies the following two distinct baseline CCG lexical categories for the verb, one for each of the two orderings in (6.37):

(6.38) $\mathsf{picked} \vdash ((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{NP}_{\mathsf{obj}})/\mathsf{Prt}$

$\mathsf{picked} \vdash ((\mathsf{S}\backslash\mathsf{NP}_{\mathsf{sbj}})/\mathsf{Prt})/\mathsf{NP}_{\mathsf{obj}}$

However, as Baldridge points out, a single multiset category can be used to capture both orderings:

(6.39) $\mathsf{picked} \vdash (\mathsf{S}\{\backslash\mathsf{NP}_{\mathsf{sbj}}\})\{/\mathsf{NP}_{\mathsf{obj}}, /\mathsf{Prt}\}$

Baldridge assumes the following versions of forward and backward application, redrafted to take into account the multisets:

(6.40) $X(\alpha \uplus \{/Y\}) \quad Y \stackrel{>}{\Rightarrow} X\alpha$

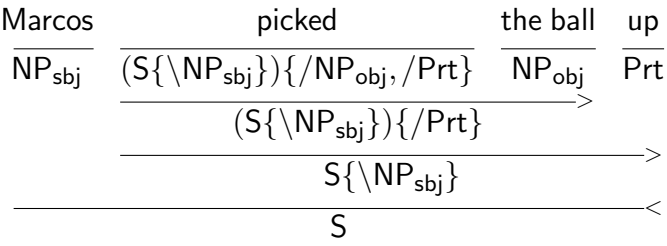
$Y \quad X(\alpha \uplus \{\backslash Y\}) \stackrel{<}{\Rightarrow} X\alpha$

Note that $\uplus$ denotes multiset union. Using these rules and the multiset category in (6.39), the two sentences in (6.37) can be derived as follows:

(6.41)

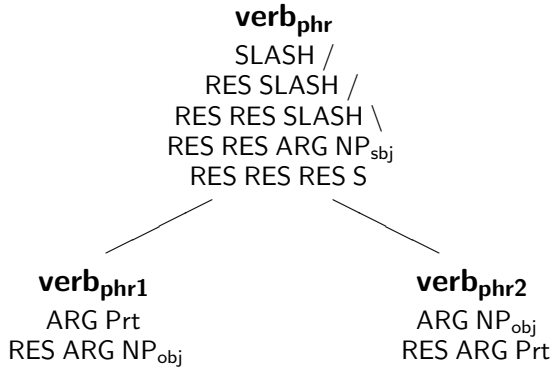
$$\dfrac{\dfrac{\mathsf{Marcos}}{\mathsf{NP}_{\mathsf{sbj}}} \quad \dfrac{\dfrac{\mathsf{picked}}{(\mathsf{S}\{\backslash\mathsf{NP}_{\mathsf{sbj}}\})\{/\mathsf{NP}_{\mathsf{obj}}, /\mathsf{Prt}\}} \quad \dfrac{\mathsf{up}}{\mathsf{Prt}}}{(\mathsf{S}\{\backslash\mathsf{NP}_{\mathsf{sbj}}\})\{/\mathsf{NP}_{\mathsf{obj}}\}}> \quad \dfrac{\mathsf{the\ ball}}{\mathsf{NP}_{\mathsf{obj}}}}{\mathsf{S}\{\backslash\mathsf{NP}_{\mathsf{sbj}}\}}> }{\mathsf{S}}<$$

(6.42)



It is clear that the point of including multisets in CCG category notation is to construct lexicons which come closer to satisfying the ideal of functionality. The multiset notation comes in particularly useful for languages like Turkish, Japanese and Tagalog, where there is a high degree of local argument scrambling. However, it should be equally clear that the multiset notation does not allow any lexical generalisations to be captured which cannot also be expressed in an I-CCG lexical inheritance hierarchy. Take for example the following simple hierarchy for English phrasal verbs:

(6.43)



This hierarchy contains two maximal types ‘verb$_{phr1}$’ and ‘verb$_{phr2}$’ corresponding to each of the baseline CCG categories in (6.38). Since there is also a common supertype ‘verb$_{phr}$’ capturing the properties common to all English phrasal verbs, we can use this type in the lexical assignment:

(6.44) picked $\vdash$ verb$_{phr}$

The first sentence in (6.37) can thus be derived as in Figure 6.3. The other sentence can be derived similarly.

In conclusion, I argue that the I-CCG formalism, which incorporates an inheritance hierarchy of lexical types renders the multiset category notation formally redundant, since every multiset CCG lexicon can be converted into an I-CCG lexicon with the same functionality ratio.

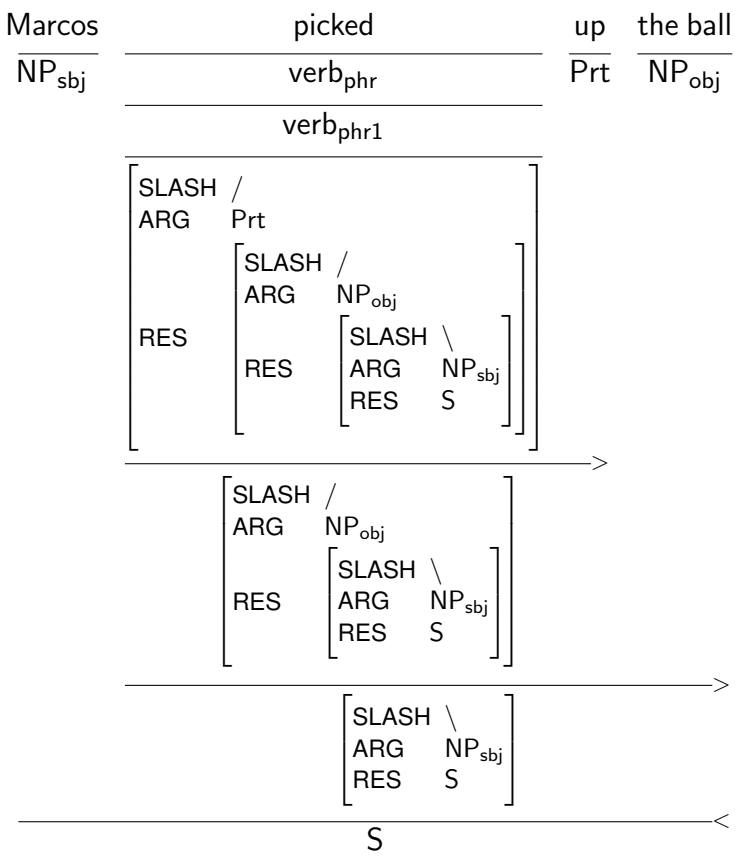


Figure 6.3: An I-CCG derivation

6.6 Summary

This chapter presented I-CCG , the 'inheritance-driven' CCG formalism, a version of CCG where: (a) the alphabet of saturated category symbols is organised into a type hierarchy; and (b) there is an additional alphabet of *lexical types*, itself organised into an inheritance hierarchy over a simple feature-based constraint language. The I-CCG formalism allows maximally efficient lexicons to be constructed for human languages, thus permitting a much wider range of linguistically significant generalisations about words and morphemes to be expressed.

Section 6.1 introduced the feature-based category description language underlying the I-CCG formalism. This represents a radical departure from the previous category description notation from section 2.1 and demonstrates the usefulness of taking a model-theoretic approach to category notation in CCG . Using this new description language we can specify unsaturated categories in a much more subtle manner, for example the set of expressions seeking an argument to the left, or the set of expressions whose result is a sentence or whose argument itself seeks an argument to the left.

Section 6.2 then exemplified how this flexible category description notation could be put to use in a redefinition of the T-CCG formalism from chapter 5. Whereas previously a T-CCG lexicon was conceptualised as a mapping from forms to categories descriptions, a redefined T-CCG lexicon maps forms to *sets* of flexible category descriptions.

Section 6.3 introduced the notion of an 'inheritance hierarchy', which I conceptualised as a type hierarchy where types are optionally associated with a set of constraints drawn from some constraint language. The intuition underlying this kind of hierarchy is that objects of a given type are characterised by all constraints associated with that type or with any type which subsumes it.

Section 6.4 turned to a formal definition of the I-CCG formalism itself. The main components of an I-CCG are a type hierarchy of saturated category symbols, an inheritance hierarchy of lexical types over the flexible category description language, and an I-CCG lexicon, the latter defined as a *function* from from to *atomic* lexical types.

Section 6.5 demonstrated that the I-CCG formalism allows maximally efficient lexicons to be constructed for human languages. In particular, it is possible to formulate I-CCG lexicons which satisfy both the ideal of functionality and the ideal of atomicity.

I then argued that the I-CCG category notation system subsumes the 'multiset' category notation of Hoffman (1995) and Baldridge (2002), in that every multiset lexicon can be converted into an I-CCG lexicon of the same functionality ratio. In other words, the I-CCG category notation can capture all the generalisations that multiset categories can express.

I conclude this chapter with a brief discussion of a couple of interesting points. The first of these concerns the nature of constraints which may be associated with the types in an I-CCG lexical inheritance hierarchy. In subsection 6.4.1, I characterised these as flexible category descriptions over $A \cup B$ where A is the alphabet of saturated category symbols and B is the set of lexical types themselves. This means that lexical types may themselves be incorporated in constraints on lexical types, leading to the possibility of recursive lexicons. One problem with this approach is that the base of the combinatory projection of an I-CCG lexicon may not be a finite set, and thus the generative capacity of the I-CCG formalism as defined in section 6.4 *may* be greater than mildly context-sensitive.

This problem can easily be rectified by restricting lexical constraints to include only saturated category symbols and not lexical types. The resulting formalism would still allow functional, atomic lexicons to be constructed for human languages, though the lexical hierarchies would have to include many more constraints. I have chosen to go with the more general system here, because, as we shall see in chapter 7, recursively defined lexical inheritance hierarchies allow us to capture the distinction between 'arguments' and 'modifiers' more easily.

The second point concerns how we might go about generalising the simple feature-based category description language so as to capture even more linguistically significant facts about the behaviour of the words and morphemes of a human language. The first obvious addition would be boolean connectives such as conjunction, disjunction and negation. Of these, disjunction is already incorporated within the geometry of the inheritance hierarchy itself, and negation is known to seriously complicate the corresponding proof system. On the other hand, boolean conjunction does appear to be necessary in order to implement the kind of subcategorisation hierarchies used in HPSG analyses of the English verbal lexicon.[1]

Another extension of the basic feature-based category description language which

[1] See for example chapter 8 of Pollard and Sag (1987).

appears to be necessary to capture the full range of linguistically significant generalisations involves the 'iteration' operator of propositional dynamic logic (Harel et al., 2000). Recall that in our flexible category description language, we have formulas like RES S (expressions whose *immediate* result is an S), RES RES S (expressions whose *second* result is an S), RES RES S (expressions whose *third* result is an S). We can use the iteration operator $\star$ to schematise over these, i.e. $\mathsf{RES}^{\star}$ S then denotes the set of all expressions whose *ultimate result* is an S. Something like this appears to be necessary to capture the concept 'verb' in English. In a similar way, the concept 'transitive' can be captured by the category description $\mathsf{RES}^{\star}$ ARG $\mathsf{NP}_{\mathsf{obj}}$, denoting the set of expressions selecting a direct object, either as their first argument or as some subsequent one.

Chapter 7

I-CCG and linguistic competence

Chapter 2 of this thesis defined a 'baseline' CCG formalism, which was characterised as the *intersection* of the formal assumptions underlying almost all recent CCG work. In other words, everybody working with CCG accepts *all* the features of the baseline formalism even though nobody accepts *only* these. Indeed, there is a general consensus that the baseline category notation needs to be supplemented in some way, and chapters 3, 4 and 5 examined in detail the reasons behind this consensus.

Chapter 3 argued that the primary aim of the CCG programme is to provide a theory of human linguistic competence, one aspect of which involves capturing the kind of implicational universals found in the language typology literature. I reviewed two instances from the CCG literature of implicational universals which the formalism is claimed to predict, and concluded that, at least in some cases, this kind of claim is justified by an appeal to the *length* of different grammars. This form of argumentation is not of course restricted to work in CCG; rather it is evident in analyses from a range of different frameworks dating back to the dawn of generative grammar. The idea is that a grammar formalism F can be argued to predict implicational universal $P \subseteq Q$ if: (a) there are equally short F-grammars for equivalent $P \cap Q$, $P' \cap Q$ and $P' \cap Q'$ languages; but (b) the shortest F-grammar of any $P \cap Q'$ language is longer than at least one of its $P \cap Q$, $P' \cap Q$ or $P' \cap Q'$ equivalents.

Chapter 4 then discussed a couple of implicational word order universals which are not captured in this way by the baseline category notation version of the CCG formalism. The first of these was the familiar Greenberg-derived observation correlating VO basic clausal order with the existence of prepositions and OV order with postpositions. The second involved some data from Travis (1989) suggesting that VO

languages with postverbal PP adverbials also have necessarily postverbal PP objects. In the case of both these universals, I argued that there is no sense in which the baseline CCG formalism can be argued to predict them in terms of an acquisition-based preference for shorter grammars. However, both universals do appear to be based on considerations of grammatical economy, and this kind of analysis has been proposed in phrase structure-based grammar formalisms such as Government and Binding Theory and Generalised Phrase Structure Grammar, where the relevant measure of size has involved counting, for example, the number of parameters set or the number of linear precedence statements in the grammar.

This then was identified as the primary motivation for generalising the category notation in CCG, i.e. to come up with a version of the formalism which can formally capture a wider range of implicational word order universals. Chapter 5 went on to consider a secondary motivation, involving the elimination of redundancy in CCG lexicons for human languages. I argued that the general desire for non-redundant lexicons is motivated both by computational considerations, involving storage space, and by linguistic considerations of missed generalisations. This desire can be partially reduced to two idealised constraints on human language lexicons: (a) the ideal of 'functionality', which holds that the ideal lexicon assigns every word or morpheme to exactly one lexical category, modulo ambiguity; and (b) the ideal of 'atomicity', according to which the ideal lexicon assigns open class words or morphemes to categories denoted by atomic symbols, possibly macro-like abbreviations for traditional unsaturated categories.

I demonstrated that the baseline CCG notation falls well short in this respect. Even for the smallest non-trivial fragments of human languages, it is impossible to come up with baseline CCG lexicons which satisfy either functionality and atomicity. This led to the formulation of a version of the CCG formalism where saturated category symbols are organised into a type hierarchy, i.e. T-CCG, which allows for the construction of human language lexicons which come come closer to satisfying the functionality requirement. This idea was inspired by the use of such structures in unification-based formalisms like HPSG, and also in computational implementations of categorial grammars such as the work reported in Villavicencio (2002), Beavers (2004) and Erkan (2003). I argued that this simple idea subsumes a range of other CCG notations, for example the use of partially ordered, 'morphosyntactic' unary modalities in Bozsahin (2002), and the conceptualisation of saturated category symbols as typed feature struc-

tures described by AVMs, from Erkan (2003).

Chapter 6 took the idea behind T-CCG a step further, proposing an extended formalism involving an inheritance hierarchy of *lexical types* in addition to the type hierarchy of saturated category symbols. The lexical types are constrained by formulas from a simple but flexible, feature-based category description language. The resulting formalism, I-CCG, allows us to construct non-redundant CCG lexicons for human languages, satisfying both the ideals of functionality and atomicity. Thus, the I-CCG formalism provides, I believe, a fairly definitive solution to the problem of lexical redundancy in CCG, although it remains to be fully implemented computationally.

Which brings us back to the primary motivation for generalising CCG category notation — Does the I-CCG formalism capture the kind of implicational word order universals discussed in Chapter 4 as formal universals involving a minimal amount of substantive specification? In this chapter I argue that it does, or at least that a slightly generalised notion of I-CCG does. This extended I-CCG incorporates not only a type hierarchy of saturated category symbols and an inheritance hierarchy of lexical types, but also an inheritance hierarchy of 'argument types' as well. Enumerating the argument types in a grammar will allow us to formalise notions such as 'subject', 'object' and 'modifier' as well as generalisations such as 'NP arguments follow heads but PP arguments precede', or 'modifiers precede heads but indirect objects follow'. The ability to quantify statements of this kind will prove crucial to the claim that the I-CCG formalism formally predicts word order universals.

This chapter will proceed as follows:

Section 7.1 presents the finalised definition of the I-CCG formalism, where grammars include an inheritance hierarchy of argument types such as 'subject' and 'object'. This involves stratifying our language of flexible category descriptions from section 6.1 over two distinct levels — the level of arguments and the level of categories.

Section 7.2 then demonstrates that the revised I-CCG formalism formally captures the implicational word order universals from chapter 4, with the minimal amount of substantive stipulation. In addition, I discuss the extent to which the approach argued for here can capture the full range of word order universals discussed in the typology literature, as well as how this approach relates to other proposed explanations involving processing cost.

7.1 I-CCG with argument hierarchies

In this section I redefine the I-CCG grammar formalism from section 6.4 to incorporate an inheritance hierarchy of argument types such as subject, object and modifier. Such an innovation is necessary to allow distinctions to be made between, for example, objects and modifiers. The I-CCG formalism developed in chapter 6, although able to express generalisations such as "dependents follow heads" (using a 'SLASH /' constraint on the root lexical type) or "dependents follow verbal heads", cannot be used to make statements like "PP modifiers follow heads".

7.1.1 Redefining I-CCG grammars

Recall from subsection 6.4.1 that an I-CCG grammar over alphabet Σ was previously defined as an ordered 7-tuple $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, S, L\rangle$, where: (a) $\langle A, \sqsubseteq_A\rangle$ is a type hierarchy of saturated category symbols; (b) $\langle B, \sqsubseteq_B, b\rangle$ is an inheritance hierarchy of lexical types over the set of flexible category descriptions over $A \cup B$; (c) S is a distinguished element of A; and (d) lexicon L is a function from Σ to $A \cup B$. From now on, we will assume that an I-CCG also includes a second inheritance hierarchy in addition to that of lexical types, i.e. a hierarchy of 'argument' types such as 'subject' and 'object'. In other words, henceforth we will be working with the following, finalised definition:

An I-CCG over alphabet Σ is an ordered 10-tuple $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, C, \sqsubseteq_C, c, S, L\rangle$, with the following properties:

1. $\langle A, \sqsubseteq_A\rangle$ is a type hierarchy of saturated category symbols
2. $\langle B, \sqsubseteq_B, b\rangle$ is an inheritance hierarchy of lexical types over the set of I-CCG category descriptions over A, B and C, with B disjoint from A
3. $\langle C, \sqsubseteq_C, c\rangle$ is an inheritance hierarchy of argument types over the set of I-CCG argument descriptions over A, B and C, with C disjoint from both A and B
4. S is a distinguished element of A
5. lexicon L is a function from Σ to $A \cup B$

A careful reading of this definition should reveal that the flexible description language from section 6.1 needs to be modified slightly. It now operates at two different levels, i.e. the lexical inheritance hierarchy is annotated with constraints from a language of

I-CCG *category* descriptions, while the hierarchy of argument types carries constraints from a language of I-CCG *argument* descriptions. Let's now consider each of these in turn.

7.1.2 I-CCG category descriptions

The set of I-CCG category descriptions over alphabets A of saturated category symbols, B of lexical types and C of argument types is defined as the smallest set Φ such that:

1. $A \subseteq \Phi$, i.e. every saturated category symbol is a category description
2. $B \subseteq \Phi$, i.e. every lexical type is a category description
3. for all $\phi \in \Phi$, $(\mathsf{RES}\ \phi) \in \Phi$
4. for all I-CCG *argument* descriptions α over A, B and C, $(\mathsf{ARG}\ \alpha) \in \Phi$

This definition is very similar to that of the flexible category descriptions in subsection 6.1.1. The only difference is that whereas previously we had formulas like $(\mathsf{ARG}\ \phi)$ where ϕ was another category description, now ϕ must come from another language altogether — the language of I-CCG *argument* descriptions. This has the advantage of reifying arguments and assigning them properties such as direction and category, as we shall see in the following section.

7.1.3 I-CCG argument descriptions

The set of I-CCG argument descriptions over alphabets A of saturated category symbols, B of lexical types and C of argument types is defined as the smallest set Ω such that:

1. $C \subseteq \Omega$, i.e. every argument type is an argument description
2. for all $\delta \in \{/, \backslash\}$, $(\mathsf{SLASH}\ \delta) \in \Omega$
3. for all I-CCG category descriptions ϕ over A, B and C, $\phi \in \Omega$

The easiest way to clarify the difference between the languages of category descriptions and argument descriptions is probably by means of exemplification. Let's assume

the following alphabets A, B and C of saturated category symbols, lexical types and argument types respectively:

(7.1) $A = \{\mathsf{S}, \mathsf{NP}\}$

$B = \{\mathsf{verb}, \mathsf{det}\}$

$C = \{\mathsf{sbj}, \mathsf{obj}\}$

Thus the following are all I-CCG *category* descriptions over A, B and C:

(7.2) NP

verb

RES verb

ARG sbj

ARG NP

ARG SLASH /

ARG RES S

RES ARG RES ARG SLASH /

The interpretation of these should be familar from section 6.1. The only difference is that previous constraints of the form ARG ϕ, where ϕ is another category description, now take the form ARG α, where α is an argument description.

On the other hand, the following are all I-CCG *argument* descriptions over A, B and C:

(7.3) obj

SLASH \

NP

RES det

ARG RES ARG sbj

Note that, although every category description is also an argument description, the reverse is not the case.

7.1.4 An example I-CCG

In subsection 7.1.1, I defined what one of our redefined I-CCG grammars actually is. In this section I present an example I-CCG for the fragment of English schematised in

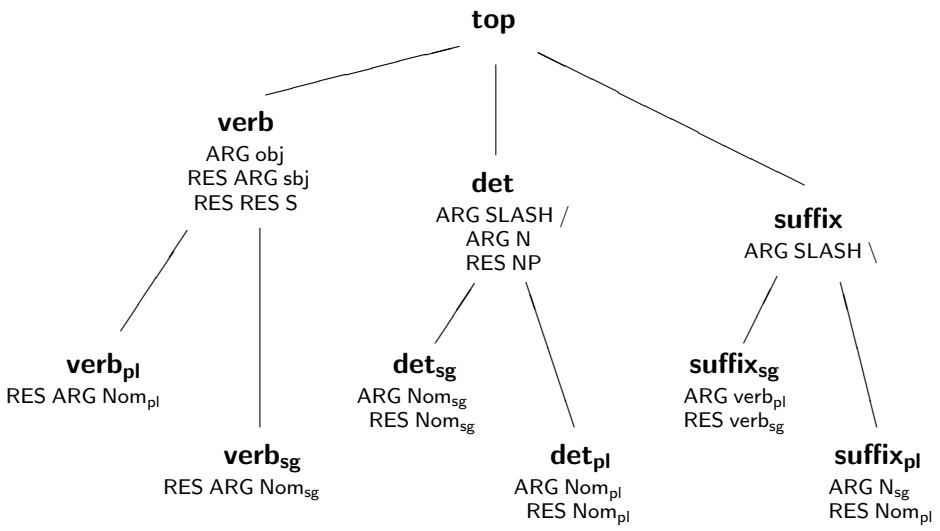


Figure 7.1: A redefined lexical inheritance hierarchy for the language in Table 5.1

Table 5.1 on page 145. The first component of an I-CCG is a type hierarchy of saturated category symbols. In this case we can just reuse that in Figure 5.2 on page 154. The second component is an inheritance hierarchy of lexical types over the language of category descriptions. Recall the lexical inheritance hierarchy in Figure 6.2 on page 202. Since the language of category descriptions has changed somewhat, we need to redraft the hierarchy to use the constraints from the new language. The new lexical inheritance hierarchy is in Figure 7.1. The most important difference between the two versions of the lexical inheritance hierarchy is to be found on the 'verb' type, where the constraints now make use of the alphabet of argument types such as 'subject' and 'object'.

The third component of an I-CCG grammar is an inheritance hierarchy of *argument* types over the language of argument descriptions. Let's assume the one in Figure 7.2. This hierarchy recognises two types of argument, i.e. subjects and objects, with the obvious properties. The final component of an I-CCG is a lexicon, which must be a function which maps each lexical form to exactly one saturated category symbol or lexical type. Since the two latter alphabets are unchanged since we last formulated an I-CCG for the language in Table 5.1, we can retain exactly the same lexicon, i.e. that in Table 6.1 on page 203.

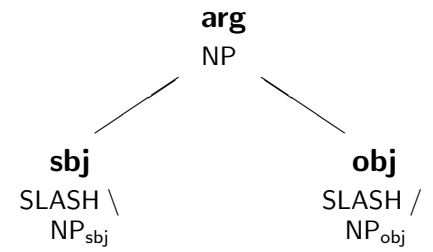


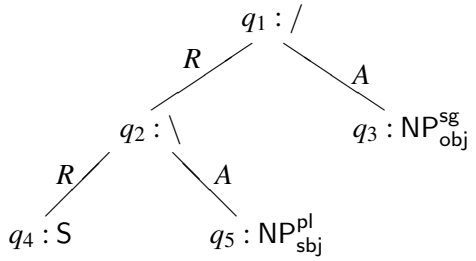
Figure 7.2: An inheritance hierarchy of argument types for the language in Table 5.1

7.1.5 I-CCG category models

The category models for the redrafted I-CCG formalism are exactly the same as before — see the definition of T-CCG category models in section 5.3.1. In other words, I-CCG category models over type hierarchy $\langle A, \sqsubseteq \rangle$ of saturated category symbols are ordered 5-tuples $\langle Q, Res, Arg, V_S, V_A \rangle$, where: (a) Q is a set of points; (b) Res and Arg are the 'result' and 'argument' relations on Q; (c) V_S assigns a slash to every non-end point in Q; and (d) V_A assigns a *maximal* saturated category symbol from the type hierarchy $\langle A, \sqsubseteq \rangle$ to each end point in Q.

An example of an I-CCG category model over the type hierarchy of saturated category symbols in Figure 6.1 on page 187 is as follows:

(7.4)



Recall that this category model can be denoted as $(\mathsf{S}\backslash\mathsf{NP}^{\mathsf{pl}}_{\mathsf{sbj}})/\mathsf{NP}^{\mathsf{sg}}_{\mathsf{obj}}$ in traditional CCG notation.

7.1.6 Redefining I-CCG satisfaction

As before, I-CCG global satisfaction is local satisfaction relative to the root point of a category model. In other words, I-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A \rangle$ over type hierarchy $\langle A, \sqsubseteq_A \rangle$ of saturated category symbols satisfies category description ϕ

over A, B and C, with respect to lexical inheritance hierarchy $\mathcal{H}_1 = \langle B, \sqsubseteq_B, b\rangle$ and argument hierarchy $\mathcal{H}_2 = \langle C, \sqsubseteq_C, c\rangle$, written $\mathcal{M} \models_{\mathcal{H}_1,\mathcal{H}_2} \phi$, if and only if $\mathcal{M}$ locally satisfies ϕ, with respect to $\mathcal{H}_1$ and $\mathcal{H}_2$, from its root point, i.e. $\mathcal{M}, q \models_{\mathcal{H}_1,\mathcal{H}_2} \phi$.

I-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A\rangle$ over type hierarchy $\langle A, \sqsubseteq_A\rangle$ of saturated category symbols locally satisfies category description ϕ over A, B and C, with respect to lexical inheritance hierarchy $\mathcal{H}_1 = \langle B, \sqsubseteq_B, b\rangle$ and argument hierarchy $\mathcal{H}_2 = \langle C, \sqsubseteq_C, c\rangle$, from point $q \in Q$, written $\mathcal{M}, q \models_{\mathcal{H}_1,\mathcal{H}_2} \phi$, if and only if:

1. where $\phi \in A$: for some $\psi \in A$ such that $\phi \sqsubseteq_A \psi$, $V_A(q) = \psi$, i.e. point q is labelled with some saturated category symbol subsumed by ϕ

2. where $\phi \in B$: for some maximal subtype χ of ϕ in $\mathcal{H}_1$, and for all constraints ψ in the constraint set of χ in $\mathcal{H}_1$, $\mathcal{M}, q \models_{\mathcal{H}_1,\mathcal{H}_2} \psi$, i.e. every category description inherited by some maximally specific subtye of lexical type ϕ is satisfied from point q

3. where $\phi = (\mathsf{RES}\ \psi)$: $\mathcal{M}, Res(q) \models_{\mathcal{H}_1,\mathcal{H}_2} \psi$, i.e. category description ψ is satisfied from the result point of q

4. where $\phi = (\mathsf{ARG}\ \alpha)$: $\mathcal{M}, q \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \alpha$, i.e. *argument* description α is satisfied from q

I-CCG category model $\mathcal{M} = \langle Q, Res, Arg, V_S, V_A\rangle$ over type hierarchy $\langle A, \sqsubseteq_A\rangle$ of saturated category symbols locally satisfies *argument* description α over A, B and C, with respect to lexical inheritance hierarchy $\mathcal{H}_1 = \langle B, \sqsubseteq_B, b\rangle$ and argument hierarchy $\mathcal{H}_2 = \langle C, \sqsubseteq_C, c\rangle$, from point $q \in Q$, written $\mathcal{M}, q \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \phi$, if and only if:

1. where $\alpha \in C$: for some maximally specific subtype γ of α in $\mathcal{H}_2$, and for all constraints β in the constraint set of γ in $\mathcal{H}_2$, $\mathcal{M}, q \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \beta$, i.e. every argument description inherited by some maximally specific subtype of argument type α is satisfied from point q

2. where $\alpha = (\mathsf{SLASH}\ \delta)$: $V_S(q) = \delta$, i.e. point q carries the correct directionality symbol

3. where α is a *category* description over A, B and C: $\mathcal{M}, Arg(q) \models_{\mathcal{H}_1,\mathcal{H}_2} \alpha$, i.e. the argument point satisfies α as a *category* description

Here are some examples based on I-CCG category premodel $\mathcal{M}$ in (7.4), given the lexical inheritance hierarchy $\mathcal{H}_1$ in Figure 7.1 and the argument inheritance hierarchy $\mathcal{H}_1$ in Figure 7.2:

(7.5)
$$\begin{aligned}
&\mathcal{M},q_2 \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{sbj}\\
&\mathcal{M},q_1 \not\models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{sbj} \quad \text{because } \mathcal{M},q_1 \not\models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{SLASH} \backslash\\
&\mathcal{M},q_1 \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{arg} \quad \text{because } \mathcal{M},q_1 \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{obj}\\
&\mathcal{M},q_2 \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{SLASH} \backslash\\
&\mathcal{M},q_1 \models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{NP}^{\textsf{sg}}_{\textsf{obj}} \quad \text{because } \mathcal{M},Arg(q_1) \models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{NP}^{\textsf{sg}}_{\textsf{obj}}\\
&\mathcal{M},q_2 \not\models^{arg}_{\mathcal{H}_1,\mathcal{H}_2} \textsf{RES S} \quad \text{because } \mathcal{M},Arg(q_2) \not\models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{RES S}
\end{aligned}$$

And here are some illustrative examples involving satisfaction of *category* descriptions in the same premodel:

(7.6)
$$\begin{aligned}
&\mathcal{M},q_1 \models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{RES ARG SLASH} \backslash\\
&\mathcal{M},q_1 \models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{RES ARG sbj}\\
&\mathcal{M},q_1 \not\models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{RES ARG S} \quad \text{because } \mathcal{M},Arg(Res(q_1)) \not\models_{\mathcal{H}_1,\mathcal{H}_2} \textsf{S}
\end{aligned}$$

7.1.7 The combinatory projection of an I-CCG lexicon

In subsection 7.1.1 I redefined the set of I-CCG grammars to incorporate an inheritance hierarchy of argument types. We are now ready to think about the combinatory projection of such a lexicon, in exactly the same way as before:

Given I-CCG $\langle A,\sqsubseteq_A,B,\sqsubseteq_B,b,C,\sqsubseteq_C,c,S,L\rangle$ over alphabet Σ, the combinatory projection of lexicon L is defined as the smallest set L', such that:

1. for all $\langle s,\phi\rangle \in L$ and all I-CCG category models $\mathcal{M} = \langle Q,Res,Arg,V_S,V_A,V_B,V_C\rangle$ over $\langle A,\sqsubseteq_A\rangle$ such that $\mathcal{M} \models_{\langle B,\sqsubseteq_B,b\rangle,\langle C,\sqsubseteq_C,c\rangle} \phi$, $\langle s,\mathcal{M}\rangle \in L'$

2. for all $\langle s,\mathcal{M}_1\rangle \in L'$ and all I-CCG category models $\mathcal{M}_2$ over type hierarchy $\langle A,\sqsubseteq_A\rangle$, where $\mathcal{M}_3$ is the result of either forward or backward raising $\mathcal{M}_1$ over $\mathcal{M}_2$, then $\langle s,\mathcal{M}_3\rangle \in L'$

3. for all $\langle s_1,\mathcal{M}_1\rangle,\langle s_2,\mathcal{M}_2\rangle \in L'$ such that $\mathcal{M}_1$ and $\mathcal{M}_2$ reduce to $\mathcal{M}_3$ by means of one of the binary CCG combinatory operations defined in section 2.2, then $\langle s_1 s_2,\mathcal{M}_3\rangle \in L'$

Since this definition is almost exactly the same as the previous one in subsection 6.4.4, I will pass over it without much comment. Suffice to say that the combinatory projection of an I-CCG lexicon is closed under the CCG combinatory operations from section 2.2, which again can be safely assumed to simply ignore the argument types on points. In addition the base of the combinatory projection of I-CCG lexicon L simply compiles out the various lexical entries — if form s is mapped to lexical type ϕ in the lexicon, and I-CCG category model $\mathcal{M}$ carries label ϕ on its root point, then s is mapped to $\mathcal{M}$ in the combinatory projection of the lexicon.

7.1.8 I-CCG generation

Finally we come to the 'generates' relation for the new version of the I-CCG formalism incorporating argument hierarchies. This is almost exactly the same as the previous definition in subsection 6.4.5:

I-CCG $\langle A, \sqsubseteq_A, B, \sqsubseteq_B, b, C, \sqsubseteq_C, c, S, L\rangle$ over alphabet Σ generates string $s \in \Sigma^*$ if and only if for some X such that $S \sqsubseteq_A X$, the ordered pair $\langle s, X\rangle$ is an element of the combinatory projection of lexicon L.

7.2 Explaining the data

Having finalised the definition of the I-CCG formalism to include both a hierarchy of lexical types and one of argument types, it is time to return to the implicational word order universals from chapter 4, in order to evaluate to what extent the I-CCG formalism is a better theory of linguistic competence than the version which incorporated simply the baseline category notation.

Subsection 7.2.1 deals with the bidirectional implicational universal discussed in section 4.1, involving interactions between the relative position of a direct object with respect to its verb in the basic word order of a language and the predominance of prepositions or postpositions in the language.

Subsection 7.2.2 then turns to the implicational universal discussed in section 4.2, which related the positions of indirect objects and PP adverbials in VO languages.

Subsection 7.2.3 then considers the extent to which lexical inheritance hierarchies can be used to explain word order universals in general, focusing on the data in Dryer

(1992) and Hawkins (1980).

Subsection 7.2.4 relates the approach to explaining word order universals in terms of lexical economy with an alternative based on average processing cost.

7.2.1 Clausal order, prepositions and postpositions

Recall the discussion from section 4.1. I assumed the domain of all and only the natural languages: (a) which manifest a basic ordering of direct objects with respect to their head verbs in matrix declarative clauses, i.e. either the verb overwhelmingly precedes its direct object, or it overwhelmingly follows it; and (b) have a class of adpositions, with a demonstrable preference for either prepositions or postpositions. This domain was partitioned along two independent dimensions. The first of these involved verb-object ordering:

VO the direct object follows the verb

OV the direct object precedes the verb

And the second partition involved the existence of prepositions or postpositions in the language:

PO lots of prepositions but few if any postpositions

OP few if any prepositions but lots of postpositions

Based on this binary partition, evidence from attested languages suggests the following bidirectional universal over the domain:

(7.7) $\text{VO} \approx \text{PO}$

I argued that the baseline CCG formalism fails to explain this universal in terms of an acquisition-based preference for shorter grammars, since there is no sense in which the lexicon of a $\text{VO} \cap \text{OP}$ language is longer than the equivalent $\text{VO} \cap \text{PO}$ language. On the other hand, it is clear that the universal is motivated by some notion of grammatical economy, when expressed in abstract terms of heads and dependents:

VO ∩ PO dependents follow heads

VO ∩ OP dependents follow verbal heads; dependents precede adpositional heads

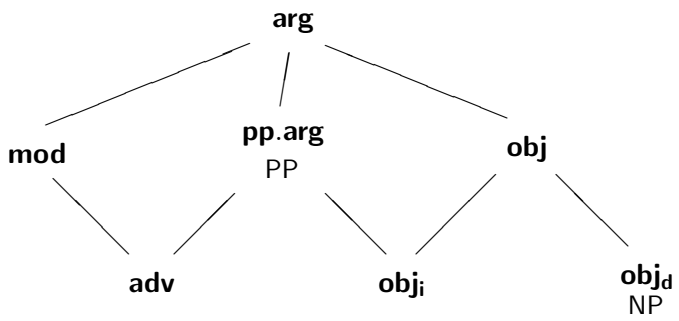
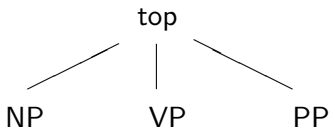


Figure 7.3: A partial argument inheritance hierarchy for human language

The aim at the outset was to deconstruct the CCG category notation so as to be able to express this kind of generalisation directly. This process resulted in the I-CCG category notation defined and interpreted in section 7.1, where grammars incorporate inheritance hierarchies of both lexical types, such as 'verb' and 'determiner', and argument types like 'subject' and 'object'.

Let us start by assuming the following simple type hierarchy of saturated category symbols:

(7.8)



As before, 'NP', 'VP' and 'PP' stand for 'noun phrase', 'verb phrase', and 'preposition phrase' respectively. We also assume the hierarchy of argument types in Figure 7.3. The various argument types in this hierarchy are to be understood as follows:

arg an argument/dependent in general

obj an object

mod a modifier/adjunct

pp.arg an argument consisting of a preposition phrase

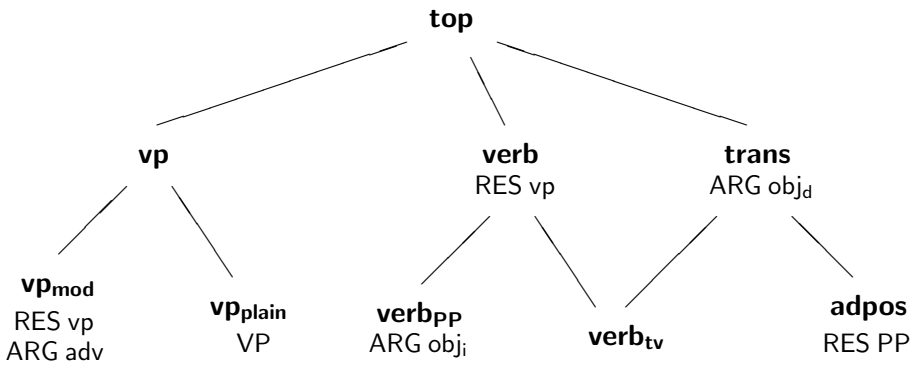


Figure 7.4: A partial lexical inheritance hierarchy for human language

adv an adverbial (i.e. verbal modifier)

obj$_i$ an indirect object

obj$_d$ a direct object

In this hierarchy, objects and modifiers are disjoint. This hierarchy includes one crucial substantive assumption — there is no argument type which subsumes both 'direct objects' and 'adverbials' but not 'indirect objects'. The fact that general linguistic theory has no name for such a concept provides a certain amount of justification for assuming this constraint to be part of a system of substantive universals of human language.

Thirdly, let us also assume the hierarchy of lexical types in Figure 7.4. The various lexical types are to be interpreted as follows:

verb a lexical verb

trans a transitive expression, i.e. taking a direct object

verb$_{pp}$ an intransitive verb taking an indirect object

verb$_{tv}$ a (strict) transitive verb

adpos an adposition

vp a verb phrase

$\mathsf{vp_{plain}}$ a 'plain' (i.e. unmodified) verb phrase

$\mathsf{vp_{mod}}$ a modified verb phrase

In the hierarchy of lexical types, the 'vp' type is defined recursively, thus allowing an unbounded number of modifiers, although I am ignoring the implications of this for the parsing algorithm. In addition, I am ignoring subjects, so as to keep the hierarchy relatively simple.

One last assumption involves the lexicon itself, which I will assume is a superset of the following:

(7.9) $\mathsf{love, crucify, devour} \vdash \mathsf{verb_{tv}}$

$\mathsf{beyond, without, during} \vdash \mathsf{adpos}$

With these substantive assumptions in mind, we need to consider what needs to be added to this universal base, in order to provide an I-CCG for each of the four language types: VO ∩ PO, VO ∩ OP, OV ∩ PO, and OV ∩ OP.

In order to provide an I-CCG for a (verb-initial, prepositional) VO ∩ PO language, it is sufficient to add just the one constraint to the argument inheritance hierarchy in Figure 7.3 (ignoring modifiers for the moment):

(7.10) $\mathsf{obj} \rightarrow \mathsf{SLASH}\ /$

In other words, the 'object' type is annotated with the argument description SLASH /, stating that all objects follow their head. Similarly, in order to provide an I-CCG for a (verb-final, postpositional) OV ∩ OP language, it is sufficient to add just the following constraint to the argument inheritance hierarchy:

(7.11) $\mathsf{obj} \rightarrow \mathsf{SLASH}\ \backslash$

In other words, all objects precede their head.

Turning to the case of the (verb-initial, postpositional) VO ∩ OP languages, an I-CCG for this kind of language would have to add the following *two* category descriptions to the hierarchy of lexical types in Figure 7.4 (or equivalently add two new constrained argument types to the argument hierarchy):

(7.12) $\mathsf{verb_{tv}} \rightarrow \mathsf{ARG\ SLASH}\ /$

$\mathsf{adpos} \rightarrow \mathsf{ARG\ SLASH}\ \backslash$

In other words, transitive verbs take following direct objects, but adpositions take preceding ones.

In a similar fashion, the I-CCG for a (verb-final, prepositional) OV ∩ PO language would require the following two constraints to be added:

(7.13) $\mathsf{verb_{tv}} \rightarrow \mathsf{ARG\ SLASH}\ \backslash$

$\mathsf{adpos} \rightarrow \mathsf{ARG\ SLASH}\ /$

In other words, transitive verbs take preceding direct objects, but adpositions take following ones.

It is clear that for any I-CCG of a VO ∩ OP or OV ∩ PO language, there will be a shorter I-CCG of an equivalent VO ∩ PO or OV ∩ OP language, i.e. one where the information in the inheritance hierarchies is organised more efficiently, with fewer constraints. Thus, we can argue that the I-CCG formalism, in conjunction with the substantive constraints encoded in the basic inheritance hierarchies in Figures 7.3 and 7.4, successfully predicts the universal in (7.7).

7.2.2 Direct objects, indirect objects and modifiers

Recall now the discussion from section 4.2. I assumed a domain consisting of a subset of the VO ∩ PO languages discussed in the previous section, where prepositional phrases can act as both indirect objects and as adverbs. Again, this domain was partitioned along two independent dimensions:

VI indirect objects *follow* the verb

IV indirect objects *precede* the verb

The second partition involved the ordering of verbs and preposition phrase adverbials:

VX prepositional adverbials are *post*modifiers

XV prepositional adverbials are *pre*modifiers

Based on these binary partitions, evidence suggests the following implicational universal:

(7.14) VX ⊆ VI

In other words, if prepositional adverbials follow the verb in VO language *L* then indirect objects also follow the verb in *L*.

I argued that the baseline CCG formalism fails to explain this universal in terms of an acquisition-based preference for shorter grammars, since there is no sense in which the lexicon of a VX∩IV language is longer than that of equivalent VX∩VI, XV∩VI or XV∩IV languages. On the other hand, it is clear that the universal is motivated by some notion of grammatical economy, when expressed for example in terms of number of Travis' word order parameters that need to be set to non-default values:

VX∩VI head-first

VX∩IV head-first, theta-left, case-right

XV∩VI head-last, theta-right

XV∩IV head-last, case-right

Let us consider again the inheritance hierarchy of argument types in Figure 7.3 and that of lexical types in Figure 7.4. We also assume the following lexicon:

(7.15) transitive verbs ⊢ $\mathsf{verb_{tv}}$
other verbs ⊢ $\mathsf{verb_{PP}}$
prepositions ⊢ adpos

Table 7.1 specifies which constraints need to be added to the two inheritance hierarchies in order to construct an I-CCG which generates equivalent languages from each of the four types. The unmarked VX∩VI language type requires the addition of just the one constraint SLASH / to the root argument type, thus capturing the generalisation that dependents follow heads in the languages. The 'impossible' VX∩IV language type, on the other hand, needs *three* constraints to be added to argument types, since in this case it is direct objects and modifiers which share the important properties, and the hierarchy contains no argument type which subsumes these but not indirect objects. The two marked but possible language types, XV∩VI and XV∩IV, each require the addition of *two* constraints to capture their word order behaviour, since in each case there is a non-maximal argument type (respectively obj and pp.arg) available to assign the generalisation to.

It is clear that for any I-CCG of a VX∩IV language, the I-CCGs of equivalent VX∩VI, XV∩VI and XV∩IV languages will be shorter, with the information in the

VX∩VI	arg → SLASH /
VX∩IV	obj_d → SLASH / mod → SLASH / obj_i → SLASH \
XV∩VI	obj → SLASH / mod → SLASH \
XV∩IV	obj_d → SLASH / pp.arg → SLASH \

Table 7.1: I-CCG constraints for Travis' language types

inheritance hierarchies organised more efficiently, with fewer constraints. Thus, we can argue that the I-CCG formalism, in conjunction with the substantive constraints encoded in the basic inheritance hierarchies in Figures 7.3 and 7.4, successfully predicts the universal in (7.14). In addition, the I-CCG formalism successfully predicts the observed markedness ordering among these four language types, where VX∩VI languages constitute the unmarked option.

One final point relates to the fact that, in any realistically sized lexical inheritance hierarchy, the kind of comparisons that need to be made are to be considered as being *local* rather than *global*. In other words, during the process of language acquisition, the child aims to ensure that particular sub-hierarchies are organised as economically as possible, rather than needing to constantly evaluate the hierarchy as a whole.

Recall the discussion towards the end of section 3.2.2. There I discussed how Steedman (2000) argues that the baseline CCG formalism predicts the *pro*-drop parameter in terms of shorter lexicons. The existence of relative clause forms in English which apparently violate this generalisation (e.g. *the man who Bill thinks loves Mary*) were argued to be the result of a later stage of language acquisition, taking place *after* the basic generalisations have been set in stone, so to speak. From this perspective, it was pointed out that this explanation assumes crucuially that comparisons of lexical shortness must be local to particular parts of the lexicon, and to particular stages of language acquisition.

I am assuming the same kind of dynamic model of language acquisition applies to the development of lexical inheritance hierarchies, where particular parts of the hierarchy are 'fixed' at certain points in the language acquisition process, with further elaboration limited to specific sub-hierarchies only.

One further related point which needs to be emphasised concerns the distinction between those abstract lexical types which are assumed to be present at the start of the language acquisition process (generally functional types such as 'head' and 'argument'). These are also considered to be irrelevant to the simplicity judgements made by the language-learner as he or she formulates a language-specific lexical hierarchy.

7.2.3 Other word order universals

This thesis has argued in favour of adding lexical inheritance hierarchies to the basic CCG formalism of Steedman (2000) and Baldridge (2002), and has proposed a constraint language which is expressive enough to capture significant linguistic generalisations about the words and morphemes found in human languages, without being so expressive as to cause problems for the basic CCG chart parsing algorithms. The most immediately obvious argument in favour of this approach concerns the elimination of redundancy from CCG lexicons, and was discussed in detail in chapters 5 and 6. To this extent, the argument parallels the motivation for incorporating inheritance hierarchies into feature-structure based formalisms such as HPSG, as elaborated in chapter 8 of Pollard and Sag (1987).

This thesis departs from the HPSG tradition in one important respect, arguing that we can do more with lexical inheritance hierarchies than just capture lexical generalisations inside individual languages. Assuming some universal vocabulary of linguistic types, and assuming that, other things being equal, the human language faculty prefers compact inheritance hierarchies, it is possible to use lexical hierarchies as the basis for a theory of word order typology. In effect, I am arguing that one notion of 'markedness' in word order typology is a matter of compactness of the representation of linguistic knowledge in the mind/brain.

Of course, other notions of linguistic markedness are also at play in the acquisition of human language, and it an interesting research question to pursue how far we can take the approach proposed here. In other words, how many word order universals can be plausibly explained with reference to the number of constraints which need to be

assigned to universal argument types?

In essence, it is those word order universals which can loosely be described as 'mirror-image' universals (though not necessarily 'bidirectional' universals) which are the most straightforward to capture in I-CCG. The data concerning the position of direct objects, indirect objects and adverbials discussed in Travis (1989) provides a nice illustration of a mirror-image universal, since along with the generalisation concerning VO languages which was the subject of sections 4.2 and 7.2.2, there is a parallel universal involving OV languages.

Just as there are no VO languages in which verbs precede PP adverbials and follow indirect objects (i.e. the class VO $\cap$ VX $\cap$ IV), so Travis claims that there are no OV languages in which verbs follow PP adverbials and precede indirect objects. In other words, the complete set of data discussed in Travis (1989) involves the following *two* word order universals (where 'X' denotes PP adverbials and 'I' denotes indirect objects, as before):

- VX $\cap$ VO $\subseteq$ VI
- XV $\cap$ OV $\subseteq$ IV

And in particular, just as VO $\cap$ VI $\cap$ XV and VO $\cap$ IV $\cap$ XV languages are considered to be marked alternatives to VO $\cap$ VI $\cap$ VX languages, so among the OV languages OV $\cap$ IV $\cap$ XV is the unmarked option.

Assuming the inheritance hierarchies in Figure 7.3 and Figure 7.4, the simplest I-CCGs for each of the four OV language types involve adding the following constraints to the argument hierarchy:

OV $\cap$ XV $\cap$ IV	$\mathsf{arg} \rightarrow \mathsf{SLASH} \backslash$
OV $\cap$ XV $\cap$ VI	$\mathsf{obj_d} \rightarrow \mathsf{SLASH} \backslash$ $\mathsf{mod} \rightarrow \mathsf{SLASH} \backslash$ $\mathsf{obj_i} \rightarrow \mathsf{SLASH} /$
OV $\cap$ VX $\cap$ IV	$\mathsf{obj} \rightarrow \mathsf{SLASH} \backslash$ $\mathsf{mod} \rightarrow \mathsf{SLASH} /$
OV $\cap$ VX $\cap$ VI	$\mathsf{obj_d} \rightarrow \mathsf{SLASH} \backslash$ $\mathsf{pp.arg} \rightarrow \mathsf{SLASH} /$

Again, the more 'marked' a language-type is considered to be by Travis, the more constraints are needed in its I-CCG lexical inheritance hierarchy. Thus, the same method

as was used to explain Travis' word order generalisations for VO languages also captures the 'mirror-image' observations for OV languages as well, using the same basic argument type hierarchy and simply reversing the SLASH statements on each node.

On the other hand, there are a number of word order universals which do not lend themselves to this kind of 'mirror-image' statement. In other words, there are some universals involving VO languages which have no corresponding mirror-image universal in OV languages. One example involves complementiser (a.k.a. 'subordinating conjunctions') positioning, assuming the following partition:

CompS complementisers precede complement clauses

SComp complementisers follow complement clauses

Hawkins (1990) points out that the following universal statement is valid:

(7.16) VO $\subseteq$ CompS

However, the corresponding mirror-image universal, OV $\subseteq$ SComp is not valid, since OV languages are more or less evenly distributed among SComp and CompS.

Note first of all that the VO $\subseteq$ CompS universal can be partially explained by the I-CCG formalism — the grammar of a VO $\cap$ CompS language will require just the one direction statement (SLASH $/$) assigned to some common supertype of direct objects and complement clauses in the argument hierarchy, whilst the grammar of a corresponding VO $\cap$ SComp language would require different direction statements for each argument type. However, a full explanation of the VO $\subseteq$ CompS universal would require that grammars of OV $\cap$ CompS languages be no more complex than those of OV $\cap$ SComp languages, and crucially that grammars of OV $\cap$ CompS languages involve fewer constraints than those of VO $\cap$ SComp ones. A simple argument type hierarchy involving distinct 'direct object' and 'complement clause' types, along with a common supertype, is not able to make this prediction correctly.

There are two ways we might go about rectifying this situation. One way would be to have a more complicated universal argument type hierarchy encoding substantive linguistic knowledge. We might go about doing this by stating that 'heavy' argument types like complement-clauses carry a universal, defeasible 'SLASH /' constraint, which can be overridden from above in special circumstances. Such an approach would be assuming that language acquisition is able to process default constraints, although

these are not part of the actual grammar itself. This appears similar to the way overridable constraints are used in metagrammatical theories such as Optimality Theory, but very different to their use in HPSG, where their main function is to make lexical inheritance hierarchies smaller by reducing the need for multiple inheritance.

An alternative approach would be to drop the requirement that all language universals be capturable in terms of relative complexity of lexical inheritance hierarchies, and instead propose that grammar size is just one among many competing, orthogonal constraints on language acquisition. This is essentially what is proposed in Hawkins (1990), where independent measures of the processing complexity of a sentence (Early Immediate Constituents and Minimal Attachment) are involved in the explanation of the VO $\subseteq$ CompS universal. Such an approach would involve linguistic competence potentially tolerating a larger than necessary hierarchy of argument types, in order to satisfy some more general principle such as "focused elements occur late in a sentence".

Dryer (1992) presents data from a sample of 625 genetically and geographically diverse languages, based on which he identifies the following six valid statistical correlations, which he argues can be explained simply as a function of consistent head-dependent ordering:

1. In VO languages nouns precede genitives; in OV languages genitives precede nouns

2. In VO languages adjectives precede standards; in OV languages standards precede adjectives

3. In VO languages verbs precede adpositional phrases; in OV languages adpositional phrases precede verbs

4. In VO languages verbs precede manner adverbs; in OV languages manner adverbs precede verbs

5. In VO languages copulas precede predicates; in OV languages predicates precede copulas

6. In VO languages verbs meaning 'want' precede subordinate verbs; in OV languages subordinate verbs precede verbs meaning 'want'

All of these 'mirror-image' universals are straightforwardly predicted by I-CCG, in exactly the same way that the correlation between VO and PO languages was captured in section 7.2.1. It should be noted that a certain amount of substantive information is assumed in each case, determining in each construction which element is the head or functor. In certain cases, this might even be controversial, since for example most categorial grammarians would prefer to treat free modifiers as functors rather than arguments. However, since Baldridge and Kruijff (2003) present a version of CCG which allows functors and heads to be distinguished by means of slash modalities, it may be possible to restate the word order constraints on the argument hierarchy directly in terms of the head-dependent distinction rather than the functor-argument one.

In addition, Dryer (1992) cites the following results about various non-correlations among the languages in his sample:

1. there is no statistical correlation between VO/OV and the ordering of nouns and adjectives

2. there is no statistical correlation between VO/OV and the ordering of nouns and demonstratives

3. there is no statistical correlation between VO/OV and the ordering of adjectives and intensifiers

4. there is no statistical correlation between VO/OV and the ordering of verbs and negative particles

5. there is no statistical correlation between VO/OV and the ordering of verbs and tense/aspect particles

Dryer claims that these are all 'false positives' for any theory of word-order universals which relies on consistent head-dependent ordering. In other words, if we are to assume: (a) that all linguistic constructions have heads; and (b) that the knowledge of which element is the head in any particular construction is either innate, universal or determined by non-syntactic factors, then any theory based on consistent head-dependent ordering must predict at least some kind of correlation in each of these five cases. Although both of these assumptions about heads appear to me to be at least questionable, it is clear that the I-CCG formalism can be argued to predict the lack of correlation in each of these cases, since any language is free to choose whether it

treats adjectives, demonstratives, intensifiers and particles as functors or arguments in the relevant constructions. In other words, the claim here would be the null hypothesis that there are no substantive universals that narrow down the search space for language acquisition in these five cases.

Finally, Dryer discusses the following eight word order correlations:

1. In VO languages tense/aspect auxiliary verbs precede content verbs; in OV languages content verbs precede tense/aspect auxiliary verbs

2. In VO languages negative auxiliary verbs precede content verbs; in OV languages content verbs precede negative auxiliary verbs

3. In VO languages complementisers precede subordinate clauses

4. In VO languages question particles precede the sentence; in OV languages sentences precede question particles

5. In VO languages adverbial subordinators precede the sentence; in OV languages sentences precede adverbial subordinators

6. In VO languages articles precede nouns; in OV languages nouns precede articles

7. In VO languages plural words precede nouns; in OV languages nouns precede plural words

8. In OV languages subjects precede verbs

Dryer claims that these correlations are 'controversial' in the sense that it is unclear whether or not they are explained by consistent head-dependent ordering. The problem involves the fact that there is disagreement among linguists as to which element is the head in each case. However, categorial grammar approaches almost uniformly treat auxiliary verbs, complementisers, question particles, adverbial subordinators and articles as functor categories, and thus even the controversial correlations discussed by Dryer can be explained in I-CCG by means of consistent functor-argument serialisation, i.e. fewer SLASH constraints in the lexical inheritance hierarchy.[1]

Thus, it appears that a significant proportion of the word order universals catalogued in for example Dryer (1992) and Hawkins (1980) are captured by the I-CCG

[1]Note however that correlations 3 and 8 here are not 'mirror-image' universals and hence are only partially captures by lexical economy on its own.

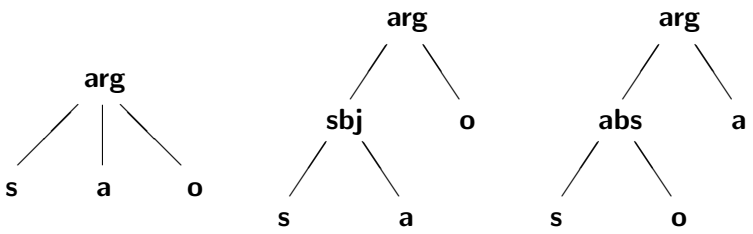


Figure 7.5: Accusative and ergative languages

formalism, in terms of an acquisition-based preference for simpler lexical inheritance hierarchies. This explanation relies on the assumption that for some, but crucially not all, syntactic constructions, the element which constitutes the functor and/or head is determined semantically. In addition, a number of other word order universals are at least partially explained by a preference for simpler inheritance hierarchies. In these cases, a full explanation of the relevant phenomena requires reference to a system of competing, orthogonal constraints on language-acquisition, of which the preference for simpler inheritance hierarchies is one of the most important factors.

However, it is not only in the context of word order universals that the explanatory potential of the I-CCG formalism is evident. The effects of lexical economy are to be found in many other aspects of human linguistic competence, where a particular problem (e.g. effectively encoding which roles are played by the various participants in an event) often has a number of distinct solutions (e.g. case, agreement, or fixed word order). One example of this comes from the study of 'ergativity' in human language (Dixon, 1994).

Discussions of ergativity generally make the following assumptions:

1. human languages have 'intransitive' verbs, taking just the one NP argument, conventionally called 'S' (mnemonic for 'single argument')

2. human languages have 'transitive' verbs, taking two NP arguments

3. of the two arguments of a transitive verb, the one whose role is more 'agent-like' is called 'A', and the other is called 'O' (or sometimes 'P' for 'patient-like')

There are three possible ways of organising these three argument types, S, A and O,

into an inheritance hierarchy, as shown in Figure 7.5. The first of these is a flat hierarchy, where all three argument types are marked in a distinct manner, for example using three separate case-markers. The second is an example of an 'accusative' system, where the sole argument of an intransitive verb and the more 'agent-like' argument of a transitive verb are marked in the same manner (as 'subjects'), but the other argument of a transitive verb is marked differently. In the third hierarchy it is the 'other' argument of transitive verbs which is marked similarly to the argument of an intransitive verb, as a so-called 'absolutive' argument, and the more agent-like argument of the transitive verb is marked in a distinct manner.

If we assume that it is a case system which is used to solve the problem of getting across who did what to whom, then the following constraints will be needed to be added to the above argument hierarchies:

accusative languages the 'subject' argument type is constrained to consist of a nominative case NP; the 'o' argument type is constrained to consist of an accusative case NP

ergative languages the 'absolutive' argument type is constrained to consist of an absolutive case NP; the 'a' argument type is constrained to consist of an 'ergative' case NP

other languages the 's' argument type is constrained to consist of a nominative case NP; the 'o' argument type is constrained to consist of an accusative case NP; the 'a' argument type is constrained to consist of an ergative case NP

Note here how the argument hierarchies in accusative and ergative languages are more compact than those in the other type, since fewer constraints need to be added. Indeed, the literature on syntactic ergativity makes clear that there are many more attested accusative and ergative languages than there are those of the other type. This illustrates that there is a general preference for systems of linguistic competence where argument types are organised into a hierarchy of supertypes rather than simply listed. In short, the I-CCG formalism appears to predict the existence of both accusative and ergative languages, as well as the relative infrequency of languages which are neither accusative nor ergative.[2]

[2]Obviously the existence of so-called 'split ergative' systems, where for example pronouns follow a nominative-accusative pattern and other NPs follow an ergatove-absolutive one, requires a more complex explanation, perhaps in terms of competing constraints on language acquisition.

7.2.4 Comparison with processing-based explanations

This thesis has assumed an approach to the theory of human linguistic competence, which can be summarised as follows. We have a grammar formalism, i.e. baseline CCG, which is expressive enough to generate all attested constructions in human languages, but is restrictive enough to make some strong predictions about the kind of constructions which are too complex to be part of a human language. However, though restrictive, this formalism does not appear to be sufficient in itself as a theory of human linguistic competence. This is because there are languages that can be generated by a CCG (e.g. VSO languages with postpositions) which are distinctly 'improbable', in the sense that they are unattested.

The approach we have taken to resolving this paradox is to propose that the grammars provided by a formalism such as CCG are 'ranked' in some way, with highly-ranked grammars more likely to generate human languages than low-ranked ones. In addition, following Chomsky (1965), we have taken the simplest approach to comparing two grammars, i.e. in terms of length. From this perspective, the task is straightforward — to generalise the language of linguistic descriptions so that meaningful linguistic statements are transformed into considerations of length.

However, measurement of the length of a grammar is not the only way we might go about ranking the grammars provided by a formalism. An alternative, associated with the work of Hawkins (1990), and recently revisited in Newmeyer (2005), involves ranking grammars according to how easily they can be parsed.

Hawkins does not present his theory in the terminology of formal language theory, but it is instructive to translate his ideas into our terms. Hawkins appears to assume a grammar formalism which, to all intents and purposes, is context-free grammar, along with a particular algorithm for parsing with these grammars. Assuming grammar G, if we have some means of measuring the cost of parsing each sentence generated by G, then we can define the ranking value of grammar G as the *average cost* of parsing the sentences it generates. Hawkins argues that this method, in common with his parsing oracle favouring early recognition of immediate constituents, provides an explanation for many of the Greenbergian word order universals. In other words, a universal like $\mathrm{VO} \approx \mathrm{PO}$ discussed in sections 4.1 and 7.2.1, is explained by the fact that, for example, $\mathrm{VO} \cap \mathrm{OP}$ languages are on average more expensive to parse than $\mathrm{VO} \cap \mathrm{PO}$ ones.

One way of integrating Hawkins' insight into a CCG-based theory of linguistic

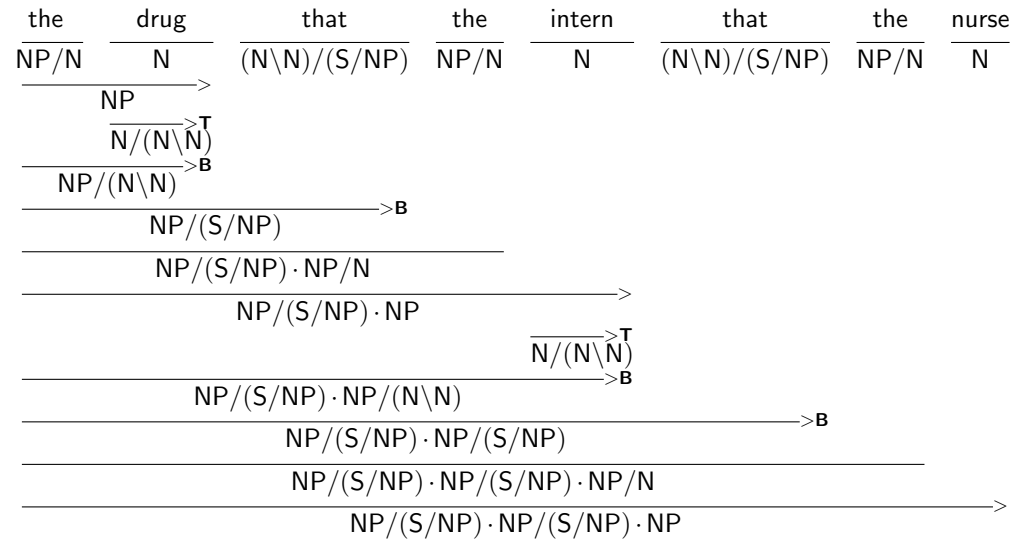
competence is inspired by the work of Johnson (1998) on the parsing cost of categorial derivations. The problem which motivates this work derives from the structural completeness of flexible categorial formalisms like the Lambek calculus — since there are many possible derivations for the same sentence, there is no one structure corresponding to a CFG phrase-structure tree from which the parsing complexity of the sentence may be read off. Johnson proposes using 'proof nets' to solve this problem, since every structurally unambiguous sentence, although having many distinct Lambek derivations, will have just the one proof net. Since the unresolved dependencies at any intermediate point in the parsing of a sentence are modelled by arcs which cross that point in the proof net, a simple measure of the cost of parsing a sentence can be found by computing the average number of arcs which cross each point. Morrill (2000) demonstrates that a number of phenomena in human sentence processing can be explained in such terms, for example the complexity of centre-embedding, garden paths, and quantifier scope phenomena.

We can make use of the capacity for CCG to provide left-to-right incremental derivations to mimic Johnson's proof net analysis. Take the following two synonymous sentences of English:

1. The drug that the intern that the nurse supervised administered cured the patient

2. The patient was cured by the drug that was administered by the intern that was supervised by the nurse.

The first sentence involves multiple centre embeddings and is very difficult to parse. The second sentence on the other hand exhibits multiple peripheral embeddings, and is much easier to parse. The following derivation is intended to represent the partial trace of an incremental shift-reduce parse of the centre-embedding sentence:



This derivation differs from normal CCG derivations in two ways. First of all, backtracking is explicitly represented, for example where the prefix *the drug* is reanalysed as a forward-seeking functor. Secondly, where a word is added to the stack but no reduction can take place, I use a Lambek-style product connective ($\cdot$) to denote this.

If we assume that the number of unresolved dependencies at some point in the derivation is defined as the number of saturated category symbols on the stack at that point, then we get the following figures, which are identical to the number of arcs in Johnson's proof net which cross that point:

the [2] drug [1] that [3] the [5] intern [4] that [6] the [8] nurse [7] supervised administered cured the patient

The fact that this sentence is difficult to process is due to the fact that at one point in its incremental parsing, there are eight unresolved dependencies, which exceeds the bounds of short-term memory. On the other hand, when parsing the synonymous peripheral embedding construction, short term memory load never exceeds four unresolved dependencies; hence the sentence is far easier to process.

In sum, this technique of measuring the short-term memory load at any point in the incremental parsing of a sentence according to a CCG is the non-phrase-structure-based counterpart to Hawkin's idea of measuring the cost of parsing sentences, although the two proposals may well differ in detail. Just as Hawkins generalises his measure of the parsing cost of a sentence to provide a means of determining the cost of parsing a grammar, by averaging over the costs of parsing all the sentences it generates, it seems like a good idea to do the same for flexible categorial formalisms, so as to provide a *parsing-based theory of linguistic competence in CCG*.

Note however that even if it does prove to be possible to find such an explanation for word order universals in terms of a parsing-based ranking of CCGs, this does not negate the original motivation for generalising the CCG category notation. The key point is that the technical innovations of I-CCG, i.e. the lexical inheritance hierarchies and the attribute-value constraint language, are independently required in CCG so as to eliminate lexical redundancy and allow for the expression of a wider range of significant linguistic generalisation. The fact that they also have a role to play in aspects of linguistic explanation is a significant bonus. Indeed, having two ways of ranking CCGs based on grammar-length and parsability is probably to be desired, as long as the rankings coincide. It is after all not implausible that the language faculty evolved in such a way that the most easily processed grammars are also the shortest, and the shortest the most easily processed. This would make an interesting topic for further research.

Interestingly, Newmeyer (2005) has recently revisited the problem of distinguishing the probable and improbable human languages, reaching essentially the same conclusion as Hawkins. Newmeyer takes the Principles and Parameters model as his starting point, arguing (correctly) that it is inadequate as a theory of probable and improbable languages. He goes on to propose a 'division of labour', according to which UG is concerned solely with defining the set of *possible* languages, and the theory of performance accounts for probabilistic language universals. What is lacking from Newmeyer's account is, of course, a theory of the lexicon — there is no discussion of how one might capture the 'medium-sized' lexical generalisations identified by Koenig (1999), i.e. those which are linguistically significant, but nonetheless have important exceptions.

7.3 Summary

The I-CCG formalism defined in chapter 6, which incorporated both a type hierarchy of saturated category symbols and an inheritance hierarchy of lexical types constrained by the feature-based category description language, allowed us to construct maximally non-redundant lexicons for human languages, i.e. lexicons which satisfy the ideals of functionality and atomicity. This chapter turned to the question as to whether the I-CCG formalism also fulfils the original motivation from chapter 4 for generalising the category notation of baseline CCG — to come up with a formalism which predicts uni-

versals of word order in human language in terms of an acquisition-based preference for 'shorter' grammars.

Section 7.1 presented, to this end, a revised definition of the I-CCG formalism, where grammars include an inheritance hierarchy of *argument types* such as 'subject' and 'object'. This step necessitated a stratification of the language of flexible category descriptions across two distinct levels — argument descriptions constrain argument types, and category descriptions constrain lexical types.

Section 7.2 demonstrated that this revised I-CCG formalism is a better theory of word order universals in human language than baseline CCG. First of all, I-CCG predicts that verb-initial languages will be prepositional and vice versa, and also that verb-final languages will be postpositional and vice versa, since in each case the smallest I-CCG for the unattested or rare language types is larger than some I-CCG for an equivalent language from one of the attested types. Secondly, I demonstrated that the observations in Travis (1989) concerning interactions between adverb and indirect object positioning are also predicted by I-CCG in the same way, assuming that there is no argument type which subsumes both direct objects and adverbials but not indirect objects. Finally, I discussed the extent to which the approach considered here can be argued to be a theory of word-order universals in general. A substantial proportion of the word order universals described in Hawkins (1980) and Dryer (1992) are capturable by I-CCG, dependant of course on an appropriate theory of substantive argument types. Notably, I-CCG is much more successful at explaining 'mirror-image' universals than 'non-mirror-image' ones. In order to account for the latter, we must embed both the I-CCG formalism and the preference for simpler inheritance hierarchies inside a system of competing constraints on lexical acquisition.

An interesting question concerns the extent to which the use of argument type hierarchies with SLASH constraints relates to the use of ordering constraints in other formalisms, such as the *linear precedence* (LP) rules in GPSG (Gazdar et al., 1985).

Take the following excerpt from a GPSG grammar involving one ID rule and three LP constraints:

(7.17) $\mathsf{VP} \rightarrow \mathsf{V}, \mathsf{NP}, \mathsf{PP}$

$\mathsf{V} \prec \mathsf{NP}$

$\mathsf{V} \prec \mathsf{PP}$

$$\mathsf{NP} \prec \mathsf{PP}$$

The ID rule here licenses a subtree whose root is labelled VP, with exactly three children labelled V (the head), NP (the direct object) and PP (the indirect object) respectively. The ID rule places no restrictions on the ordering of the child nodes, leaving this to the three LP rules.

The first two LP rules regulate the relative orderings of the head and the two objects, stating that the head must always precede both objects. These two constraints can be encoded effectively in an I-CCG argument hierarchy such as that in Figure 7.3, where the appropriate SLASH feature is added to some common supertype of the direct object ($\mathsf{obj_d}$) and indirect object ($\mathsf{obj_i}$) types, in this case SLASH /. The last LP rule is different however. Rather than regulating head-dependent ordering, it specifies the relative ordering of two dependents of the same head, in this case stating that the direct object must always precede the indirect object.

This latter kind of LP rule *cannot* be encoded in terms of constraints in an I-CCG argument type hierarchy. This information must be encoded in the lexical inheritance hierarchy proper, as a set of constraints on the relevant type:

(7.18)
$$\begin{array}{l}\mathsf{ARG\ obj_d}\\ \mathsf{RES\ ARG\ obj_i}\\ \mathsf{RES\ RES\ S}\end{array}$$

Since the direct object is specified as combining first with the verb, and the indirect object second, and since the argument hierarchy encodes the information that both *follow* the verb, both hierarchies taken together will provide an *indirect* specification of the relative ordering of the two arguments. Thus, it appears that the use of argument types in the I-CCG formalism involved assuming some kind of ontological distinction between these two kinds of linear ordering constraints.

Chapter 8

Conclusion

> *To the extent that* [more traditional theories of grammar] *provide a systematic account of the relation between interpretations ... and syntactic categories, they provide what amounts to a theory of the categorial lexicon — a component of the present theory that continues to be lacking in this and preceding discussions of CCG.* (Steedman, 2000)
>
> *CCG's notion of universal grammar just got more universal, and we now await a fuller and more cross-linguistically articulated theory of the lexicon.* (Baldridge, 2002)

It is clear from the discussion in chapter 3 of Baldridge (2002) that the perceived need for a theory of the CCG lexicon actually involves two distinct problems. The first of these is the CCG analogue of the traditional problem of linguistic competence, i.e. the fact that even if it does turn out to be the case that there is a CCG for every human language, it is certainly not true that every language for which there is a CCG is a probable human language. Baldridge suggests that this problem may be solved in two steps. First of all, he specifies a universal alphabet of saturated category symbols, conceptualised as typed feature structures described by attribute-value matrices and constrained by a universal type signature of the kind discussed in subsection 5.5.2 of this thesis. Secondly, he suggests that aspects of Government and Binding Theory (Chomsky, 1981) may be utilised to distinguish those sets of unsaturated categories which are possible human language lexicons from those which are not. In short, this involves finding a formal system which generates finite sets of elementary trees over the universal alphabet of saturated category symbols, and then applying a tree-to-category algorithm, such as that proposed in Hockenmaier (2003), so as to convert this set of trees into a set of unsaturated categories.

The second problem involving the CCG lexicon revolves around the fact that, as Baldridge puts it (p.64), "CCG still lacks an explicit theory of the lexicon which declares how the lexicon is structured so that systematic relationships are encoded with minimal redundancy". Following Villavicencio (2002), he suggests that the obvious way to solve this problem is to incorporate lexical inheritance hierarchies into the CCG formalism, provided that some appropriate constraint language is formulated.

This thesis has in essence argued that these two aspects of the problem of the CCG lexicon are more closely related than previously assumed. In particular, I contend that in solving the problem of redundancy in CCG lexicons by means of lexical inheritance, we also have a partial solution to the problem of distinguishing probable CCGs from improbable ones.

The initial motivation for the work reported here was to explore the incorporation of techniques from 'unification-based' grammar formalisms such as HPSG into the CCG formalism, as a means of solving the lexical redundancy problem. It became clear at a very early stage that the most natural way to go about doing this was to assume a 'model-theoretic' perspective on CCG category notation. This involves distinguishing category 'descriptions' from underlying category 'models', in exactly the same way that unification-based formalisms distinguish feature structure descriptions (i.e. attribute-value matrices) from the underlying feature structures. At the same time, we can retain the traditional CCG combinatory system by defining rules such as application, raising and composition as operations on category models. By taking such a 'partially' model-theoretic approach to the formalisation of CCG, we can retain the essential 'CCG-ness' of the resulting formalism, compared to those approaches which regard CCG as simply another application of typed feature logic (Villavicencio, 2002), (Beavers, 2004).

It soon became clear that, provided the logic of categories is set up appropriately, a CCG formalism based around the concept of lexical inheritance can also be argued to capture certain well-known word order universals in terms of a preference for 'simpler' hierarchies. In other words, taking a model-theoretic, or 'constraint-based', perspective on CCG category notation, with its accompanying distinction between descriptions and their underlying models, allows for the construction of a hybrid grammar formalism which:

1. provides non-redundant lexicons for human languages

2. captures a range of implicational word order universals as formal universals, i.e. in terms of an acquisition-based preference for shorter grammars

Chapter 1 set out the background for the work contained within this thesis, i.e. the search for a grammar formalism which in some way 'embodies' human linguistic competence. In particular, I discussed two contrasting perspectives on grammar formalisms: (a) the generative-enumerative perspective, according to which a grammar is a set of instructions for generating sentences; and (b) the model-theoretic perspective, where a grammar is conceptualised as a set of constraints on structures.

Chapter 2 introduced the CCG formalism, and attempted to define a 'baseline' version, constituting the intersection of the formal assumptions made by all recent work involving CCG, for example Steedman (1996), Steedman (2000), Baldridge (2002), Baldridge and Kruijff (2003), Bozsahin (2002), Trechsel (2000), Hockenmaier (2003), Hoffman (1995), Beavers (2004). The baseline formalism includes the modalised bidirectional unsaturated category notation, with no assumptions made about the possible internal structure of saturated category symbols. This baseline CCG formalism fulfils two functions in the overall narrative of this thesis. First of all, it made it easier to analyse the rationale behind the various proposals to generalise CCG category notation found in the literature. Secondly, it provided a formal basis on which to build the 'inheritance-driven' generalised CCG formalism in an incremental manner.

One important aspect of the formalisation of the baseline CCG category notation was that it was presented 'model-theoretically', in terms of a basic distinction between category descriptions and category models, the two notions related by logical satisfaction. Whilst this distinction was not really made to do any kind of interesting work in chapter 2, it was crucial to the deconstruction of the baseline notation and the development of 'flexible' category description notations in chapters 6 and 7.

Chapter 3 discussed the CCG formalism as a theory of human linguistic competence. I reviewed a number of arguments from the CCG literature concerning the relationship between the generative capacity of CCG and predictions it makes about the permitted complexity of unbounded dependency constructions in human languages. I then turned to the subject of implicational universals and considered two examples which CCG has been argued to capture formally. The first of these was the observation that human languages with backward-gapping constructions are generally verb-final, and

the second involved a correlation between the extractability of embedded subjects in a language and the possibility of free subject inversion. I paid particular attention to the underlying rationale behind the claim that CCG predicts these universals, and concluded that, at least in the second case, the CCG analysis is based on the notion of grammar-ranking in terms of an evaluation measure of grammar length.

Chapter 4 drew attention to a couple of implicational universals which are not captured by the baseline CCG formalism in terms of lexical economy. The first of these was the familiar Greenbergian correlation between basic clausal ordering in a human language and the existence of prepositions or postpositions in its lexicon. The second was a lesser-known observation from the Principles and Parameters literature which posits a link between verb-object and verb-modifier serialisation in human languages. I argued that, although there is no real sense in which baseline CCG can be argued to predict these universals formally, the generalisations do appear to be based on some notion of grammatical economy, i.e. the number of descriptive statements required to characterise the phenomena. This insight provided the motivation for the next three chapters — to extend the definition of the baseline CCG formalism so as to be able to express the relevant kinds of generalisation.

Chapter 5 introduced the 'type-hierarchical' CCG (T-CCG) formalism, an extension of CCG where the alphabet of saturated category symbols is organised into a type hierarchy. The motivation for this extra machinery derived from the requirement that the information contained within a human language lexicon be stored in a reasonably efficient manner, i.e. children acquiring human languages aim to construct lexicons which satisfy the ideals of functionality and atomicity. I argued that the T-CCG formalism renders redundant a number of other proposed generalisations of the baseline CCG category notation, where saturated categories either are conceptualised as typed feature structures or are prefixed by unary modalities.

Chapter 6 introduced the 'inheritance-driven' CCG (I-CCG) formalism, and extension of T-CCG incorporating an alphabet of lexical category symbols organised into an inheritance hierarchy over a simple attribute-value-style category description language. I demonstrated that I-CCG allows for the construction of highly efficient human language lexicons, satisfying both the ideals of functionality and atomicity. I argued that I-CCG renders redundant generalisations of the CCG category notation involving multisets of arguments.

Chapter 7 argued that the I-CCG formalism also captures, as primarily *formal* universals, a substantial proportion of those implicational universals that were shown to be problematic for the baseline CCG formalism. This explanation takes the form of a basic preference for lexical inheritance hierarchies with fewer constraints. Although a number of 'non-mirror-image' universals can only be partially explained in this way, all of the word order universals described in Dryer (1992) involving consistent operator-operand ordering, are predicted by a combination of the I-CCG formalism and a preference for simpler lexical inheritance hierarchies, assuming an appropriate extra-grammatical specification of which elements are functors in which constructions. In essence, I argued that lexical inheritance hierarchies are a natural representation for capturing concepts such as the Natural Serialisation Principle of Vennemann (1972) and the principle of Cross Category Harmony from Hawkins (1980), both of which are intended to describe the role of analogy in word order acquisition.

It should be emphasised here that the generalisations which form the basis of this work are all 'cross-linguistic' ones, rather than observations about syntactic asymmetries found in particular languages. In this respect, this dissertation constitutes, at least to a certain extent, a departure from the methodology of much previous work in the CCG formalism. However, it is important to point out that observations about the distribution of syntactic phenomena across groups of languages has been touched upon in previous CCG literature — section 3.2 discussed two putative language universals which are discussed at length in Steedman (1996) and Steedman (2000).

I-CCG and HPSG

It should be clear from the discussion in chapters 5 through 7 that many aspects of the I-CCG formalism are influenced by the HPSG formalism of Pollard and Sag (1987) and Pollard and Sag (1994), just as HPSG was itself heavily informed by the categorial grammar formalisms popular in the mid-1980s. First of all, both I-CCG and HPSG are model-theoretic formalisms, in the sense that they are most naturally conceptualised as constraint-satisfaction systems rather than as derivational systems. The main reason for this is that both formalisms rely on lexical inheritance hierarchies over attribute-value style constraint languages to organise lexical information in an efficient manner and to capture significant linguistic generalisations about the behaviour of words and morphemes.

However, it is also the case that significant differences remain between I-CCG and HPSG. The most important of these is the fact that I-CCG has a clear distinction between its formal and substantive aspects, something which is lacking in the HPSG formalism. Recall again that HPSG is built upon an underlying feature logic, for example the 'speciate reentrant logic' of King (1989) or the typed feature logic of Carpenter (1992). The HPSG theory itself constitutes an alphabet of universal linguistic types with appropriate features, and a set of constraints common to the grammars of all human languages, the analogue to the Chomskyan notion of UG. An HPSG grammar of a particular language is then a consistent superset of UG, adding language-specific types, features and constraints.

From this perspective, the most important fact about HPSG is that it is not a grammar formalism (i.e. theory of human linguistic competence) at all, but is rather an 'application' of feature logic, in the same sense that web technologies like XHTML and RSS are applications of an underlying XML standard. The I-CCG formalism stands in complete contrast to this. It is clear exactly what is an I-CCG and what is not. In addition, it is clear whether or not a particular sentence is generated by a given I-CCG. In short, I-CCG makes clear, empirically falsifiable predictions about human linguistic competence, and thus counts as a theory in the strong sense of the term, in counterpart to HPSG which is essentially a very general description language for relating facts about different dimensions of human language.

The differences between I-CCG and HPSG are a result of the different approaches to the problem of human languistic competence that they developed within. The HPSG tradition has assumed a 'top-down' approach, starting with an extremely expressive, Turing-complete formalism and attempting to constrain the class of languages it generates by means of a complex network of substantive universals. The CCG paradigm, on the other hand, has pursued a 'bottom-up' approach, starting out with the basic context-free apparatus provided by application-only categorial grammar, and attempting to add just enough extra formal machinery to capture human language phenomena. This thesis has attempted to follow this path, by starting out with the baseline CCG vocabulary of atomic saturated category symbols and lexicons as mappings from morphemes to categories, and adding the minimal amount of descriptive machinery to capture the kinds of generalisations which are found in human language grammars.

Thus, I consider this thesis as a contribution to both the categorial grammar literature and the unification-based grammar literature. I propose I-CCG as an alternative

to HPSG, incorporating the best things about the latter (the structure/description dichotomy and lexical inheritance hierarchies) whilst maintaining a simple, restrictive formalism, with far fewer degrees of freedom. In particular, I argue that lexical inheritance has a role to play not only in the organisation of an individual's linguistic competence, but also in the theory of language universals itself, in the sense that they embody the role that analogy has to play in the process of language acquisition.

I-CCG and default inheritance

The essential feature of the I-CCG formalism is that a human language grammar includes an inheritance hierarchy of lexical types, and the notion of inheritance assumed in this thesis is strictly 'monotonic'. In other words, the set of constraints inherited by a lexical type is consistent, and constraints on lower types cannot 'override' those inherited from higher types. In this respect, I-CCG as formulated here differs from theories of human language which assume that lexical inheritance is non-monotonic, with some inherited constraints being mere defaults which can be discarded if they conflict with a competing constraint from a more specific supertype.

One point which is important here is that any generalisation which can be stated in a non-monotonic multiple inheritance hierarchy can also be stated in a monotonic multiple inheritance hierarchy, albeit with the addition of extra types. In other words, default inheritance does not add any expressivity to a grammar formalism, it just allows for the more compact encoding of a lexical hierarchy. However, non-monotonic inheritance adds significant computational overheads to any parsing system, since it is possible for a subtype to inherit conflicting constraints from supertypes, and some means is required to resolve such conflicts. This is particularly important in cases where neither of the two supertypes subsumes the other.

On the other hand, should it be decided at some future date that default reasoning is needed in I-CCG, the elaboration of such a formalism should be no more difficult than for similar versions of HPSG (Lascarides and Copestake, 1999). Note however that the demands of strong competence on grammar formalisms as models of human linguistic competence necessitate that the human sentence processing model resolve any conflicts on-line, rather than during some kind of precompilation step translating default inheritance hierarchies into monotonic ones (Copestake, 2002).

One criticism to the use of strictly monotonic reasoning in lexical inheritance hierarchies is relevant to the assumption that the I-CCG formalism will be used as the

basis for a future grammar engineering system, and concerns the *scalability* of I-CCG grammars. In other words, in what ways will the lexical inheritance hierarchies need to change as coverage is increased in the course of grammar development.[1]

In this kind of scenario, having access to a system which allows constraint overriding is extremely helpful to a grammar engineer. Rather than constantly having to redraft a subhierarchy to include a class of words which are related to, but not identical to, an already existing class, a default contraint can be placed on a common supertype, which can then be overridden in the new subclass. This has the advantage of restricting the set of lexical types to a smaller and more manageable number, thus making the hierarchy easier to grasp by the engineer.

However, one lesson that development of recent grammar engineering environments like the LKB (Copestake, 2002) has taught us is that we need a distinction between the *tools* that we use for constructing and editing hierarchies, and the underlying structure of the hierarchies themselves. It is important to note here that the use of default inheritance in the LKB is to be seen as a tool for the convenience of the grammar engineers rather than an intrinsic part of the underlying formalism, since default hierarchies are always expanded into non-monotonic ones when the grammar is compiled. The development of tools for lexical hierarchy editing and expansion is an important task, which thus can be argued to subsume the use of default constraints as well as other concepts such as 'conjunctive type construction' (Carpenter, 1992), whereby HPSG-style partition diagrams are converted automatically into type hierarchies.

I-CCG and constraint ranking

The strong competence requirement on grammar formalisms purporting to be theories of human linguistic competence is also relevant to the discussion of so-called 'optimality-theoretic' grammar formalisms.

Recall that, in the I-CCG formalism developed in chapter 6 and 7, constraints on linguistic structures are organised hierarchically, in terms of an alphabet of lexical types. Optimality-theoretic formalisms are comparable to this in the sense that they also are expressed in terms of constraints on structures. However, important differences remain.

[1] Of course, this consideration is equally significant from the perspective of language acquisition.

First of all, whereas I-CCG follows HPSG in assuming that constraints are recursively defined in terms of attributes and values, optimality-theoretic formalisms assume a fixed, finite alphabet of *atomic* constraints. Perhaps the best known application of the optimality-theoretic ideas to the domain of syntax is the work of Grimshaw (1997), who assumes that linguistic structures are $\bar{\text{X}}$-theoretic trees of the kind found in 1980s 'Government and Binding' theory. These trees are constrained by atomic constraints such as 'OP-SPEC' (operators should be specifiers) and 'STAY' (traces are discouraged). The local satisfaction definition specifies which constraints are associated at which nodes in a tree, i.e. OP-SPEC is satisfied at a node if either the node is not headed by an operator (i.e. quantifier or *wh*-element) or the node is a specifier.

Secondly, whereas an I-CCG grammar consists of a lexicon and a set of constraints organised hierarchically in terms of lexical types, an optimality-theoretic grammar consists of a lexicon and a weak ordering relation on the alphabet of atomic constraints. For example, a grammar can specify that either OP-SPEC is more important than STAY (i.e. movement of operators is encouraged), or that STAY is more important than OP-SPEC (i.e. movement of operators is discouraged). Thus, in optimality-theoretic formalisms, constraints are violable, but crucially differ in degree of violability.

Thirdly, determining the well-formedness of a category structure in I-CCG is a strictly local process, simply involving a check that every point in the structure satisfies every relevant constraint. On the other hand, determining well-formedness with an optimality-theoretic grammar requires universal quantification over the class of structures. A structure cannot be said to be well-formed in isolation, since well-formedness is not a property but rather a relation. Structure $\mathcal{M}_1$ is *better*-formed than structure $\mathcal{M}_2$ according to grammar G just in case:

- $\mathcal{M}_1$ and $\mathcal{M}_2$ are equivalent in some way, e.g. made up from the same lexical items, expressing the same proposition, ...
- the highest-ranked constraint not satisfied by $\mathcal{M}_1$ is ranked below some constraint not satisfied by $\mathcal{M}_2$

The *best*-formed structure is then the member of the equivalence class which is better-formed than all the others.

Thus, if we are to assume the strong competence criterion for theories of human linguistic competence, it should be clear that optimality-theoretic grammars will have

great difficulty satisfying this requirement. Indeed, in the case where the set of candidate structures is infinite, determining the grammaticality of a string will be undecidable. In addition, constraint-ranking in an optimality-theoretic grammar is not independently motivated, unlike lexical inheritance hierarchies which are required to eliminate lexical redundancy.

In sum, it is not clear that many proposed optimality-theoretic formalisms are theories of linguistic competence at all. Rather they appear to constitute a kind of linguistic meta-theory, describing how conflicting communicative demands interact in processes such as child language acquisition, language change and language evolution. From this perspective, the work in this thesis can be interpreted as proposing a general, violable constraint of 'shortness' on human language grammars.

I-CCG and lexical rules

Finally, the I-CCG formalism as defined in this thesis does not incorporate the notion of 'lexical rules', i.e. statements of the form 'if lexicon L assigns form π to type α, then it also assigns form ϕ' to type α''. This omission is quite deliberate. Firstly, Bozsahin (2002) has argued that inflectional and derivational suffixes must be handled as normal unsaturated categories in CCG, if highly agglutinating languages such as Turkish are to be described. In this sense, the effect of a lexical rule saying something like 'if π is a N_{sg} then $\pi + \mathsf{s}$ is a N_{pl}' is captured by assigning the plural suffix to the unsaturated category $\mathsf{N}_{\mathsf{pl}} \backslash \mathsf{N}_{\mathsf{sg}}$. Evidence from the language universals literature suggesting that head-initial languages are much more likely to have prefixes than suffixes supports this view.

Secondly, recent work in the HPSG formalism (Koenig, 1999) has moved towards using the disjunction implicit in inheritance hierarchies to replace 'homophonous' lexical rules of the form 'if lexicon L assigns form π to type α, then it also assigns it to type α''. Thus, a lexical rule like 'if π is an infinitive then it is also a non-third-person-singular present tense verb' can be eliminated simply by assigning such forms to an underspecified type which subsumes both verbal infinitives and non-third-person-singular present tense verb forms.

Bibliography

Ajdukiewicz, K. (1935). Die syntaktische Konnexität. *Studia Philosophica*, 1:1–27.

Bach, E. (1979). Control in Montague Grammar. *Linguistic Inquiry*, 10:515–531.

Baldridge, J. (2002). *Lexically Specified Derivational Control in Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh.

Baldridge, J. and Kruijff, G.-J. (2002). Coupling CCG with Hybrid Logic Dependency Semantics. In *Proceedings of the 40th Meeting of the Association for Computational Linguistics (ACL), Philadelphia*, pages 319–326.

Baldridge, J. and Kruijff, G.-J. M. (2003). Multi-Modal Combinatory Categorial Grammar. In *Proceedings of the Tenth Conference of the European Chapter of the Association for Computational Linguistics*, pages 211–218.

Baldridge, J. M. (1998). Local Scrambling and Syntactic Asymmetries in Tagalog. Master's thesis, Department of Linguistics, University of Pennsylvania.

Bar-Hillel, Y. (1953). A Quasi-Arithmetical Notation for Syntactic Description. *Language*, 29:47–58.

Bar-Hillel, Y., Gaifman, C., and Shamir, E. (1964). On Categorial and Phrase Structure Grammars. In Bar-Hillel, Y., editor, *Language and Information*, pages 99–115. Addison-Wesley, Reading MA.

Beavers, J. (2004). Type-inheritance Combinatory Categorial Grammar. In *Proceedings of the 20th International Conference on Computational Linguistics, University of Geneva*.

Blackburn, P. (1993). Modal Logic and Attribute-Value Structures. In de Rijke, M., editor, *Diamonds and Defaults*, pages 19–65. Kluwer Academic Publishers, Dordrecht NL.

Blackburn, P. (2000). Representation, reasoning and relational structures: A hybrid logic manifesto. *Logic Journal of the IGPL*, 8(3):339–365.

Blackburn, P., Gardent, C., and Meyer-Viol, W. (1993). Talking About Trees. In *Proceedings of the Sixth Conference of the European Chapter of the Association for Computational Linguistics, Utrecht*, pages 21–29.

Bos, J., Clark, S., Steedman, M., Curran, J. R., and Hockenmaier, J. (2004). Wide-coverage semantic representations from a CCG parser. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING-04), Geneva, Switzerland*, pages 1240–1246.

Bozsahin, C. (2002). The combinatory morphemic lexicon. *Computational Linguistics*, 28(2):145–186.

Bresnan, J. (1977). Variables in transformations. In Culicover, P., Wasow, T., and Akmajian, A., editors, *Formal Syntax*. Academic Press, New York.

Bresnan, J., editor (1982). *The Mental Representation of Grammatical Relations*. MIT Press, Cambridge MA.

Bresnan, J. and Mchombo, S. A. (1987). Topic, pronoun, and agreement in Chicheŵa. *Language*, 63(4):741–782.

Carpenter, B. (1992). *The Logic of Typed Feature Structures*. Cambridge Tracts in Theoretical Computer Science. Cambridge University Press.

Chomsky, N. A. (1956). Three models for the description of language. *IRE Transactions on Information Theory*, 2:113–124.

Chomsky, N. A. (1957). *Syntactic Structures*. Number 4 in Janua Linguarum. Mouton, Den Haag NL.

Chomsky, N. A. (1959). On certain formal properties of grammars. *Information and Control*, 2:137–167.

Chomsky, N. A. (1963). Formal properties of grammars. In Luce, R. D., Bush, R., and Galanter, E., editors, *Handbook of Mathematical Psychology, Volume 2*. Wiley, New York.

Chomsky, N. A. (1964). *Current Issues In Linguistic Theory*. Number 38 in Janua Linguarum. Mouton, Den Haag NL.

Chomsky, N. A. (1965). *Aspects of the Theory of Syntax*. MIT Press, Cambridge MA.

Chomsky, N. A. (1972). *Studies on Semantics in Generative Grammar*. Mouton, Den Haag.

Chomsky, N. A. (1981). *Lectures on Government and Binding*. Foris, Dordrecht NL.

Clark, S. and Curran, J. R. (2004). Parsing the WSJ using CCG and log-linear models. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04), Barcelona, Spain*, pages 104–111.

Comrie, B. (1989). *Language Universals and Linguistic Typology*. Basil Blackwell, 2 edition.

Copestake, A. (2002). *Implementing Typed Feature Structure Grammars*. CSLI Publications, Stanford CA.

Copestake, A., Flickinger, D., Sag, I., and Pollard, C. (2005). Minimal recursion semantics: an introduction. *Journal of Research on Language and Computation*, 3:281–332.

Curry, H. B. and Feys, R. (1958). *Combinatory Logic*, volume 1. North Holland, Amsterdam NL.

Dixon, R. (1994). *Ergativity*. Cambridge University Press.

Dryer, M. S. (1992). The Greenbergian word order correlations. *Language*, 68(1):81–138.

Erkan, G. (2003). A Type System for Combinatory Categorial Grammar. Master's thesis, Middle East Technical University, Ankara.

Flickinger, D. P. (1987). *Lexical Rules in the Hierarchical Lexicon*. PhD thesis, Stanford University.

Flynn, M. (1985). *Structure Building Operations and Word Order*. Outstanding Dissertations in Linguistics. Garland Publishing, Inc.

Fodor, J. D. and Crain, S. (1990). Phrase Structure Parameters. *Linguistics and Philosophy*, 13:619–659.

Foster, J. (1990). *A Theory of Word-Order in Categorial Grammar, with Special Reference to Spanish*. PhD thesis, University of York.

Gazdar, G. (1988). Applicability of indexed grammars to natural languages. In Reyle, U. and Rohrer, C., editors, *Natural Language Parsing and Linguistic Theories*. D.Reidel, Dordrecht, NL.

Gazdar, G., Klein, E., Pullum, G. K., and Sag, I. (1985). *Generalised Phrase Structure Grammar*. Blackwell, Oxford UK.

Gazdar, G., Pullum, G. K., Carpenter, R., Klein, E., Hukari, T. E., and Levine, R. D. (1988). Category structures. *Computational Linguistics*, 14:1–19.

Greenberg, J. (1963). Some universals of grammar with particular reference to the order of meaningful elements. In Greenberg, J., editor, *Universals of Language*, pages 73–113. MIT Press, Cambridge MA.

Grimshaw, J. (1997). Projections, Heads and Optimality. *Linguistic Inquiry*, 28(3):373–422.

Harel, D., Kozen, D., and Tiuryn, J. (2000). *Dynamic Logic*. MIT Press, Cambridge MA.

Hawkins, J. A. (1980). On implicational and distributional universals of word order. *Journal of Linguistics*, 16:193–235.

Hawkins, J. A. (1990). A parsing theory of word order universals. *Linguistic Inquiry*, 21(2):223–261.

Hepple, M. (1995). Mixing Modes of Linguistic Description in Categorial Grammar. In *Proceedings of the Seventh Conference of the European Chapter of the Association for Computational Linguistics – EACL-7, Dublin IRL*, pages 127–132.

Hockenmaier, J. (2003). *Data and Models for Statistical Parsing with Combinatory Categorial Grammar*. PhD thesis, University of Edinburgh.

Hockenmaier, J. and Steedman, M. (2002). Acquiring Compact Lexicalized Grammars from a Cleaner Treebank. In *Proceedings of Third International Conference on Language Resources and Evaluation, Las Palmas, ESP*.

Hoffman, B. (1995). *The Computational Analysis of the Syntax and Interpretation of "Free" Word Order in Turkish*. PhD thesis, University of Pennsylvania.

Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages and Computation*. Addison Wesley.

Ingram, D. (1989). *First Language Acquisition: Methods, Description and Explanation*. Cambridge University Press.

Jackendoff, R. S. (1977). *X-Bar Syntax: A Study of Phrase Structure*. Linguistic Inquiry Monographs. MIT Press, Cambridge MA.

Johnson, M. (1988). *Attribute-Value Logic and the Theory of Grammar*. Number 16 in CSLI Lecture Notes. Center for the Study of Language and Information, Stanford CA.

Johnson, M. (1998). Proof nets and the complexity of processing center embedded constructions. *Journal of Logic, Language and Information*, 7:433–447.

Joshi, A., Rambow, O., and Becker, T. (2000). Complexity of scrambling — a new twist to the competence-performance distinction. In Abeillé, A. and Rambow, O., editors, *Tree Adjoining Grammars: Formalisms, Linguistic Analysis and Processing*. CSLI Publications, Stanford CA.

Joshi, A. K., Levy, L., and Takahashi, M. (1975). Tree adjunct grammars. *Journal of the Computer and System Sciences*, 10:136–163.

Kamp, H. and Reyle, U. (1993). *From Discourse to Logic*. Kluwer.

Kasper, R. T. and Rounds, W. C. (1986). A Logical Semantics for Feature Structures. In *Proceedings of the 24th Annual Conference of the Association for Computational Linguistics*, pages 257–266.

King, P. J. (1989). *A logical formalism for head-driven phrase structure grammar*. PhD thesis, University of Manchester.

Koenig, J.-P. (1999). *Lexical Relations*. Stanford Monographs in Linguistics. CSLI Publications, Stanford CA.

Kruijff, G.-J. (2001). *A Categorial-Modal Architecture of Informativity: Dependency Grammar, Logic and Information Structure*. PhD thesis, Charles University, Prague.

Lambek, J. (1958). The Mathematics of Sentence Structure. *American Mathematical Monthly*, 65:154–170.

Lascarides, A. and Copestake, A. (1999). Default Representation in Constraint-Based Frameworks. *Computational Linguistics*, 25(1):55–105.

Lehmann, W. P. (1973). A structural principle of language and its implications. *Language*, pages 47–66.

Lyons, J. (1991). *Chomsky*. Fontana Modern Masters. Fontana Press, 3rd edition.

McCawley, J. D. (1968). Concerning the Base Component of a Transformational Grammar. *Foundations of Language*, 4:243–269.

McConville, M. (2006). Inheritance and the CCG lexicon. In McCarthy, D. and Wintner, S., editors, *EACL 2006: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics, Trento, Italy, 3-7 April 2006*, pages 1–8.

Moortgat, M. (1997). Categorial type logics. In van Benthem, J. and ter Meulen, A., editors, *Handbook of Logic and Language*, pages 93–177. North Holland, Amsterdam, NL.

Morrill, G. (1994). *Type Logical Grammar*. Kluwer, Dordrecht, NL.

Morrill, G. (2000). Incremental processing and acceptability. *Computational Linguistics*, 26(3):319–338.

Newmeyer, F. J. (2005). *Possible and Probable Languages: A Generative Perspective on Linguistic Typology*. Oxford University Press.

Pentus, M. (1993). Lambek Grammars are Context-Free. In *Proceedings of the IEEE Symposium on Logic in Computer Science, Montreal*, pages 429–433.

Perlmutter, D. (1971). *Deep and Surface Structure Constraints in Syntax*. Hole, Rinehart and Winston, New York.

Pollard, C. J. and Sag, I. A. (1987). *Information-Based Syntax and Semantics, Volume 1 – Fundamentals*. Number 13 in CSLI Lecture Notes. CSLI Publications, Stanford CA.

Pollard, C. J. and Sag, I. A. (1994). *Head-Driven Phrase Structure Grammar*. The University of Chicago Press.

Prince, A. and Smolensky, P. (1993). Optimality theory: Constraint interaction in generative grammar. Technical Report 2, Center for Cognitive Science, Rutgers University.

Pullum, G. K. and Scholz, B. C. (2001). On the distinction between model-theoretic and generative-enumerative syntactic frameworks. In de Groote, P., Morrill, G., and Retoré, C., editors, *Logical Aspects of Computational Linguistics: 4th International Conference*, number 2099 in Lecture Notes in Artificial Intelligence, pages 17–43. Springer Verlag.

Rambow, O. (1994). *Formal and Computational Aspects of Natural Language Syntax*. PhD thesis, Department of Computer and Information Sciences, University of Pennsylvania.

Rizzi, L. (1982). *Issues In Italian Syntax*. Studies in Generative Grammar. Foris Publication, Dordrecht, Holland.

Rogers, J. (1996). A model-theoretic framework for theories of syntax. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics, Santa Cruz CA*, pages 10–16.

Rogers, J. (1998). *A Descriptive Approach to Language-Theoretic Complexity*. Studies in Logic, Language and Information. CSLI Publications, Stanford CA/FoLLI.

Rogers, J. (2003). wMSO theories as grammar formalisms. *Theoretical Computer Science*, 293(2):291–320.

Ross, J. R. (1970). Gapping and the order of constituents. In Bierwisch, M. and Heidolph, K., editors, *Progress in Linguistics*, pages 249–259. Mouton, Den Haag.

Shieber, S. (1985). Evidence againast the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343.

Shieber, S. M. (1986). *An Introduction to Unification-Based Approaches to Grammar*. CSLI Publications, Stanford CA.

Steedman, M. (1985). Dependency and Coordination in the Grammar of Dutch and English. *Language*, 61:523–568.

Steedman, M. (1990). Gapping as constituent coordination. *Linguistics and Philosophy*, 13:207–263.

Steedman, M. (1996). *Surface Structure and Interpretation*. Number 30 in Linguistic Inquiry Monographs. MIT Press, Cambridge MA.

Steedman, M. (2000). *The Syntactic Process*. MIT Press, Cambridge MA.

Stromswold, K. (1995). The acquisition of subject and object *Wh*-questions. *Language Acquisition*, 4:5–48.

Traat, M. and Bos, J. (2004). Unification Combinatory Categorial Grammar: Combining information structure and discourse representations. In *Proceedings of COLING 2004*.

Travis, L. (1984). *Parameters and Effects of Word Order variation*. PhD thesis, MIT.

Travis, L. (1989). Parameters of Phrase Structure. In Baltin, M. R. and Kroch, A. S., editors, *Alternative Conceptions of Phrase Structure*, chapter 11, pages 263–279. University of Chicago Press.

Trechsel, F. (2000). A CCG account of Tzotzil pied piping. *Natural Language and Linguistic Theory*, 18:611–663.

Uszkoreit, H. (1986). Categorial Unification Grammars. In *Proceedings of the 11th International Conference on Computational Linguistics, Bonn*, pages 187–194.

van Riemsdijk, H. and Williams, E. (1986). *Introduction to the Theory of Grammar*. MIT Press, Cambridge MA.

Vennemann, T. (1972). Analogy in generative grammar, the origin of word order. In Heilmann, L., editor, *Proceedings of the Eleventh International Congress of Linguists*, volume 2, pages 79–93. Il Mulino, Bologna, Italy.

Villavicencio, A. (2002). *The Acquisition of a Unification-Based Generalised Categorial Grammar*. PhD thesis, Computer Laboratory, University of Cambridge.

Weir, D. and Joshi, A. (1988). Combinatory Categorial Grammars: Generative power and relation to linear cf rewriting systems. In *Proceedings of the 26th Annual Meeting of the Association for Computational Linguistics, Buffalo, NY*, pages 278–285. Morgan Kaufmann, San Francisco CA.

White, M. and Baldridge, J. M. (2003). Adapting chart realization to CCG. In *Proceedings of the Ninth European Workshop on Natural Language Generation (EWNLG)*.

Zeevat, H., Klein, E., and Calder, J. (1987). Unification Categorial Grammar. In Haddock, N., Klein, E., and Morrill, G., editors, *Categorial Grammar, Unification Grammar and Parsing*, Working Papers in Cognitive Science. Centre for Cognitive Science, University of Edinburgh.