

**UCLA**

**UCLA Electronic Theses and Dissertations**

**Title**

Slaying the Great Green Dragon: Learning and modelling iterable ordered optional adjuncts

**Permalink**

<https://escholarship.org/uc/item/640605fb>

**Author**

Fowlie, Meaghan

**Publication Date**

2017

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Slaying the Great Green Dragon:  
Learning and modelling iterable ordered optional adjuncts

A dissertation submitted in partial satisfaction  
of the requirements for the degree  
Doctor of Philosophy in Linguistics

by

Meaghan Fowlie

2017

© Copyright by

Meaghan Fowlie

2017

ABSTRACT OF THE DISSERTATION

Slaying the Great Green Dragon:  
Learning and modelling iterable ordered optional adjuncts

by

Meaghan Fowlie

Doctor of Philosophy in Linguistics

University of California, Los Angeles, 2017

Professor Edward P. Stabler, Chair

Adjuncts and arguments exhibit different syntactic behaviours, but modelling this difference in minimalist syntax is challenging: on the one hand, adjuncts differ from arguments in that they are optional, transparent, and iterable, but on the other hand they are often strictly ordered, reflecting the kind of strict selection seen in argument application. The former properties mean the derivation proceeds the same way whether or not the adjuncts are present, but the latter means the derivation must know which adjuncts have already been adjoined, to avoid adjoining new ones out of order. This dissertation proposes a precise minimalist model of adjuncts that accounts for both behaviours. The second half considers the learnability of two closely related properties of adjuncts: their optionality and iterability. Many formal learning models predict a relationship between optionality and iterability, and any learning model of human language needs to be able to generalise from limited to indefinite repetition, since many languages include such sentences as “I really really really ... really love linguistics”. All of the formal models I examine make this generalisation. A study of people learning an artificial language indicates that people also make this generalisation.

The dissertation of Meaghan Fowlie is approved.

Edward Keenan

Martin Monti

Carson Schutze

Edward P. Stabler, Committee Chair

University of California, Los Angeles

2017

To  
*my dad, who taught me to love science,*  
and  
*my mum, who taught me to love language*

## TABLE OF CONTENTS

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Formal grammars	3
1.2	Tools	5
1.2.1	Formal Language Theory: The Chomsky Hierarchy	6
1.2.2	X-bar grammars	14
1.2.3	Bare phrase structure trees	19
1.2.4	Artificial Language Learning	20
<b>2</b>	<b>Adjuncts: the phenomenon</b>	<b>24</b>
2.1	Structural definitions of adjuncts	24
2.2	Semantic definition of adjuncts and arguments	28
2.3	Subcategorisation definition of adjunction	28
2.4	Syntactic diagnostics	29
2.5	Obligatory adjuncts	32
2.6	Adjuncts in phrase structure grammars	36
2.7	Transparency	41
2.8	Ordering restrictions	45
2.8.1	Adjective ordering	45
2.8.2	Adverb Order	65
2.8.3	Ernst 2002	69
2.9	Selectability	71
2.10	Adjuncts of adjuncts	72

2.11	Unordered adjuncts . . . . .	76
2.12	Iterability . . . . .	77
2.13	Conclusion . . . . .	81
<b>3</b>	<b>Minimalism . . . . .</b>	<b>83</b>
3.1	The Minimalist Program . . . . .	84
3.1.1	A note on trees and graphs . . . . .	86
3.1.2	Features . . . . .	92
3.2	MGs: Informal introduction to a formalisation . . . . .	94
3.3	MG Formalism . . . . .	100
3.3.1	String-generating MG . . . . .	104
3.3.2	Tree-generating grammar . . . . .	108
3.4	How MGs implement traditional minimalism . . . . .	111
3.4.1	Tree search implementation . . . . .	111
3.4.2	MG extensions . . . . .	114
3.4.3	Overt, covert, and copy movement . . . . .	119
3.4.4	Combine and Store . . . . .	122
3.4.5	Constraints . . . . .	124
<b>4</b>	<b>Minimalist Grammars with Adjunction . . . . .</b>	<b>127</b>
4.1	Introduction . . . . .	127
4.2	Minimalist Grammars . . . . .	129
4.3	Cartography . . . . .	132
4.4	Desiderata . . . . .	134



4.5	Previous Approaches to Adjunction . . . . .	136
4.5.1	Traditional MG solution . . . . .	137
4.5.2	Frey & Gärtner . . . . .	139
4.5.3	Selectional approaches . . . . .	141
4.5.4	Fowlie (2013) . . . . .	146
4.6	Proposal 1: Minimalist Grammars with Adjunction . . . . .	150
4.6.1	Example . . . . .	152
4.6.2	Definition: MGAs . . . . .	153
4.6.3	Properties . . . . .	155
4.7	Discussion and Extensions . . . . .	157
4.7.1	Pre- and post-head adjunction . . . . .	158
4.7.2	Obligatory Adjuncts . . . . .	159
4.8	Formal Properties . . . . .	163
4.9	Interim Summary . . . . .	167
4.10	What should move? . . . . .	169
4.11	Graf 2014 . . . . .	171
4.11.1	Adverbs and Functional Heads . . . . .	174
4.12	Proposal 2: Minimalist Grammars with Hierarchies . . . . .	176
4.12.1	A combined approach: Minimalist Grammars with Hierarchies (MGH) . . . . .	179
4.12.2	Discussion . . . . .	180
4.12.3	Formal Properties . . . . .	184
4.13	Conclusion . . . . .	185
<b>5</b>	<b>Learnability of adjuncts . . . . .</b>	<b>187</b>

5.1	Introduction . . . . .	187
5.2	Learnability . . . . .	192
5.3	Repetition and Optionality . . . . .	195
5.4	N-gram learners . . . . .	197
5.5	0-reversible learner . . . . .	202
5.6	Substitutable context free languages . . . . .	206
5.6.1	Learnability of substitutable CF languages . . . . .	207
5.6.2	Suitability of substitutable CF languages for human language . . . . .	212
5.7	Clark & Thollard . . . . .	213
5.8	Context free languages with finite kernel and context properties . . . . .	216
5.9	Conclusion . . . . .	232
5.9.1	A note on learners not analysed here . . . . .	234
<b>6</b>	<b>Artificial language learning experiment . . . . .</b>	<b>237</b>
6.1	Introduction . . . . .	237
6.2	Design . . . . .	241
6.2.1	Stimuli . . . . .	244
6.2.2	Testing stimuli . . . . .	246
6.2.3	Procedure . . . . .	248
6.2.4	Participants . . . . .	249
6.2.5	Analysis . . . . .	250
6.3	Results . . . . .	251
6.3.1	Ungrammatical subtypes . . . . .	253
6.3.2	Duplicate items . . . . .	257

6.4	Discussion . . . . .	259
6.4.1	Scrambled stimuli . . . . .	260
6.4.2	Surface effects . . . . .	261
6.5	English survey . . . . .	263
6.6	Limitations . . . . .	268
6.7	Conclusion . . . . .	269
<b>7</b>	<b>Conclusion . . . . .</b>	<b>272</b>
7.1	Adjuncts vs Arguments . . . . .	272
7.2	Learnability . . . . .	276
7.3	Final Thoughts . . . . .	277
<b>A</b>	<b>Mathematical definitions and symbols . . . . .</b>	<b>278</b>
<b>B</b>	<b>Adjective ordering . . . . .</b>	<b>282</b>
<b>C</b>	<b>Tagalog Experiment: Probabilistic Finite State Automata . . . . .</b>	<b>284</b>
<b>D</b>	<b>Tagalog Stimuli . . . . .</b>	<b>287</b>
D.1	Training stimuli . . . . .	287
D.2	Testing stimuli . . . . .	291
D.2.1	Grammatical . . . . .	291
D.2.2	Ungrammatical . . . . .	294
D.3	English stimuli . . . . .	295

## LIST OF FIGURES

1.1	Refined Chomsky Hierarchy of Languages . . . . .	7
1.2	Sample finite state machine generating <i>the (very* bright) star</i> $\cup$ <i>the (very* bright) mechanic</i> . . . . .	8
1.3	CFG generating the same language as the FSM in figure 1.2 . . . . .	12
1.4	NP specifier precedes adjuncts . . . . .	17
1.5	Building VP <i>she put the book on the table</i> with a minimalist grammar . . . . .	19
1.6	Bare phrase structure tree . . . . .	20
2.1	Adjuncts: sister and daughter of XP . . . . .	25
2.2	Bare phrase structure adjuncts are indistinguishable from arguments . . . . .	25
2.3	Which <i>as</i> are maximal projections? . . . . .	26
2.4	Which <i>as</i> are maximal projections? . . . . .	27
2.5	All NPs can be replaced with <i>one</i> . . . . .	31
2.6	According to the diagnostic, the PPs are adjuncts of <i>picture</i> , but arguments of <i>destruction</i> . . . . .	31
2.7	$a^n b^n$ . . . . .	37
2.8	Categorial Grammars . . . . .	40
2.9	T selecting a VP . . . . .	43
2.10	Constituency of ordinary adjunction of modified adjuncts . . . . .	77
3.1	Graphs can distinguish internal and external Merge . . . . .	88
3.2	Derivation tree for <i>The Firefly sailed</i> . . . . .	90
3.3	Annotated derivation tree . . . . .	91

3.4	Annotated derivation tree. Derived trees are in yellow boxes. . . . .	91
3.5	<i>sailed</i> takes <i>Serenity</i> as argument . . . . .	93
3.6	wh-movement . . . . .	94
3.7	Unannotated and annotated derivation trees for <i>She sang</i> . . . . .	96
3.8	Partial derivation tree for <i>who sang?</i> . . . . .	97
3.9	Partial derivation tree for <i>who sang?</i> . . . . .	98
3.10	Derivation tree for <i>who sang?</i> . . . . .	98
3.11	Features on a lexical item . . . . .	99
3.12	String-generating grammar derivation tree for VP <i>Kaylee fix Serenity</i> . . . . .	108
3.13	Bare phrase structure tree and equivalent bare tree . . . . .	109
3.14	Derivation tree, term over $\langle \Sigma, f \rangle$ , and derived bare tree of <i>the cowboy</i> . . . . .	110
3.15	Derivation tree, term over $T_{\langle \Sigma, f \rangle}$ and derived bare tree for <i>Who did Jayne shoot?</i> . . . . .	111
3.16	<i>x</i> is the head of <i>x</i> , <i>z</i> , and <i>y</i> . . . . .	112
3.17	Derivation in a tree-search MG . . . . .	113
3.18	Derivation tree with successive cyclic movement . . . . .	115
4.1	Merge . . . . .	131
4.2	Annotated and unannotated derivation trees . . . . .	132
4.3	Desiderata . . . . .	136
4.4	Traditional MG approach . . . . .	137
4.5	Derivation tree and derived <i>bare tree</i> . The $\langle$ points to the head, <i>big</i> . . . . .	138
4.6	Selected adjective, modifier adjective, adjoined-to adjective . . . . .	139
4.7	Frey & Gärtner: derivation tree and derived <i>bare tree</i> . The $\rangle$ points to the head, <i>wolf</i> . . . . .	140

4.8	F & G derivations of <i>the big bad wolf</i> and <i>*the bad big wolf</i> . . . . .	140
4.9	Selectional approach . . . . .	142
4.10	Valid derivation of <i>bad wolf</i> . . . . .	147
4.11	Valid derivation of <i>the big bad wolf</i> and attempted derivation of <i>*the bad big wolf</i>	147
4.12	Adjoin. The category feature of the new phrase is the first two elements of the adjoined-to phrase followed by the second element of the adjunct . . . . .	152
4.13	Adjunct of adjunct; functional head merge; adjunction after functional head merge . . . . .	153
4.14	Adjunct ordering: valid and invalid derivations . . . . .	154
4.15	<i>Who always slept?:</i> required functional categories . . . . .	156
4.16	Unordered English PPs . . . . .	159
4.17	Obligatory adjunct extention . . . . .	160
4.18	Determiners of modified NPs could have their own category DM . . . . .	161
4.19	Derivation and derived trees for vP <i>makes a good father</i> . . . . .	162
4.20	Non-canonical adjective order . . . . .	169
4.21	Adjonee can move independently using $Adjoin_3$ . . . . .	170
4.22	Slices by colour . . . . .	172
4.23	To determine if the root node is a valid application of Merge, we need to look 3 nodes away, to =A=BC. . . . .	172
4.24	The validity of this derivation tree cannot be determined. . . . .	173
4.25	Adjoin interrupts the locality of Merge . . . . .	174
4.26	Derivation of ungrammatical sentence due to Merge resetting the hierarchy level . . . . .	177
4.27	Derivation tree of <i>the three kings sang</i> using HopMerge . . . . .	178

4.28	The three old kings derived by an MGH . . . . .	180
4.29	Optionality of internal functional head in French DPs . . . . .	181
4.30	Complete Cinque hierarchy . . . . .	183
5.1	Repetition is optionality . . . . .	190
5.2	Learning . . . . .	192
5.3	Finite language learner . . . . .	193
5.4	A run of the 0-reversible learner . . . . .	204
5.5	Derivations of $a, aa$ in the target grammar . . . . .	210
5.6	Ambiguity arises in the hypothesis grammar . . . . .	210
5.7	$L = ab^*c$ . . . . .	213
5.8	Grammar . . . . .	218
5.9	Grammar . . . . .	219
5.10	Two parses of $abc$ . . . . .	219
5.11	Algorithm for learning $CFG_{E,K}$ . . . . .	222
5.12	$G_{ab^*c}$ . . . . .	223
5.13	$G_1$ . . . . .	224
5.14	Human-readable $G_1$ . . . . .	225
5.15	$abc$ is structurally ambiguous . . . . .	225
5.16	$b$ is optional and repeatable in $G_1$ . . . . .	226
5.17	Generating $a^+b$ . . . . .	235
6.1	Grammatical sentence templates . . . . .	242
6.2	Design of Stimuli . . . . .	243
6.3	Results by category and group . . . . .	253

6.4	Results by category and group . . . . .	257
6.5	Testing stimuli repeated from the training phase . . . . .	259
6.6	Correlation between amount of repetition of adjectives and grammaticality judgement. The row for <i>definitely not English</i> is empty . . . . .	266
6.7	Correlation between amount of repetition of intensifiers and grammaticality judgement . . . . .	267
7.1	BBC tweet . . . . .	275
C.1	Finite State Automata that generated the training stimuli . . . . .	285
C.2	Finite State Automata that generated the testing stimuli . . . . .	286
D.1	Illustration by Martín Villalba . . . . .	296



## LIST OF TABLES

2.1	Sample lexicon for 1980s-era grammar . . . . .	38
2.2	Adding adjuncts into the grammar . . . . .	39
2.3	Adding adjuncts into the lexicon . . . . .	39
2.4	Six languages' adjective order in <a href="#">Hetzron (1978)</a> . . . . .	63
2.5	Three languages' pre- and post-nominal adjective orders in <a href="#">Hetzron (1978)</a> . . . . .	63
3.1	Features . . . . .	95
3.2	Features . . . . .	102
3.3	Combine and Store possibilities . . . . .	122
4.1	Summary of models re: desiderata . . . . .	141
4.2	Summary of models re: desiderata . . . . .	146
4.3	Summary of models re: desiderata . . . . .	150
4.4	Summary of models re: desiderata . . . . .	168
4.5	Summary of models re: desiderata . . . . .	186
5.1	Grammar 1 . . . . .	199
5.2	Grammar 2 . . . . .	199
5.3	Substrings and contexts in abc . . . . .	216
5.4	Old and new category names . . . . .	218
5.5	Summary of learners . . . . .	232
6.1	Vocabulary . . . . .	245
6.2	Training stimuli . . . . .	245

6.3	Testing stimuli . . . . .	247
6.4	English survey stimuli . . . . .	264
B.1	Approximate alignment of adjective ordering claims from different sources .	283

## ACKNOWLEDGMENTS

This project was only possible because of the patience, kindness, and generosity of a great many people. My committee has been endlessly patient and consistently helpful for these many years. Ed Keenan is forever inspiring me with the joy and simplicity he brings to mathematical linguistics, and entertaining me with hilarious stories about wonderful people and places. Martin Monti enthusiastically accepted the role of connection between linguistics and psychology of language, and provided crucial help with the experimental portion of my dissertation. Carson Schutze has devoted many hours to detailed commentary, greatly improving the quality of my writing and connection to mainstream minimalism. And finally, Ed Stabler, my chair, is truly extraordinary in his inspiration, ruthless clarity, and kindness. I am very lucky to have had the opportunity to work with a scholar and a human like Ed.

Beyond my committee, I find help everywhere I go. I was educated very well by the undergraduate program at McGill. In particular, my advisor there, Lisa deMena Travis, is truly a perfect advisor: smart, generous, demanding, and unfailingly kind. I spent the 2015-16 academic year back at McGill working with Lisa again, where I learned a great deal. I am now at the University of Saarland, where Alexander Koller has not only been a fantastic teacher to me, but has patiently allowed me to finish my dissertation while working on my new research.

UCLA must be the best place in the world to study linguistics. I have been very happy in the department, which is full of bright, lively people who create a close and vibrant community. My cohort deserves special mention, as they inspired and helped me all throughout my time there, and are my friends to this day: Laura Kalin, Laura McPherson, Kathleen O'Flynn, Michael Tseng, Floris van Vugt, Kaeli Ward, and David Wemhaner. I thank fellow students and friends for many useful and inspiring conversations and a great deal of fun: Natasha Korotkova, Michael Lefkowitz, Yu Tanaka, Jesse Zymet, Mal-

colm Roberts, Remkes Kooistra, Christopher Anderson, Aislin Jackson, and surely many others I am forgetting. I also thank the UCLA Syntax-Semantics Seminar and MathLing Circle for helpful feedback on research. The Tagalog experiment was made possible by Seth Ronquillo's recordings of sentences, and Shaya Lurya's dedicated (and voluntary!) research assistanceship.

The great green dragon illustration is by Martín Villalba, an artist and fellow computational linguist.

I haven't room here to do justice to the thanks I owe to Floris van Vugt, who has provided constant support, has taught me a great deal both personally and scientifically, and was instrumental in the experimental portion of this work, writing the code that ran the actual experiment and providing detailed guidance in the statistical analysis. Without him, I suspect I would not have graduated before reaching retirement age.

Finally, there is my family, who have always loved and supported me and have never given me grief for my failings. Few people are so lucky in their parentage, to be raised by such warm and intelligent people, and to be supported whenever it was needed, even in adulthood.

Thank you, everyone, from the very bottom of my foolish enthusiastic heart.

## VITA

- 2016-2017      Researcher and Instructor, Department of Computational Linguistics and Phonetics, Saarland University
- 2016            Researcher, Department of Linguistics, McGill University
- 2015            Instructor, Department of Linguistics, McGill University
- 2010–2015      Teaching Assistant, Department of Linguistics, UCLA.
- 2007            Bachelor of Arts, First Class Honours in Linguistics, McGill University.
- 1997-1998,      Physics, Mathematics, and History, University of Alberta.  
1999-2001,  
2005-2006

## PUBLICATIONS

Fowlie, M. and Koller, A. (2017). Parsing Minimalist Languages with Interpreted Regular Tree Grammars. In Proceedings of the 13th International Workshop on Tree Adjoining Grammars and Related Formalisms (TAG+13) (pp. 1120). Umeå, Sweden.

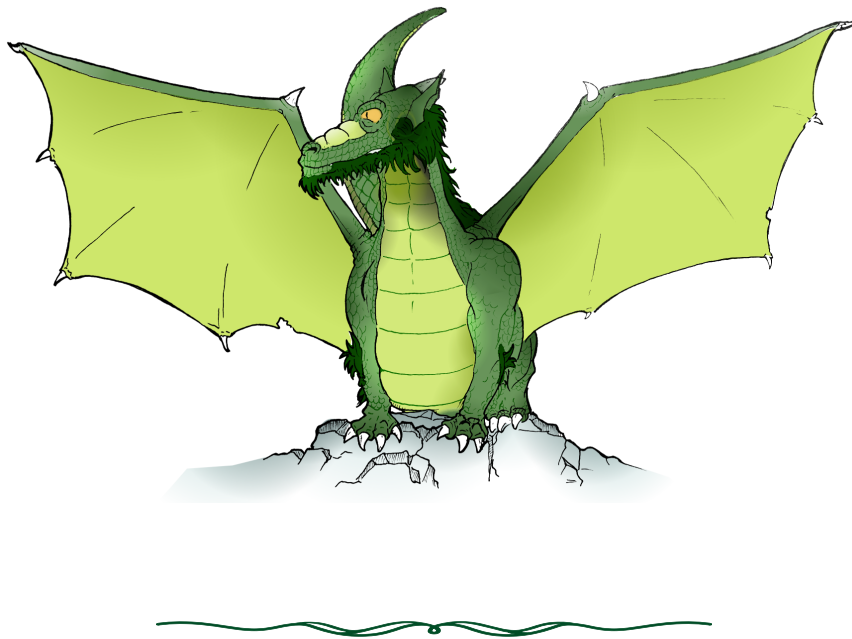
Fowlie, M. (2014). Adjuncts and minimalist grammars. In International Conference on Formal Grammar (pp. 34-51). Springer, Berlin, Heidelberg.

Fowlie, M. (2013). Order and optionality: Minimalist grammars with adjunction. In The 13th Meeting on the Mathematics of Language (Vol. 12).

Fowlie, M. (2013). Multiple multiple spellout. *Challenges to Linearization*, 114, 129.


# CHAPTER 1

## Introduction



*I first tried to write a story when I was about seven. It was about a dragon. I remember nothing about it except a philological fact. My mother said nothing about the dragon, but pointed out that one could not say “a **green great** dragon”, but had to say “a **great green** dragon”. I wondered why, and still do. The fact that I remember this is possibly significant, as I do not think I ever tried to write a story again for many years, and was taken up with language.*


– J.R.R. Tolkien in a letter to W.H Auden **Carpenter** (1981)



*It's been a long long long time  
How could I ever have lost you  
When I loved you*

*It took a long long long time  
Now I'm so happy I found you  
How I love you*

– George Harrison, *Long long long*, The White Album



It has long been observed that modifiers behave differently from other syntactic elements. Classically, adjuncts are optional (*a dragon* and *a green dragon* are both grammatical) and iterable (*a great green dragon*; *a long long long time*); they have also been observed to be ordered (*\*a green great dragon* sounds wrong). In this dissertation I consider the properties of adjuncts and how to model their behaviour in the syntax (chapters 2 and 4), and how two of their basic properties – their optionality and iterability – may be learned (chapters 5 and 6).

The three primary findings are as follows: first, Chapters 2 and 4 observe the differences between the behaviour of adjuncts and arguments, and argue that syntactic models should indeed treat them distinctly, contra modern cartographic models such as Cinque (1999). In particular, Chapter 4 proposes a minimalist syntax that captures both the traditionally observed differences between adjuncts and arguments and the ordering observations in the literature (e.g. Vendler, 1968; Cinque, 1999, 2010). Second, Chapter 5 shows that many formal learning algorithms assume a relationship between iterability and optionality, and considers the merits and faults of learners that draw such a conclusion.



Third, the artificial language learning experiment in Chapter 6 shows that in learning a new language, people use exposure to limited amounts of repetition to generalise to indefinite repetition.

## 1.1 Formal grammars

At the heart of linguistic inquiry is this: What is the nature of language and what is the nature of the mind/brain such that language is learnable? Language is so complex that not a single language has been fully described despite the best efforts of a great many researchers, yet babies learn language nearly perfectly.

Would it be possible for the baby to learn its native language if language could be literally anything? If it wasn't even expecting to encounter language? Does a baby completely re-invent not only her native language but the very idea of language? Does the baby have to re-invent her native language completely, or does she expect some things to be true of it? How restricted is the space of possible human languages? If a baby's only linguistic exposure was to a made-up "language" that shared very few of its properties with any known real language and yet was used in such a way that it drew her attention, would she learn it?

The brain is a natural object and is subject to the laws of nature, and the laws of nature can be described mathematically, so far at least. We should expect to be able to describe language in a mathematically precise way, and ultimately to describe a baby's brain in a mathematically precise way. The descriptions, if accurate, should explain how a baby learns language, and should answer the questions posed above.

Linguists have observed patterns in human languages and worked to describe those patterns precisely. They have also worked to characterise the space of possible human languages. All of these characterisations are, aim to be, or can be re-written as, mathematical objects.

Similarly, the study of real children acquiring their first languages, and adults learning new languages, aims to describe the process(es) through which humans learn language. Ultimately, these processes are algorithms:

Linguists have long been interested in the mathematical properties of human language. Indeed, some of Chomsky's earliest work established a hierarchy of syntax in general, and sought the right place for human language within it (Chomsky, 1957). Since then progress has been made in describing human language in general mathematical terms (Huybregts, 1984; Schieber, 1985; Joshi, 1985; Ades and Steedman, 1982; Steedman, 2001; Joshi et al., 1991; Weir, 1988; Stabler, 1997; Heinz, 2010). Much remains to be done, however, as much of what we have learned still picks out far too large a class of languages, of which human language is but a small subset.

There is currently a significant gulf between formal approaches to language (Formal Language Theory, which is more a branch of computer science or math) and linguistics. Both sides have gaps that the other does not. Formal approaches are very precise, which is a quality often lacking in mainstream linguistics. Linguists, on the other hand, have made a great many observations and generalisations that formal language theory is barely aware of, much less ready to incorporate into its models. It is in this gap that this dissertation exists. I formalise observations of syntacticians and explore the mathematical properties of these formalisations. In particular, I offer a model of adjunction that formalises the work of linguists such as Cinque and Rizzi regarding ordered adjuncts and functional heads, and explore the mathematical properties of the formal model. I also present a formal model of minimalism that explicitly models Move as Internal Merge, as envisioned in Chomsky and Collins (2001) (section 3.4.3 of chapter 3).

Ultimately, my aim is to bring traditional and mathematical syntax just a little bit closer together by creating an explicit minimalist model of adjuncts that accounts for the myriad empirical generalisations discovered by syntacticians, exploring learning algorithms' treatment of specific empirical phenomena, and testing real human learners'

behaviour of the same phenomena.

## 1.2 Tools

The formal tools I make use of in this thesis are primarily formal learnability – the study of mathematical models of language learning –, and Minimalist Grammars – formal models of contemporary syntax. These tools are set in the milieu of Formal Language Theory, the study of the complexity of sets of sequences, called *languages*, of which human language is but one part.

For those unfamiliar with any of the mathematical notation and definitions I make use of, I provide a brief overview in appendix A.

Learnability is the study of algorithms that generate grammars – or subparts of grammars, such as rule probabilities – using a sample of the language. Chapter 5 looks at several learning algorithms and considers how they treat two aspects of adjunction – optionality and repetition.

Minimalist Grammars are Stabler’s (1997 etc) formalisation of Chomsky’s (1993, 1995 etc) minimalist program feature grammars. Chapter 3 introduces the grammars and chapter 4 offers expansions that incorporate adjunction. I choose this model for several reasons. Mathematically, MGs are arguably the best characterisation we have so far, in terms of the languages they generate, of human language (Jäger and Rogers, 2012).<sup>1</sup> Also of importance is that MGs are designed to be formalisations of the kind of feature-driven syntax that most of current mainstream syntax is working with, and so it provides a good bridge between the abstract mathematical questions and the actual work of syntacticians. Finally, MGs are intuitive and simple to work with. Although there are other formalisms that can generate the same languages (such as Multiple Context-Free Gram-

---

<sup>1</sup>Human language seems to involve copying, so a slight extension of the basic MG formulation is required to include copying as a form of movement.

mas), these are not nearly so intuitive, and they fail to capture basic generalisations that syntacticians have identified (Stabler, 2013).

My experimental tool is Artificial Language Learning (ALL). In ALL paradigms, the idea is simply to make up a language and teach it to people, and see what they do with it: whether they learn the language, how they generalise, how their brains look when they're processing the artificial language, etc.

My experiment explores learning of adjuncts, specifically how people generalise repetition: trained to recognise sentences of a language with limited repetition, do people also accept sentences with more repetition?

### 1.2.1 Formal Language Theory: The Chomsky Hierarchy

Chomsky (1957) defines a hierarchy of languages, which this dissertation makes frequent reference to, differentiated by the kind of grammar needed to describe, or generate, it. It is a hierarchy because a language lower in the hierarchy can always be described by a mechanism from higher in the hierarchy, but not vice-versa. For two classes  $C, D$  where  $C \subset D$ , we say a language  $L$  is *properly*  $D$  just in case  $L \in D - C$ .

**Notation: Kleene star** Let  $a$  be a symbol and  $n \in \mathbb{N}$ .  $a^n$  is a sequence of  $n$   $a$ 's.  $a^*$  (pronounced *a-star*) is the set of sequences made up of 0 or more  $a$ 's. Let  $\Sigma$  be a set of symbols.  $\Sigma^*$  (pronounced *sigma-star*) is the set of all sequences made up of 0 or more symbols from  $\Sigma$ .

**Definition 1.2.1** (Alphabet, Language, Sentence). An *alphabet* or *lexicon*  $\Sigma$  is a finite set of symbols. A *language* over an alphabet/lexicon is a set of sequences of symbols from the alphabet; i.e.  $L \subseteq \Sigma^*$  is a language over  $\Sigma$ . Members of a language can be called *words* or *sentences*, on analogy with phonology ( $\Sigma$  is a set of sounds) and syntax ( $\Sigma$  is a set of words or morphemes). Throughout **I will call the members of a language sentences** and the members of the alphabet/lexicon *lexical items*, or sometimes *words* or *morphemes*.

The original Chomsky hierarchy had four levels Chomsky (1959): regular, context free, context sensitive, and recursively enumerable. I will briefly describe here a refinement of this original hierarchy.

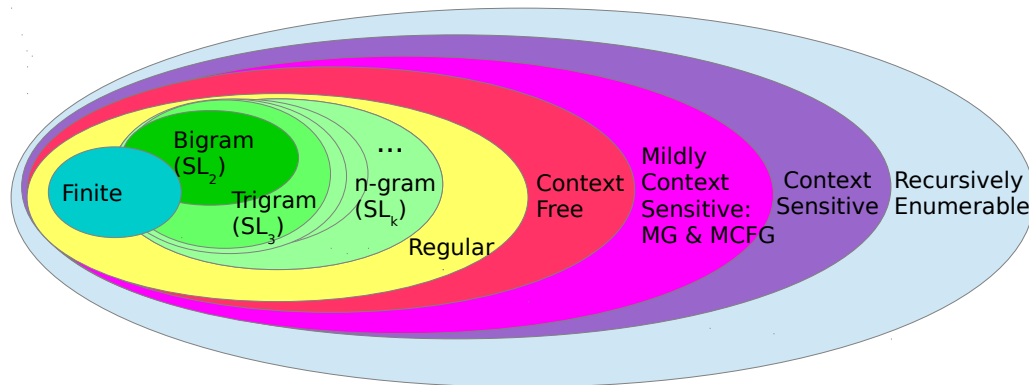


Figure 1.1: Refined Chomsky Hierarchy of Languages

At the bottom of the Chomsky hierarchy are finite languages, which are languages with a finite number of sentences. Finite languages can be defined by a grammar that simply lists all the sentences.

### 1.2.1.1 Strictly local languages

Bigram grammars, which can be described entirely by a set of legal pairs of words that can be next to each other, are higher up than Finite. Probabilities can be added to these bigrams, yielding grammars based entirely on Transitional Probabilities. Above bigram grammars is an infinite hierarchy of n-gram grammars, defined by legal strings of three words (trigrams), four words (4-grams), and so on. These languages are also called Strictly Local languages (SL), or specifically, Strictly  $k$ -local languages (SK <sub>$k$</sub> ) for some  $k > 0$ . For example, bigram languages are SL<sub>2</sub>.

### 1.2.1.2 Regular languages

Next up are the regular, or finite state, languages, which can be described by a finite state machine. A finite state machine can remember only a finite amount of information in determining what word can come next. Finite languages can be described with a finite state machine, but finite-state languages with an infinite number of sentences cannot be described just by listing.

A finite state machine, intuitively, is a set of nodes with paths between them. The nodes are the states that you can be in, and the paths are associated with lexical items. You can build a sentence by starting at any initial state, and hopping from state to state, adding the symbol from the path you took for each hop. You can stop when you get to a final state, indicated with a double circle. In the FSM in Figure 1.2, 0 is the start state and 3 is the final state. The grammar generates  $the (very^* bright) star \cup the (very^* bright) mechanic$ , which is *the* followed by *star* or *mechanic* or by 0 or more *verys*, followed by *bright*, followed by either *star* or *mechanic*. (See Definition 1.2.2 below for a formal definition.)

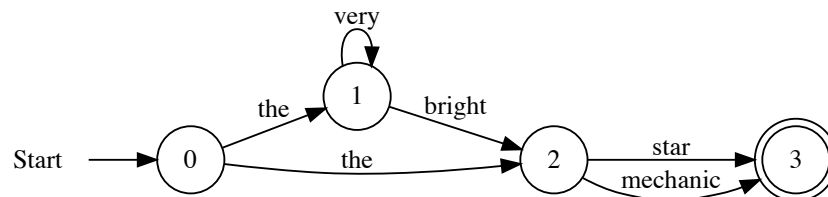


Figure 1.2: Sample finite state machine generating  $the (very^* bright) star \cup the (very^* bright) mechanic$

The language generated by this machine is infinite since there is no bound on the number of *verys*. The “finite” in *finite state* refers to the number of states (the circles with the numbers in them). Formally:

**Definition 1.2.2** (Finite State Automaton). A finite state automaton (DFSA) is a five-tuple

$$\langle \Sigma, Q, q_0, F, \delta \rangle$$

where:

$\Sigma$  is an alphabet

$Q$  is a finite set (*states*)

$q_0 \in Q$  is the designated *start state*

$F \subseteq Q$  is the set of *final states*

$\delta : \mathcal{P}(Q) \times \Sigma \rightarrow \mathcal{P}(Q)$  is the *transition function*

### 1.2.1.3 Context Free languages

Higher than finite-state languages we find context-free languages, which can be described with a particular sort of phrase structure, or re-write, grammar. The particular sort is one that can only have one symbol on the left hand side (LHS) of a rule. It's called *context-free* because limiting the LHS to one symbol means you don't have to refer to its context (what surrounds it) to determine if the rule applies.

**Definition 1.2.3** (Re-write rule). Let  $N$  and  $\Sigma$  be alphabets, and call  $N$  the *non-terminal* symbols, and  $\Sigma$  the *terminal* symbols. A re-write rule is a rule of the form  $s \rightarrow t$  where  $s, t \in (N \cup \Sigma)^*$ ; i.e.  $s$  and  $t$  are sequences of terminals and non-terminals.

**Definition 1.2.4.** A *context free grammar* is a 4-tuple  $\langle N, \Sigma, P, S \rangle$  such that

$N$  is a set of non-terminal symbols

$\Sigma$  is a set of terminal symbols

$P$  is a set of re-write rules of the form  $n \rightarrow s$  where  $n \in N$  and  $s \in (N \cup \Sigma)^*$

$S \in N$  is the designated *start symbol*

For example,  $G$  is a context-free grammar. I've numbered the rules so I can refer to them later.

- (1)  $G =$   
 $\langle N = \{S\},$   
 $\Sigma = \{a, b\},$   
 $P = \{$   
    1.  $S \rightarrow \epsilon$   
    2.  $S \rightarrow a S b$   
 $\},$   
 $S \rangle$

A context free grammar generates a language by starting with the start symbol and at each step re-writing a non-terminal using one of the rules in  $P$ . That re-writing step is notated with a double arrow  $\Rightarrow$ . A sample derivation using  $G$  above to generate  $aaabbb$  is given in (2). The subsequences in bold were added in the last step. I notated each arrow with the number of the rule that applied.<sup>2</sup>

$$(2) \quad S \Rightarrow_2 \mathbf{a}Sb \Rightarrow_2 \mathbf{aa}Sbb \Rightarrow_2 \mathbf{aaa}Sbbb \Rightarrow_1 \mathbf{aaa\epsilon}bbb = \mathbf{aaabbb}$$

If we want to say that a sequence  $t$  was generated in some finite number of steps from another sequence  $s$  we can write  $s \Rightarrow^* t$ ; for example,  $S \Rightarrow^* aaabbb$  summarises (2).

**Definition 1.2.5** (Context-free language). A context free grammar  $G = \langle N, \Sigma, P, S \rangle$  generates a context free language  $L(G)$  if, for all  $s$  in  $L(G)$ ,  $s$  consists only of terminal symbols ( $s \in \Sigma^*$ ) and there is a derivation  $S \Rightarrow^* s$ . A language is context free just in case it can be generated by a context-free grammar.

---

<sup>2</sup> $\epsilon$  is the symbol for an empty sequence.



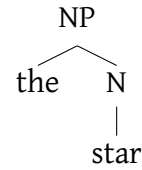
Famously, the grammar in (1) generates  $\{a^n b^n | n \in \mathbb{N}\}$  (the language consisting of sentences with some number of *as* followed by the same number of *bs*), usually just written  $a^n b^n$ , which provably cannot be generated by a regular grammar. A context-free grammar, or higher, is required. Because of centre-embedding in languages like Japanese, human language is thought to be at least context free. Indeed, ultimately it is thought to be Mildly Context-Sensitive, which is yet higher in the hierarchy (Joshi, 1985). This is due to the existence of crossing dependencies in languages such as Dutch and Swiss German.<sup>3</sup>

Since a finite-state language can also be described by mechanisms higher up the hierarchy, such as a phrase-structure grammar, sometimes we find that a language, while finite-state, is more efficiently described by a PSG. We saw in 1.2 above a finite state grammar that can generate *the (very\* bright) star*  $\cup$  *the (very\* bright) mechanic*. The same language can be represented by the following Phrase Structure Grammar:

---

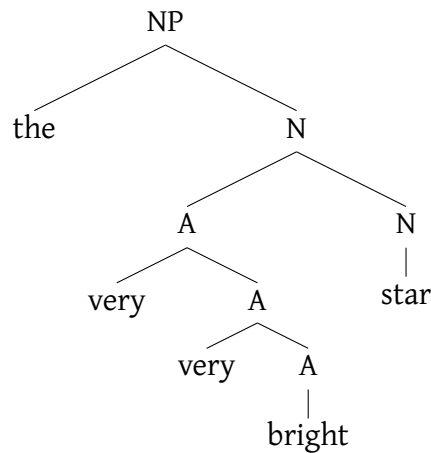
<sup>3</sup>However, as we will see, the mechanisms available at the Mildly Context Sensitive level (such as Tree Adjoining Grammars (Joshi, 1985) and Minimalist Grammars (Stabler, 1997)) more concisely and intuitively describe human language, regardless of the issue of crossing dependencies.

1.  $NP \rightarrow \text{the } N$
2.  $N \rightarrow A N$
3.  $N \rightarrow \text{star} \mid \text{mechanic}$
4.  $A \rightarrow \text{very } A$
5.  $A \rightarrow \text{bright}$



(b) Derivation of *the star*

(a) PSG



(c) Derivation of *the very very bright star*

Figure 1.3: CFG generating the same language as the FSM in figure 1.2

These two grammars illustrate two important things. First, just because something can be represented by a context free PSG doesn't mean it is properly Context Free. In this case, the language is regular, but the grammars are regular (figure 1.2) and context free (figure 1.3) respectively.

Second, just because a language is regular doesn't mean a regular grammar is the "best" representation of it. For example, often a PSG is more succinct than a FSM. In this case, the FSM is perhaps very slightly more succinct than the PSG (4 states and 6 transitions vs. 6 rules), but it does not take much for a PSG to become far more succinct. Relatedly, FSMs often miss generalisations that PSGs capture. In this case, the FSM has

two arcs labelled *the*. It misses the generalisation that the two *thes* are really the same, it's just that the modifiers are optional.

#### 1.2.1.4 Mildly context-sensitive languages

Higher still are Mildly Context Sensitive languages, which include human languages and can be described with, among other things, a Minimalist Grammar (see Chapter 3) which includes Merge and Move (Joshi, 1985) (Stabler, 1997). MCFLs can also be described with a *multiple context-free grammar*, which is a type of re-write grammar. For a formal definition of multiple context free grammars, see definition 4.8.3 in chapter 4. The intuition is that while a context-free grammar re-writes its non-terminals in exactly the place where it found them (eg. in the step  $aSb \Rightarrow aaSbb$ ,  $S$  is between  $a$  and  $b$  and what it gets rewritten as –  $aSb$  – also goes between  $a$  and  $b$ ), a multiple context free grammar can keep parts of the sequence it is generating temporarily separate, so that rules might change their order. For example, you could have a rule as in (3-a) in which strings  $x$  and  $y$  of categories  $A$  and  $B$  can form a pair  $(x, y)$  of category  $S$ . Another rule like (3-b) could put  $x$  and  $y$  together in the reverse order. Rules like these can be used to model movement.

- (3) a.  $S(x, y) \leftarrow A(x)B(y)$   
b.  $S(yx) \leftarrow S(x, y)$

#### 1.2.1.5 Recursively Enumerable

At the very top of the Chomsky hierarchy are the recursively enumerable languages. These are all languages that can be effectively enumerated with any kind of computer with infinite memory. If an algorithm can be described to generate the language, then it is RE. If we aim for a restrictive model of human language, it should not be this powerful. Peters and Ritchie (1973) showed that the system of context-free grammars in combination with *transformations* (rules for changing the trees) of Chomsky (1957) is recursively

enumerable, leading to attempts to come up with a more restrictive theory. One of the fruits of these labours is Minimalism (see Chapter 3), which is the approach this dissertation takes. Minimalist Grammars without the shortest move constraint – see Section 3.3 – are also recursively enumerable (Kobele and Michaelis, 2009).

### 1.2.2 X-bar grammars

Although the syntactic world has largely moved on from X-bar grammars as a formal model, many of us still think at least half in terms of X-bar grammars. As such, I will sometimes use the vocabulary and even trees of X-bar grammars when it is convenient.

My main formalism, grammar-wise, is Minimalist Grammars. MGs are a framework which can be used to generate any number of structures, including Chomsky (1995)-style bare phrase structure trees, sets, sequences, and X-bar trees. There is therefore no contradiction here, and thinking in terms of an X-bar schema is sometimes more intuitive than other ways of thinking.

Throughout, I choose largely for simplicity a specific X-bar schema – that used in an introductory textbook (Fromkin et al., 2000) written at UCLA. I’ll refer to it as the Fromkin schema or grammar.

X-bar grammars are phrase-structure grammars that grew out of Harris (1951) and Chomsky (1970). What they have in common is a grammatical rule constraint that all or most rules must be of the following form, where X is a category and  $\forall i < m$ ,  $C_i$  is a phrase  $Y^j$  for some  $j \in \mathbb{N}$ , some nonterminal  $Y$ :

$$(4) \quad X^n \rightarrow C_0 \dots C_k X^{n-1} C_{k+1} \dots C_m$$

The superscripts are also called bar-levels and may be written as primes or bars over X. X-bar grammars vary along such dimensions as the maximal number of bar levels and whether rules can also have  $X^n$  on the right hand side rather than  $X^{n-1}$ . Modern X-bar

grammars generally settle (Carnie, 2010) on something like the following:

- (5) Arguments
- a.  $XP \rightarrow YP X' \mid X' YP$  Specifier rules
- b.  $X' \rightarrow X YP \mid YP X$  Complement rules
- (6) Possible phrase-level adjunction
- a.  $XP \rightarrow XP YP$
- b.  $XP \rightarrow YP XP$
- (7) Possible bar-level adjunction
- a.  $X' \rightarrow X' YP$
- b.  $X' \rightarrow YP X'$

The exact structure of an X-bar grammar is not very important for this discussion. The crucial thing is that they not be exactly as in Jackendoff (1977), where every new merge results in a higher bar level. Rather I will stick to X-bar grammars in which adjuncts are added with recursive rules like those in (6) or (7), in keeping with what Carnie calls “the consensus in the more recent literature” (Carnie, 2010, pp. 131).

(8) gives the schema I will use throughout when I make use of X-bar grammars. In this schema, bar levels are only for arguments – specifically for making room for a head to have more than one argument – and adjunction is at the phrase level. Note that not all grammars in this style would allow for rules of the form in (8-e), since the additional argument is not obviously a complement or specifier, and it is often argued that a phrase has maximum one of each.

- (8) a.  $XP \rightarrow X$  *no arguments*
- b.  $XP \rightarrow X YP \mid YP X$  *complement only*

- c.  $XP \rightarrow X' YP \mid YP X'$  *specifier*
- d.  $X' \rightarrow X YP \mid YP X$  *complement*
- e.  $(X' \rightarrow X' YP \mid YP X')$  *(additional arguments)*
- f.  $XP \rightarrow XP YP \mid YP XP$  *adjunction*

Another common grammar schema adjoins at the bar level. For example, another popular introductory textbook, [Carnie \(2013\)](#), uses grammars like the following:

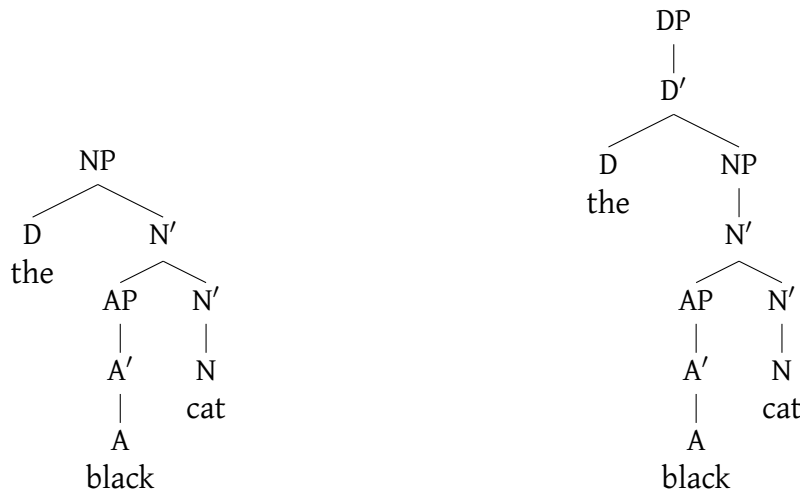
- (9) a.  $XP \rightarrow X' (YP) \mid (YP) X'$  *specifier rules*
- b.  $X' \rightarrow X (YP) \mid (YP) X$  *complement rules*
- c.  $X' \rightarrow X' YP \mid YP X'$  *adjunction*
- d.  $NP \rightarrow D N'$  *special NP specifier rule*

I choose the former because it correlates better with the minimalist grammar I propose in chapter 4. In that chapter, I propose a grammar for adjunction in which all of the requirements of the head are discharged, after which the phrase is complete, and its category visible to the derivation. Once the category is visible, it can be targeted for adjunction with another complete phrase, based on their categories. For instance, since adjectives are modifiers of nouns, once we have a complete NP built with a visible N category, and a complete AP with a visible A category, Adjoin can apply. If instead Adjoin happened before the specifier merged, the derivation would have to know when there was only a specifier remaining to be merged, pause and do any necessary adjunction, and then continue merging the specifier. This is possible, but it is simpler to always treat the requirements of the head as a stack, with only the top requirement relevant to the current operation.

Similarly, in the X-bar schema in (8), adjuncts adjoin to a complete phrase that has all of its complements and specifiers.

Moreover, I would argue that this schema is superior to the one in (9). The reason

adjunction is thought to be at the bar level in such grammars is primarily the need to put the determiner before the adjectives in a noun phrase (figure 1.4). However, this need disappears under the DP hypothesis, which also erases the special NP-specifier rule in (9-d). Using either the Carnie grammar (Fig. 1.4b) or the Fromkin (Fig. 1.4d), we no longer need adjuncts to be at the bar level.



(a) Special NP-specifier rule in Carnie grammar      (b) DP hypothesis in Carnie grammar  
(c) Special NP-specifier rule added to Fromkin grammar yields wrong word order      (d) DP hypothesis in Fromkin grammar

Figure 1.4: NP specifier precedes adjuncts

The Fromkin schema also does away with the need for rules to target bar-level phrases; specifically *one*-replacement and *do-so*-replacement now target NP and VP respectively. In this way, only complete phrases (XPs) are targeted by deletion, movement, or replacement.

Intuitively, the Fromkin x-bar schema works as follows. We have a lexicon with theta grids (lists of arguments for each word).

- X is the head of the phrase. If it has no arguments, use  $XP \rightarrow X$  and our phrase is complete.
- Otherwise, we merge the argument YP with the head and now our phrase is complete:  $XP \rightarrow X YP \mid YP X$ .
- What if we have more than one argument? No problem, we introduce an intermediate level,  $X'$ , that means “I’m building an XP but I’m not quite finished”:  $X' \rightarrow X YP \mid YP X$ ,  $X' \rightarrow YP X' \mid X' YP$ , and the phrase-level rules  $XP \rightarrow X' YP \mid YP X'$

This is exactly the intuition in a Minimalist Grammar.<sup>4</sup> A lexical item comes with a stack of features which specify the complements and specifiers. Only when those features have been checked – by merging or moving specifiers into the phrase – is the category feature on top of the feature stack. This allows the derivation to treat the phrase as a unit – to merge or move it in its entirety.

More details on MGs are given in chapter 3, but briefly: The complete feature stack of a lexical item in an MG looks like the schema in (10-a); for example, the verb *put* might have a feature stack like in (10-b).

- (10) a. {complements and specifiers, if any} Category {features that make this phrase move, if any}
- b. *put*::=D =P =D V

---

<sup>4</sup>Note that MGs do not in general restrict the number or arguments a head may select, while some X-bar grammars do. For instance, the Carnie grammar and the Fromkin grammar in (8) without rules like (8-e) allow up to two arguments. Adding the (8-e) rule schema makes the grammar more like an MG in this respect. MGs can of course be given constraints on the lexicon that limit the number of elements a head may require or move.



When the DP complement is Merged with *put*, the =D feature is deleted, revealing =P underneath. Then the PP is Merged, revealing the =D, triggering the merger of the subject DP. Only then is V revealed, and only then is the VP complete. Figure 1.5 illustrates the derivation, not including the building of the arguments *on the table* and *the book*.

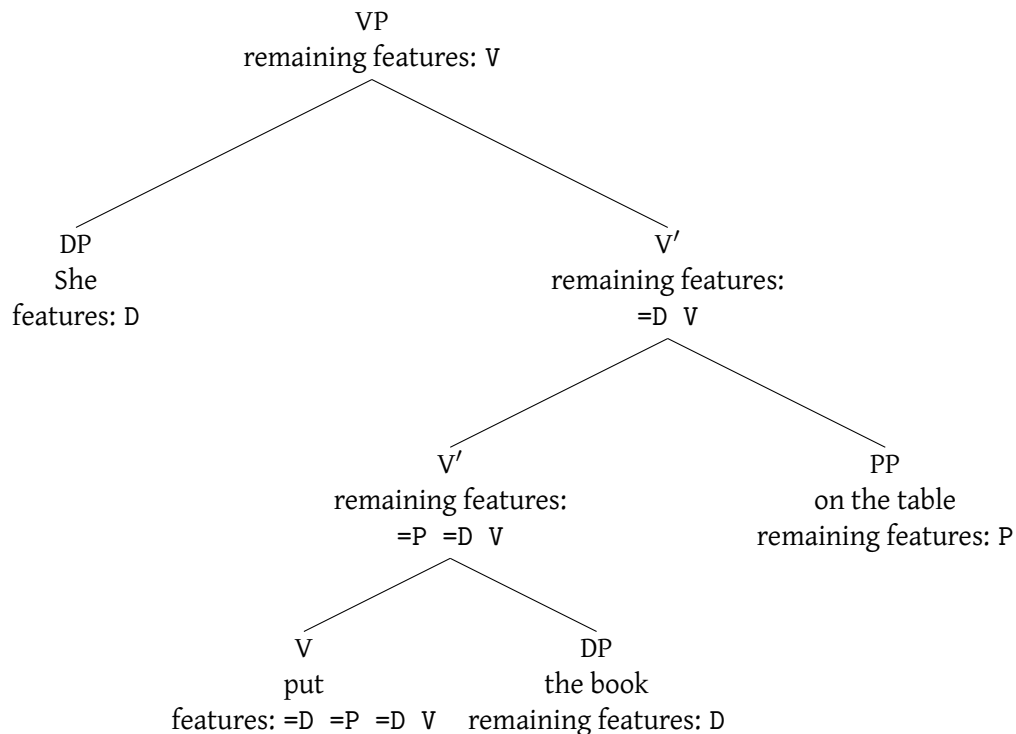


Figure 1.5: Building VP *she put the book on the table* with a minimalist grammar

### 1.2.3 Bare phrase structure trees

Chomsky (1995) introduces *bare phrase structure trees*, which are meant to be the simplest possible representation of what Merge could create. The two merged items are sisters, and the mother node is labelled with a copy of one of the two daughters; the one it is named after is the *head*. For example, in figure 1.6, *slept* is the head. Notice that bare phrase structure trees do not contain any indices to distinguish separate instances of the same word.

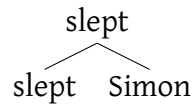


Figure 1.6: Bare phrase structure tree

## 1.2.4 Artificial Language Learning

Chapter 6 describes an Artificial Language Learning (ALL) study on a language with repetition.

In ALL studies, small, controlled grammars generate languages taught to participants. This careful control over the grammar allows for comparison of structures or patterns. My experiment tests participants' willingness to generalise a particular pattern – specifically whether people who learn a language with repeated adjectives accept sentences with more repetitions of adjectives than they heard while they were learning.

Since language learning is largely unconscious, ALL studies often include design features intended to tap into unconscious, or *implicit*, learning. For example, [Rohrmeier et al. \(2012\)](#)'s investigation of implicit learning of context-free languages used *process dissociation inclusion and exclusion tasks*. In the inclusion task, participants were presented with two items and asked to choose the grammatical one. In the exclusion task, they were asked to choose the ungrammatical one. Unconscious learning is thought to be reflected in the participant's inclination to choose the familiar item, even when the instructions are to choose the ungrammatical one. [Rohrmeier et al. \(2012\)](#) also used a distractor task to keep participants from focusing too consciously on the training stimuli.

### 1.2.4.1 A short history of artificial language learning

ALL started with [Reber \(1967\)](#), in which participants read patterns of letters generated by a regular (trigram) grammar. They were subsequently tested on new sentences gen-

erated by the same grammar (*grammatical items*) and sentences not generated by that grammar (*ungrammatical items*). Participants preferred the grammatical items to the ungrammatical, correctly accepting or rejecting on average 69.4/88 items.

Artificial Grammar/Language Learning experiments started off simply asking the question of whether people can learn some languages they were exposed to. Since that was established, new research began to examine what kind of representation people might have of the language they learned. [Perruchet and Pacteau \(1990\)](#) found that sometimes knowledge of legal bigrams could account for enough of participants' correct judgments of sentences from context-free languages. As such, more recent studies such as [Opitz and Friederici \(2007\)](#) and [Pallier et al. \(2012\)](#) make a point of controlling for simple knowledge of bigrams. Since human language cannot be accounted for by bigrams, we want to show that people learning human-language-like artificial grammars can learn more than just bigrams.

The experiment I ran does not control for bigrams – that will be done in a later experiment. Indeed, the language is so simple as to be a bigram language.

Acceptance and rejection of sentences is not the only way to get at how people represent an artificial language. [Friederici and Opitz \(2002\)](#) found that people learning a regular artificial language they generated with a human-language-like PSG were not only able to learn the language, but that learners listening to sentences with syntactic violations had fMRI scans showing activity very much like they were listening to syntactic violations in their native language; indeed, the more proficient the learners were, the more native-like the artificial language appeared in the brain. This evidence supports the possibility that people learning an artificial language that they recognise as human-language-like represent it however they represent human language.

Of particular interest to syntacticians is [Rohrmeier et al. \(2012\)](#), which tested learning of context-free languages. They found that people could indeed learn the languages, even when they controlled for bigram and trigram frequencies and surface repetition

structures. Naturally, an ALL experiment is limited to a finite chunk of a language, so demonstrating that participants were able to learn the target language is not perfect proof of their having learned a properly context-free artificial language. Regardless, this is a particularly careful study and all evidence is consistent with CF language learning, while grammars lower on the Chomsky hierarchy do not explain the results as well as a CF grammar.

ALL studies are much more common in phonology, as the patterns are lower on the Chomsky hierarchy, often making participants' knowledge easier to probe. For example, [Lai \(2011\)](#) and [Finley \(2012\)](#) examine learning of subregular patterns within words.

There are a number of studies that aim to probe learning high on the Chomsky hierarchy, but which do so with an alphabet of syllables generating a language of words, rather than an alphabet of words generating a language of sentences. For instance, [De Vries et al. \(2008\)](#) trained participants on  $a^n b^n$  patterns of syllables with nested dependencies. They found that people seemed to use extra-linguistic strategies such as counting, and that although they learned the  $a^n b^n$  pattern, they did not learn the dependencies. Famously, [Fitch and Hauser \(2004\)](#) trained humans and monkeys on  $(ab)^n$  and  $a^n b^n$  languages of syllable classes. The humans learned both languages, but the monkeys learned only the regular  $(ab)^n$  language. These experiments are difficult to interpret. On the one hand, the computational power required to do things like count is in fact context-free. However, counting does not seem to be how real human language syntax works: it achieves the same effect by recursive embedding. If ALL experiments are designed to probe how people really learn and represent language, we may not want to include learning by explicit strategies such as counting. Moreover, if it is true that human language phonology is subregular (mostly strictly local; some tier-based strictly local or strictly-piecewise) but human language syntax is mildly context-sensitive, then we would expect the language learner to be attuned to these patterns in their respective contexts. That is, we would expect a phonological learner to be a subregular learner, and a syntax learner to

be a MCS learner, and that the language learner would apply the former word-internally and the latter to sentences (Heinz, 2010). Syllables being neither sounds nor words, but being word-internal, people's learning patterns thereof are difficult to interpret.

## CHAPTER 2

### Adjuncts: the phenomenon

This dissertation is centred around the behaviour of adjuncts, but what exactly are adjuncts, and how are they different from non-adjuncts? The answer to this question, like most scientific questions, is not entirely clear. However, this need not necessarily worry us. The purpose of this chapter is to introduce the properties and behaviours that the dissertation will explore; these properties will not necessarily exhaustively describe adjunction.

We can stipulate a structural definition for the term *adjunct*, which, being stipulated, is clear and well-defined, but of course these structures are intended to be a model of a particular sort of real element with particular semantic and syntactic properties, which are less clear. Let us begin with the structural definitions.

#### 2.1 Structural definitions of adjuncts

In the X-bar grammars used in this dissertation, an adjunct is sister and daughter of a maximal projection, XP (Fig. 2.1). The crucial thing to notice is that the sister and daughter have the same label; they are instances of the same sort of constituent.

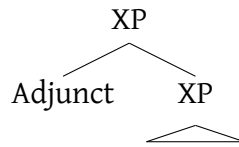


Figure 2.1: Adjuncts: sister and daughter of XP

Conversely, in a bare phrase structure grammar (Fig.2.2; see section 1.2.3 of chapter 1), there are no adjuncts,<sup>1</sup> since phrases are labelled just according to what heads the phrase; X' is not distinguished from XP. For instance, suppose *Mal* and *Jayne* are the subject and object of *hit*, respectively, and that *hard* is an adjunct of the verb phrase headed by *hit*. In the first tree in figure 2.2, *hard* is an adjunct of what would be the VP if this were an X-bar grammar, and in the second it would adjoin to V', but in either case, the phrase it adjoins to is labelled *hit*. The arguments *Mal* and *Jayne* as well as the adjunct *hard* are all sister and daughter of *hit*.

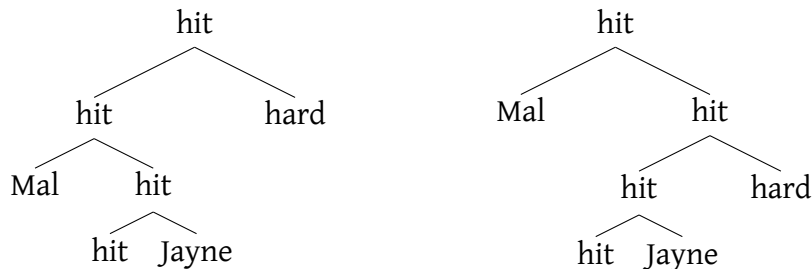


Figure 2.2: Bare phrase structure adjuncts are indistinguishable from arguments

One thing you can see in a bare phrase structure tree is which constituents are minimal projections. In the trees above, the leaf labelled *hit* is a minimal projection, which is the same thing as a head in X-bar. Maximal projections cannot always be identified. In Fig 2.2, the whole phrase is a maximal projection: it's the biggest phrase containing *hit*

<sup>1</sup>Or, equivalently, there are only adjuncts.

in which there is a path down through the tree in which every node is labelled *hit*. However, this will not always work, as bare phrase structure grammars do not distinguish separate occurrences of lexical items. For example, in Fig. 2.3, two separate instances of *a* are merged. The lower *a* clearly projects once, but the root node could be a projection of either *a*.

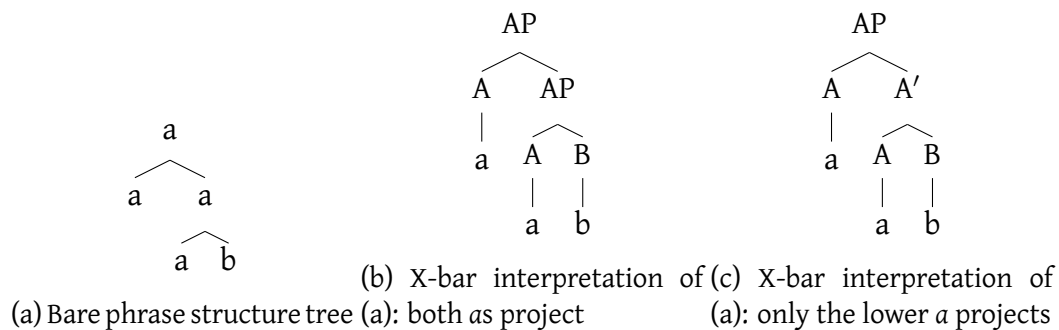


Figure 2.3: Which *a*s are maximal projections?

Even in the cases like 2.2 where maximal projections can be identified, this is not enough to distinguish adjuncts from arguments structurally. The maximal projection is just the last node labelled *hit*, and despite the fact we would like to say that there is some sense in which the first tree reached its maximum one node below the root and then stayed maximal, this is not true in a bare phrase structure tree.

Another version of bare phrase structure trees are *bare trees*, introduced in [Stabler \(1997\)](#). These phrase markers contain more information with arguably lighter notation. Instead of labelling internal nodes with a copy of the label of the head of the phrase, they are marked with arrows  $\langle$  or  $\rangle$  which indicate the path down to the head from any point within a projection.



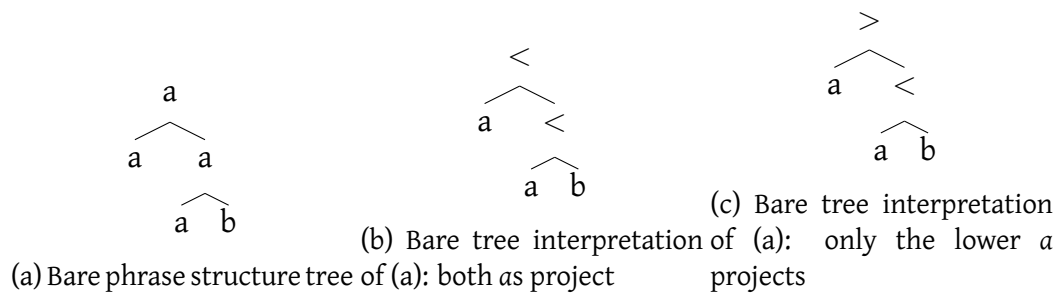


Figure 2.4: Which *as* are maximal projections?

In a bare tree, a node is a maximal projection if its mother is not labelled with an arrow pointing at it. However, knowing which projections are maximal does not distinguish adjuncts from arguments. In the tree in 2.4c, the higher *a* could have been selected as a specifier, or it could have been adjoined. The bare tree contains no record of how it was built that will distinguish these two scenarios. Structurally, then, bare trees, like bare phrase structure trees, do not have adjuncts.<sup>2</sup>

The structural definition of adjuncts as sister and daughter of a maximal projection is specific to a particular kind of grammar, but can sometimes be useful even when not wedded to these particular grammars, to illustrate basic ideas. For example, with X-bar trees, it is particularly easy to illustrate the notion that the optionality of adjuncts is about the fact that you have the same kind of phrase whether you have an adjunct present or not.

A second structural definition comes from Cinque (1999): an adjunct is a selected (not moved) specifier of certain functional projections. This approach is explored in more depth in Section 2.8.2.

---

<sup>2</sup>Or, equivalently, only have adjuncts.

## 2.2 Semantic definition of adjuncts and arguments

Arguments are usually thought to be semantic arguments of the function (eg verb) that is subcategorised for them. For example, *shoot* is a verb subcategorised for a DP. The verb phrase *shoot Jayne* has the interpretation  $\llbracket \text{shoot} \rrbracket(\llbracket \text{Jayne} \rrbracket)$  – the meaning of the DP *Jayne* is the argument of the function denoted by the verb *shoot*.

The semantics of adjuncts on the other hand is often thought to be intersective. For example, the meaning of *orange hat* is the intersection of the set denoted by *orange* and the set denoted by *hat* ( $\llbracket \text{orange} \rrbracket \cap \llbracket \text{hat} \rrbracket$ ). Not all adjuncts are strictly intersective – for example *big* is not strictly intersective: its interpretation depends on the context. The set of big things arguably contains no mice at all, yet *big mouse* is meaningful.

There are other ways of defining adjuncts too, some of which define the modifiers as functions (for example, second-order logic models that define adjectives as functions from unary predicates to unary predicates), leading to the possibility of type-raising creating modifiers that are arguments of functions, just like canonical arguments.

## 2.3 Subcategorisation definition of adjunction

Adjuncts are traditionally distinguished from arguments by their optionality. Predicates are *subcategorised* for their arguments: each predicate in the lexicon comes with a set of requirements, without which the predicate is incomplete. For example, *kiss* requires two arguments, both nominals (in contemporary syntax we would say DPs), one of which will be interpreted as the theme/object/kiss-ee and the other of which will be interpreted as the agent/subject/kisser.

**Definition 2.3.1** (Argument). A phrase semantically required by some predicate

**Definition 2.3.2** (Adjunct). A phrase that is not an argument

We run into some trouble, diagnostically, when we encounter optional arguments and

obligatory adjuncts. The former are well-defined in the grammar; they are simply listed in the subcategorisation frame as arguments of the verb, and the definition of *argument* is shifted slightly to Definition 2.3.3.

**Definition 2.3.3 (Argument-2).** A phrase for which a predicate is subcategorised

However, distinguishing optional arguments from adjuncts from real data – to build the grammar in the first place – is not always easy. Syntactic diagnostics can help with this task, and they can also help with the problem of diagnosing optional arguments and obligatory adjuncts.

## 2.4 Syntactic diagnostics

A number of diagnostics are available for distinguishing arguments from adjuncts.

The first has already been alluded to: optionality. *Can the phrase be dropped from the sentence?* In (1), *quickly* can be dropped, but *Kaylee* cannot.

- (1) a. Kaylee worked quickly
- b. Kaylee worked.
- c. \*Worked quickly.

However, this diagnostic can also fail both positively and negatively. In (2), there is reason to believe *the cake* is an argument, even though it is optional, and in (3), there is reason to think *good* is an adjunct, even though it is required. (These reasons are discussed in section 2.5 below.)

- (2) a. Simon ate the cake.
- b. Simon ate.
- (3) a. Simon makes a good brother.

- b. \*Simon makes a brother.

Adjuncts can sometimes change places with other adjuncts, but rarely with arguments. For example, in (4-a), the PP *on the table* is an argument of *put*, so it sticks close to the verb, but *on Thursday* is an adjunct so it can be further. In contrast, in (4-c) and (4-d), neither PP is an argument of *saw*, so both orders are acceptable.

- (4) a. Inara put a cup of tea on the table on Thursday.  
b. \*Inara put a cup of tea on Thursday on the table.  
c. Inara saw Mal and Saffron from the doorway on Thursday.  
d. Inara saw Mal and Saffron on Thursday from the doorway.

Most other diagnostics are phrase-specific. For low verbal adjuncts, often the sentence can be re-phrased without the adjunct, and with the adjunct in an embedded clause like *...which occurred [Adjunct]*. For instance, the PP *in her shuttle* is an adjunct in (5-a) but an argument in (5-c), making (5-b) much better than (5-d) (or at least (5-d) cannot mean the same thing as (5-c)).

- (5) a. Inara drank a cup of tea **in her shuttle**  
b. ?Inara drank a cup of tea, which occurred **in her shuttle**  
c. Inara left a cup of tea **in her shuttle**  
d. \*Inara left a cup of tea, which occurred **in her shuttle**

English has replacement tests for VPs and NPs (or V's and N's, if adjuncts adjoin to bar-levels). VP can often be replaced with *do so* and NP can often be replaced with *one*. This opens the possibility of stranding adjuncts with the replacement. If the highest NP can be replaced with *one*, then so can a lower one.

- (6) a. Give me that big hat.  
 b. Give me that one.  
 c. Give me that big one.

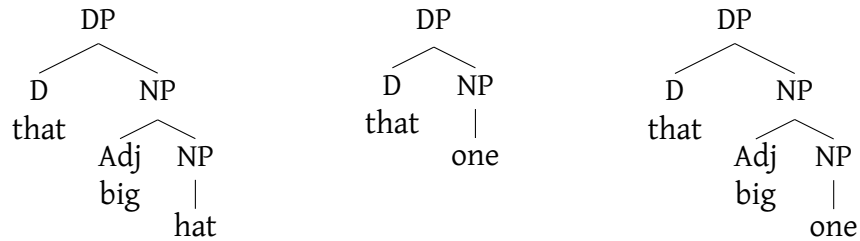


Figure 2.5: All NPs can be replaced with *one*

Adjectives are usually fairly easy to diagnose as adjuncts, but PPs are harder. Here the NP-replacement test diagnoses *of*-PPs as arguments of *destruction* but adjuncts of *picture*.

- (7) a. Did you see the picture of the bridge?  
 b. No, but I saw the one of the road.  
 c. Did you see the destruction of the bridge?  
 d. \*No, but I saw the one of the road

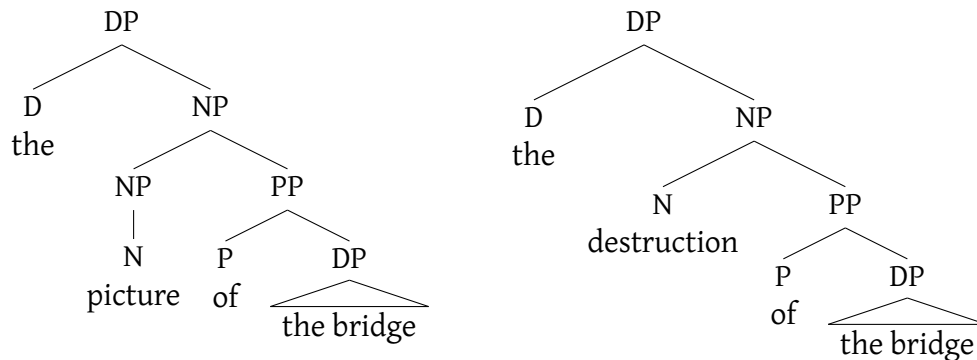


Figure 2.6: According to the diagnostic, the PPs are adjuncts of *picture*, but arguments of *destruction*

A similar diagnostic is available in German and Dutch, with deletion in place of substitution. For example, in a context where we are already talking about different hats, the following are all possible in Dutch:<sup>3</sup>

- (8) a. Geef mij die hoed met de pluim  
give me that hat with the feather  
'Give me that hat with the feather'
- b. Geef mij die  
give me that  
'Give me that one'
- c. Geef mij die met de pluim  
give me that with the feather  
'Give me that one with the feather'

## 2.5 Obligatory adjuncts

We saw in (3) above that adjectives can be obligatory adjuncts. We can use *one*-replacement to test the hypothesis that *good* is an adjunct.

- (9) a. Simon makes a good brother.  
b. ...but Jayne would make a terrible one.

This test supports the hypothesis that the adjective really is an adjunct even though it is obligatory.

VP in English can be replaced with *do so*, which can help diagnose obligatory adjuncts as adjuncts, not arguments. The sentences in (10) show that in the sentences *He worded the letter carelessly* and *She worded the letter carefully*, the adverbs *carefully* and *carelessly* are obligatory adjuncts of *worded*.<sup>4</sup> We contrast *wrote* with *worded*. (10-a) and (10-c) show

---

<sup>3</sup>Data from Floris van Vugt, p.c.

<sup>4</sup>There is some inter-speaker variation here. Carson Schutze, for example, finds the examples lacking

*carefully* is optional with *wrote* but not with *worded*. This would seem to diagnose argumenthood of *carefully/carelessly*, but (10-b) and (10-d) show do-so replacement works for both verbs, stranding the adjuncts. This indicates *carefully/carelessly* are adjuncts of *worded* despite their obligatoriness.

- (10) a. He wrote the letter.  
b. He wrote the letter carelessly, but she did so carefully.  
c. \*He worded the letter  
d. He worded the letter carelessly, but she did so carefully.

Not all adjuncts work here.

- (11) a. \*He worded the letter at his writing desk  
b. \*He worded the letter with help from his sister  
c. \*He probably worded the letter  
d. \*He worded the letter in ten minutes

The requirement seems to be for a manner adverb such that the meaning of the sentence

---

the adverb perfectly grammatical (p.c.). Goldberg and Ackerman (2001) also find some variation. They comment (pp. 18, example numbers mine):

*Word* appears to be an intermediate case. It is not generally possible to use *word* without an adverb (i), although speakers we have queried vary in their judgments of (i-c) (indicated by us with %):

- (i) a. #She didnt WORD the letter!  
b. #She finally worded the speech.  
c. %We've figured out the content of the exam questions, but we havent worded them yet. (Ernst 1984).

*Word* used to appear readily without an adverb meaning simply, to express in or put into words (OED online) as in Burton's Diary (1828) IV. 225, *I would have the question worded, before you rise...* However today it appears almost exclusively with an adverb. It thus appears that *word* has evolved through a process of grammaticalization.

is *the wording of the letter* was ADJ, for the adjectival equivalent ADJ of the adverb.

- (12)
- a. He worded the letter carefully
  - b. The wording was careful.
  - c. He worded the letter enthusiastically
  - d. The wording was enthusiastic.
  - e. ??He worded the letter early
  - f. ??The wording was early
  - g. \*He (probably) worded the letter (, probably)
  - h. ??The wording was probable

Gross (1979) notes that by-phrases in passives can sometimes be obligatory.

- (13)
- a. The house was burned down Optional by-phrase
  - b. The house was burned down by an arsonist
  - c. \*The house was designed Obligatory by-phrase
  - d. The house was designed by my grandfather
  - e. \*The best tomatoes are grown Obligatory by-phrase
  - f. The best tomatoes are grown by organic farmers

However, Grimshaw and Vikner (1993) point out that other adjuncts can save the grammaticality of these sentences.

- (14)
- a. The house was designed six weeks before construction began
  - b. The best tomatoes are grown in sandy soil
  - c. The house was designed carelessly

Not all adjuncts work.



- (15) a. \*Fortunately, the house was designed.  
b. \*The house was probably designed.

In these sentences, none of our diagnostics work, as the adjuncts are obligatory and *do so* replacement is ungrammatical for passives, as for example in (16)

- (16) a. The ball was thrown over the fence, ...  
b. \*...and the bat was done so through the window

However, Grimshaw and Vikner (1993) argue that they are nonetheless adjuncts, due to their variety (it would be a strange set of alternative categories to select for: PP or MannerAdvP) and to the fact that by-phrases are adjuncts according to all other tests.

Grimshaw and Vikner (1993) suggest that the class of sentences that require adjuncts of this sort are the passives of constructive accomplishments: accomplishment events in which the event creates the theme of the predicate. For example, before tomatoes are grown, there are no tomatoes, but after they've been grown, there are tomatoes. Thus *grow* is a constructive accomplishment. In contrast, before a house is burned down, there is a house, and before a ball is kicked, there is a ball.

It is the internal structure of constructive accomplishments that explains which adjuncts save the construction: those which have two sub-events. The first sub-event is the process, and the second is the coming into existence of the theme. The theme identifies the second sub-event; the adjunct serves to identify the first sub-event.

Perhaps, though, the sentences marked with asterisks ought really to be marked with number signs, marking them as semantically odd rather than ungrammatical. Take, for example, (13-e). In a magical universe, witches and wizards might debate the merits of naturally grown versus magically conjured tomatoes<sup>5</sup> Here, (13-e) could be quite accept-

---

<sup>5</sup>Here we must of course exclude the Harry Potter universe, in which the five principal exceptions to

able. Similarly, imagine a future in which many structures are not designed by people but rather generated by algorithms. In this future, we might debate the artistry of designed versus generated houses. The adjuncts that save the sentences seem to make them pragmatically usable, by making them non-trivial. Without them, there is no situation in the present real world in which they are utterable, as all tomatoes are grown and all houses are designed. Goldberg and Ackerman (2001) make a similar argument: most – though likely not all – obligatory adjuncts are only pragmatically obligatory, required to satisfy the Maxim of Quantity.

If it is ultimately determined that no true adjuncts are truly syntactically obligatory, there is nothing to account for in the syntax. If, however, there in fact remain syntactically-obligatory adjuncts, the puzzle remains: how are these adjuncts forced into the sentence? How can adjuncts be required in some cases but optional in others, yet still be the same sorts of phrases otherwise? Section 4.7.2 of Chapter 4 offers a possible solution for the general problem of obligatory adjuncts, but it over-generates: any adjunct will do, but we have seen that in actuality, only particular adjuncts work.

## 2.6 Adjuncts in phrase structure grammars

Recall from Section 1.2.1.3 above that a *phrase structure grammar* is a set of re-write rules in which a non-terminal symbol can be re-written as a sequence of terminal and/or non-terminal symbols. One symbol is designated as the “start” symbol, and any sequence of terminal symbols that can be derived from the start symbol qualifies as a grammatical sentence of the language; it is *generated* by the grammar.

For example, let  $S$  be the start symbol in the grammar in 2.7. ( $\epsilon$  is the empty string.) This grammar can generate  $a^n b^n$ ; for example the tree in 2.7 shows how  $aaaabbbb$  is derived from  $S$ , first with expansions of  $S$  using rule 1, and then with one using rule 2. In

---

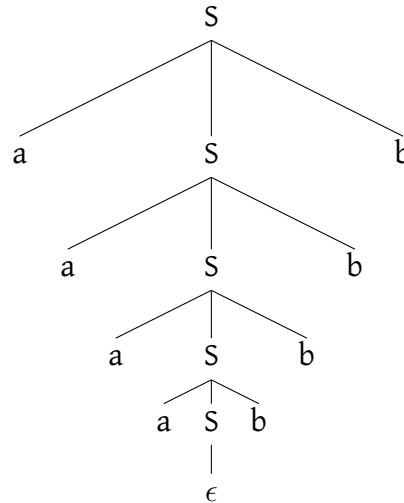
Gamp’s law of elemental transfiguration include food (Rowling, 2008).

this grammar,  $S$  is a non-terminal, and  $a$  and  $b$  are terminal symbols.<sup>6</sup>

1.  $S \rightarrow aSb$

2.  $S \rightarrow \epsilon$

(a) Phrase structure grammar for  $a^n b^n$



(b) Parse tree for  $a^4 b^4$

Figure 2.7:  $a^n b^n$

An arbitrary phrase structure grammar is not restricted in which symbols can appear on the right hand side of a given rule. For example, there is no constraint that requires there to be anything verbal on the RHS of a VP rule. A rule like  $VP \rightarrow TP D P'$  is a perfectly licit PSR. Usually in linguistically relevant grammars, however, licit PSRs are endocentric: most or all rules have somewhere on the right hand side of the arrow a symbol which is repeated on the left hand side in some way. For example, a *noun phrase* is a phrase that acts like a noun, leading to rules like the one in (17), which is used to represent the fact that when a prepositional phrase follows a noun, the whole thing acts like a noun. The phrase is therefore named after the noun: it's called a noun phrase.

(17)  $NP \rightarrow N PP$

---

<sup>6</sup> $a^n b^n$  is the language consisting of all sentences starting with some number of  $a$ s followed by the same number of  $b$ s. The name is short for  $\{a^n b^n | n \in \mathbb{N}\}$

However, this is not quite true: noun phrases do not always behave exactly like nouns. The name “noun phrase” also says something about the “completeness” of the phrase. For instance, sometimes adding a DP after a verb makes the verb “complete”, which means it has everything it needs to be allowed to participate in the sentence. For this reason we don’t want to just call it a verb, even though it acts rather like a verb. It’s something new: a verb plus all its arguments.

(18)  $VP \rightarrow DP V_{\text{intransitive}}$

Stowell (1981) points out that phrase structure rules and the lexicon are largely redundant. All PSRs for your average 1980s phrase structure grammar that included categories could be replaced by the lexical entry and a general rule naming the category of the constituent formed. For example, the PSRs in (19) encode a subset of the information encoded in the lexicon in Table 2.1, except the label on the left-hand side of the PSRs. The lexicon includes more information, namely which words can occur with which rules.

(19) a.  $VP \rightarrow V (NP)$

b.  $NP \rightarrow N (PP)$

word	category	arguments
devour	V	NP
eat	V	(NP)
in	P	NP
destruction	N	(PP)
cat	N	

Table 2.1: Sample lexicon for 1980s-era grammar

If we add to Table 2.1 a rule naming the new internal node, say *the node dominating a phrase containing a single category X (and optionally some phrases) is labelled XP*, then the

arguments are taken care of, without the need for phrase structure rules like those in (19). Notice however that modifiers are not taken care of by this schema. Modifiers in this system are defined as phrases with sister and mother having the same (phrasal) label. For example, AP is a modifier of NP:

$$(20) \quad NP \rightarrow AP \ NP$$

In order to include modifiers in the grammar, we need either to keep these phrase structure rules, to add to the grammar a set of modifiers and their categories as in Table 2.2, or to incorporate modification into the lexical entries, as in Table 2.3.

category	modifiers
V	{AdvP, PP, CP}
N	{AP, PP, CP}
A	{AdvP, IntensifierP}

Table 2.2: Adding adjuncts into the grammar

word	category	arguments	modifiers
devour	V	NP	{AdvP, PP, CP}
eat	V	(NP)	{AdvP, PP, CP}
in	P	NP	
destruction	N	(PP)	{AP, PP, CP}
cat	N		{AP, PP, CP}

Table 2.3: Adding adjuncts into the lexicon

Missing here for both the arguments and modifiers is order relative to the head. Here we could argue for keeping phrase structure rules just for order ( $NP \rightarrow N \ PP$ ;  $NP \rightarrow AP \ NP$ ;  $NP \rightarrow NP \ PP$  etc), to include in the grammar a general rule, perhaps language specific, and claim other orders are the result of movement (eg: arguments are to the left; adjuncts are to the right), to include some of the order information in the lexicon (distinguish left and right adjuncts, distinguish left and right arguments), or some combination.

Categorial Grammars (Ajdukiewicz, 1935; Bar-Hillel, 1953; Lambek, 1958, etc.) formalise this same observation. The derivation is driven entirely by argument fulfilment. The order of the argument and selector is notated by the order of categories in the type of a selector, and which is the selector and which is the argument is notated by direction of the slashes that separate categories. For example, a grammar might have three categories or basic types, N, NP, and S. A noun might have a simple type, just N, while a determiner like *the* could have a complex type NP/N, meaning that it selects an N on its right to form an NP. S is the type of a complete sentence. Here are two sample derivations.

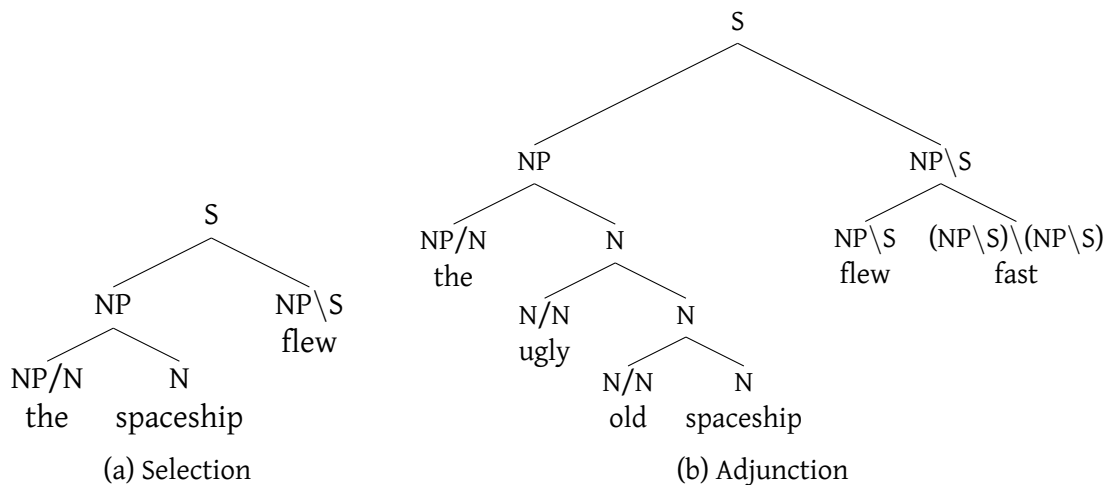


Figure 2.8: Categorial Grammars

Modifiers are given types in which the same types appear on both sides of the slash,  $x/x$  or  $x\backslash x$  for some type  $x$  (Fig. 2.8b). However, categorial grammars do not distinguish adjuncts from selectors that just happen to select something of the same type as them: both have category  $x/x$  or  $x\backslash x$ .

The Merge component of Minimalist Grammars works just like categorial grammars, except for the fact that in a basic MG the order of selector and selectee is not notated in the feature list. Traditionally, they solve the argument order problem with two Merge

rules: Merge of a complement is to the right; Merge of a specifier is to the left. This works well if the goal is to create Kayne-style X-bar trees, but they work poorly for adjuncts, as they treat adjuncts as arguments. “Adjuncts” therefore appear on either side of the phrase, depending on whether they are the first “argument” merged and therefore treated as a complement. The model I present in Chapter 4 solves the adjunct ordering problem with different Adjoin instances for different classes of adjuncts.

The observation that everything you need to know about phrase structure rules can be encoded in the lexicon becomes particularly important in Minimalism (chapter 3), where everything is encoded in the lexicon. There are but two rules, Merge and Move, and their validity depends on, if anything, features in the lexicon. Traditionally, minimalist models encode modification as identical to argument selection, driven by category features. Throughout this dissertation, I argue that modifiers should not be treated identically to arguments in the syntax. Even in the earliest days of modern syntax, any attempt to encode phrase structure properties in the lexicon needed to treat modifiers as distinct from arguments.

## 2.7 Transparency

I argue that adjuncts have a property I call *transparency to selection*. What this means is that when a phrase that an adjunct is adjoined to is selected, the selector can see right through the adjunct to the selectee; the adjunct does not normally affect the selectability of the phrase. (A possible exception is obligatory adjuncts – see Section 2.5.)

One aspect of this is the fact that no matter how many adjuncts are adjoined to a phrase, it remains selectable by the same selectors. For example, in English, the noun phrase *hunk of junk* is selectable by the determiner *a* no matter what is adjoined to *hunk of junk*.

- (21) That spaceship is **a rusty old hunk of junk** lying in the trash heap that I wouldn't trust to take me to the store and back.

Languages with gender agreement between determiner and noun, which is not necessarily reflected on the adjuncts, show this transparency well. For example, in Dutch definite DPs, gender is not represented on the adjective.<sup>7</sup> The determiner must nonetheless agree with the gender of the noun.

- (22) a. Het boek  
the<sub>NEU</sub> book(NEU)  
'the book'
- b. De man  
the<sub>DE</sub> man(DE)  
'the man'
- c. Het grot-e boek  
the<sub>NEU</sub> big-DEF book(NEU)  
'the big book'
- d. \*Het grot-e man  
the<sub>NEU</sub> big-DEF man(DE)  
Intended: 'the big man'
- e. de grot-e man  
the<sub>DE</sub> big-DEF man(DE)  
'the big man'

Similarly, in Q'anjob'al<sup>8</sup>, classifiers must agree with the class/gender of their noun.

- (23) a. te' mexha  
CL<sub>WOOD</sub> table  
'the table'

---

<sup>7</sup>NEU=neuter/"het" gender, DE= common/"de" gender.

<sup>8</sup>Data from Bervoets et al. (2011)



- b. te' yalixh mexha  
CL<sub>WOOD</sub> small table  
'the small table'
- c. \*no' yalixh mexha  
CL<sub>ANIMAL</sub> small table

In the verbal sphere, I argue that even functional heads (argued to be intricately tied to adverbs (Cinque, 1999)) select as though adjuncts were not present, as long as those adjuncts are not higher in the Cinque hierarchy<sup>9</sup> than the functional heads themselves. For instance, suppose in English T selects V, as per the traditional analysis.<sup>10</sup>

(24) Kaylee can fix Serenity

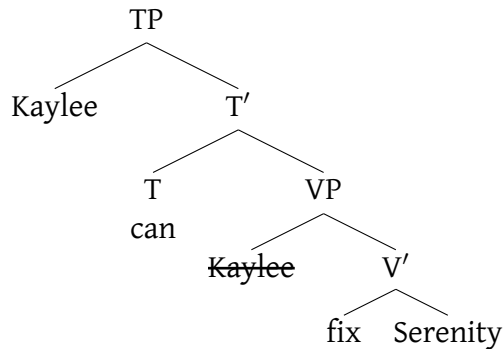


Figure 2.9: T selecting a VP

Suppose *definitely* adjoins to VP, giving us

(25) Kaylee can definitely fix Serenity.

Despite the presence of *definitely*, the T head *can* still selects the VP as usual. It still re-

<sup>9</sup>See section 2.8.1.1 below

<sup>10</sup>Or something similar. The crucial point is that there be room for adjuncts between the selector and the selectee.

quires that the verb be in bare infinitival form (not *fixes* or *fixed*), just as it did when the adverb was not present.

Now, this notion of transparency is not merely a surface fact; it is a claim about the structure of the sentences and the grammar. It could instead be the case that selection *is* interrupted by the presence of adjuncts, and the selector is designed to select *either* the un-adjoined-to phrase *or* one with adjuncts (see chapter 4, section 4.5.3.2). It could be that the adjunct actually *selects* the so-called adjoin-to phrase, and the phrase that is selected by the usual selector is actually the “adjunct”’s phrase (see chapter 4, section 4.5.1). It could be that there are silent, meaningless versions of the adjuncts (or functional heads that select them) present in every sentence, and the idea that, say, D selects N is an illusion (see chapter 4, section 4.5.3.1).

However, I argue that adjunction *is* transparent. The only sense in which phrases with adjuncts behave syntactically differently from phrases without adjuncts is in terms of ordering restrictions on adjuncts.<sup>11</sup> Otherwise, they might as well not be there. Agreement is uninterrupted, selection is uninterrupted. For most of linguistic history, adjunction has been analysed to be transparent. It is only with the advent of the cartographic approach that adjunction has been analysed as not adjunction at all, but rather selection by functional heads. The cartographic approach gains us two major things: the relative ordering of adjuncts, and the relative ordering of functional morphemes. In chapter 4 I argue for a model that can account for this ordering and still capture the apparent transparency of adjunction.

---

<sup>11</sup>If obligatory adjuncts are a true syntactic phenomenon – see section 2.5 – then there is a second case in which phrases with adjuncts behave differently from those without. However, this does not preclude transparency. A head selecting a phrase of category X that has an adjunct is still selecting a phrase of category X; it’s just that that phrase must have other properties as well. In section 4.7.2 of chapter 4 I propose a way to keep adjuncts transparent to selection yet still require an adjunct, using the same machinery that solves the other problem, that of adjunct ordering.

## 2.8 Ordering restrictions

Despite their optionality and transparency, adjuncts have been argued to have ordering restrictions. Most famously, Cinque (1999) proposes a 30-level hierarchy of adverbials. Deviation from the hierarchy is due to movement, and results in an informationally different meaning – usually focus – and often a different intonation.

### 2.8.1 Adjective ordering

Adjectives in many languages have a default order. I follow Hetzron (1978) in considering only adjective sequences other than the following: adjectives connected by conjunctions (*red and pretty/pretty and red*), adjectives pronounced with comma intonation, due to their being possible parentheticals and afterthoughts (*red, pretty/pretty, red*), adjectives with stress that have a contrastive focus meaning (*I mean the red pretty ball, not the green one*), and adjectives that are modifiers of other adjectives in the sequence (*the light-blue sky*). This leaves sequences of adjectives with no special intonation, which mean simply that the noun has all of these properties. For example, (26) refers to a ball that is both pretty and red.

(26) the pretty red ball

Adjuncts connected by conjunctions are not adjoining sequentially to the phrase, but rather form a constituent of their own by whatever selectional mechanism forms conjunctions. Adjectives with comma intonation may be conjoined in a list – also a constituent of their own and therefore not subject to adjunction restrictions (Cinque, 2010) – or moved. Contrastive adjuncts may have moved from their usual position to a focus head (Giusti, 1996), and adjuncts of adjuncts are subject to ordering restrictions within their own constituent, so they do not play into the restrictions on the ordering of ad-

juncts in the main constituent.

Adjectives can be divided into classes, and these classes are ordered with respect to each other. The questions are how to divide them into classes, why those classes are ordered as they are, whether these same classes hold across languages, and whether their ordering is consistent across languages.

Most attempts to divide adjectives into classes do so on semantic grounds. For example, all colour adjectives belong to the same class; I have seen no one claim that *red* and *green* belong to different classes, say with *green* ordered before *pretty* and *red* ordered after it.

Often attempts to account for ordering of semantic classes appeal to some supposedly inherent quality of the adjective and its relationship with the noun, such as how much the adjective's meaning varies by what it modifies, or how "intrinsic" to the noun the adjective's property is. We will see that while semantic classes of adjectives are probably the right division, attempts to explain the order semantically fail.

Sweet (1898), the earliest English reference I found, proposes that adjectives are ordered from least to most "closely connected with [the noun] in meaning" (§1789). This close connection, or "specialisation", is demonstrated by the existence of a single-word synonym of the adjective+noun combination. Examples given include (with apologies for the old-fashioned and therefore now-offensive language):

- (27) a. the three **wise men** = the three **sages**  
b. a tall **black man** = a tall **negro**

In (a) the author conflates numerals with true adjectives. Modern grammars generally model numerals separately from adjectives. Since *three* is not an adjective, there is no surprise that there is no single word that can stand in for "three sages", so (a) is not a good example. In (b), *black man* has compound intonation, so it is not a clear case of an

adjunct either. Even if we had a good example, it would be quite easy to reject as the correct generalisation. Counterexamples abound; for example, *an enormous fancy blue house* is a *blue mansion*, with the farther-out adjective combining with the noun to form a single word. There is no word in English meaning *blue house*. In the literature review in [Martin \(1969\)](#), this “specialisation” is interpreted as how inherent to the noun the property the adjective denotes is.

Sweet’s attempt fails. In the example given with two adjectives, the implication is that a man’s race is more inherent to him than his height. Biologically, this is false: race is a social construct based loosely on biological facts, while height is a measurable quantity. Whether a man’s race being more inherent to him than his height is emotionally or culturally true depends on the social milieu in which the man finds himself, but adjective ordering has not changed over time or across cultures to reflect this. In *the big bad wolf*, which is a more inherent property of the wolf, his size or badness? In *the big red table*, surely the size of the table is more inherent when a simple coat of paint can transform it into a big blue table.

Another dimension proposed in the literature to influence adjective order is “definiteness of denotation” ([Martin, 1969](#)) which means the extent to which the meaning of the adjective depends on the meaning of the noun. Adjectives that are more stable in their denotation are said to appear closer to the noun. For instance, *Chinese* is fairly stable in its interpretation: it means relating to China. The exact relationship (*from, spoken in, happening in, made in...*) varies, but otherwise the meaning is the same. Size measures, conversely, depend on a standard for their meaning. A short mouse is much shorter than a short elephant; a large subatomic particle is much smaller than a large star.

This is a better-defined notion, but it is inadequate. For example:

(28) The small ancient triangular green Irish pagan metal artefact

While *small* and *ancient* are variable, it is not clear how *green* is more variable in meaning than *metal*, when it comes to the noun *artefact*. *Green* means its colour falls in a particular range of the the visible light spectrum, and *metal* means it is made of a material with particular chemical properties, like gold or iron. These meanings do not vary according to whether the noun is *artefact*, *dish*, or *statue*.

- (29) a. The green metal artifact/dish/statue  
b. ?the metal green artifact/dish/statue

Ziff (1960) observes a semantic distributional fact: some adjectives make sense with far more nouns than others. He calls this *privilege of occurrence*. For instance, adjectives of material only make sense with things that can be made of something, while evaluative adjectives like *good* can work with just about anything.

- (30) a. A good dress  
b. A good shawl  
c. A good speech  
d. A lace dress  
e. A lace shawl  
f. #A lace speech  
g. A good lace dress  
h. \*A lace good dress

He proposes that adjectives with greater privilege of occurrence occur farther from the noun. This works for some pairs (such and *lace* and *good* in (30-g) and (30-h)) but not, as Ziff himself notes, with others, such as *young* and *pious*, where *young* has the greater privilege of occurrence, but follows *pious*.

- (31) a. A young girl

- b. A young turnip
- c. A pious girl
- d. #A pious turnip
- e. A pious young girl
- f. ?A young pious girl

Hill (1958) notes that in English, adjectives derived from nouns like *wooden* tend to be closer to the noun. He orders the rest as follows:

(32) [other] size shape age colour noun-derived N

Brown (1965), a social psychologist, lists an adjective order without attempt at analysis. I created the example sentence mostly from his examples of adjectives in each category.

- (33) a. characteristic size shape temperature/humidity age colour origin  
 b. a witty short stocky warm young black American man

Danks and Glucksberg (1971) propose ordering based on *intrinsicness*, and give it an interpretation consistent with Brown's ordering. By way of explanation they write (Danks and Glucksberg, 1971, pp.63):

Given a big red Swiss table, the property Swiss is more intrinsic to the table than the property of color. Further, bigness is not an intrinsic property at all. A table is big only in relation to some other object. For our present purposes we can explicitly order some of the adjective classes on an intrinsicness scale as follows: inherent and central to the nature of the object (e.g., place of origin); superficial property (e.g., color); relative to some reference object or condition (e.g., size); and relative to a personal judgment (e.g., possession).

Again, it is not clear that there is anything more "intrinsic" about origin than size or

colour. What is true is that it is a matter of fact, not of opinion. It is not relative, and that is perhaps the authors' point. However, this lack of relativity begins to fall apart quite quickly. Is origin less relative than colour, given that nations are social constructs that change throughout history? Is age less relative than size, given the arguably equal disparity between *an old galaxy vs an old fruitfly* and *a big galaxy vs a big fruitfly*?

The grounds being shaky for intrinsicness as a general rule for adjective ordering does not, of course, change empirical facts about adjective ordering. To test whether people really do prefer the apparent usual adjective order over other orders the authors ran an experiment. They examined the order of three adjective classes: Size/Shape > Colour > Origin. Participants ranked pairs of sentences drawn from the following 6 types.

- (34) a. A big red Swiss table Normal order  
b. A big Swiss red table  
c. A red big Swiss table  
d. A red Swiss big table  
e. A Swiss big red table  
f. A Swiss red big table

They found that participants were quite consistent in their rankings of sentence types, and that their overall ranking was as in the order in (34) above. However, they did find that even the order in (a) was not accepted at as high a rate as their grammatical filler sentences without the three adjectives.

Vendler (1961) finds four adjective classes, ordered as follows:

- (35) a. exclamatory verb-like adverb-like noun-like  
b. bulky yellow catlike American

Vendler (1968) proposes to explain these orderings by the transformations required to



attain them in a transformational grammar, along with an (ad hoc) ordering on the transformations. For example, the derivation in (36-a) is an example of the transformation in (36-b). Similarly, the transformation in (37-a) is an application of the transformation rule in (37-b).

(36) a. the chair is red  $\rightarrow$  the red chair

b. N is A  $\rightarrow$  AN

(37) a. the chair is large for a chair  $\rightarrow$  the large chair

b. N is A for N  $\rightarrow$  AN

Vendler divides adjectives into categories according to which constructions, other than the basic pre-nominal, they can be used in. In this way he expands his classes from 4 to 9, named  $A_1$  to  $A_9$ . I list here some of the relevant constructions. Most of the names were added by me.

1. N which is A (*red rose*: the rose is red)
2. N whose measure is A (a *tall man's* height is tall)
3. N such that it Vs A-ly (a *wise king* rules wisely)
4. N that is A to V (an *easy book* is easy to read)
5. Control: (an *eager* participant is eager to participate)
6. Subject-oriented sentential: (the *stupid man* was stupid to have done that)
7. Possibility (a *possible candidate's* running for parliament is possible (\*for me))
8. Necessity: a *useful tool's* fixing the problem is useful for me
9. Probability: that a *probable winner* will win is probable

$A_1$ 's are further subdivided into 13 subclasses as follows.

$A_a$  : material *wooden, iron*

$A_b$  : (mostly) origin: N-ic, N-an, N-ine, N-ese, N-ish, N-al *metallic, Wagnerian, canine, Japanese, Turkish, tonal*

$A_c$  : N-y, N-like *silky, cat-like*

$A_d$  : N-al, N-ical *radical, logical*

$A_e$  : N-ish *foolish, childish*

$A_f$  : shape, colour, & texture *rectangular, granular*

$A_g$  : has N *humorous, colourful, greasy*

$A_h$  : present participles: *floating, dormant*

$A_i$  : past participles: *broken, aged*

$A_j$  : inclination: *active, rabid, shiny*

$A_k$  : passive inclination: *breakable, fragile*

$A_m$  : contrastive: *young, old, warm, cold, happy, sad*

$A_x$  : emotive (causing emotions): *terrible, delightful*

The complete order hypothesised by Vendler is:

(38)  $A_9 A_8 \dots A_2 A_x A_m \dots A_a N$

He does not provide evidence for every relative ordering. It may turn out that the classes are not quite right, or too finely articulated. Since he is working within a transformational grammar, he is looking for classes that can be described and explained transformationally. Regardless, these classes provide a good starting hypothesis, well worth exploring experimentally. Some of his examples are given in (39). The subscripts correspond to his categories.

- (39) a. easy<sub>4</sub> short<sub>2</sub> mathematical<sub>d</sub> demonstration  
 b. tiny<sub>2</sub> rectangular<sub>g</sub> yellow<sub>f</sub> spots  
 c. comfortable<sub>4</sub> upholstered<sub>j</sub> brown<sub>f</sub> mahogany<sub>a</sub> chair

The title of this dissertation is:

- (40) Slaying the Great<sub>2</sub> Green<sub>f</sub> Dragon: learning and modelling iterable<sub>k</sub> ordered<sub>i</sub> optional<sub>d</sub> adjuncts

The five orderings explored above are descriptively fairly consistent. The classes preceding Size for Hill (“other”) and Brown (“Characteristic”) could be considered superclasses for Vendler’s more finely articulated  $A_2 - A_9$ . Vendler’s “Contrastive” ( $A_{1m}$ ) seems wrong in that it lumps together temperature and age, which are quite low and which Brown and Hill put below Shape, with higher adjectives such as *happy*, and places them before Shape and various verbally-derived adjectives. Otherwise, Hill’s Noun-Derived class seems to cover Vender’s lowest  $A_1$  classes, and Vendler’s Measure and everyone else’s Size fit at the top of what everyone seems to agree is a sub-hierarchy. Above Size, Hill, Brown, and Danks and Glucksberg put only one class, Hill’s “other”, Brown’s “characteristic” – which might as well be “adjective” for all it tells us –, and Danks and Glucksberg put only the possessive. This is where Vendler separates the  $A_2 - A_9$  from the sub-hierarchy of the  $A_1$ ’s.

In summary, we have seen descriptions of ordered adjective classes from four to 22 levels. The classes are divided mostly semantically, with some morpho-syntax such as that from Vendler. The morpho-syntax is arguably a stand-in for semantic properties, however. These descriptions are relatively consistent across proposals. We have also seen several attempts to explain psychologically, semantically, or syntactically how it is that these orders came about. Vendler gives a transformational grammar, but this falls short explanatorially for two reasons. For one, it relies on stipulated orders of transformations, which makes it equivalent to a stipulated order on adjective class. Second, transformational grammars have been shown to be far too powerful for human syntax, and therefore poor candidates for real human grammars (Peters and Ritchie, 1973). Semantic/psychological explanations such as intrinsicness, definiteness, and specialisation are inadequate.

Sproat and Shih (1988) propose *apparentness* as a cognitive criterion for adjective ordering. An adjective is more apparent than another if it requires less and/or less high-level computation. For instance, determining whether a car is red requires only that one determine whether a large portion of it is red. The computation of colour perception may be complex, but it is at least low-level. To decide if a car is large, on the other hand, one must decide it's a car, perceive its size, and compare its size to other cars.

Like the other criteria we have considered in this section, this works pretty well for certain pairs of classes, but fails for others, such as shape vs. colour. Does it really require more computation to decide if an object is round than to decide if it is red?

However, the authors make much more convincing use of a concept from Kamp (1975), that of *predicativeness*.

**Definition 2.8.1.** An adjective A is *predicative* if the following syllogism holds for all X,Y,Z:

All Xs are Ys  
Z is an A(X)

Therefore Z is an A(Y)

Adjectives of shape, colour, and provenance are predicative, while quality and size adjectives are not.

- (41)
- a. All mice are mammals  
Freddy is a **white** mouse  
✓ Therefore Freddy is a white mammal
  - b. All tables are pieces of furniture  
This is a **square** table  
✓ Therefore this is a square piece of furniture
  - c. All vases are ornaments  
This is a **Chinese** vase  
✓ Therefore this is a Chinese ornament
  - d. All mice are mammals  
Freddy is a **large** mouse  
✗ Therefore Freddy is a large mammal
  - e. Agatha Christie Novels are literature  
*The Murder of Roger Ackroyd* is a **good** Agatha Christie novel  
✗ Therefore *The Murder of Roger Ackroyd* is good literature

If entailment is well-defined then so is predicativeness, giving this proposal a substantial advantage over the ones considered heretofore. Moreover, it explains a large division in adjective classes, and one that [Sproat and Shih \(1988\)](#) claim accounts for the strongest judgements. Non-predicative adjectives precede predicative adjectives. This is the same division noted above: Quality/Characteristic/Other/ $A_3$ - $A_9$  + Size/ $A_2$  vs everything else/ $A_1$ . There are still unexplained divisions within the large classes of predicative and non-, but this at least does seem to hold. Indeed, I suggest that it is this distinction that researchers have in mind when they propose notions such as “apparentness”,

“specificity”, “intrinsicness”, and “definiteness”.

As evidence for strength of judgement, the authors note that when judgements are weaker, phonological weight can sometimes allow ordering inversions, such as heavy *beautiful*, a quality adjective, appearing after, as well as before, lighter *large*:

- (42) a. A good large house  
b. %A large good house  
c. A beautiful large house  
d. A large beautiful house

However, this inversion is much worse when mixing predicative and non-predicative adjectives such as non-predicative *humongous* and predicative *red*. The non-predicative adjective stays before predicative *red* despite it being much longer.

- (43) a. A humongous red peach  
b. \*A red humongous peach

Chinese offers interesting evidence for the primacy of the distinction between predicative and non-predicative adjuncts (Sproat and Shih, 1988). Direct modification of nouns (without the complementiser *de*) shows ordering like in English, but this sort of modification allows only up to two adjectives. The first must be non-predicative and the second predicative.

- (44) a. hao hong pan-zi  
good red plate  
'a good red plate'  
QUALITY > COLOUR = NON-PRED > PRED  
b. \*hong hao pan-zi  
red good plate

- c. hao yuan pan-zi  
good round plate  
QUALITY > SHAPE = NON-PRED > PRED
- d. \*yuan hao pan-zi  
round good plate
- e. xiao hong pan-zi  
small red plate  
SIZE > COLOUR = NON-PRED > PRED
- f. \*hong xiao pan-zi  
red small plate
- g. \*xiao yuan / yuan xiao pan-zi  
small round / round small plate  
\*SIZE + SHAPE = \*PRED + PRED
- h. \*yuan hong / hong yuan pan-zi  
round red / red round plate  
\*SHAPE + COLOUR = \*PRED + PRED
- i. \*hao xiao / xiao hao pan-zi  
good small / small good plate  
\*QUALITY + SIZE = \*NON-PRED + NON-PRED

Predicative-non-predicative sequences are possible, but they require a different construction, which [Sproat and Shih \(1988\)](#) argue is likely a relative clause, and therefore not subject to the same ordering restrictions.

- (45) a. hao-de xiao pan-zi  
good-DE small plate  
QUALITY-DE > SIZE
- b. xiao-de hao pan-zi  
small-DE good plate  
SIZE-DE > QUALITY

It may be, then, that this semantic property of predicativeness determines a partial order on adjective classes. Languages like English (and unlike Chinese) that allow multiple

modifiers of the same class are subject to an additional, and it would seem cognitively and semantically undefinable, ordering within these two classes. However, even this semantic division explains only the division into the classes, not their relative order: why do predicative adjectives occur farther from the noun, rather than closer? Worse, it is not clear that entailment is indeed a well-defined property for human language (Putnam, 1970; Etchemendy, 1990, e.g.).

Despite a great many attempts to account for adjective ordering extra-syntactically, none thus far are convincing. Although an account may indeed be still in our future, I find the likelihood high enough that a purely syntactic mechanism is needed to propose a general schema for this in chapter 4.

### 2.8.1.1 Cross-linguistic evidence of adjunct ordering

Cinque (1994) proposes that there is a cross-linguistically universal unmarked order for adjectives. He gives two separate hierarchies, one for event nominals and one for object-denoting nominals.

- (46) a. speaker-oriented subject-oriented manner thematic event  
 b. quality size shape colour nationality object

He provides the following examples from Italian:

- (47) a. le sue due altre probabili goffe realizzazioni immediate  
 his two other probable<sub>sp-orien</sub> clumsy<sub>sub-orien</sub> reactions<sub>N</sub> immediate<sub>manner</sub>  
 alla tua lettera  
 to your letter  
 ‘His two other probable clumsy immediate reactions to your letter’<sup>12</sup>

---

<sup>12</sup>or perhaps the English should be *...immediate clumsy*; the Italian ordering does not make predictions about the English, as *immediate* is the sole post-nominal adjective here.



- b. Suoi due altri bei grandi quadri tondi grigi  
His two other beautiful large paintings round gray  
'His two other beautiful large round grey paintings'

Hetzron (1978) reports that generally, when languages do have a consistent order for adjectives, pre-nominal adjectives are ordered as in English, and post-nominal adjectives have a mirror-image ordering.

- (48) schoener grosser roter Ball  
beautiful big red ball  
'beautiful big red ball' German

- (49) a. bola reah besar tjantik  
ball red big beautiful  
'beautiful big red ball' Bahasa Indonesia

- b. ma daam may  
dog black big  
'big black dog' Thai (Sproat and Shih, 1988)

Cinque argues that the mirror-image could be in some cases a rightward-modifier parameter setting (perhaps by “modifier” he means specifier, as his claim is that these adjectives are specifiers) and in other cases, such as Italian, a matter of secondary predication. In (50), the mirror-image order of post-nominal adjectives can be explained if at least one of the adjectives is a secondary predicate, where the adjective is not part of the usual stack of adjectives but is rather part of a separate subclause. In such a case, the outer adjective at least would not be able to intervene between the noun and a PP complement of the noun. This is what is found.<sup>13</sup>

---

<sup>13</sup>It is not made clear that this PP is indeed a complement, and if it is how anything can intervene between it and the noun. Indeed, in English the PP would be an adjunct, given the grammaticality of (i-b), with *one*-substitution for NP.

- (i) a. The car for racing  
b. The one for racing

(50) Italian

- a. una macchina rossa bellissima  
a machine red beautiful  
'a beautiful red car'
- b. \*una macchina rossa bellissima da corsa  
a machine red beautiful for racing  
'a beautiful red car for racing'
- c. una macchina rossa da corsa, bellissima  
a machine red for racing, beautiful
- d. una macchina da corsa, rossa, bellissima  
a machine for racing, red, beautiful
- e. una macchina da corsa, bellissima, rossa  
a machine for racing, beautiful, red

In a footnote, he gives Kayne's analysis, which is of roll-up movement, wherein the adjective moves out of the noun phrase and the remnant moves up above it, yielding an inverted order as this pair of movements is repeated throughout the whole phrase.

Cinque (1994) proposes that adjectives are specifiers of functional heads between N and D; he does not however give any suggestions as to what these functional heads might be. His arguments are as follows:

First, adjunction is generally thought to be free, and these adjectives are not freely ordered. Second, he claims there is a "clear limit on the number of non-coordinated attributive APs within DP (apparently not exceeding six or seven)." (Cinque, 1994, pp.96). He gives no evidence for this, however. Third, adjunction is (arguably) to the left or right, while specifiers are (arguably) only to the left. These adjectives are (again, arguably) always to the left. That the adjectives are underlyingly to the left is in fact Cinque's primary interest here, and he explains the adjectives sometimes being on the right with head movement of N to some functional head between N and D.

I find none of these arguments very convincing. The first depends on adjunction being fully free; I argue in chapter 4 that it can be defined to incorporate order. Second,

I do not see a clear limitation on the number of adjectives. Consider the repeatability for emphasis in some languages, including English:

(51) Look at that big, big, big, big ... green tree!

Outside of repetition, however, there is arguably a limit, something like 24 classes as in [Vendler \(1968\)](#). This is a much larger limit than Cinque's six or seven. Naturally, outside of repetition there will be a limit on the number of adjectives that can appear because the lexicon is finite. Whether the correct model is one like Cinque's, in which adjectives are specifiers, or one that uses a recursive adjunction rule, there must be some limit on the number of distinct adjectives that can appear in a noun phrase. Moreover, a model like the one I propose in [Chapter 4](#), if taken to require that sequential adjectives move strictly up the Cinque hierarchy (rather than allowing sequential adjectives at the same level in the hierarchy) would account for a limit on the number of adjectives without requiring that they be specifiers.

The third argument assumes specifiers are strictly leftward and adjuncts are free; this need not be so, and indeed the author suggests what seem to be rightward specifiers as an explanation for some mirror-image adjective orders: mirror-image orders are accounted for by the very same hierarchy as those in a "regular" order, and no additional machinery is introduced, leading me to conclude they must be rightward specifiers. Finally, unlike with adverbs, there is no evidence for independently-motivated functional heads to act as hosts for these adjective specifiers.

[Sproat and Shih \(1988\)](#) provide data from Greek, Kannada, Thai, and Mokilese in support of the QUALITY > SIZE > SHAPE > COLOUR > MATERIAL > PROVINENCE (partial) hierarchy. I provide here a sampling. All orders given are preferred over all other possible orders.

(52) Greek

- a. to mikro kokkino kineziko vazo  
the small red Chinese vase
- (53) Kannada
- a. dhɔɔɔɔᵛ gungᵛ nili goli  
large round blue marble
- (54) Thai: post-nominal direct modifiers are the mirror-image of English
- a. maa dam yai / \*maa yai dam  
dog black big / dog big black  
'big black dog'
- (55) Molikese: post-nominal direct modifiers are the mirror-image of English
- a. did sakai koro:ro:y  
wall stone white-DET  
'this white stone wall'
- b. mwək səl pwu:wu:sso  
cup black round-DET  
'this round black cup'
- c. pwo:la wa:ssa siksikko  
ball red small-DET  
'This small red ball'

Sproat & Shih's order matches the ones we've seen so far except that it reverses prominence and material, relative to [Vendler \(1968\)](#).

More evidence for cross-linguistic consistency comes from [Hetzron \(1978\)](#). The following six languages have the same ordering on evaluative, size, and colour adjectives, and have pre-nominal adjectives.

<b>English</b>	beautiful	big	red	ball
<b>German</b>	schoner	grosser	roter	Ball
<b>Hungarian</b>	szép	nagy	piros	labda
<b>Polish</b>	piekna	duza	czerwona	pilka
<b>Turkish</b>	guzel	buyuk	kirmizi	top
<b>Hindi</b>	mudar	bəsa	lal	gēd

Table 2.4: Six languages' adjective order in Hetzron (1978)

These languages have the mirror-image order, with post-nominal adjectives:

- (56) tupe qermeze bozorge qasangi  
ball red big beautiful  
'beautiful big red ball' Persian
- (57) a. exte zuri txiki polit bat  
house white little pretty a  
'a pretty little quite house' Basque
- b. soileko gorri zar motz bat  
dress red old ugly a  
'an ugly old red dress'

These languages with some adjectives preceding and some following the noun, like French, are consistent with the English ordering, but relative closeness to the noun of *big* and *red* cannot be determined.

<b>French</b>	un	joli	gros	ballon	rouge
<b>Italian</b>	una	bella	grossa	palla	rossa
<b>Ladin</b>	una	bella	granda	balla	cotchna
	a	beautiful	big	ball	red

Table 2.5: Three languages' pre- and post-nominal adjective orders in Hetzron (1978)

Hetzron also finds a few languages that have apparently free adjective order: Somali, Kurdish, Arabic, and Hebrew. However, all of these languages give reason to suspect that

these constructions do not belong to the class of stacked adjectives I am interested in. Arabic and Hebrew have definite articles on all of the adjectives, and usually have at least one conjunction, indicating these are of the conjoined list variety, which are expected to be free, as in *the red, big, and pretty ball*. Kurdish requires conjunction if the DP is indefinite, though not when it is definite, leaving open the possibility that the definite DP has truly unordered stacked adjectives. Finally, Somali requires conjunctions at least every two adjectives, making it likely these too are of the conjoined list variety.

Hetzron suggests the underlying adjective ordering rule is from subjective to objective, with adjectives whose veracity can be objectively verified closer to the noun. This is at least the best-argued proposal I have found. Material and origin are very objective, colour is objective but its classification is not, and so on. At the outer end are the “quality” or “evaluative” adjectives like beautiful which are clearly in the eye of the beholder.

On a broad scope, this criterion is a relatively good one. It is true that adjectives like *pretty* and *kind* are more subjective than *red* and *round*. However, the distinctions at finer levels are hard to argue for, particularly for them to be clear enough to people in general that they naturally order their adjectives from least to most objective with any consistency. For example, how much more subjective is shape than colour? Either order could be argued for.

The bids to explain adjective order in terms of extra-linguistic facts that I have surveyed here are so far unsuccessful. The best candidates in my mind are predicativeness and subjectivity. However, even these at best divide adjective classes into broad supersets, and fail to explain the highly articulated classes that for example Vendler observes. Moreover, even if we had a good extra-linguistic definition of the ordering of classes, how do we explain their order relative to the noun? What is it about an adjective that is the most or least *whatever* that makes it stay closest to the noun?

I speculate that the order of semantic classes of adjectives is built directly into UG. Alternatively, perhaps some ordering of larger supersets of classes (say, predicative and

non-predicative) is built in, and the ordering of adjective classes within the supersets is an historical remnant. This latter hypothesis predicts that there should be language families that, like Chinese and English, strictly order the superclasses of predicative and non-predicative adjectives (or whatever distinction we claim is built in) but which are like English in allowing adjectives to stack within the larger classes, and are unlike English in that they have an entirely different default order of subclasses. To my knowledge there is currently little or no evidence one way or the other.

Appendix B contains a table with an approximation of the alignment of the various adjective orders discussed in this section.

### 2.8.2 Adverb Order

Cinque (1999) is the classic work on the universality of adverb ordering across languages. Cinque argues that all languages have an underlying adverb order for non-VP-internal adverbs, and that it is best explained by their not being adjuncts at all, but rather specifiers of functional heads projected on the spine above the verb.

Examples (58) to (63), copied from Cinque (1999), illustrate the precedence of *always* over *completely* in English, Norwegian, Bosnian/Serbo-Croatian, Hebrew, Mandarin, and Albanian.

(58) The snow **always** **completely** covers my car

(59) Norwegian

- a. De forstår enda ikke **alltid** **helt** hva jeg snakker om  
they understand still not always completely what I talk about  
‘They do not yet always completely understand what I’m talking about’
- b. \*De forstår enda ikke **helt** **alltid** hva jeg snakker om  
they understand still not completely always what I talk about

(60) Bosnian/Serbo-Croatian (**always** > just > almost > **completely**)

- a. Kad god ga sretnem, on se **uvijek** **upravo** vraća iz grada  
 whenever him I meet he REFL always just returns from town  
 ‘Whenever I meet him, he has always just returned from town’
- b. \*Kad god ga sretnem, on se **upravo** **uvijek** vraća iz grada  
 whenever him I meet he REFL just always returns from town  
**always > just**
- c. Ja sam **upravo** **gotovo** pao  
 I am just almost fallen  
 ‘I have just almost fallen’
- d. \*Ja sam **gotovo** **upravo** pao  
 I am almost just fallen  
**just > almost**
- e. Ja sam ga **gotovo** **potpuno** zagboravio  
 I have him almost completely forgot  
 ‘I have almost completely forgotten him’
- f. \*Ja sam ga **potpuno** **gotovo** zagboravio  
 I have him completely almost forgot  
**almost > completely**

(61) Hebrew<sup>14</sup>

- a. Hu **tamid** hores **legamrey** 'et ma še-hu ose  
 he always destroy.pres.masc.3sg completely acc what that-he do.pres.masc.3sg  
 ‘He always completely destroys what he does.’
- b. \*Hu **legamrey** **tamid** hores 'et ma še-hu ose  
 he completely destroy.pres.masc.3sg always acc what that-he do.pres.masc.3sg

(62) Mandarin (**always > just > completely**)

- a. mei ci wo pengjian ta, ta **zongshi** **ganggang** cong guowai huilai  
 every time I meet him, he always just from abroad return  
 ‘Every time I see him, he’s always just come back from abroad’
- b. \*mei ci wo pengjian ta, ta **ganggang** **zongshi** cong guowai huilai  
 every time I meet him, he just always from abroad return  
**always > just**
- c. wo **ganggang** **wanquan** wang-le ta-de  
 I just completely forgot his address

<sup>14</sup>Gloss from Hadas Kotek, p.c.



'I just completely forgot his address'

- d. \*wo **wanquan** **ganggang** wang-le ta-de  
I completely just forgot his address  
**just** > **completely**

(63) Albanian

- a. Ai nuk i kupton **gjithnë tërësisht** vërejtjet  
he not them understands always completely remarks  
'He does not always understand the remarks completely'
- b. \*Ai nuk i kupton **tërësisht** **gjithnë** vërejtjet  
he not them understands completely always remarks

Cinque argues that this ordering is best explained by adverbs occurring as specifiers of functional heads above the verb, such as Aspect and Tense. He argues from succinctness – a grammar with specifiers and not adjuncts is more parsimonious than one with both – but more interestingly, he presents evidence that there are functional heads between the adverbs. For example, he uses active past participle movement in Italian to argue for several functional heads to which the participle could have head-moved.

- (64) a. Da allora, non hanno **rimesso** di solito mica più sempre completamente  
since then, not have put of usual little more always completely  
tutto bene in ordine.  
everything well in order  
'Since then, they haven't usually not any longer always put everything well in order'
- b. Da allora, non hanno di solito **rimesso** mica più sempre completamente  
since then, not have of usual put little more always completely  
tutto bene in ordine.  
everything well in order
- c. Da allora, non hanno di solito mica **rimesso** più sempre completamente  
since then, not have of usual little more put more always completely  
tutto bene in ordine.  
everything well in order
- d. Da allora, non hanno di solito mica più **rimesso** sempre completamente  
since then, not have of usual little more put always completely

tutto bene in ordine.  
everything well in order

- e. Da allora, non hanno di solito mica più sempre **rimesso** completamente  
since then, not have of usual little more always put completely  
tutto bene in ordine.  
everything well in order
- f. Da allora, non hanno di solito mica più sempre completamente **rimesso**  
since then, not have of usual little more always completely put  
tutto bene in ordine.  
everything well in order

Following Pollock (1989) in assuming adverbs are normally still and other things move around them, he proposes that the availability of these inter-adverbial slots for the verb is evidence for heads to which the verb may move. He compares this analysis to one in which adverbs are not in fixed positions, and instead freely adjoin to (or are multiple specifiers of) the past participle, which occupies a fixed position. These adjunctions or specs are on both the left and right. An analysis with freely adjoined adverbs cannot account for the ordering of the adverbs, in particular for (65) to (67). The ungrammaticality of (67-b) cannot be explained by the location of the adverbs *mica* ‘little’ and *più* ‘more’ relative to the heads, since in (65-b) we see *mica* following *mangiato*, and in (66-a), *più* appears between *hanno* and *mangiato*. The problem is also not that both adverbs are present: (67-a) has both adverbs; they are just in the other order. The remaining explanation for the ungrammaticality of (67-b) is the relative order of the two adverbs.

- (65) a. non hanno mica mangiato  
not have little eaten  
‘They haven’t not eaten’
- b. non hanno mangiato mica  
not have eaten little
- (66) a. non hanno più mangiato  
not have more eaten  
‘They haven’t eaten any longer’

- b. non hanno mangiato più  
not have eaten more
- (67)
- a. non hanno mica mangiato più  
not have little eaten more  
'They haven't not eaten any longer'
  - b. \*non hanno più mangiato mica  
not have more eaten little

Pollock rejects a filter to prevent the wrong adverb order, stating it would recapitulate the ordering fact *mica* > *più*. In fact, in chapter 4 I propose a model that essentially implements such a filter, but which also has the possibility for functional heads. No recapitulation is necessary, as the same hierarchy accounts for adverb ordering whether or not the verb moves between them, and the same hierarchy can account for functional heads as well, without requiring that we do away with adjunction.

### 2.8.3 Ernst 2002

Ernst (2002) argues against Cinque's functional heads model in favour of a semantic model. His main arguments include data such as (68), which are similar to Cinque's Italian data in (64) above. Here, he notes that English clausal predicational adverbs like *wisely* can occur anywhere before the verb. The "clausal reading" is that it was wise of them to hang back.

- (68)
- a. **Wisely**, they had been hanging back whenever the pendulum swung near.
  - b. They **wisely** had been hanging back whenever the pendulum swung near.
  - c. They had **wisely** been hanging back whenever the pendulum swung near.
  - d. They had been **wisely** hanging back whenever the pendulum swung near.

Just as Cinque used a sequence of adverbs between any of which a single verb could occur, Ernst uses a series of verbs between any of which a single adverb can occur. Cinque

argues that the adverbs stay still while the verb moves around them; Ernst argues that the multiple auxiliaries cannot all head-move around the adverb.

If *wisely* stays in place, then its place must be above the auxiliary verbs' base positions, since in for example (68-b), *wisely* precedes the auxiliaries *had* and *been*, and head-movement is always up and to the left. However, the auxiliaries must also head-move up past the adverb in (68-d). Unless they form a single head together, this is not possible because of the Head Movement Constraint (Travis, 1984). Only the higher of the two heads, *had*, would be able to raise.

Ernst's alternative is that the adverb's many positions can be explained by the semantic types of the constituents it can be sister to. *Wisely* takes as argument an event of a certain type; as long as the predicate it is sister to is of that type, the adverb can be there.

Ernst makes a great many more arguments against the purely syntactic account of Cinque (1999) and for his scope-based theory, which go far beyond the scope of this thesis.

Ernst's account may indeed be the correct one for adverbs, making a purely syntactic account of adjuncts of the verbal spine irrelevant. Despite this, in chapter 4, I make just such a proposal, for two reasons. First, Ernst's semantic account of adjuncts does not extend to most adjectives: there is no semantic type difference between a noun modified by, say, a colour adjective and one modified by a shape adjective, yet shape adjectives precede colour adjectives. As such, I argue that even if there is no arbitrary syntactic ordering whatsoever on adverbials, there is still one on adjectives. Second, my intention in chapter 4 is less to propose the right model for adjuncts than to examine what a model would have to look like if we were to account for adjunct behaviour in the syntax: what sort of syntactic model could account for both the optionality and the ordering of adjuncts?

## 2.9 Selectability

Continuing with properties of adjuncts, some phrase classes that tend to be adjuncts – CPs, PPs, and perhaps adjectives – can also occur as arguments.

CPs can of course be subjects and objects:

(69) [That he left early]<sub>CP</sub> made us think [that he was bored]<sub>CP</sub>

PPs can be arguments of nouns and verbs:

(70) a. The destruction [of the spaceship]<sub>PP</sub>  
b. Put it [on the table]<sub>PP</sub>

This means that it is not only the class of word that determines whether or not a phrase is an adjunct. Any model of adjuncts needs to allow space in the grammar for these categories to be selected.

Adjectives may be selectable by verbs like *be*, *seem*, *look*, for example:

(71) a. She is tall.  
b. She seems nice.  
c. She looks funny.

It may be instead that the verb is selecting a small clause; if so, then depending on the internal structure of a small clause, the adjective may have to be selected by its sister.

## 2.10 Adjuncts of adjuncts

That a phrase is an adjunct does not preclude adjunction to it. Adverbs and intensifiers can adjoin to adjectives and adverbs.

(72) English

- a. The [remarkably<sub>adv</sub> red] balloon
- b. He left [surprisingly<sub>adv</sub> quickly<sub>adv</sub>]
- c. He left [really<sub>int</sub> quickly<sub>adv</sub>]

(73) Dutch

- a. de [heel snelle] fiets  
the [very<sub>int</sub> fast] bike  
'the very fast bike'
- b. Hij is verassend snel weg gegaan  
he is surprisingly quickly away gone  
'He left surprisingly quickly'
- c. Hij is heel snel weg gegaan  
he is very quickly away gone  
'He left very quickly'

However, not all adverbs can modify adjectives. (74-b) to (74-g) are unacceptable.<sup>15</sup>

- (74)
- a. His exceptionally big car
  - b. \*His early big car
  - c. \*His fast big car
  - d. \*His soon big car
  - e. \*His quickly big car
  - f. \*His again big car

---

<sup>15</sup>Data from Carson Schutze, p.c.

- g. \*His carefully big car

These unacceptable phrases could be out for syntactic reasons, but I suggest the reasons are semantic: it is simply hard to find a meaning for them. Carefully chosen adjectives and nouns can improve matters, though they do not make any of these perfect, with the possible exception of *quickly*.

- (75) a. His exceptionally red car
- b. \*His early sleepy baby
- c. ?His fast obsolete cell phone
- d. (?) His quickly obsolete cell phone
- e. (?) His soon cranky baby
- f. \*His again cranky baby
- g. ?His carefully full glass

Most of these marginal adverbs are fine with participial adjectives, but it is easy to argue that the modification is in fact of the verb, before it is converted into an adjective. *Early* and *again* seem to be exceptions. *Early* seems to want to be “more adverbial”, as in “earlily”, which isn’t a word. *Again* is perhaps somehow blocked by the dramatically better *once again*.

- (76) a. ??His [early planned] vacation
- b. His fast planned vacation
- c. His soon planned vacation
- d. His quickly planned vacation
- e. ?His again closed door
- f. His carefully planned vacation

There may indeed be something much deeper happening with these exceptions than interpretability; this I will leave open. Instead, I focus on the myriad adverbs that *can* modify adjectives. The order of examples in (77) follows Cinque's hierarchy's of adverbs in English (Cinque, 1999, pp.106). Note most are perfect, especially with a felicitous choice of context.

- (77)
- a. A frankly stupid idea
  - b. A fortunately clever idea
  - c. An allegedly clever idea
  - d. A probably stupid idea
  - e. A once handsome man
  - f. My then current boyfriend
  - g. A perhaps unwise plan
  - h. A necessarily risky proposal
  - i. A possibly risky proposal
  - j. My usually playful cat
  - k. ?An again functional TV
  - l. An often functional TV
  - m. An intentionally obsolete cell phone
  - n. A quickly obsolete cell phone
  - o. An already obsolete cell phone
  - p. A no longer viable plan
  - q. A still viable plan
  - r. An always friendly dog
  - s. A just obsolete cell phone
  - t. ?A soon obsolete cell phone
  - u. A briefly sunny sky
  - v. A characteristically fancy car



- w. An almost fancy car
- x. A completely red apple

The lowest adverbs seem to be the worst:

- (78) a. \*A well old computer
- b. A well-planned vacation
- c. ?A fast obsolete cell phone
- d. \*An early tired baby

When these adverbs are stacked, they follow the same ordering restrictions as when the adverbs modify verbs. In terms of their interpretation, the adverbs behave as they do for verbal modification, rather than as adjectives do for nouns: they modify the whole phrase, rather than the head alone. For instance, in (79-a), the bag is simply both big and red. It need not mean that as red bags go, it is big. In this sense, both *red* and *big* modify *bag*. In both (79-b) and (79-d), on the other hand, we're being frank in suggesting that the riskiness of the plan is necessary. That is, in (79-b), *frankly* is modifying *necessarily risky*, not just *risky*, and in the sentence in (79-d), it is modifying the whole sentence. (79-c) sounds strange because the adverbs are in the wrong order. This is a distinction I will not address further in this work.

- (79) a. A big red bag
- b. A frankly necessarily risky plan
- c. ?A necessarily frankly risky plan
- d. Frankly, that is necessarily a risky plan.

## 2.11 Unordered adjuncts

There is a reason adjuncts are often modelled as unordered: some in fact are. In particular, PP adjuncts are not generally ordered with respect to each other.

- (80) a. Simon ate the cake with chopsticks on his birthday  
b. Simon ate the cake on his birthday with chopsticks

(81) Dutch

- a. Simon heeft de taart met chopsticks op zijn verjaardag gegeten  
Simon has the cake with chopsticks on his birthday eaten  
'Simon ate the cake with chopsticks on his birthday'
- b. Simon heeft de taart op zijn verjaardag met chopsticks gegeten  
Simon has the cake on his birthday with chopsticks eaten  
'Simon ate the cake on his birthday with chopsticks'

(82) French

- a. Simon a mangé le gâteau avec des baguettes lors de  
Simon has eaten the.MASC cake with INDEF.PL chopsticks during  
sa fête  
POSS.FEM birthday  
'Simon ate the cake with chopsticks on his birthday'
- b. Simon a mangé le gâteau lors de sa fête  
Simon has eaten the.MASC cake during POSS.FEM birthday with  
avec des baguettes  
INDEF.PL chopsticks  
'Simon ate the cake on his birthday with chopsticks'

Another place we find sometimes find unordered adjuncts is when the adjuncts themselves are modified. Stabler (p.c. 2016) points out that the following seem to be equivalent:

- (83) a. the very big extremely bad wolf

- b. the extremely bad very big wolf

It is not clear to me whether comma or list intonation is required here, particularly in (83-b), where in principle *big* and *bad* are out of order. If so, this is simply another example of adjective sequences pronounced with comma intonation, indicating the out-of-order adjunct may have moved, or list intonation, indicating each individual adjunct is not in fact adjoined to the noun, but rather gathered into a constituent as a list and that whole constituent is the adjunct. In either case we would predict unordered adjuncts.

If, on the other hand, this is in fact usual adjunction with a constituency something like that in Fig. 2.10, at this point I can't even speculate as to what is going on.

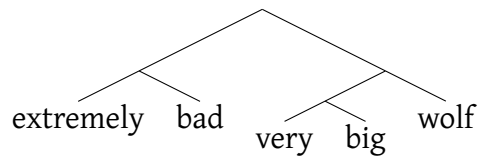


Figure 2.10: Constituency of ordinary adjunction of modified adjuncts

## 2.12 Iterability

Adjuncts are classically iterable, meaning that once you have one adjunct you may also have more. We have seen that this iterability is at least subject to ordering constraints. For instance, once a sentence has a high adverb like *frankly*, it usually can't take any more before/above it.

- (84) a. I frankly don't give a damn  
b. ??She secretly frankly doesn't give a damn

Some languages do not allow “direct” adjective stacking, in the sense of Sproat and Shih (1988) – they are not relative clauses or conjoined or listed or secondary predicates or

anything of that nature. For example, Tagalog<sup>16</sup> does not allow adjectives to stack in any order. One adjective may go before the noun and one may go after (85-d). In order to have more than one adjective on the same side of the noun, Tagalog employs conjunction as in (85-g) and (85-h)

- (85) a. ang malaki-ng babae / ang babae-ng malaki  
 NOM big-LK woman / NOM woman-LK big  
 ‘the big woman’
- b. ang masaya-ng babae / ang babae-ng masaya  
 NOM happy-LK woman / NOM woman-LK happy  
 ‘the happy woman’
- c. ang matanda-ng babae / ang babae-ng matanda  
 NOM old-LK woman / NOM woman-LK old
- d. ang malaki/masaya-ng babae-ng masaya/malaki  
 NOM big/happy-LK woman-LK happy/big
- e. \*ang malaki-ng matanda-ng babae-ng masaya  
 NOM big-LK old-LK woman-LK happy  
 Intended: ‘the big happy old woman’
- f. \*ang malaki-ng babae-ng matanda-ng masaya  
 NOM big-LK woman-LK old-LK happy  
 Intended: ‘the big happy old woman’
- g. ang malaki-ng babae-ng matanda-ng at masaya  
 NOM big-LK woman-LK old-LK and happy  
 ‘the big happy old woman’
- h. ang matanda-ng at malaki-ng babae-ng masaya  
 NOM happy-LK and big-LK woman-LK old  
 ‘the big happy old woman’

Another kind of iterability is at the word level. Some languages allow at least some of their adjuncts to iterate, with the meaning of intensification. Commonly, this is possible

---

<sup>16</sup>Data from Seth Ronquillo, p.c.

NOM=nominative, which isn’t really right. The *ang*-marked DP has the semantic role the verb is voice-marked for, and is the only extractable DP. LK=linker

with intensifiers like *very* and *really*.

- (86) a. the really really really really big house  
b. I really really really really like her.
- (87) Man Kheili Kheili Kheili Kheili shokolat doost daram  
I really really really really chocolate have friend  
'I really really really really like chocolate' Persian<sup>17</sup>
- (88) a. naneun chokhollis-eul maeu maeu maeu maeu joh-ahanda  
I chocolate-by very very very very like  
'I really really really really like chocolate' Korean<sup>18</sup>  
b. aju aju keun ge  
very very big dog  
'A very very big dog' Korean<sup>19</sup>
- (89) Realmente es muy, muy, muy difícil  
really is very very very difficult  
It really is very very very difficult Spanish<sup>20</sup>
- (90) on bezhal ochen' ochen' ochen' ochen' bystro  
he ran very very very very fast  
'He ran very very very very fast' Russian<sup>21</sup>
- (91) Ik vind het heel heel heel heel leuk  
I find it very very very very very nice  
'I really really really really really like it' Dutch<sup>22</sup>

Less commonly, in my experience, will languages allow repetition of adjectives.

---

<sup>17</sup>Data from Setareh Safavi, p.c.

<sup>18</sup>Data from Yun Jung Kim, p.c.

<sup>19</sup>Data from Donghyun Kim, p.c.

<sup>20</sup>Data from José Maria Lahoz Bengoechea, p.c.

<sup>21</sup>Data from Alexandra Grabarchuk and Natasha Korotkova, p.c.

<sup>22</sup>Data from Floris van Vugt, p.c.

- (92) natulog ang (malaki malaki ... malaki) babae  
 sleep D (big big ... big) woman  
 'The big big ... big woman is sleeping/slept' Tagalog<sup>23</sup>
- (93) Gole rose ghermeze ghermeze ghermeze ghermez  
 The rose red red red red  
 'the red red red red rose' Persian<sup>24</sup>
- (94) krasnyy krasnyy krasnyy krasnyy yabloko  
 red red red red apple  
 'red red red red apple' Russian<sup>25</sup>

Korean allows adjective repetition only when the adjective list is of the conjoined type, connected by *-go* 'and'.

- (95) a. ppalgah-go ppalgah-go ppalgan sagwa  
 red-and red-and red apple  
 'a red red red apple' Korean<sup>26</sup>
- b. \*keun keun ge  
 big big dog  
 Intended: 'a big big dog' Korean<sup>27</sup>

It may well be that word-level repeating adjectives are always of a different sort than the stacked, direct modification. More research is required here.

Iteration of (non-intensifier) adverbs is apparently even rarer.

- (96) a. \*He suddenly suddenly suddenly sneezed (Intended: 'He very suddenly sneezed')

---

<sup>23</sup>Data from Seth Ronquillo, p.c.

<sup>24</sup>Data from Setareh Safavi, p.c.

<sup>25</sup>Data from Alexandra Grabarchuk and Natasha Korotkova, p.c.

<sup>26</sup>Data from Yun Jung Kim, p.c.

<sup>27</sup>Data from Donghyun Kim, p.c.

- b. \*Frankly frankly my dear, I don't give a damn.
- c. When dealing with a potential explosive device, you need to carefully, carefully remove the housing and gently, gently detach the red wire.<sup>28</sup>

## 2.13 Conclusion

This chapter provided an overview of the behaviour of adjuncts, and the literature surrounding them. In sum:

- Generally optional (section 2.1)
- Provide “additional information” to the meaning of the adjoinee (section 2.3)
- Can be obligatory, rarely (section 2.5)
- Require separate information from arguments in subcategorisation frames (section 2.6)
- Transparent to selection (section 2.7)
- Adjectives are very often ordered, but no purely semantic or psychological account suffices to explain the order (section 2.8.1)
- Adverbs are very often ordered, and semantic type may explain their order, at least as modifiers of the verbal spine (section 2.8.2)
- Adjunct categories are often also selectable (section 2.9)
- Adjuncts can themselves be adjoined to (section 2.10)
- Some adjuncts are unordered (section 2.11)

---

<sup>28</sup>Data from Carson Schutze, p.c.

- Many languages allow word-level iteration of intensifiers (section [2.12](#))
- Fewer language allow word-level iteration of adjectives (section [2.12](#))
- Word-level iteration of adverbs is at best extremely rare (section [2.12](#))



## CHAPTER 3

### Minimalism

Minimalism (Chomsky, 1993, 1995, etc.) is one of the syntactic frameworks currently being pursued by a considerable number of theoretical syntacticians whose goal is modeling the mental faculty of language. A number of computational linguists have been implementing various versions of it; this thesis uses Stabler (1997 etc)'s as a starting point. The program takes its name from the shift in focus from explanatory adequacy, the principal driving concern of previous approaches, most recently Government and Binding (Chomsky, 1981), to “simplicity, naturalness, elegance, and parsimony” (Hornstein et al., 2005, pp. 6). Natural language data being complex, a focus on explanatory adequacy naturally gives rise to complex proposals. If under this apparent complexity lies a unifying simplicity, a shift in focus to elegance may find it.

Minimalism begins with what must absolutely be in any theory of language: principally that it is recursive, has meaning, and has sound<sup>1</sup>.

However, minimalism also had available to it from the beginning all the observations and generalisations of the previous half-century. Bringing these two factors together in Chomsky (1995), minimalism in practice became a quite specific theory of feature checking, Merge, covert and overt Movement, and transderivational constraints such as Procrastinate. From here, naturally, researchers have formulated models of language phenomena within this framework, and the term *minimalism* today generally means something much more specific than a general approach to language that emphasises theoret-

---

<sup>1</sup>or hand movement, in the case of sign languages

ical elegance.

In this thesis, I take a middle ground in working within Stabler’s (1997 etc) Minimalist Grammars (MGs), which formalise the basic specifications given in Chomsky (1995) (formal feature checking, Merge, Move, interface conditions in a broad sense, and one constraint) but which makes no particular claims about more detailed questions such as *what  $\phi$ -features should be on T?*: it is general enough to handle whatever  $\phi$ -features one might want to assign to T.

Minimalist Grammars are mildly context-sensitive, putting them in the right general class for human language grammars. They are also simple and intuitive to work with. Another useful property is that the properties of well-formed *derivations* are easily separated from the properties of *derived structures* (Kobele et al., 2007). A number of grammars have been proposed with the same set of well-formed derivations, such as the string-generating grammar in Stabler and Keenan (2003) (Section 3.3.1), the tree-generating grammar in Stabler (1997) (Section 3.3.2), and the multidominant graph-generating grammar in Fowlie (2011). I provide an overview of the above grammars to illustrate the general principles of MGs and to provide the reader with a sense of the breadth of possible structures that can be formulated with the same basic architecture.

### 3.1 The Minimalist Program

Chomsky (1993) introduced a new approach to syntax, intended to simplify the systems in place at the time. In minimalism, there is essentially one operation, called *Merge*, which simply takes two things and puts them together. The atoms of the operation have, or possibly simply are, features. At least some of the features of the atoms are interpretable by the syntax, and drive the derivation.

Merge comes in two types, *internal* and *external* (Chomsky, 2004). In external merge, two distinct things are put together. In internal merge, one of the two things is a sub-

part of the other; this operation can also be thought of as *Move*, because you're taking something from one part of the structure and putting it in a new place.

In this description so far, I have said nothing about how the two things are put together. In *The minimalist program*, Chomsky suggests that the simplest choice is to make a set; for example if the two things being merged are the words *the* and *Firefly*, the result of merging them would be the set  $\{the, Firefly\}$ . This is not the only choice, of course. Since phrases tend to behave rather like one or the other of the two things merged to form them, Chomsky proposes (a notational variant of) an ordered pair, where the first is the “head”, or the label of the whole phrase,  $(the, Firefly)$ . Another option is a sequence, either *the Firefly* or *Firefly the*. We might get an (ordered or unordered) graph labelled by *the* and *Firefly*, as in (1-e) below. Nor are these the only possibilities for putting two things together.

- |     |    |   |              |
|-----|----|---|--------------|
| (1) | a. | $\{the, Firefly\}$  | Set          |
|     | b. | $(the, Firefly)$  | Ordered pair |
|     | c. | <i>the Firefly</i>  | Sequence     |
|     | d. | <i>Firefly the</i>  | Sequence     |
|     | e. |  | Graph        |

Similarly, if the atoms of the derivation are features F and G, we might make sets, sequences, or graphs as follows:

- |     |    |           |          |
|-----|----|-----------|----------|
| (2) | a. | $\{F,G\}$ | Set      |
|     | b. | FG        | Sequence |
|     | c. | GF        | Sequence |

d. 3



Graph

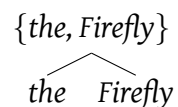
### 3.1.1 A note on trees and graphs

Formally, trees can be defined in several equivalent ways, and one is as a labelled graph. A graph is a set of things (call them *nodes*) and connections between some pairs of nodes (call them *edges*, usually represented graphically with lines). Nodes may have labels; for us these are usually the atoms of the derivation (for example, words) and possibly internal node labels (such as XP, or a set of features, or the name of the operation (internal or external merge) that applied).

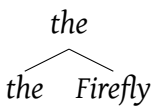
Trees are a special class of graphs, where there's one root node that has no mothers and dominates everything, and no node has more than one mother.

Trees are a notation that can be used to represent various things, not only the syntactic phrase markers familiar to most linguists. For example, set inclusion as in (1-a) above can be represented with the tree in (3). The lines mean “is a member of”, reading from bottom to top.

(3)



Indeed, the root label is determined by the information given by the leaves and the lines, so we can also just use the tree in (4-a), where daughters of the same node constitute a set. Such a tree represents the set-inclusion grammar mentioned above. The ordered-pair/head-labelling variant is in (4-b); this is a bare phrase structure tree common in mainstream Minimalist representations.

- (4) a.   
 b. 

We might also use a tree to represent rewriting with a rewrite grammar. For example, suppose we have a very simple phrase structure grammar with one rule,

$$\text{NP} \rightarrow \text{the Firefly}$$

Then we can make a tree representing “NP was rewritten as *the* followed by *Firefly*”. Such a tree is the familiar pre-minimalism phrase structure tree.

- (5) 

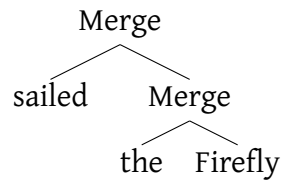
We can also draw a tree representing “*the* and *Firefly* were Merged”. Since it is a tree describing a derivation, this is called a *derivation tree*, and will be used a lot in this dissertation.

- (6) 

The derivation tree only represents “the operation called “Merge” applied to the thing we’re representing by *the* and the thing we’re representing by *Firefly*”. It does not tell us what the operation Merge does to *the* and *Firefly*. The output of the function Merge might be a set, a sequence, a tree, a graph, or something else entirely, depending on how we define Merge. This is simply a more readable notation for Merge(*the*, *Firefly*).

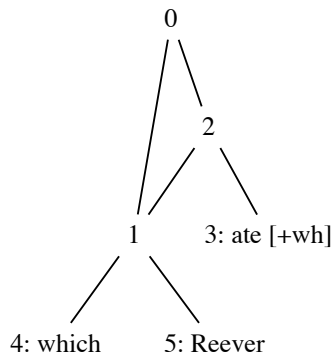
If the result of merging *the* and *Firefly* is then merged with something else, say *sailed*, we can write Merge(*sailed*, Merge(*the*, *Firefly*)) or draw:

(7)

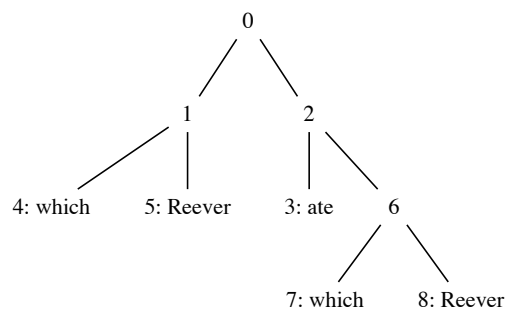


Not all graphs are trees. Another sort of graph useful for representing syntax is a *multidominant graph*, which is just like a tree, except nodes may have more than one mother. Multidominance can be used to distinguish between internal and external merge. In the copy theory of movement, these are indistinguishable without indexing; multidominance is a convenient and intuitive alternative.<sup>2</sup> In a graph, nodes are a set of things – here I’ve made them numbers – some of which may have labels.

- (8) a. Which Reever ate \_\_\_?  
b. Which Reever ate which Reever?



(a) Which Reever ate?



(b) Which Reever ate which Reever?

Figure 3.1: Graphs can distinguish internal and external Merge

<sup>2</sup>Normally both of these trees would have additional structure, for example an intermediate projection to which the subject would move (eg TP) but for simplicity I will often leave such projections out.

In (8-a), there is only one instance of *which Reeve*, but it moves, due to the *wh* feature on *ate*.<sup>3</sup> If we used copy movement, with no special labelling or indexing of copies, the result would be the tree in 3.1b. This is a problem, because this is the tree for (8-b). Multidominance, as in 3.1b, allows the derived structure to retain the information about each position of a constituent – in this case, the two positions of *which Reeve* – without confounding internal merge (move) with external merge of a new but similar element.

Crucially, there is a difference between using a tree to describe a derivation (“I merged these two things”) and using a tree to represent the thing derived (“I merged these two things and made *this tree*”).

It is clear that syntax is, in some sense, hierarchically organised. For example, in (9), *the* and *Firefly* belong together to the exclusion of *sailed*.

(9) The Firefly sailed.

This fact can be represented in a tree such as the one in (10).



However, what is not clear is whether it is enough to say that this hierarchy is true of the derivation, or if we also require that it be true of the structure that was made. That is, it is perhaps enough to know that *the* and *Firefly* were Merged, and then the result was Merged with *sailed*. This description of the derivation is hierarchically structured. Do you also need to build something that itself has hierarchical structure such as a tree, or can we get away with building, say, a sequence, with order determined by something in the definitions of Merge and Move?

---

<sup>3</sup>There would be more silent functional structure in these trees under a standard analysis, but this is the general principle: how are copies distinguished from coincidental multiple instances of the same string?

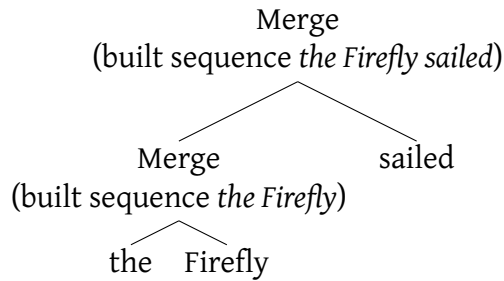


Figure 3.2: Derivation tree for *The Firefly sailed*

The answer to this question comes down to what exactly the derived structure is used for. In Minimalism, it is assumed that information must be given to the interfaces – the conceptual-intentional interface for meaning and the articulatory-perceptual interface for pronunciation. What exactly these interfaces need should govern the information available in the derived structure(s).

Ultimately, however, answering this question goes beyond the scope of this dissertation. Throughout, I will usually represent the derivation with trees, but the derived structure – the thing that was built by the derivation – as strings, simply for the sake of economy: strings take up little space on the page. It is also easy to represent both the derivation and the derived strings in an *annotated derivation tree*, but it’s much uglier to try to represent derived trees or graphs or baskets of puppies in an annotated derivation tree.

An annotated derivation tree has its internal node labels expanded to include more than just the name of the operation that applied. For example, in Figure 3.3, I represent the name of the operation, the derived string, and the remaining uncanceled features.



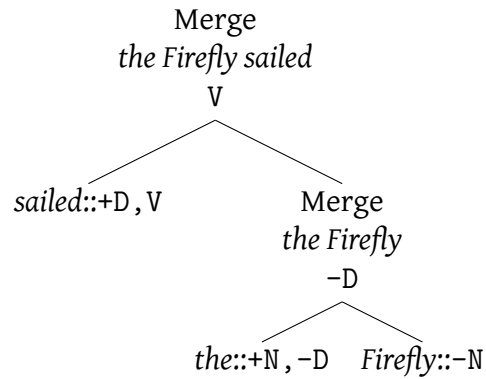


Figure 3.3: Annotated derivation tree

The diagram in figure 3.4 represents a possible derivation of the tree in (10) above. As in Figure 3.3 above, the internal nodes are labelled with the name of the operation, the derived structure (this time a tree, not a string), and the uncanceled features.

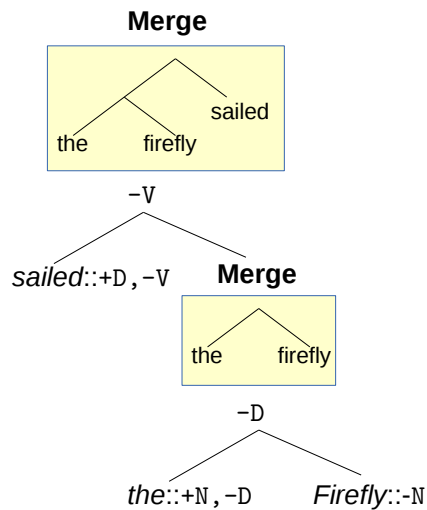


Figure 3.4: Annotated derivation tree. Derived trees are in yellow boxes.

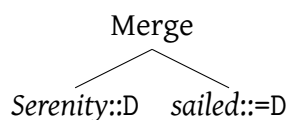
### 3.1.2 Features

We tend to think of features as properties of syntactic objects, made explicit. For example, a feature of *Firefly* might be Noun, and another might be singular. Still another feature might be its denotation, **Firefly**<sub>*e*</sub>, and perhaps **spaceship**<sub>*(e,t)*</sub>. The word is also pronounced, so perhaps it has features such as [+fricative] and [+labiodental].

However, in Minimalism this is not quite right: it's not that lexical items *have* features, but rather that they *are* bundles of features. In this sense, features are not properties of lexical items, but rather are the very content of the lexical items. Moreover, these bundles of features are probably not sets, since a set cannot have more than one of the same item. Rather, feature bundles must be multisets or sequences; this allows a feature such as “*I need a DP*” or [+DORSAL] to occur more than once.

At least some features drive the syntactic derivation. One way to model selection is to give matching, but opposite polarity, features to the selector and the selectee. For example, *sailed* can come with feature =D, meaning it needs a DP, and *Serenity* can come with feature D, meaning it is a DP. The matching features of opposite polarity license the application of Merge.

(11)



Note, however, that semantics plays a huge role in the availability of syntactic selection. For instance, arguably *Serenity* and *sailed* are possible sisters largely because the denotation of *sailed* is a function that takes things of type *e* as argument, and *Serenity* is of type *e*.

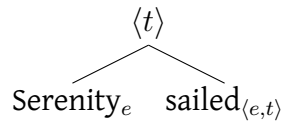


Figure 3.5: *sailed* takes *Serenity* as argument

Moreover, semantic properties of the sisters affect the coherence of the sentence. In (12), the difference in semantical acceptability is due to the fact that horses are the kinds of things that gallop, but spoons are not.

- (12) a. The horse galloped  
 b. #The spoon galloped

For these reasons, some syntacticians assume that Merge is not driven by features, but rather occurs freely, only resulting in interpretable sentences if sisters are interpretable as function and argument, or modification. In this thesis **I will always treat Merge as being driven by features.**

Move, on the other hand, is usually thought to be driven by features. The mover is modelled as having a feature, say *wh*, that tells it to move, and some higher lexical item has a matching feature that can only be satisfied if the *wh* mover moves up to re-merge with it.

For example, in Figure 3.6, *who* comes with a feature *-wh*, meaning “*I need to move on account of my being a wh-word*” and *did* comes with a feature *+wh*, meaning “*I need to move a wh-word into my specifier*”. I use a multidominant graph here because it is easy to see exactly what moved where.

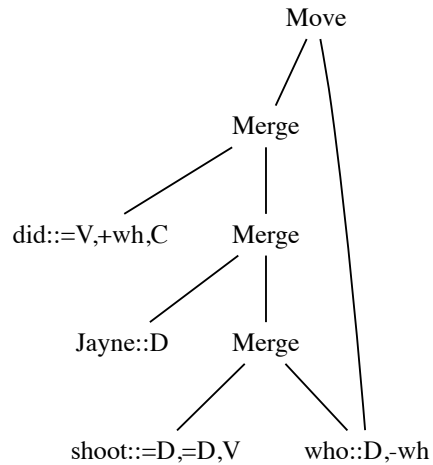


Figure 3.6: wh-movement

### 3.2 MGs: Informal introduction to a formalisation

This section is an informal description of the formalism I have been leading up to: [Stabler's \(1997\)](#) Minimalist Grammars. Since MGs are a formalism, we are forced to make decisions on the issues raised above. In MGs, Merge is feature-driven, features have two polarities (which we'll call positive and negative, but they can be thought of as interpretable and uninterpretable or black and white, or whatever you like), features are symmetrically cancelled, and the derivation is interpretable if all features but one are cancelled, and that feature is a special designated feature such as C (meaning we've built a CP). Move features are disjoint from Merge features. Negative Move features trigger the derivation to put aside the moving phrase and then re-attach it when the corresponding positive feature arises in the course of the derivation. For each Move feature, a set maximum of movers can be waiting to move at any one time (usually we say one, but any finite number will do).

Interestingly, one thing we do not need to decide on is what the grammar builds (a set, a sequence, a tree, a graph, etc.) A given grammar will need to make this decision, but the general idea of MGs is about the derivation (“take these two things and put them together”), not the generated structure (“...and make a set/sequence/tree/graph/basket of puppies”).

In Minimalist Grammars, the features enter the derivation through the lexicon, and guide the building of a grammatical sentence. Each lexical item (LI) comes with a stack of features that need to be checked. Checking happens either through the operation Merge, where an LI selects an element not already in the tree, or Move, where an LI selects an element already in the tree. In either case, the selector and selectee, or licensor and licensee, have matching features of opposite polarity.

For example, *wh*-movement is traditionally modelled with a *+wh* feature in C and a *-wh* feature on the mover. This triggers movement, and the phrase with the negative licensing feature moves to the specifier of the head with the positive licensing feature. When Move happens, the matching features are cancelled, or “checked”.

Selection in MGs is modelled just the same way. A verb that is subcategorised for a DP has a positive selectional feature, written =D, and a DP by definition has a negative selectional D feature, written just D. When V selects DP, the matching features are checked.

	Positive	Negative
Selectional (Merge)	=X	X
Licensing (Move)	+f	-f

Table 3.1: Features

An LI has a feature stack, so phrases are selected or moved in the order in which the features appear. Although grammars can be augmented with rules for determining what features may occur and in what orders, these feature stacks ultimately are simply stipulated by the definition of a particular lexicon; this reflects the arbitrariness of the

sign. Since checked features are deleted, new features are brought to the front of the stack with each check.

For example, take the following mini-lexicon:

- (sang, =D V): *sang* will select a D, and it's a V
- (she, D): *she* is a D

The operation **Merge** puts these two LIs together because the Selectional feature D is on the top of each stack, and they have opposite polarities. We can write this with a Derivation Tree, where the leaves are labelled with LIs and the internal nodes with the names of the operations that applied. For illustration I've notated the checked features with ~~strikethrough~~, but this is not standard. For a string-generating grammar, we can also (redundantly but conveniently) annotate the tree with the resultant string, and the remaining features. Notice that in this case, once D is checked, *she* has no features left, so we're just left with V from *sang*.



Figure 3.7: Unannotated and annotated derivation trees for *She sang*

**Move** is driven by a different set of features, Licensing features. Consider the following lexicon:

- (sang, =D V): *sang* will select a D, and it's a V
- (she, D): *she* is a D
- (who, D -wh): *who* is a D that will Move because it's also -wh

- ( $\epsilon$ , =V +wh C): a silent C that selects a V in its complement and moves a wh-phrase into its specifier.<sup>4</sup>

Since *who* has a D on top of its feature stack just like *she* does, we can **Merge** *sang* and *who* just like we did *sang* and *she*. However, this time *who*, unlike *she*, still has a feature left: -wh. This means it's going to Move later. Even though we'll check the features, we won't put *who* together with *sang* since it's just going to Move anyway. Instead we keep two separate expressions, (sang, V) and (who, -wh).

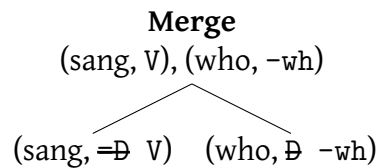


Figure 3.8: Partial derivation tree for *who sang*?

Thus, Merge forms *lists* of (derived item, feature stack) pairs. These lists are called *expressions*. The first element of the expression is always the “main” structure, and the remaining are movers waiting to be re-attached in their final position.

Now we Merge the silent complementiser with what we've just created, which means we put it together with the first element in the list, (sang, V). The moving expression (who, -wh) just stays in the list until the right feature configuration arises. (Note that  $\epsilon$  *sang* = *sang*, so I'll just write the string as *sang*.)

---

<sup>4</sup> $\epsilon$  is the symbol for an empty sequence, used here to mean the LI has no phonological features. For simplicity I've let the C select V directly, rather than have an intervening T.

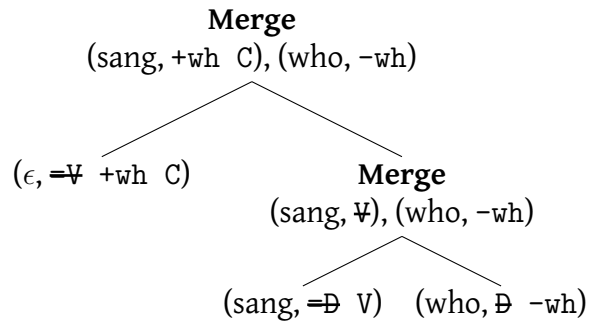


Figure 3.9: Partial derivation tree for *who sang?*

Now the top feature on the first expression (sang, +wh C) is a positive licensing feature +wh. This means the only possibility is for something to Move. In particular, the expression waiting to Move that has the matching negative licensing feature -wh on top of its feature stack has to move: (who, -wh). (If there is no such expression, the derivation crashes.) The wh features are checked and since *who* is now out of features, it is put together with *sang*.

Because the operation Move is putting together two expressions that are already in the derivation, Move is a unary function, so it is added to the derivation tree as a unary branching node.

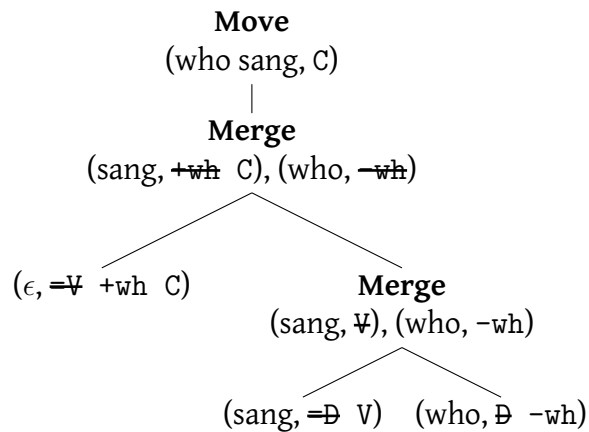


Figure 3.10: Derivation tree for *who sang?*



An LI's feature stack must have the following format for a derivation to succeed:

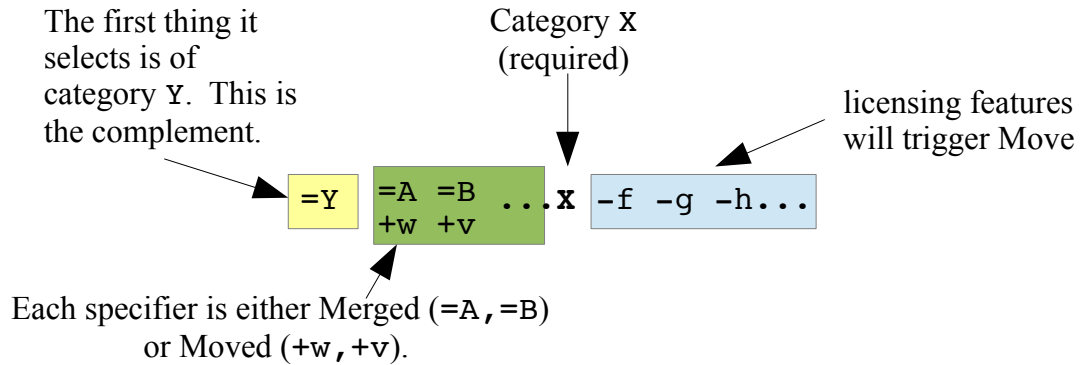


Figure 3.11: Features on a lexical item

Examples:

- (13)
- a. (kick, =D =D V): a transitive verb needs two DPs
  - b. (who, D -nom -wh): a subject wh-word will move twice, once for case and once for -wh.
  - c. (will, =V +nom T): a T selects a VP complement and moves a -nom subject into its specifier
  - d. (cat, N): a noun

Notice that once the complement and specifiers, if any, are added to the phrase, the head has all its positive features satisfied and we have what we would call in X-bar theory a complete XP.

In other words, this formalism just lists the requirements a head has, and once they are satisfied, the phrase is complete, and can be selected and then moved as necessary. It is exactly when the category feature X comes to the front of the stack that all the requirements of the XP are met. Exposing this feature allows the phrase to be selected. Cancelling that category feature reveals the negative licensing features, if any, allowing

the phrase to be moved.

### 3.3 MG Formalism

Now that we have a semi-formal basis for MGs, this section gives a properly formal definition of MGs. (Please see Appendix A for mathematical notation basics.) The first part gives a formal definition for the general case of MGs. It does not specify what sorts of objects are being built. Sections 3.3.1 to 3.3.2 give the formal definitions for three MGs, two that build strings, one that builds bare phrase structure trees. The core of the grammars (Merge and Move take two things and put them together, and are driven by opposite-polarity feature cancellation) is the same; what varies is the derived structure: what is built by this putting-together.

I start with a general MG, one which leaves open what is built. MGs are defined here over complex objects called *expressions*, which are sequences of pairs. In these pairs, the second element is a stack of features, and the first is the derived structure. The derived structure can be thought of as the interpretation of a term of an algebra with signature  $\langle \Sigma, \mathbf{Com}^{(2)} \rangle$ . Common interpretations of the signature are a string algebra, with **Com** interpreted as string concatenation, and a *bare tree* algebra, with **Com** interpreted as tree formation with parent node labelled  $>$  or  $<$ .

**Definition 3.3.1** (Terms over a signature).  $T_{\text{sig}}$  is the set of terms over signature  $\text{sig}$ ; that is, for all  $f \in \text{sig}$ ,  $f^{(i)}(t_1, \dots, t_i) \in T_{\text{sig}}$ .

For concreteness, in examples I will often use a string algebra and illustrate the elements as [Stabler and Keenan \(2003\)](#)'s grammar would generate them,<sup>5</sup> but this is just a place-holder for any algebra values the grammar might be defined to map into.

---

<sup>5</sup>Stabler & Keenan's grammar also incorporates an additional element: lexical items are triples of string, features, and lexical status, which allows derivation of Spec-Head-Complement order. I will leave this out for simplicity, as it is not relevant here.

Definition 3.3.2 gives the formal definition of a minimalist grammar, without yet defining Merge and Move. It lays out the pieces of a grammar – the vocabulary (aka “alphabet”)<sup>6</sup>, the two sets of features (**selectional** and **licensing**), how the lexicon is built by pairing vocabulary items with lists of features, Merge and Move, and the combining function **Com**. Finally, we can designate certain features to define the complete sentences, such as T and C. The language is everything we can build with one remaining feature, and if we have indeed designated a set of complete sentence features, that remaining feature must be one of those features.

**Definition 3.3.2.** A *Minimalist Grammar* is a six-tuple

$$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, Lex, M, \mathbf{Com} \rangle$$

$\Sigma$  is the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, X \mid f \in \mathbf{lic}, X \in \mathbf{sel}\}$  be the *features*.  $Lex \subseteq \Sigma \times F^*$ , and  $M$  is the set of operations **Merge** and **Move**. A set  $C \subseteq F$  of designated features can be added; these are the types of complete sentences.

The closure of the lexicon under Merge and Move is the lexicon as expressions, plus everything you can build from the lexicon using Merge and Move:

$$CL(Lex) = \{\langle l \rangle \mid l \in Lex\} \cup \{\mathbf{Merge}(e_1, e_2) \mid e_1, e_2 \in CL(Lex)\} \\ \cup \{\mathbf{Move}(e) \mid e \in CL(Lex)\}$$

To get the *language* of the grammar we take the subset of the closure that has exactly one remaining feature and interpret the terms of the algebra with a function  $\llbracket \cdot \rrbracket$  that will be defined in various ways below.

$$L(g) = \{\llbracket t \rrbracket \mid \exists S \in \mathbf{sel} \text{ s.t. } ((t, S)) \in CL(Lex)\}$$

---

<sup>6</sup>Instead of giving the phonological features and semantic features as collections of features, I will just give the word. This is standard in formal language approaches, and should be construed as shorthand for all of the sound-related and meaning-related features.

If we have defined  $C \subseteq \mathbf{sel}$  then

$$L(g) = \{\llbracket t \rrbracket \mid \exists S \in C \text{ s.t. } ((t, S)) \in CL(Lex)\}$$

Common algebras are:

- *Strings*:  $(\Sigma, \frown^{(2)})$  where  $\frown$  is string concatenation
- *Bare trees*:  $(\Sigma, f^{(2)})$  where the values of the algebra are terms over  $\{<^{(2)}, >^{(2)}\} \cup \Sigma^{(0)}$  and  $\llbracket f(t_1, t_2) \rrbracket = <(t_1, t_2)$  if  $t_1 \in \Sigma$  and  $\llbracket f(t_1, t_2) \rrbracket = >(t_1, t_2)$  otherwise

Minimalist Grammars are *feature-driven*, meaning features of lexical items determine which operations can occur and when. There are two disjoint finite sets of features, **selectional** features **sel** which drive the operation **Merge** and **licensing** features **lic** which drive **Move**. **Merge** puts two elements of the algebra together; **Move** operates on the already built structure. Each feature has a positive and negative version.

<i>Polarity</i> →	Pos	Neg	
for <b>Merge</b>	=X	X	X ∈ <b>sel</b>
for <b>Move</b>	+f	-f	f ∈ <b>lic</b>

Table 3.2: Features

In a minimalist grammar with alphabet  $\Sigma$ ,  $Lex \subseteq (\Sigma \cup \{\epsilon\}) \times F_{\text{fin}}^*$  is the **lexicon**: each word comes with a finite stack of features. In the course of the derivation the features will be *checked*, or deleted, by the operations **Merge** and **Move**.

Merge and Move are defined over **expressions**: sequences of pairs  $\langle \text{algebra element, feature stack} \rangle$ . The first pair can be thought of as the “main” structure being built; the remaining are waiting to move. An expression **displays** feature **f** just in case that feature is the first feature in the feature stack of the first pair.

An MG essentially works as follows: **Merge** is a binary operation driven by **sel**. It takes two expressions and combines them into one just in case the first structure displays =X and the second displays X for some  $X \in \text{sel}$ . These matching features are deleted, or “checked”. Once the second structure is selected, it may still have features remaining; these are always negative licensing features and mean that the second structure is going to move. As such is it stored separately by the derivation. When the matching positive licensing feature comes up later in the derivation, the moving structure is combined again. This is **Move**.

**Move** also carries the requirement that for each  $f \in \text{lic}$  there be at most one element waiting to move. This is the *shortest move constraint (SMC)*.<sup>7</sup>

**Notation** :: adds an element to a list. · appends two lists. – removes an element from a list. Also,

$$e :: \text{list}_1 \cdot \text{list}_2 = e :: (\text{list}_1 \cdot \text{list}_2) = (e :: \text{list}_1) \cdot \text{list}_2$$

Definition 3.3.3 gives the formal definition of Merge. It comes in two cases. In both cases, the two expressions being Merged must display matching selectional features of opposite polarity, like =X and X. These two features get deleted. In the first case, the thing being selected now has no more features left, and will therefore not move. In this case the two elements of the algebra are combined by **Com**. In the second case, the selected element still has features remaining, so it’s going to move.<sup>8</sup> In this case, we still cancel the features, but we add the selected element to the mover list. (In both cases, we combine the mover lists for the two Merging elements. Intuitively, this means things can move

---

<sup>7</sup>The SMC is based on economy arguments in the linguistic literature (Fox, 2000), but it is also crucial for a type of finiteness: the valid derivation trees of an MG form a regular tree language (Kobele et al., 2007). The number of possible movers must be finite for the automaton to be finite-state. The SMC could also be modified to allow up to a particular (finite) number of movers for each  $f \in \text{lic}$ .

<sup>8</sup>It will move because all lexical items that can participate in successful derivations will have the feature structure in figure 3.11. This Merge operation will cancel the category feature X, so if any features remain they will be negative licensing features, which drive Move.

out of both sisters. Note also that the order of movers doesn't matter, since there will only ever be one mover headed by a given feature. I've put the movers from the selector first, but this is arbitrary.)

**Definition 3.3.3 (Merge).** For  $\alpha, \beta$  sequences of features,  $s, t \in T_{\langle \Sigma, \text{Com} \rangle}$ ,  $\text{movers}_s, \text{movers}_t$  expressions:

$$\text{Merge}(\langle s, \text{X}\alpha \rangle :: \text{movers}_s, \langle t, \text{X}\beta \rangle :: \text{movers}_t) = \begin{cases} \langle \text{Com}(s, t), \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \langle s, \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta \neq \epsilon \end{cases}$$

Definition 3.3.4 gives the formal definition of Move. It is very similar to Merge, except that we have to extract the second element from the mover list. Suppose the expression displays feature  $+f$ . We look at the list called *movers*, and if we find exactly one pair whose first feature is  $-f$ , then we can apply Move. Call that unique pair *the mover*. We cancel the  $+f, -f$  features, and if the mover is now out of features, then it's finished moving. We combine it with the main structure. This is the first case. In the second case, the mover has features left, so it isn't finished moving. Instead of attaching it to the main structure, we put it back in the mover list.

**Definition 3.3.4 (Move).** For  $\alpha, \beta, \gamma \in F^*$ ,  $s, t \in T_{\langle \Sigma, \text{com} \rangle}$ , suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = -f\gamma$ . Then:

$$\text{Move}(\langle s, +f\alpha, \rangle :: \text{movers}) = \begin{cases} \langle \text{Com}(s, t), \alpha \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \gamma = \epsilon \\ \langle s, \alpha \rangle :: \langle t, \gamma \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \gamma \neq \epsilon \end{cases}$$

### 3.3.1 String-generating MG

This section describes the string-generating MG defined in [Stabler and Keenan \(2003\)](#). **Com** is given as  $\frown$ , which just means string concatenation. Section 3.3.1.1 gives a simpler version which follows the generic MG template above exactly. Section 3.3.1.2 gives the

more complex version, which is given in Keenan and Stabler (2003), designed to create word orders as Kayne (1994)'s antisymmetric trees would define them. This is a slight extension of the MG definition in Definition 3.3.2 as it adds an extra item: expression are built of *triples* that include a feature  $\pm lex$  that indicate whether the structure is lexical or not.

### 3.3.1.1 Non-antisymmetric version

This grammar is defined to always put selected strings on the right and moved strings on the left. There is no way to distinguish heads and derived phrases, because everything we build is flat strings. As such, there is no way to Merge specifiers on the left and complements on the right; we can however Move and Merge to different directional specifiers, since Move and Merge are defined separately. The right and left choices here are, formally speaking, arbitrary.

**Definition 3.3.5.** A *String-generating Minimalist Grammar* is a seven-tuple

$$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, Lex, M, \frown \rangle$$

$\Sigma$  is the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, X | f \in \mathbf{lic}, X \in \mathbf{sel}\}$  be the *features*.  $Lex \subseteq \Sigma \times F^*$ , and  $M$  is the set of operations **Merge** and **Move**.  $\frown$  concatenates strings, so the MG builds terms over the signature  $(\Sigma, \frown)$ . A set  $C \subseteq F$  of designated features can be added; these are the types of complete sentences. If so, normally the language is defined as  $\{\llbracket w \rrbracket | \exists c \in C \text{ s.t. } \langle \langle w, c \rangle \rangle \in CL(Lex)\}$ .

I define Merge to always put the selected string on the right. This means complements are rightward, but so are selected specifiers. The part of the definition that gives this directionality is  $\langle s \frown t, \alpha \rangle$ .  $s$  was the string belonging to the selector and  $t$  to the selectee.

**Definition 3.3.6 (Merge).** For  $\alpha, \beta \in F^*$ ,  $s, t$  strings:

$$\text{Merge}(\langle s, \text{X}\alpha \rangle :: \text{movers}_s, \langle t, \text{X}\beta \rangle :: \text{movers}_t) = \begin{cases} \langle s \frown t, \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \langle s, \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta \neq \epsilon \end{cases}$$

Move is always to a specifier; we can choose to always put moved specifiers on the left.  $\exists!$  means "there is a unique".

**Definition 3.3.7 (Move).** For  $\alpha, \beta, \gamma \in F^*$ ,  $s, t$  strings, suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = -f\gamma$ . Then:

$$\text{Move}(\langle s, +f\alpha \rangle :: \text{movers}) = \begin{cases} \langle t \frown s, \alpha \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \gamma = \epsilon \\ \langle s, \alpha \rangle :: (\langle t, \gamma \rangle :: \text{movers} - \langle t, \beta \rangle) & \text{if } \gamma \neq \epsilon \end{cases}$$

For example derivations see figures 3.2, 3.3, 3.7, and 3.10 above.

Chapter 4 will use this grammar as a basis, but I will usually give the derived strings as if they had been generated by the grammar in section 3.3.1.2, since that makes them more readable.

### 3.3.1.2 Antisymmetric version

This MG is very much like that in Section 3.3.1.1, except that this MG has an additional element, a feature  $\pm 1\text{ex}$  that indicates whether the string is lexical, as in just taken straight from the lexicon, or non-lexical, as in derived. This distinction is used to allow complements and specifiers to be on opposite sides of the head, à la Kayne (1994). When the selector is  $+1\text{ex}$ , it is a head, so it goes on the left (putting the complement on the right). When the selector is  $-1\text{ex}$ , it is a derived phrase, so it goes on the right (putting the specifier on the left). Note that these directions could be redefined for a different shape of phrase if desired.



**Definition 3.3.8.** A *String-generating Minimalist Grammar* is a six-tuple

$$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, \{+lex, -lex\}, Lex, M, \frown \rangle$$

$\Sigma$  is the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, X | f \in \mathbf{lic}, X \in \mathbf{sel}\}$  be the *features*.  $Lex \subseteq \Sigma \times \{+lex\} \times F^*$ , and  $M$  is the set of operations **Merge** and **Move**.  $\frown$  concatenates strings, so the MG is defined over the algebra  $(\Sigma^*, \frown)$ . A set  $C \subseteq F$  of designated features can be added; these are the types of complete sentences. If so, the language is normally defined at  $\{\llbracket w \rrbracket | \exists c \in C, l \in \{+lex, -lex\} \text{ s.t. } \langle \langle w, l, c \rangle \rangle \in CL(Lex)\}$ .

Merge distinguishes between lexical and non-lexical phrases. The lexicon has only  $+lex$  features, and Merge always outputs  $-lex$ , so we can keep track of whether something is straight from the lexicon or if it's been built by the derivation. This information is used to determine order. To model Kayne (1994)'s antisymmetry, heads ( $+lex$  strings) put their selectee on their right, and non-heads ( $-lex$  strings) put their selectee on their left. This gets complements on the right of the head and specifiers on the left. Move is always to a specifier and therefore always to the left.

**Definition 3.3.9 (Merge).** For  $\alpha, \beta \in F^*$ ,  $s, t$  strings,  $l, k \in \{+lex, -lex\}$ :

$$\mathbf{Merge}(\langle s, l, =X\alpha \rangle :: \mathbf{movers}_s, \langle t, k, X\beta \rangle :: \mathbf{movers}_t) = \begin{cases} \langle s \frown t, -lex, \alpha \rangle :: \mathbf{movers}_t & \text{if } l = +lex, \beta = \epsilon \\ \langle t \frown s, -lex, \alpha \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } l = -lex, \beta = \epsilon \\ \langle s, -lex, \alpha \rangle :: \langle t, k, \beta \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } \beta \neq \epsilon \end{cases}$$

Move is always to a specifier, so it always puts the selected phrase on the left.

**Definition 3.3.10 (Move).** For  $\alpha, \beta, \gamma \in F^*$ ,  $s, t$  strings, suppose  $\exists! \langle t, l, \beta \rangle \in \mathbf{movers}$  such that  $\beta = -f\gamma$ . Then:

$$\text{Move}(\langle s, -\text{lex}, +f\alpha, \rangle :: \text{movers}) = \begin{cases} \langle t \curvearrowright s, -\text{lex}, \alpha \rangle :: \text{movers} - \langle t, \text{l}, \beta \rangle & \text{if } \gamma = \epsilon \\ \langle s, -\text{lex}, \alpha \rangle :: (\langle t, \text{l}, \gamma \rangle :: \text{movers} - \langle t, \text{l}, \beta \rangle) & \text{if } \gamma \neq \epsilon \end{cases}$$

Figure 3.12 shows the derivation of a VP *Kaylee fix Serenity*. It assumes that *Kaylee* won't move, in order to demonstrate the spec-head-complement order. Notice that since *fix* is +lex, Merge generates *fix Serenity*, with the selector *fix* on the left. But then *fix Serenity* is -lex, so when it selects *Kaylee*, it puts the selectee, *Kaylee*, on the left.

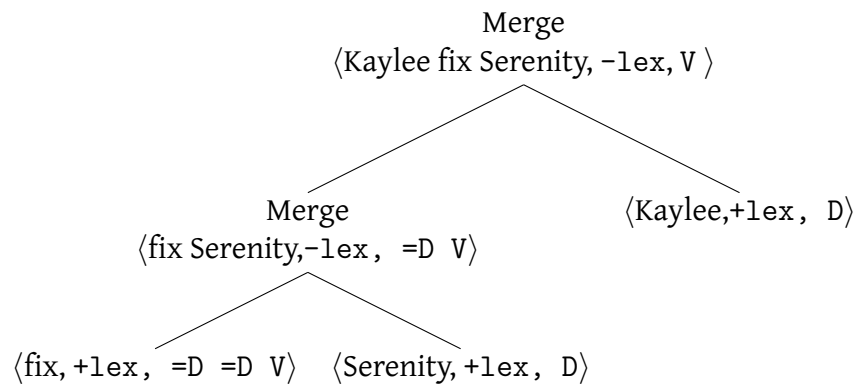


Figure 3.12: String-generating grammar derivation tree for VP *Kaylee fix Serenity*

Chapter 4 will use this grammar as a basis, but I will leave out the  $\pm\text{lex}$  distinction for the sake of simplicity. However, I will usually give the derived strings as if they had been generated by this grammar, since that makes them more readable.

### 3.3.2 Tree-generating grammar

This section describes a bare tree-generating MG simplified from [Stabler \(1997\)](#).<sup>9</sup> “Bare trees” are yet simpler versions of the *bare phrase structure trees* proposed by [Chomsky](#)

<sup>9</sup>The grammar in [Stabler \(1997\)](#) does not make a mover list, and does not pair the trees with their features. Rather, it leaves the features in the bare tree, cancelling them as it goes. Move works by a tree search for the relevant licensing feature. These two approaches give the same language, so I have stuck with the version that is more similar to the other grammars we will consider.

(1995). Chomsky suggests that internal nodes of the derived tree be labelled by a copy of the head label; bare trees instead incorporate just an arrow pointing the way to the head. I will call the set of bare trees over an alphabet  $\Sigma$   $T_{\langle \Sigma, <, > \rangle}$ .



Figure 3.13: Bare phrase structure tree and equivalent bare tree

**Definition 3.3.11.** A *bare tree-generating Minimalist Grammar* is a five-tuple

$$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, Lex, M, f \rangle$$

$\Sigma$  is the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, X \mid f \in \mathbf{lic}, X \in \mathbf{sel}\}$  be the *features*.  $Lex \subseteq \Sigma \times F^*$ , and  $M$  is the set of operations **Merge** and **Move**. A set  $C \subseteq F$  of designated features can be added; these are the types of complete sentences. If so, normally the language is defined as  $\{\llbracket w \rrbracket \mid \exists c \in C \text{ s.t. } \langle \langle w, c \rangle \rangle \in CL(Lex)\}$ .  $f$  builds trees, as defined below.

Bare trees have internal nodes labelled  $>$  and  $<$ . To find the head of a phrase, follow the internal node labels as if they are arrows.

A tree  $t$  is *simple* if it has exactly one node. Otherwise it is *complex*.

We define the tree-combining function  $f$  as follows:

$$\mathbf{Definition 3.3.12} (f). f(t_1, t_2) = \begin{cases} < (t_1, t_2) & \text{if } t_1 \text{ is simple} \\ > (t_2, t_1) & \text{otherwise} \end{cases}$$

This just means that we label the mother of the combining trees with  $<$  if the first tree is simple, and we reverse their order and label the mother with  $>$  if it is complex. This models Kayne (1994); again,  $f$  could be redefined for other spec-head-complement orders.

**Definition 3.3.13 (Merge).** For  $\alpha, \beta$  sequences of features,  $s, t \in T_{\langle \Sigma, <, > \rangle}$ :

$$\text{Merge}(\langle s, =X\alpha, \rangle :: \text{movers}_s, \langle t, X\beta \rangle :: \text{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers}_t & \text{if } \beta = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_t & \text{if } \beta \neq \epsilon \end{cases}$$

Move will always end up making a  $>$ -rooted tree, since the element that drives Move with its +f feature is never just a head, but is rather always part of the tree.

**Definition 3.3.14 (Move).** For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  trees, suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = -f\gamma$ . Then:

$$\text{Move}(\langle s, +f\alpha \rangle :: \text{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers} - (t, \beta) & \text{if } \gamma = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \gamma \rangle :: \text{movers} - (t, \beta) & \text{if } \gamma \neq \epsilon \end{cases}$$

Figure 3.14 shows the derivation tree and derived bare tree for *the cowboy*, and Figure 3.15 shows the derivation and derived trees for *Who did Jayne shoot?*.

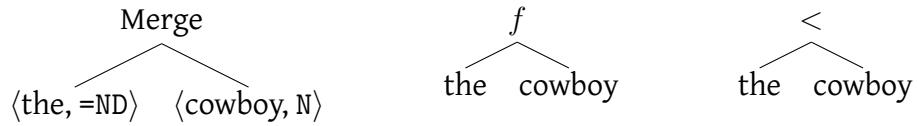


Figure 3.14: Derivation tree, term over  $\langle \Sigma, f \rangle$ , and derived bare tree of *the cowboy*

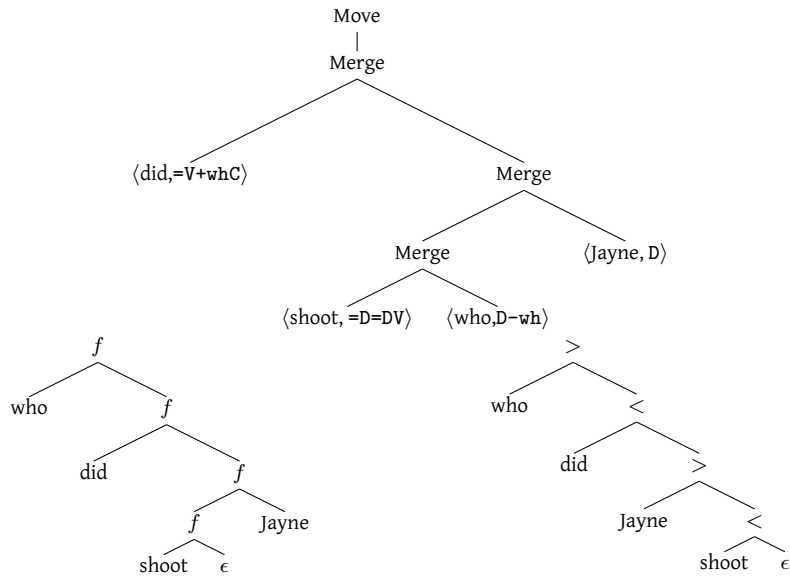


Figure 3.15: Derivation tree, term over  $T_{\langle \Sigma, f \rangle}$  and derived bare tree for *Who did Jayne shoot?*

## 3.4 How MGs implement traditional minimalism

### 3.4.1 Tree search implementation

Stabler (1997) originally interprets MGs in a bare tree algebra, and it works without a mover list. Instead of the mover list, the mover is left in the tree, along with its remaining features. When its matching positive licensing feature comes up, we search the tree for the mover, delete it from its current position and move it to its new position at the top of the tree. The grammars in Stabler (1997) and Stabler and Keenan (2003) are strongly equivalent, so I have been using the mover-list implementation in the latter throughout. However, for certain considerations, having the tree-search implementation in mind can be informative. In particular, mainstream syntax normally approaches syntactic derivations from a tree-search perspective. Moreover, a tree-search implementation opens up certain possibilities that a mover-list implementation does not, such as late adjunction and certain ways of implementing optional movement. I will briefly define this approach

here.

**Definition 3.4.1.** In a bare tree  $t$ , node  $x$  projects over its sister  $y$  iff  $t$  contains  $\langle(x, y)$  or  $\rangle(y, x)$ . A node also projects over itself, and we can extend projection to the transitive closure thereof: node  $x$  projects over node  $y$  iff  $x$  projects over  $y$  or  $x$  projects over  $z_1$  which in turn projects over  $z_2 \dots$  which in turn projects over  $y$ .

**Definition 3.4.2.** Let  $x$  be a node.  $x$  is the *head* of  $y$  if:

- $y = x$  and  $y$  is simple (a leaf) or,
- $y$  has a daughters  $z$  and  $w$  where  $z$  projects over  $w$ , and  $x$  is the head of  $z$ .

Intuitively, a head projects over everything in its phrase.

For example, in Figure 3.16, node  $y$  has daughters  $w$  and  $z$ , and  $z$  projects over  $w$  since  $y$  is labelled  $\rangle$ .  $z$  has daughters  $x$  and  $c$ , and  $x$  projects over  $c$ .  $x$  is the head of  $x$  since it is a leaf.  $x$  is therefore the head of  $x$ ,  $z$ , and  $y$ .

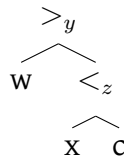


Figure 3.16:  $x$  is the head of  $x$ ,  $z$ , and  $y$

**Definition 3.4.3.** A bare tree subtree is *maximal* if it is a left sister and its mother is labelled  $\rangle$  or it is a right sister and its mother is labelled  $\langle$ .

**Definition 3.4.4.** A (sub)tree  $t$  has *feature*  $f$  if the head of  $t$  is labelled  $\langle s, f :: \alpha \rangle$  for some string  $s$  and feature list  $\alpha$ .

**Definition 3.4.5 (Merge (tree-search implementation)).** Let  $X \in \mathbf{sel}$ , and let  $t_1, t_2$  be bare trees having feature  $=X$  and  $X$  respectively.

$$\text{Merge}(t_1, t_2) = \begin{cases} < (t'_1, t'_2) \text{ where } t'_1 \text{ is just like } t_1 \text{ except that } =X \text{ is deleted} \\ & \text{and } t'_2 \text{ is just like } t_2 \text{ except that } X \text{ is deleted} & \text{if } t_1 \text{ is a leaf} \\ > (t'_1, t'_2) \text{ where } t'_1 \text{ is just like } t_1 \text{ except that } =X \text{ is deleted} \\ & \text{and } t'_2 \text{ is just like } t_2 \text{ except that } X \text{ is deleted} & \text{otherwise} \end{cases}$$

**Definition 3.4.6** (*Move* (tree-search implementation)). Let  $f \in \mathbf{lic}$  and let  $t$  be a bare tree with feature  $+f$  with exactly one maximal subtree having feature  $-f$ . Call that subtree  $t_0$ .

$\text{Move}(t) = > (t'_0, t')$  where  $t'$  is just like  $t$  except that  $+f$  is deleted and the subtree  $t_0$  is deleted, and  $t'_0$  is just like  $t_0$  except  $-f$  is deleted.

Figure 3.17 illustrates the steps of a derivation in a tree-search type MG.

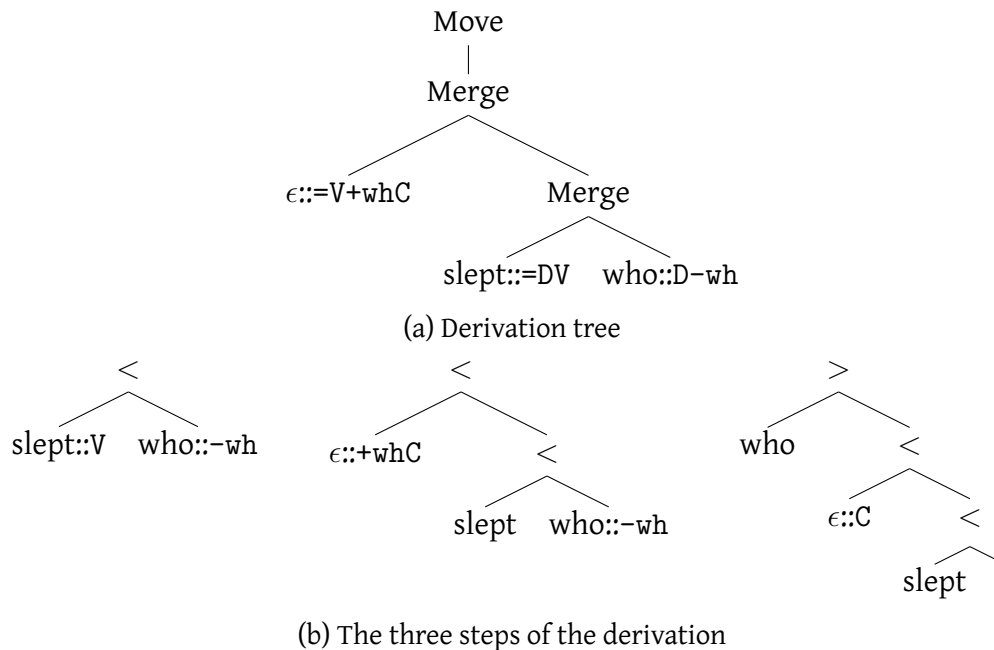


Figure 3.17: Derivation in a tree-search MG

### 3.4.2 MG extensions

MGs can be extended to include aspects of minimalism not included in the simple implementations I described here.

#### 3.4.2.1 Head Movement

MGs can be defined to include head movement; it is modelled as a variant on Merge. For details, see [Stabler \(1997\)](#). I have left head movement completely out of this dissertation, as it is not relevant to the parts of the grammar I look at, and it complicates the picture: Merge has 11 cases rather than just 3.

#### 3.4.2.2 Successive cyclic movement

MGs can be extended to include successive cyclic movement by modifying Move to make deletion of the negative feature optional. Only the positive feature of the goal is deleted, and the entire mover stays in the mover list.

**Definition 3.4.7 (Move).** For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  trees, suppose  $\exists! \langle t, \beta \rangle \in$  movers such that  $\beta = -f\gamma$ . Then:

$$\mathbf{Move}_{\text{final}}(\langle s, +f\alpha, \rangle :: \text{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers} - (t, \beta) & \text{if } \gamma = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \gamma \rangle :: \text{movers} - (t, \beta) & \text{if } \gamma \neq \epsilon \end{cases}$$
$$\mathbf{Move}_{\text{sc}}(\langle s, +f\alpha, \rangle :: \text{movers}) = \langle f(s, \epsilon), \alpha \rangle :: \text{movers}$$

For example (skipping TP levels), the following derivation includes a successive cyclic step at the [blue](#) Move node. The +wh feature is checked but nothing else changes. Naturally, in the tree-search implementation described in [Section 3.4.1](#), the mover also lands temporarily in the interim position.



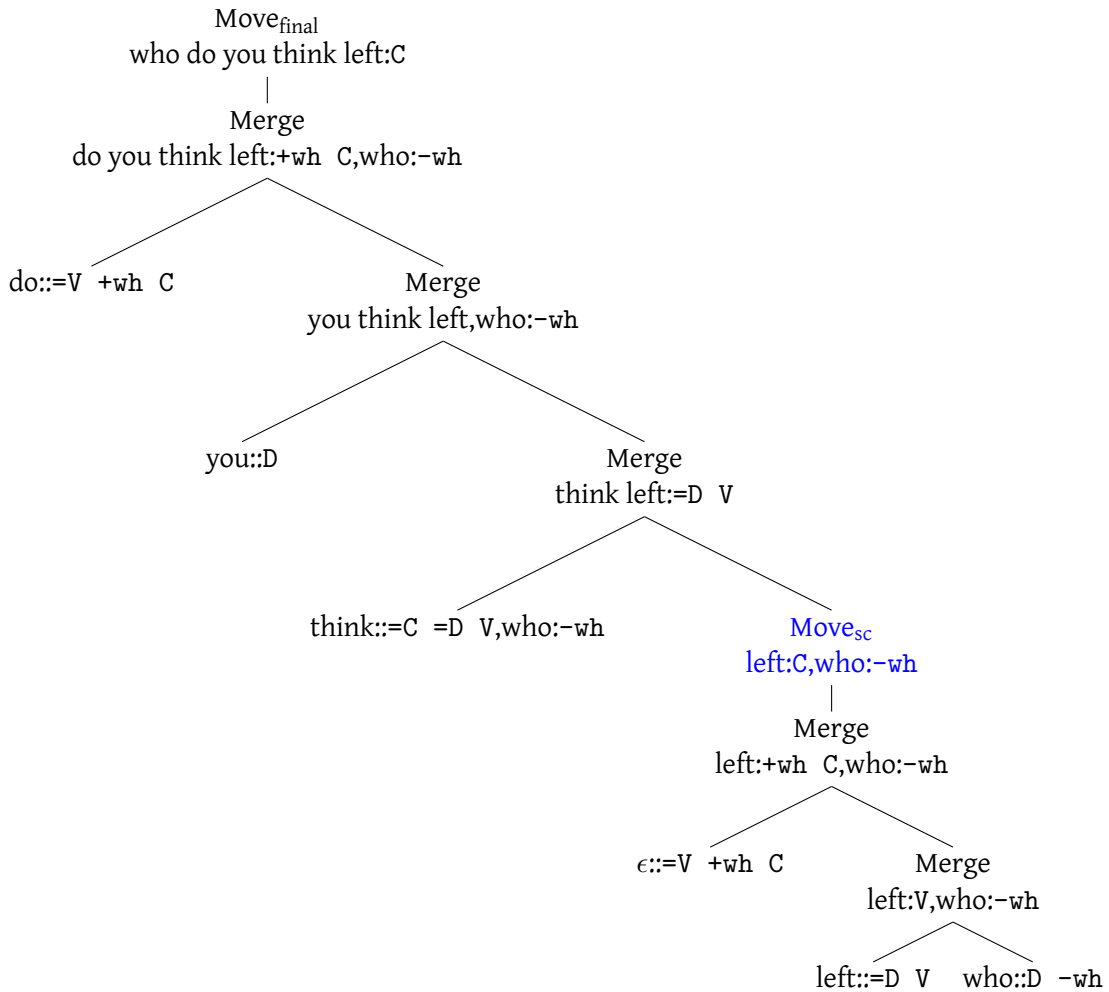


Figure 3.18: Derivation tree with successive cyclic movement

Depending on exactly what we want our derived structures to look like, we can define  $\text{Move}_{\text{sc}}$  a little differently. If we're building sequences or trees with potentially empty nodes, the above definition is right. If we want traces in the trees, we use  $\mathbf{Move}_{\text{sc}}(\langle s, +f\alpha \rangle :: \text{movers}) = \langle f(s, \text{trace}), \alpha \rangle :: \text{movers}$ .

Note that a system with optional feature deletion does not have deterministic  $\text{Move}$ , unlike in traditional MGs.

One thing that MGs illuminate is that it is only in the tree search implementation

rather than a mover list implementation for movement, that successive-cyclic movement has any obvious derivational use. If the derivation uses a tree search, successive-cyclic movement brings the goal closer to the next probe. In all other cases, the only thing that happens is that the intermediate probe has its positive licensing feature checked. Since often this intermediate probe feature is only posited to be there to license successive-cyclic movement, the intermediate landing becomes purposeless, unless additional mechanisms are added, such as phases.

There are some MG extensions that require a tree-search, rather than mover-list, implementation, such as those with late adjunction. Most, however, are strongly equivalent with either tree-search or mover-list implementations, and the mover-list implementation does not require an expensive tree search.

### 3.4.2.3 Phases

Phases, like successive-cyclic movement, are intuitively pragmatic under a tree-search implementation, by constraining the size of the search tree. Under a mover-list implementation, however, they have the intuitive feel of extra steps in the derivation for no obvious reason.

Phases without escape hatches can be implemented by forbidding movers at the merge of a phase head. This is done by simply not including in the definition of Merge a case for a phase boundary with movers. If the derivation reaches a phrase boundary with a mover list, Merge is undefined so the derivation crashes. In these implementations Phase heads are marked with an additional feature; another way to do it is to have a set of phase head categories; if, say, CP is always a phase, the phase case is when the category is C rather than when the phase feature is True.

**Definition 3.4.8** (*Merge of a phase head*). For  $\alpha, \beta$  sequences of features,  $s$  a lexical item,  $t$  a derived structure,  $p, q \in \{T, F\}$  marking phasehood, movers an expression,  $X \in \mathbf{sel}$ :

$$\mathbf{Merge}(\langle s, p, =X\alpha \rangle, \langle t, q, X\beta \rangle :: \mathbf{movers}) = \begin{cases} \langle f(s, t), F, \alpha \rangle :: \mathbf{movers} & \text{if } \beta = \epsilon, p = F \\ \langle f(s, \epsilon), F, \alpha \rangle :: \langle t, q, \beta \rangle :: \mathbf{movers} & \text{if } \beta \neq \epsilon, p = F \\ \langle f(s, t), T, \alpha \rangle & \text{if } \beta = \epsilon, p = T, \mathbf{movers} = \epsilon \\ \langle f(s, \epsilon), T, \alpha \rangle :: \langle t, q, \beta \rangle & \text{if } \beta \neq \epsilon, p = T, \mathbf{movers} = \epsilon \end{cases}$$

If a phase head without an escape hatch is not the selector of the frozen phrase but rather the head of the frozen phrase, we use almost the same definition, except it's the selectee's phasehood feature that must be true, and the selector need not be a head, so there are potentially two mover lists:

**Definition 3.4.9** (*Merge of phase*). For  $\alpha, \beta$  sequences of features,  $s, t$  derived structures,  $p, q \in \{T, F\}$  marking phasehood,  $\mathbf{movers}_s$  and  $\mathbf{movers}_t$  expressions,  $X \in \mathbf{sel}$ :

$$\mathbf{Merge}(\langle s, p, =X\alpha \rangle :: \mathbf{movers}_s, \langle t, q, X\beta \rangle :: \mathbf{movers}_t) = \begin{cases} \langle f(s, t), p, \alpha \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } \beta = \epsilon, q = F \\ \langle f(s, \epsilon), p, \alpha \rangle :: \langle t, F, \beta \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } \beta \neq \epsilon, q = F \\ \langle f(s, t), p, \alpha \rangle :: \mathbf{movers}_s & \text{if } \beta = \epsilon, q = T, \mathbf{movers}_t = \epsilon \\ \langle f(s, \epsilon), p, \alpha \rangle :: \langle t, T, \beta \rangle :: \mathbf{movers}_s & \text{if } \beta \neq \epsilon, p = T, \mathbf{movers}_t = \epsilon \end{cases}$$

If on the other hand, phases have escape hatches, things are more complicated. Let us call an escape hatch a specifier of a phase head, and say that nothing can move out of the sister of a phase head. A specifier may be newly Merged, in which case our definition will not have to worry about it, or it can be moved. It is these latter phrases that are of concern, as they have come from the part of the phase from which nothing is supposed to be able to move. We build the exception into the Merge rule, which must unfortunately look deeper into the feature stack than the top element as is usual. We need to ensure all phase-internal movers have a matching positive licensing feature on the phase heads to

move them out into a specifier.<sup>10</sup>

**Definition 3.4.10** (*Merge a phase head with escape hatches*). For  $\alpha, \beta, \gamma, \delta$  sequences of features,  $s$  a lexical item,  $t$  a derived structure,  $X, Y \in \mathbf{sel}$ ,  $f \in \mathbf{lic}$ ,  $p, q \in \{T, F\}$  marking phasehood, movers an expression:

$$\mathbf{Merge}(\langle s, p, =X\alpha \rangle, \langle t, q, X\beta \rangle :: \mathbf{movers}) = \begin{cases} \langle f(s, t), F, \alpha \rangle :: \mathbf{movers} & \text{if } \beta = \epsilon, p = F \\ \langle f(s, \epsilon), F, \alpha \rangle :: \langle t, q, \beta \rangle :: \mathbf{movers} & \text{if } \beta \neq \epsilon, p = F \\ \langle f(s, t), T, \alpha \rangle :: \mathbf{movers} & \text{if } \beta = \epsilon, p = T, \forall \langle u, -f\delta \rangle \in \mathbf{movers}, \exists +f \in \alpha \\ \langle f(s, \epsilon), T, \alpha \rangle :: \langle t, q, \beta \rangle :: \mathbf{movers} & \text{if } \beta \neq \epsilon, p = T, \forall \langle u, -f\delta \rangle \in \mathbf{movers}, \exists +f \in \alpha \end{cases}$$

Again, under a mover-list interpretation, phases lose a lot of their appeal. They add something extra, rather than simplify by constraining the search space. Phases of course are not only argued to increase computational efficiency; they are also argued to account for data. They are also sometimes taken to be Spellout domains in multiple spellout models.

### 3.4.2.4 Spellout

In traditional string-generating MGs, Spellout is not an operation but rather simply the relationship between the derivation and the derived structure. In the case of a tree- or graph-generating grammar, Spellout can also be the operation on the *derived* structure that makes it pronounceable.

If we want phases not to be defined as arbitrary barriers but rather as the result of multiple spellout, the enterprise changes immediately. To begin with, we have a third operation – Spellout – whose output must be determined. It is not obvious what exactly the derived structure of such a derivation is. Is it a set of constituents, passed on to PF?

---

<sup>10</sup>If there is only a single escape hatch, this usually follows from there being only a single specifier; this is enforced by a constraint on the lexicon: all lexical items must be of the form  $s::\alpha X\beta$  where  $|\alpha| \leq 3$

A single constituent as usual, but spellout adds to the numeration an atom with all the features of the root of the spelled-out constituent, and a new derivation occurs? How is the sentence put back together in the end?

A minimalist grammar with spellout goes beyond the scope of this dissertation. Of interest here is rather how the well-definition of MGs forces precision in the definition of spellout.

### 3.4.3 Overt, covert, and copy movement

A traditional MG defines only overt movement, in which a moving constituent is pronounced only in its final position. We can add covert movement by adding a third arity to the licensing features. I will follow [Kobele \(2010\)](#) in notating a covert-moving item with  $\ominus f$ . We mark the mover because in a mover-list implementation, it is at the point of Merge that the choice between overt and covert must first be made: do we pronounce it here, or add it to the mover list and pronounce it in a higher position? In fact, in either case we add the features to the mover list; overt and covert move differ only in how they treat the derived element (e.g., the string): is the pronunciation of  $t$  added to the main structure or the mover list? We also modify Move similarly for the case of a mover that will continue moving.

**Definition 3.4.11** (*Merge with covert movers*). For  $\alpha, \beta$  sequences of features,  $s, t$  derived or atomic structures:

$$\text{Merge}(\langle s, X\alpha \rangle :: \text{movers}_s, \langle t, X\beta \rangle :: \text{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = -f\gamma \\ \langle f(s, t), \alpha \rangle :: \langle \epsilon, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \ominus f\gamma \end{cases}$$

**Definition 3.4.12** (*Move with covert movers*). For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  derived or atomic elements, suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = -f\gamma$  or  $\beta = \ominus f\gamma$ . Then:

$$\mathbf{Move}(\langle s, +f\alpha \rangle :: \mathbf{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \mathbf{movers} - \langle t, \beta \rangle & \text{if } \gamma = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \gamma \rangle :: \mathbf{movers} - \langle t, \beta \rangle & \text{if } \beta = -f\gamma \\ \langle f(s, t), \alpha \rangle :: \langle \epsilon, \gamma \rangle :: \mathbf{movers} - \langle t, \beta \rangle & \text{if } \beta = \ominus f\gamma \end{cases}$$

We can also implement copying in a similar way. If we want, all movement can be copy movement. This is a common interpretation of traditional Minimalism. As discussed in section 3.1.1 above, it must be decided whether and how to mark copies in the derived structure if we don't want to necessarily pronounce and/or interpret all copies. If we do not mark copies and all movement is copy movement, the definitions of Merge and Move change very little. The only difference is, in the second Merge case, the selected phrase  $t$  appears in both the main structure  $f(s, t)$  and the mover list.

**Definition 3.4.13** (*Merge for copy move*). For  $\alpha, \beta$  sequences of features,  $s, t$  derived or atomic structures:

$$\mathbf{Merge}(\langle s, X\alpha \rangle :: \mathbf{movers}_s, \langle t, X\beta \rangle :: \mathbf{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } \beta = \epsilon \\ \langle f(s, t), \alpha \rangle :: \langle t, \beta \rangle :: \mathbf{movers}_s \cdot \mathbf{movers}_t & \text{if } \beta = -f\gamma \end{cases}$$

**Definition 3.4.14** (*Move with copy move*). For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  derived or atomic elements, suppose  $\exists! \langle t, \beta \rangle \in \mathbf{movers}$  such that  $\beta = -f\gamma$ . Then:

$$\mathbf{Move}(\langle s, +f\alpha \rangle :: \mathbf{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \mathbf{movers} - \langle t, \beta \rangle & \text{if } \gamma = \epsilon \\ \langle f(s, t), \alpha \rangle :: \langle t, \gamma \rangle :: \mathbf{movers} - \langle t, \beta \rangle & \text{if } \beta = -f\gamma \end{cases}$$

If on the other hand we want to mark our copies, we need to add to the system a set of indices, and when we determine the pronunciation of the derived structure we need an algorithm to decide which copy/ies to pronounce. This only makes sense if our grammar generates trees or graphs; strings have no hierarchical structure to mark copies in. Note that the indices on the derived structures are not variables in the definition but rather actual parts of the derived structure.

**Definition 3.4.15** (*Merge for copy move with indices*). For  $\alpha, \beta$  sequences of features,  $s, t$  members of the algebra,  $j \in \mathbb{N}$  such that  $j$  is new to the derivation:

$$\text{Merge}(\langle s, X\alpha \rangle :: \text{movers}_s, \langle t, X\beta \rangle :: \text{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \langle f(s, t_j), \alpha \rangle :: \langle t_j, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = -\mathbf{f}\gamma \end{cases}$$

**Definition 3.4.16** (*Move with copy move*). For  $\alpha, \beta, \gamma$  sequences of features,  $s, t_i$  members of the algebra, suppose  $\exists! \langle t_i, \beta \rangle \in \text{movers}$  such that  $\beta = -\mathbf{f}\gamma$ . Then:

$$\text{Move}(\langle s, +\mathbf{f}\alpha \rangle :: \text{movers}) = \begin{cases} \langle f(s, t_i), \alpha \rangle :: \text{movers} - (t_i, \beta) & \text{if } \gamma = \epsilon \\ \langle f(s, t_i), \alpha \rangle :: \langle t_i, \gamma \rangle :: \text{movers} - \langle t_i, \beta \rangle & \text{if } \gamma \neq \epsilon \end{cases}$$

I claim that we can also use this mechanism for actual copying. By adding another arity to **lic**, we can tell the derivation to copy the mover to the mover list. I will denote this with  $\otimes\mathbf{f}$ . I will give this definition in terms of a grammar that does not have copy move as its default move; rather copying is different from moving.

**Definition 3.4.17** (*Merge for overt move plus phrasal copying*). For  $\alpha, \beta$  sequences of features,  $s, t$  derived or atomic structures:

$$\text{Merge}(\langle s, X\alpha \rangle :: \text{movers}_s, \langle t, X\beta \rangle :: \text{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = -\mathbf{f}\gamma \\ \langle f(s, t), \alpha \rangle :: \langle t, \beta \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \otimes\mathbf{f}\gamma \end{cases}$$

**Definition 3.4.18** (*Move with overt move plus phrasal copying*). For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  derived or atomic elements, suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = -\mathbf{f}\gamma$  or  $\beta = \otimes\mathbf{f}\gamma$ . Then:

$$\text{Move}(\langle s, +\mathbf{f}\alpha \rangle :: \text{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers} - (t, \beta) & \text{if } \gamma = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \gamma \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \beta = -\mathbf{f}\gamma \\ \langle f(s, t), \alpha \rangle :: \langle t, \gamma \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \beta = \otimes\mathbf{f}\gamma \end{cases}$$

Adding copying to the string yield of a minimalist grammar, whether via a copy variation on Move as in definitions 3.4.17 and 3.4.18 above, or by any copy move that does not ultimately spell out exclusively to sentences with only one copy, slightly changes the place of the language in the Chomsky Hierarchy. We move up to being weakly equivalent to *parallel* Multiple Context Free Grammars, which have copying [Stabler \(2004\)](#).

### 3.4.4 Combine and Store

Since the definitions of Merge and Move vary so slightly depending on whether we choose overt or covert movement, and it is so easy to vary the definition again along the very same lines to derive copying, we can also consider these possibilities in a full  $2 \times 2$  matrix, according to whether a moving element is pronounced in the place it is being merged, and whether its pronunciation is being stored in the mover list:

	pronunciation in mover list	$\epsilon$ in mover list
pronunciation main structure	Copy ( $\otimes f$ )	Covert move ( $\ominus f$ )
$\epsilon$ in main structure	Overt move ( $-f$ )	Delete ( $\text{del } f$ )

Table 3.3: Combine and Store possibilities

The bottom right quadrant is new. What happens if a constituent’s pronunciation is nowhere? It is not pronounced at all. We have now added a new capability to the grammar, deletion, using only elements we already had available to us.

We can even imagine a grammar in which Move features are not distinguished by polarities but rather by pairs of polarities, the first telling the system whether to include the pronunciation of the mover in the main structure, and the second whether to include the pronunciation of the mover in the mover list:  $(++)f$  for copy,  $(+-)f$  for covert move,  $(-+)f$  for overt move, and  $(-- )f$  for deletion. We define general functions for combining a mover with the main structure (Combine) and for storing the mover in the mover list (Store). Note we always do both of these things in an MG, we just haven’t explicitly made



it part of the definition of Merge and Move. Whether we are Merging a new thing that will move (external Merge of a mover) or Moving an existing thing that will move again (internal Merge of a mover), we do something with the features of the main structure and possibly add the mover to the pronunciation thereof, and we add something to the mover list.

Operations Combine and Store are defined for each of the polarities of the relevant item in the pair  $(\pm\pm)$ . Combine will take the merging elements that have already had their active features checked, and determine whether the main structure will have the pronunciation of the  $t$  or if it will be silent. Store will take the moving element that has already had its active feature checked and determine whether it will be added to the mover list with its pronunciation or if it will be silent.

**Definition 3.4.19** (Combine and Store). For  $\alpha, \gamma$  sequences of features,  $s, t$  derived or atomic structures,  $x, y \in \{+, -\}$ ,  $f$  the combining function for the grammar:

$$\text{Combine}(\langle s, \alpha \rangle, \langle t, (\mathbf{xy})\mathbf{f}\gamma \rangle) = \begin{cases} \langle f(s, t) \rangle & \text{if } \mathbf{x} = + \\ \langle f(s, \epsilon) \rangle & \text{if } \mathbf{x} = - \end{cases}$$

$$\text{Store}(\langle t, (\mathbf{xy})\mathbf{f}\gamma \rangle) = \begin{cases} \langle t, (\mathbf{xy})\mathbf{f}\gamma \rangle & \text{if } \mathbf{y} = + \\ \langle \epsilon, (\mathbf{xy})\mathbf{f}\gamma \rangle & \text{if } \mathbf{y} = - \end{cases}$$

These function are now used in Merge and Move.

**Definition 3.4.20** (Merge for 4 kinds of Move). For  $\alpha, \beta$  sequences of features,  $s, t$  algebraic elements,  $f$  the combining function for the grammar:

$$\text{Merge}(\langle s, \mathbf{X}\alpha \rangle :: \text{movers}_s, \langle t, \mathbf{X}\beta \rangle :: \text{movers}_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta = \epsilon \\ \text{Combine}(\langle s, \alpha \rangle, \langle t, \beta \rangle) :: \text{Store}(\langle t, \beta \rangle) :: \text{movers}_s \cdot \text{movers}_t & \text{if } \beta \neq \epsilon \end{cases}$$

**Definition 3.4.21** (Move with 4 kinds of Move). For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  derived or atomic elements, suppose  $\exists! \langle t, \beta \rangle \in \text{movers}$  such that  $\beta = (\mathbf{xy})\mathbf{f}\gamma$  with  $x, y \in \{+, -\}$ .

Then:

$$\text{Move}(\langle s, +f\alpha, \rangle :: \text{movers}) = \begin{cases} \langle f(s, t), \alpha \rangle :: \text{movers} - \langle t, \beta \rangle & \text{if } \gamma = \epsilon \\ \text{Combine}(\langle s, \alpha \rangle, \langle t, \gamma \rangle) :: \text{Store}(\langle t, \gamma \rangle) :: \text{movers} - \langle t, \beta \rangle & \text{if } \gamma \neq \epsilon \end{cases}$$

This MG model I am proposing in this chapter may not in fact have the properties we want. Should, in fact, copying and deletion be treated as variations on Move? Are there empirical properties these operations all share?

A Java implementation using Combine and Store will shortly be available as part of the algebraic language toolkit Alto at <https://bitbucket.org/tclup/alto> and on its own at <https://github.com/megodoonch/MG>.

### 3.4.5 Constraints

MGs overtly contain one constraint: the shortest move constraint, or SMC. It is based on one of the economy considerations laid out in Chomsky (1993), in which movements should be in short steps. The SMC implements the idea that a probe moves the closest compatible goal by allowing only one mover per feature at a time.

The SMC is a transderivational constraint in that we are to prefer a derivation with a shorter move. Transderivational constraints can be taken to be in competition with one another, à la OT, as for example the competition between fewest and shortest moves – if you move everything phrase by phrase you get short but many moves; if you move something in one fell swoop, you get a single long move – or they can be taken to be absolute rules. MGs implement the SMC as an absolute rule, and can therefore build it into the definition of Merge and Move. *Fewest moves*, on the other hand, is built in only in that the application of Merge or Move is deterministic: it depends entirely on the featural configuration of the derivation. Therefore a derivation will have exactly the number of moves as the number of move features in the numeration.

Graf (2013) showed that in fact all transderivational constraints can be implemented as derivational rules in an MG and all derivational rules can be implemented as transderivational constraints. Therefore the distinction is mathematically unnecessary, though it can still be enlightening to keep them conceptually separate. These are big questions for future work.

### 3.4.5.1 Constraints we don't need

Some minimalist constraints fall out of the way an MG is designed: they are theorems, not restrictions. The No Tampering Condition and the Extension Condition<sup>11</sup> are natural examples. Because the derivation is bottom-up and Merge and Move are defined only between constituents, there is no way for a tree to grow downward and no way to change an existing structure Collins and Stabler (2015).

Heads cannot move into specifier position unless they are also complete phrases. This comes from the fact that the features triggering a constituent to move – the negative licensing features such as *-wh* – always follow all of the positive features that build up a complete phrase. Since the features are treated by the derivation as a stack, the move triggers are not usable until the phrase is complete. Since features percolate up, the constituent targeted by Move is the complete phrase. The fact that the negative licensing features always follow the category feature, which in turn follows all positive features, is not a constraint on the lexicon but rather a theorem about the grammar if a sentence is defined as something generated by the grammar that has exactly one feature, and that feature is a “final” category feature like *C*.

*Procrastinate* requires that movement that can wait until after Spellout waits. A traditional MG has no reflex of such a notion at all. Covert movement is the reflex of post-Spellout movement. The result of waiting until after Spellout to move is the strings being

---

<sup>11</sup>For MGs without head movement, or for the Extension Condition and/or head movement defined in such a way that head movement does not by its very nature violate it

in their original position, but the features still being present. In an MG with covert move, The MG equivalent of “weak” features is  $\ominus f$ , and Procrastinate simply means that  $\ominus f$  can only trigger covert move; the choice between covert and overt is not optional.

*Greed* requires that movement always satisfies some requirement of the mover. This is true of MGs because lexical items must have their move-triggering features – their negative licensing features – after the category feature. This means the whole phrase will always move, and nothing else can trigger movement, so movement is always to satisfy a requirement of the mover.

## CHAPTER 4

### Minimalist Grammars with Adjunction

Sections 4.1 to 4.8 of this chapter are based on a published paper, Fowlie (2014).

#### 4.1 Introduction

Adjuncts are optional,<sup>1</sup> meaning the sentence is grammatical without them. For example, in (1-a), *red* is optional. They are transparent to selection in that the selector seems to select for the features of the head, not those of the intervening adjunct. For example, in (1-b), the gender of *boek* ‘book’ is neuter. The intervening adjective does not have gender agreement, so *het* selects *boek* for its gender, regardless of the intervening adjunct.

- (1) a. The (red) rose **Optionality**  
b. Het mooi-e boek  
the.NEU beautiful-DEF book  
‘The beautiful book’ **(Dutch) Transparency**

Many languages have a default order for adjuncts, with unmarked intonation and without special scopal meaning. For example, English has ordered adjectives.

- (2) a. Wear the enormous ugly green hat  
*Wear the hat that is enormous, ugly, and green*

---

<sup>1</sup>Arguments have been made for (rare) obligatory adjuncts (Grimshaw and Vikner, 1993), which I will address in section 4.7.2 below.

b. #Wear the ugly enormous green hat

*Of your enormous green hats, wear the ugly one.*

Adjuncts differ from arguments primarily in the direction of their dependencies. Arguments and their selectors are mutually dependent: it is the fact that the argument is needed by the selector that licenses the presence of the argument, and it is the presence of the argument that makes the selector complete. For example in the sentence *She slept*, the verb *slept* requires a sleeper and is not allowed unless it has one, and the DP *she* is allowed in the sentence because *slept* requires it. Adjuncts, on the other hand, require a phrase to adjoin to, but are not required by anything. For example, in *She slept soundly*, the adverb *soundly* requires a VP (or something along those lines) to adjoin to, but the VP *slept* does not require an adverb. It is for this reason that in a classic phrase structure grammar, the internal structure of specific XPs can be determined entirely by the requirements of the head, but phrase structure rules (or some equivalent) are still required to describe the distribution of adjuncts. In an MG, the requirements of the head are modelled with the feature stack preceding the category feature; however, no mechanism equivalent to phrase structure rules for adjuncts exists in MGs.

To replace this missing mechanism, I propose a separate function that I dub *Adjoin*, along with a listing in the grammar of what categories can adjoin to what categories. This listing, together with the action of *Adjoin* that builds the right structure, is equivalent to the missing phrase structure rules. For example, if in a PSG we had the rule  $NP \rightarrow AP NP$ , we could just as easily say *AP's are among the adjuncts of NPs* if we also had an understanding that adjuncts always adjoin after the requirements of the head are fulfilled and if we knew what side of the NP the AP should appear on.

I argue that without such a separate mechanism we are forced to treat adjunction as selection, which makes wrong predictions.

On top of this selection/adjunction distinction, there is a second, more difficult issue:

that of adjunct ordering. We have already seen that adjuncts tend to follow quite a strict order cross-linguistically, and that this order cannot always be explained by semantic factors. Which adjuncts behave the same way as each other does seem to be semantic, but how these classes of adjuncts are ordered is not always semantically explicable. This chapter supposes that we do in fact need to account for at least some of these ordering facts in the syntax. The intention is to take a careful look at what kinds of mechanisms the grammar would have to include for this to be possible.

I propose a grammar that uses a mechanism of numerical indexing of hierarchy levels. I do not argue that this is necessarily the right model, but rather that any syntactic model of hierarchies requires something to do the work of the numerical indices.

## 4.2 Minimalist Grammars

This section provides a brief overview of minimalist grammars. For a more complete picture, please see chapter 3.

I formulate my model as a variant of *Minimalist Grammars* (MGs), which are [Stabler \(1997\)](#)'s formalisation of [Chomsky \(1995\)](#)'s notion of feature-driven derivations using the functions Merge and Move. MGs are mildly context-sensitive, putting them in the right general class for human language grammars. They are also simple and intuitive to work with. Here I will briefly run over the definition of MGs.

I will give derived structures as strings as [Stabler and Keenan \(2003\)](#)'s grammar would generate them.<sup>2</sup> See chapter 3 for a more thorough discussion.

**Definition 4.2.1.** A *Minimalist Grammar* is a five-tuple  $G = \langle \Sigma, \text{sel}, \text{lic}, Lex, M \rangle$ .  $\Sigma$  is

---

<sup>2</sup>Stabler & Keenan's grammar also incorporates an additional element: lexical items are triples of string, features, and lexical status, which allows derivation of Spec-Head-Complement order. I will leave this out for simplicity, as it is not initially relevant here, as our interest is in spec/adjunct placement, which will always be on the left. Post-head adjuncts are discussed in Section 4.7.1.1. For convenience of English reading, I will give sentences in head-spec-complement order, but the formal definition I give here always puts the selected on the left and the selector on the right.

the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, X \mid f \in \mathbf{lic}, X \in \mathbf{sel}\}$  be the *features*.  $Lex \subseteq \Sigma \times F^*$ , and  $M$  is the set of operations **Merge** and **Move**. The language  $L_G$  is the closure of  $Lex$  under  $M$ . A set  $C \subseteq F$  of designated features can be added; these are the types of complete sentences.

Minimalist Grammars are *feature-driven*, meaning features of lexical items determine which operations can occur and when. There are two finite sets of features, *selectional* features  $\mathbf{sel}$  which drive the operation **Merge** and *licensing* features  $\mathbf{lic}$  which drive **Move**. **Merge** puts two derived or atomic structures together; **Move** operates on the already built structure. Each feature has a positive and negative version. Positive  $\mathbf{sel}$  and  $\mathbf{lic}$  features are  $=X$  and  $+f$  respectively, and negatives are  $X$  and  $-f$ . Intuitively, negative  $\mathbf{sel}$  features are the categories of lexical items. Merge and Move are defined over *expressions*: sequences of pairs  $\langle \text{derived structure, feature stack} \rangle$ . The first pair can be thought of as the “main” structure being built; the remaining are waiting to move. Positive  $\mathbf{sel}$  features are not unlike the “uninterpretable” category features of classical minimalism: features that had to be cancelled because they could not be interpreted by any interface.

An MG essentially works as follows: **Merge** takes two expressions and combines them into one if the first structure displays  $=X$  and the second  $X$  for some  $X \in \mathbf{sel}$ . The  $X$  features are deleted, after which the second structure may still have features remaining, meaning the second structure is going to move. It is stored separately by the derivation until the matching positive licensing feature comes up later in the derivation, when the moving structure is combined again; this is **Move**. **Move** also carries the requirement that for each  $f \in \mathbf{lic}$  there be at most one structure waiting to move. This is the *shortest move constraint (SMC)*.<sup>3</sup>

---

<sup>3</sup>The SMC is based on economy arguments in the linguistic literature (Chomsky, 1995), but it is also crucial for a type of finiteness: the valid derivation trees of an MG form a regular tree language (Kobele et al., 2007). The number of possible movers must be finite for the automaton to be finite-state. The SMC could also be modified to allow up to a particular (finite) number of movers for each  $f \in \mathbf{lic}$  in order to account for phenomena such as multiple wh-fronting.



For example, we can put *saw* and *him* together to make *saw him* if the features of *saw* are  $=D =D V$  and the features of *him* are just  $D$  (Figure 4.1a).

If we Merge *saw* with *who*, we don't put them together since *who* is going to wh-move anyway. Instead we store it for later (Figure 4.1b).



Figure 4.1: Merge

Merge is defined formally as follows:

**Definition 4.2.2 (Merge).** For  $\alpha, \beta$  sequences of features,  $s, t$  derived structures,  $\text{mvr}_{s,t}$  expressions:<sup>4</sup>

$$\text{Merge}(s : =X\alpha :: \text{mvr}_s, t : X\beta :: \text{mvr}_t) = \begin{cases} ts : \alpha :: \text{mvr}_s \cdot \text{mvr}_t & \text{if } \beta = \epsilon \\ (s : \alpha) :: (t : \beta) :: \text{mvr}_s \cdot \text{mvr}_t & \text{if } \beta \neq \epsilon \end{cases}$$

The function Adjoin is based on Merge, so I will not go into detail on Move here. See Chapter 3 for a thorough introduction. Here I give just the definition.

**Definition 4.2.3 (Move).** For  $\alpha, \beta, \gamma$  sequences of features,  $s, t$  derived structures,  $\text{mvr}_s$  an expression, suppose there is a unique  $\langle t, \beta \rangle \in \text{mvr}_s$  such that  $\beta = -f\gamma$ . Then:

$$\text{Move}(s : +f\alpha :: \text{mvr}_s) = \begin{cases} ts : \alpha :: \text{mvr}_s & \text{if } \gamma = \epsilon \\ s : \alpha :: t : \gamma :: \text{mvr}_s - t : \beta & \text{if } \gamma \neq \epsilon \end{cases}$$

In this chapter I will make use of *derivation trees*, which are trees describing the derivation. They may also be annotated: in addition to the name of function, I (redundantly)

<sup>4</sup>:: adds an element to a list;  $\cdot$  appends two lists.

include for clarity the derived expressions in the form of strings and features. For example, figure 4.2 shows derivation trees (annotated and unannotated) of *the wolf* with feature D.

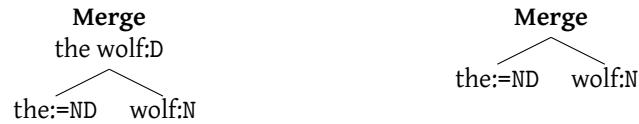


Figure 4.2: Annotated and unannotated derivation trees

### 4.3 Cartography

The phenomena my model is designed to account for are modifiers and other apparently optional projections such as those in (3). (For more data and discussion, please see chapter 2.) We will see, though, that it is more successful with adjectives and adverbs than functional projections. Section 4.12 improves on the adjunction model presented in section 4.6.

- (3) a. The small ancient triangular green Irish pagan metal artifact was lost.  
 b. \*The metal green small artifact was lost. **Adjectives**  
 c. Frankly, John probably once usually arrived early.  
 d. \*Usually, John early frankly once arrived probably. **Adverbs**  
 e.  $\begin{bmatrix} \text{DP} & \text{zhe} & [\text{NumP} & \text{yi} & [\text{ClP} & \text{zhi} & [\text{NP} & \text{bi}]]] \\ \text{DP} & \text{this} & [\text{NumP} & \text{one} & [\text{ClP} & \text{CL} & [\text{NP} & \text{pen}]]] \\ \text{'this pen'} \end{bmatrix}$  **Functional projections**

These three phenomena all display *optionality*, *transparency to selection*, and *strict ordering*. By *transparency* I mean that despite the intervening modifiers, properties of the selected head are relevant to selection. For example, in a classifier language, the correct classifier selects the noun even if adjectives intervene.

- (4) a. te' mexha  
 CL<sub>WOOD</sub> table  
 'the table' Q'anjob'al<sup>5</sup>
- b. te' yalixh mexha  
 CL<sub>WOOD</sub> small table  
 'the small table'
- c. \*no' yalixh mexha  
 CL<sub>ANIMAL</sub> small table

The hypothesis that despite their optionality these projections are strictly ordered is part of *syntactic cartography* (Rizzi, 2004). Cinque (1999, 2010), in particular proposes a universal hierarchy of functional heads that select adverbs in their specifiers, yielding an order on both the heads and the adverbs. He proposes a parallel hierarchy of adjectives modifying nouns. These hierarchies are very deep. The adverbs and functional heads incorporate 30 heads and 30 adverbs.

Cinque argues that the surprising universality of adverb order calls for explanation. For example, Italian, English, Norwegian, Bosnian/Serbo-Croatian, Mandarin Chinese, and more show strong preferences for *frankly* to precede *unfortunately*. These arguments continue for a great deal more adverbs.<sup>6</sup>

- (5) Italian
- a. **Francamente** ho *purtroppo* una pessima opinione di voi.  
**Frankly** have *unfortunately* a bad opinion of you  
 'Frankly I unfortunately have a very bad opinion of you.'
- b. \**Purtroppo* ho francamente una pessima opinione di voi.  
*Unfortunately* have **frankly** a bad opinion of you
- (6) English
- a. **Frankly**, I *unfortunately* have a very bad opinion of you

---

<sup>5</sup>Data from Bervoets et al. (2011)

<sup>6</sup>Data from Cinque (1999)

- b. %*Unfortunately* I **frankly** have a very bad opinion of you
- (7) Norwegian
- a. Per forlater [**rerlig talt**] [*heldigvis*] [nil] selskapet.  
Peter leaves [**honestly spoken**] [*fortunately*] [now] the.party.  
'Frankly, Peter is fortunately leaving the party now.'
- b. \*Per forlater [*heldigvis*] [**rerlig talt**] [nil] selskapet.  
Peter leaves [*fortunately*] [**honestly spoken**] [now] the.party.
- (8) Bosnian/Serbo-Croatian
- a. **Iskreno**, ja *naialost* imam jako lose misljenje o vama  
**Frankly**, I *unfortunately* have very bad opinion of you.  
'Frankly, I unfortunately have a very bad opinion of you.'
- b. \**Naialost*, ja **iskreno** imam jako lose misljenje o varna.  
*unfortunately* I **frankly** have very bad opinion of you.
- (9) Mandarin Chinese
- a. **laoshi-shuo** wo *buxing* dui tamen you pian-jian.  
**Frankly**, I *unfortunately* to them have prejudice  
'Honestly I unfortunately have prejudice against them.'
- b. \**buxing* wo **laoshi-shuo** dui tamen you pian-jian.  
*unfortunately* I **Frankly** to them have prejudice

#### 4.4 Desiderata

In addition to these three main properties, an account of adjuncts should ideally also account for the following: a lack of adjunction to multiple categories at once (Graf, 2014), selectability of adjunct categories, adjuncts of adjuncts, unordered adjuncts, and so-called obligatory adjuncts. These phenomena are discussed in more detail in Chapter 2.

We also have the following data:

- (10) Mary is **tall** *tall* is selected by *is*<sup>7</sup>

- (11) The **surprisingly short** basketball player *surprisingly* modifies *short*<sup>8</sup>
- (12) a. The alliance officer shot Kaeli **in the cargo hold** *with a gun*.  
 b. The alliance officer shot Kaeli *with a gun* **in the cargo hold**. *in situ* English PP adjuncts are unordered
- (13) a. He makes a **good** father *good* is an adjunct but is not optional  
 b. \*He makes a father  
 c. She worded the letter **carelessly**.  
 d. ...and Marc did so **carefully**. *carefully* is an adjunct  
 e. \*She worded the letter. yet it is not optional

Please see Chapter 2 for a careful empirical survey of these and other properties of adjunction. In sum, an account of adjuncts in minimalist grammars should ideally have the properties in Fig. 4.3:

---

<sup>7</sup>Or, a small clause *Mary tall* is selected by *is*; in this case there is some kind of selection that brings *Mary* and *tall* together, perhaps a silent head selecting both ( $\epsilon ::= A \Rightarrow D$  SC). If selectable adjectives are not convincing, consider instead selectable and adjoinable PPs and CPs:

- (i) a. Kaylee was shot [**in the cargo hold**]<sub>PP</sub> PP can be adjoined  
 b. Kaylee put the contraband [**in the cargo hold**]<sub>PP</sub> PP can be selected
- (ii) a. River, [**who is crazy**]<sub>CP</sub>, laughed. CP can be adjoined  
 b. Jayne thinks [**that River is crazy**]<sub>CP</sub> CP can be selected

<sup>8</sup>Not all adverbs can modify adjectives. In chapter 2 I argue that most of the restriction is semantic, so in this chapter I will simplify and aim to model adverbs modifying adjectives in general.

### Desiderata

1. **Optionality:** sentences should be grammatical with or without adjuncts
2. **Transparency to selection:** If a phrase P is normally selected by head Q, when P has adjuncts Q should still select P.
3. **Order:** there should be a mechanism for forcing an order on adjuncts
4. **Selectability (10)**
  - (a) **Efficiency:** All<sup>9</sup> adjectives are possible arguments of the same predicates, so there should be as few distinct categories of adjectives as possible (ideally just one, say A).
5. **Adjuncts of adjuncts (11)**
  - (a) **Efficiency:** Similarly to selection, there are large classes of adjuncts that adjoin to the same things; their members should not be listed separately. For example, most or all English adverbs are adjuncts of adjectives. There should be as few separate categories of adverbs as possible.
6. **Unordered** Some adjuncts are unordered with respect to each other. (12)
7. **Obligatory adjuncts** Occasionally, adjuncts are obligatory. (13)

Figure 4.3: Desiderata

## 4.5 Previous Approaches to Adjunction

This section provides a brief overview of four approaches to adjunction. The first two are from a categorial grammar perspective and account for the optionality and, more or less,

---

<sup>9</sup>or at least almost all. Consider the following (Carson Schutze, p.c.)

- (i) a. Jayne seems tired
- b. \*Jayne seems alleged

Again, we simplify slightly for the time being.

transparency to selection; however, they are designed to model unordered adjuncts. The next two are MG formalisations of the cartographic approach. Since the cartographic approach takes adjuncts to be regular selectors, unsurprisingly they account for order, but not easily for optionality or transparency to selection. Finally, I will briefly outline an earlier (published) version of my own solution.

#### 4.5.1 Traditional MG solution

To account for the optionality and transparency, a common solution is for a modifier to combine with its modified phrase, and give the result the same category as the original phrase. Traditionally in MGs, an X-modifier has features =XX: it selects an X and the resulting structure has category feature X. Similarly, in categorial grammars (see section 2.6 of chapter 2), an X-modifier has category X/X or X\X. As such, the properties of traditional MG and CG models of adjunction are the same.<sup>10</sup>

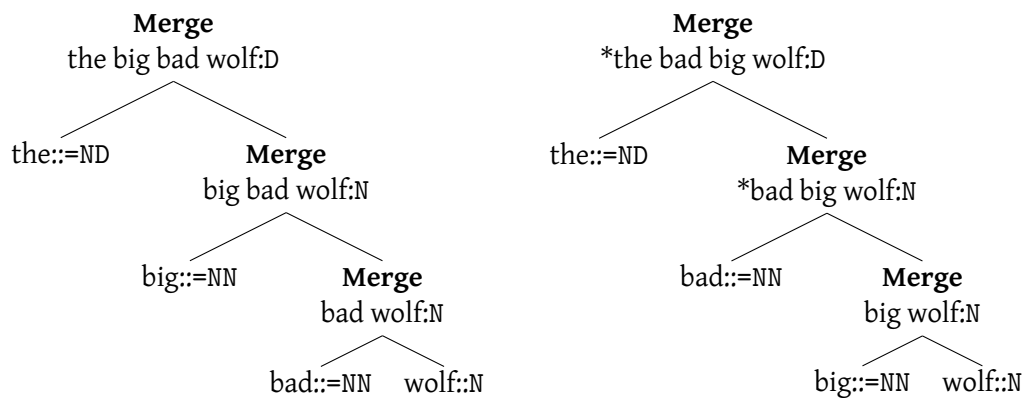


Figure 4.4: Traditional MG approach

<sup>10</sup>This is not the only possible solution using the MG architecture, but rather the traditional solution. Section 4.5.3 gives a model within MGs that accounts for order.

An anonymous reviewer of Fowlie (2014) suggested a different solution, with a set of silent, meaningless heads that turn categories into selectors of their adjuncts, for example  $\epsilon::=N =Adj =N$ . Such a solution does much better on desiderata 4 and 5 than the one given here, but shares with the cartographic solution given in section 4.5.3 the problem of linguistic undesirability of silent, meaningless elements.

This is the same as the traditional categorial grammar approach, where an N-modifier is given the slash category  $N/N$  (meaning it needs an N on the right to make an N) or  $N \backslash N$  (meaning it needs an N on the left to make an N).

This system makes adjuncts truly optional, and allows for unordered adjuncts, but does not capture anything else. It cannot account for ordering. This is because the category of the new phrase is the same regardless of the modifier's place in the hierarchy. That is, the very thing that accounts for the optionality and the transparency of modifiers (that the category does not change) is what makes strict ordering impossible. Moreover, the modifier is not truly transparent to selection: the modifier in fact becomes the new head; it just happens to share a category with the original head. This can be seen in tree-generating grammars such as [Stabler \(1997\)](#) (figure 4.5).



Figure 4.5: Derivation tree and derived *bare tree*. The < points to the head, *big*.

1. **Optionality:** ✓ the original category is kept
2. **Transparency to selection: Sort of:** in Fig.4.4, *the* selects N, but the N it checks is the one introduced by *bad*, not the one on *wolf*. This makes *big* the head of the noun phrase *big wolf*, but it should be *wolf*, as shown in Fig. 4.5.
3. **Order:** ✗ The original category is kept so any adjunct may adjoin at any time.
4. **Selectability** Adjuncts need two versions, one for being adjuncts and the other for being selected. For example, *bad::=NN* cannot be selected by anything until it has itself selected an N. We need a second version of *bad*, with, say, category A, so that it can be selected by such words as *is* (Fig. 4.6).
  - (a) **Efficiency:** ✗ Adjuncts have two versions.



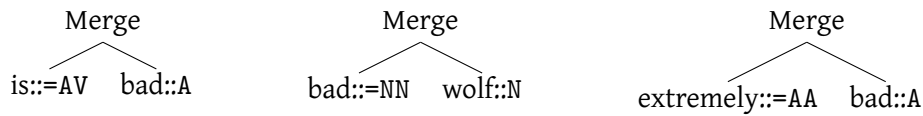


Figure 4.6: Selected adjective, modifier adjective, adjoined-to adjective

5. **Adjuncts of adjuncts:** ✗ Since adjunction is selection in this model, we have the same problem, but with the same solution: the feature for selection is also the feature for being adjoined to (Fig. 4.6).

(a) **Efficiency:** The homophony for selection covers adjunction too.

6. **Unordered:** ✓ All adjuncts are unordered in this model
7. **Obligatory adjuncts:** ✗ There is no way to distinguish between an phrase with an adjunct and one without.

#### 4.5.2 Frey & Gärtner

Frey and Gärtner (2002) propose an improved version of the categorial grammar approach, one which keeps the modified element as the head, giving true transparency to selection. They do this by asymmetric feature checking. To the basic MG formalism a third polarity is added for **sel**,  $\approx X$ . This polarity drives the added function **Adjoin**. **Adjoin** behaves just like **Merge** except that instead of cancelling both  $\approx X$  and  $X$ , it cancels only  $\approx X$ , leaving the original  $X$  intact. This allows the phrase to be selected or adjoined to again by anything that selects or adjoins to  $X$ . This model accounts for optionality and true transparency, but it is not designed to capture ordered adjuncts. Also, since adjuncts don't have categories of their own (just  $\approx X$ ), it is not clear how best to model selection of and adjunction to adjuncts.

1. **Optionality:** ✓ the original category is kept



Figure 4.7: Frey & Gärtner: derivation tree and derived *bare tree*. The > points to the head, *wolf*.

2. **Transparency to selection:** ✓ For example, in the example in Figure 4.8, *the* selects N, which is introduced by *wolf*.
3. **Order:** ✗ The original category is kept so any adjunct may adjoin at any time (Fig. 4.8).

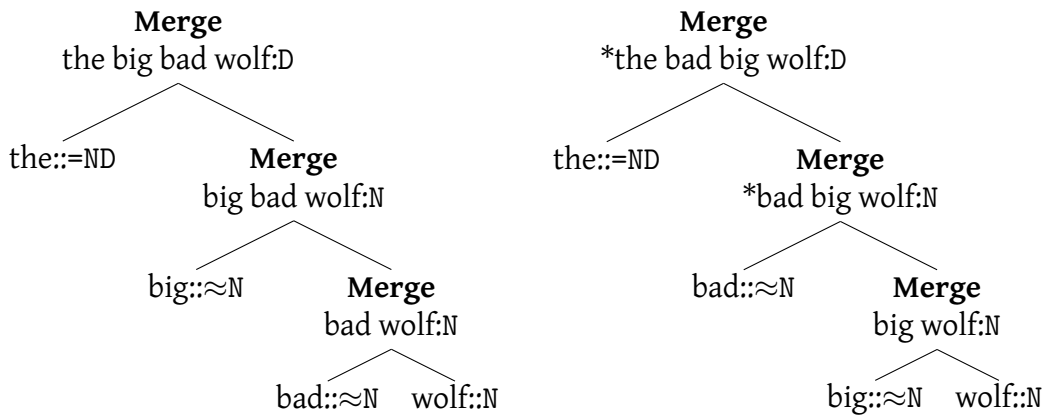
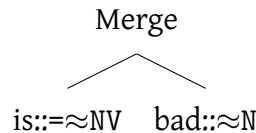


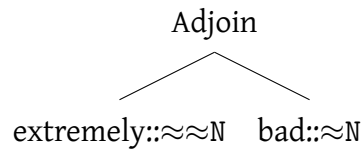
Figure 4.8: F & G derivations of *the big bad wolf* and *\*the bad big wolf*

4. **Selectability** ✓ Frey & Gärtner allow (in one version at least)  $\approx X$  to be selected, yielding  $=\approx X$  features for items that select adjuncts of X. This solution is not ideal, as not all adjuncts of nouns can be selected by *be*, for example relative clauses, as in *\*John is that left*.



- (a) **Efficiency:** ✓ No homophony required

5. **Adjuncts of adjuncts** Not clear: F& G do not give an account of adjuncts of adjuncts, but their account could presumably be analogously extended to include a limited number of  $\approx$  symbols, say up to 3 or 4.



(a) **Efficiency:** ✓

6. **Unordered** ✓ All adjuncts are unordered in this model
7. **Obligatory adjuncts** ✗ There is no way to distinguish between a phrase with an adjunct and one without.

The models seen so far are summarised in Table 4.1.

<b>Model Section</b>	Trad 4.5.1	F&G 4.5.2
Optional	✓	✓
Efficiency (opt)	✓	✓
Transparent	?	✓
Order	✗	✗
Selectability	✓	✓
Efficiency (Sel)	✗	✓
Adj of adj	✓	?
Efficiency (Adj of adj)	✗	✓?
Unordered	✓	✓
Oblig	✗	✗

Table 4.1: Summary of models re: desiderata

### 4.5.3 Selectional approaches

A third approach is to treat adjuncts just like any other selector. This is the approach taken by syntactic cartography. Such an approach accounts straightforwardly for order,

but not for optionality or transparency; this is unsurprising since the phenomena I am modelling share only ordering restrictions with ordinary selection.

The idea is to take the full hierarchy of modifiers and functional heads, give each their own category names, and have each select the one below it; for example, *big* selects *bad* but not vice versa, and *bad* selects *wolf*. (See Figure 4.9.) However, here we are left with the question of what to do when *bad* is not present, and the phrase is just *the big wolf*. *big* does not select *wolf* (Fig. 4.9). I will give two solutions, one in which the full structure is always present and one in which it is not.

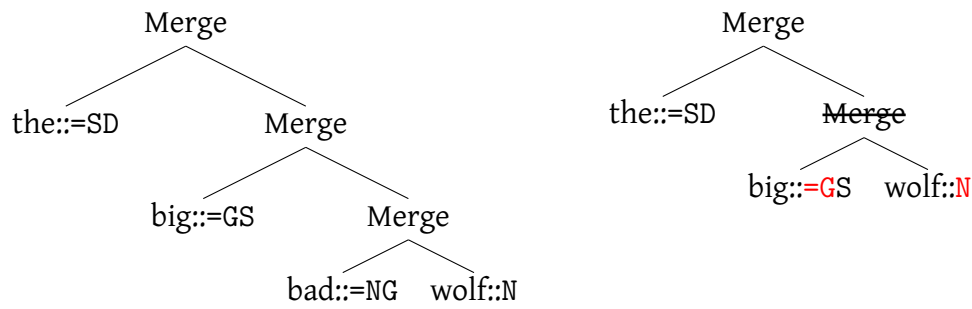


Figure 4.9: Selectional approach

#### 4.5.3.1 Silent, meaningless heads

We give each modifier and functional head a silent, meaningless version that serves only to tie the higher modifier to the lower, like syntactic glue. For example, we add to the lexicon a silent, meaningless “size” modifier that goes where *big* and *small* and other LIs of category S go.

$\langle \text{the}, =S D \rangle$      $\langle \epsilon, =S D \rangle$      $\langle \text{wolf}, N \rangle$   
 $\langle \text{big}, =G S \rangle$      $\langle \epsilon, =G S \rangle$      $\langle \text{bad}, =N G \rangle$      $\langle \epsilon, =N G \rangle$

This solution doubles substantial portions of the lexicon. Doubling is not computationally significant, but it does indicate a missing generalisation: somehow, it just hap-

pens that each of these modifiers has a silent, meaningless doppelgänger. Relatedly, the ordering facts are epiphenomenal. There is no universal principle predicting the fairly robust cross-linguistic regularity. Moreover, normally when something silent is in the derivation, we want to say it is contributing something semantically. Here these morphemes are nothing more than a trick to hold the syntax together.

In favour of these heads lacking in both phonetic and semantic content is the efficiency gains silent elements can give a grammar. For example, to remove a silent element from a context-free grammar and still generate the same language, we find every category that could end up with a silent pronunciation (*nullable* category), and then for every rule in which it occurs, a new rule that lacks that nullable category is added to the grammar. For example if we have a rule  $A \rightarrow \epsilon$  and another  $C \rightarrow A B$ , then we add a new rule  $C \rightarrow B$ . Thus silent elements can in fact make grammars *more* efficient.

However, the silent elements that can increase efficiency of grammars need not necessarily be meaningless. These silent heads in human language could just as easily be universally meaningful. For instance, suppose proper names are always Ns, and in Italian they are selected by a D *il* or *la* to make a DP, while in English are selected by a silent D, also forming a DP. The silent English D could be argued to carry the same meaning as the overt Italian one.

- (14) a. *il* Paolo  
the<sub>M</sub> Paul  
'Paul'
- b. *la* Paola  
the<sub>F</sub> Paula  
'Paula'

Moreover, items that have nothing interpretable at either interface run counter to minimalist principles.

1. **Optionality:** ✓ Choose the right version of an LI. Note this is **inefficient**.
2. **Transparency to selection:** ✗ Selection is of the adjunct, not the head. For example, in the lexicon above, *the* selects the (possibly empty) adjunct with features =GS, not the noun.
3. **Order:** ✓ This is Merge, so order is determined by the particular lexical items' feature stacks.<sup>11</sup>
4. **Selectability** ✓ This is ordinary Merge, so selection proceeds as usual.
  - (a) **Efficiency:** ✗ We need to add selectable versions of adjuncts; eg *big::=GS* needs to be paired with *big::S*. Selectors of adjectives and adverbs also need multiple versions, for example *be::=SV*, *be::=GV*, etc.
5. **Adjuncts of adjuncts** ✓ Adjuncts of adjuncts are simply selectors of adjuncts.
  - (a) **Efficiency:** ✗ Adjuncts need to select all the adjuncts they adjoin to. eg: *very::=S Int*, *very::=G Int* etc.
6. **Unordered** ✓ Here we use the traditional features, =XX, but we might need a version for each category of, say, adjectives, depending on where we think unordered adjuncts adjoin.
7. **Obligatory adjuncts:** ✗ The same thing that makes adjuncts optional makes it impossible to require them.

If we are willing to accept a model with large amounts of structure that contribute nothing to either interface, we can stop here. This model does succeed in capturing the ordering facts, and the other unmet desiderata are arguably not fatal. If not, we must look further.

---

<sup>11</sup>Note that there is a missing generalisation here: nothing forces a particular order cross-linguistically, but rather the order is an epiphenomenon of the particular lexical items. This shortcoming is an example of a general problem for MGs: nothing forces, say, verbs to select DP.

### 4.5.3.2 Massive homophony

A second selectional approach is for each morpheme in the hierarchy to have versions that select each level below it. For example, *the* has a version which selects N directly, one that selects “goodness” adjectives like *bad*, one that selects “size” adjectives like *big*, and indeed one for each of the ten or so levels of adjectives.

⟨the, =SD⟩      ⟨the, =GD⟩    ⟨the, =SD⟩    ⟨the, =ND⟩  
 ⟨big, =GS⟩      ⟨big, =NatS⟩    ⟨big, =NS⟩  
 ⟨bad, =NatG⟩    ⟨bad, =NG⟩  
 ⟨Canadian, =NNat⟩  
 ⟨wolf, N⟩

This second solution lacks the strangeness of silent, meaningless elements, but computationally it is far worse. To compute this we simply use Gauss’s formula for adding sequences of numbers, since an LI at level  $i$  in a hierarchy has  $i$  versions. For example, in the model above, *the* is at level 4 (counting from 0), and there are 4 versions of *the*. For a lexicon *Lex* without these duplicated heads, and a language with  $k$  hierarchies of depths  $l_i$  for each  $1 \leq i \leq k$ , adding the duplicated heads increases the size of the lexicon. The increase is bounded below by a polynomial function of the depths of the hierarchies as follows:<sup>12</sup>

$$|\text{Lex}'| \geq \sum_{i=1}^k 1/2(l_i^2 + l_i) + |\text{Lex}|$$

This approach has similar properties as the silent meaningless heads approach in terms of our desiderata. Table 4.2 summarises the models seen so far.

---

<sup>12</sup>I say “bounded below” because this formula calculates the increase to the lexicon assuming there is exactly one LI at each level in the hierarchy. If there are more, each LI at level  $i$  of a hierarchy has  $i$  versions as well.

<b>Model Section</b>	Trad 4.5.1	F&G 4.5.2	Syn. glue 4.5.3.1	Homoph 4.5.3.2
Optional	✓	✓	✓	✓
Efficiency (opt)	✓	✓	✗	✗✗
Transparent	?	✓	✗	✗
Order	✗	✗	✓	✓
Selectability	✓	✓	✓	✓
Efficiency (Sel)	✗	✓	✗	✗
Adj of adj	✓	?	✓	✓
Eff (Adj of adj)	✗	✓?	✗	✗
Unordered	✓	✓	✓	✓
Oblig	✗	✗	✗	✗

Table 4.2: Summary of models re: desiderata

#### 4.5.4 Fowlie (2013)

In an earlier paper I proposed a way to handle adjuncts that combined some of the benefits of Frey & Gärtner’s approach with those of Cinque. The grammars I will propose in this paper are improvements on my previous work. I will briefly outline the previous work here for comparison. It is also the version Graf (2014) analyses (see Section 4.11), though the results are the same for the new model.

Like Frey & Gärtner, I add to MGs an operation **Adjoin**. Unlike their proposal, Fowlie (2013) does not add a polarity but rather a function **Ad** that maps categories to their adjuncts. It also changes categories from single category names to pairings of the category of the head with the category of the last adjunct adjoined to the phrase:  $[X, A]$ . For example, in 4.10, the category of *wolf* is  $[N, N]$ , but when the category  $G$  adjunct *bad* adjoins to it, the resulting phrase is of category  $[N, G]$ : a noun phrase whose last adjunct was of category  $G$ .



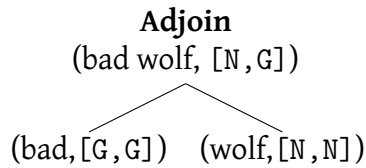


Figure 4.10: Valid derivation of *bad wolf*

Two additional structures are added to the grammar: a partial order on the category features and a partial function **Ad** from category features to sets of their adjuncts. The validity of an application of **Adjoin** is determined by two factors: whether the adjunct is an adjunct of the phrase it is adjoining to, and whether the category of the adjunct is at least as high in the partial order as the last adjunct adjoined, which is encoded as the second element of the pair of category features. For example, suppose  $G, S \in \text{Ad}(N)$ , and suppose  $S \geq G \geq N$ . Then the first derivation in 4.11 is valid because  $G$  is an adjunct of  $N$  and  $G \geq N$ . The second derivation is bad because the second adjunct is too low ( $G > S$ ).

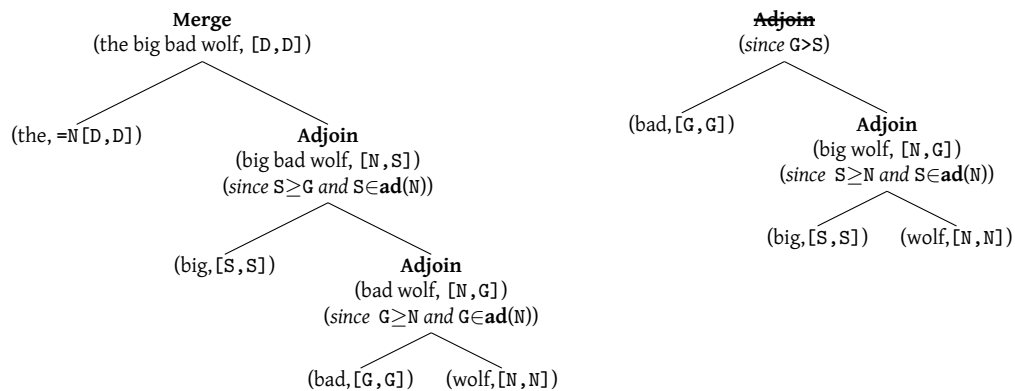
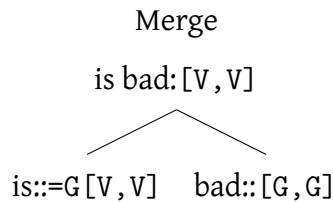


Figure 4.11: Valid derivation of *the big bad wolf* and attempted derivation of *\*the bad big wolf*

1. **Optionality:** ✓ the original category is kept as the first element of the category pair
2. **Transparency to selection:** ✓ the original category is kept

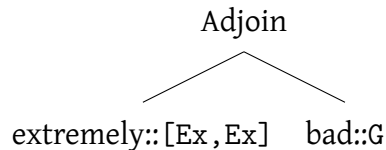
3. **Order:** (✓) The second element of the category is the last adjunct adjoined. There is an order on the adjunct categories and the Adjoin rule requires that the adjunct be at least as high in the order as that second element of the category pair. However, as we will see in section 4.11.1.1, both this model and the following run into a problem when Merge intervenes between adjunctions.

4. **Selectability** ✓ Adjuncts have regular categories



(a) **Efficiency:** ✗ Each adjunct has its own category so an LI that selects all, say, adjectives, needs a version for each adjective category, for example is::=S V, is::=G V, is::=Nat V, is::=M V, ...

5. **Adjuncts of adjuncts** ✓ Adjunct categories are ordinary categories so they can have adjuncts too.



(a) **Efficiency:** ✗ Each adjunct has its own category so the adjunct sets will be inefficiently listed, missing the generalisation that, say, all adverbs modify all adjectives. e.g.  $\text{Ad(S)=Ad(G)=Ad(Nat)=Ad(M)=...}$   
 $=\{\text{Adv}_1, \text{Adv}_2, \text{Adv}_3 \dots, \text{Int(ensifier)}\}$ .

6. **Unordered** ✓ An extension of this model handles unordered adjuncts. It will be recapitulated for the new proposal in section 4.7.1.1 below.

7. **Obligatory adjuncts: Maybe.** Until a modified phrase is selected, it is distinguishable from an unmodified phrase. For example, an unmodified noun has category  $[N, N]$  while a modified noun has category  $[N, A]$  for some  $A \neq N$ . This might be exploitable for a special case of Merge. This question will be raised again in the new proposal, and the shape of any solution would be similar (Section 4.7.2).

Where this approach falls short is in adjunction to and selection of categories that are usually adjuncts. While it is possible, a generalisation is missed. The problem is that there is no unifying feature for adjectives or adverbs, since the distinctions are accounted for by giving them different categories. To define adjuncts of Adjectives, we have to give **Ad** as assigning the same adjunct set to all adjectives:

$$\mathbf{Ad}(S)=\mathbf{Ad}(G)=\mathbf{Ad}(\text{Nat})=\mathbf{Ad}(M)=\dots=\{\text{Adv}_1, \text{Adv}_2, \text{Adv}_3 \dots, \text{Int}(\text{ensifier})\}.$$

Similarly, selection of adjectives must be specified for each adjective. For example, *be* selects all adjectives, as in:

(15) She is silly/Canadian/brown-haired/tall/ugly...

As it stands, *be* must be cross-classified across adjective categories, multiplying the lexicon and missing a generalisation:

(16)  $\langle \text{is}, =S \ V \rangle, \langle \text{is}, =G \ V \rangle, \langle \text{is}, =\text{Nat} \ V \rangle, \langle \text{is}, =M \ V \rangle, \dots$

The proposal in this chapter fixes this shortcoming. Table 4.3 summarises the models seen so far.

<b>Model Section</b>	Trad 4.5.1	F&G 4.5.2	Syn. glue 4.5.3.1	Homoph 4.5.3.2	Fowlie13 4.5.4
Optional	✓	✓	✓	✓	✓
Eff (opt)	✓	✓	✗	✗✗	✓
Transparent	?	✓	✗	✗	✓
Order	✗	✗	✓	✓	(✓)
Select	✓	✓	✓	✓	✓
Eff (Sel)	✗	✓	✗	✗	✗
Adj of adj	✓	?	✓	✓	✓
Eff (Adj)	✗	✓?	✗	✗	✗
Unordered	✓	✓	✓	✓	✓
Oblig	✗	✗	✗	✗	?

Table 4.3: Summary of models re: desiderata

## 4.6 Proposal 1: Minimalist Grammars with Adjunction

In this section and in section 4.12.1 I propose solutions, which I call *Minimalist Grammars with Adjunction (MGAs)*<sup>13</sup> and *Minimalist Grammars with Hierarchies (MGHs)*. The former model fails to fully capture the behaviour of functional heads; the latter solves this problem, but at the expense of a fourth operation, a variation on Merge designed for hierarchically organised functional heads. Both models account for ordering by indexing phrases according to the hierarchy level of the last adjunct adjoined to them; MGHs use these hierarchies in one of their Merge operations as well.

A given adjunct phrase  $P$  needs four pieces of information:  $P$ 's category, what  $P$  is an adjunct of, what level adjunct  $P$  is, and what level the last adjunct that adjoined to  $P$  had. We need to know what a category is an adjunct of because that will determine whether, say, an adjective can adjoin to a noun phrase. I include in the grammar a set of adjuncts for each category. The hierarchy level of the adjunct (encoded as a number) is needed

---

<sup>13</sup>My earlier paper Fowlie (2013) described in section 4.5.4 used this name as well; this model was designed to improve on it.

for when it acts as an adjunct. If the phrase it is adjoining to already has an adjunct, we need to check that the new adjunct is higher (or at the same level) in the hierarchy. For this purpose, every phrase carries with it an additional number, indexing the level of its last adjunct. The two numbers are kept separate so that adjuncts can have adjuncts, as in *bright blue*. *Bright blue* has an adjunct *bright*, which may affect what further adjuncts can adjoin to it, but which does not affect what the phrase *bright blue* can adjoin to.

To index hierarchy levels I use natural numbers and the usual order  $\leq$  on  $\mathbb{N}$ .<sup>14</sup> Any index set would do, and in fact I claim these numbers correspond to semantic classes of adjuncts such as “Size” and “Goodness”. This reflects the similarity of adjunct ordering across languages; without semantic classes, the apparent universality of Cinque’s hierarchies would be accidental. For example, level 6 categories might be the Size class, and level 4 the Goodness class. This hierarchy level–semantic class correspondence is universal across languages, making, for example, *good* and *bad* belong to level 6 in English, and *goed* ‘good’ and *slecht* ‘bad’ belong to level 6 in Dutch.

To track hierarchy level, each category feature is expanded into a triple consisting of the category feature, the *adjunct level*, and the *phrase level*, where the adjunct level is the level of the hierarchy of adjuncts the head belongs to, and the phrase level is the level of hierarchy the whole phrase is at. These numbers are lexically specified; for example *bad*::[A, 4, 0] would be in the lexicon. Since these numbers are really meant to represent semantic categories, the lexical item should look something more like *bad*::[A, Goodness, Base], where Goodness is a semantic category of adjective, and Base stands for whatever the lowest level of adjunct is. The grammar then includes, rather than the ordering on the natural numbers, an ordering on semantic categories. However, throughout this chapter I will use numbers, for ease of reading, since we know what order natural numbers come in without having to check the order every time.

---

<sup>14</sup> $\mathbb{N}$  is simply acting as an index set, and the maximal depth of hierarchies in a language bounds the actual index set for the grammar.

By splitting the category into its category and its level as adjunct, we can allow all, say, adjectives, to have the same category. This extends the efficiency gains in Fowlie (2013) to selection of adjuncts and adjuncts of adjuncts.

When adjunct  $[Y, n, m]$  adjoins to something of category  $[X, i, j]$ , the resulting phrase is of category  $[X, i, n]$ ,  $i, j, n, m \in \mathbb{N}$ . The second number is the phrase level, which tracks what tracks the level of the hierarchy the phrase is at; it is the only thing that can change.



Figure 4.12: Adjoin. The category feature of the new phrase is the first two elements of the adjoined-to phrase followed by the second element of the adjunct

Note that rather than having all category names include indices, we could reserve them just for relevant categories: those which can adjoin and/or be adjoined to. Doing so would work exactly the same way; the difference is that the operations would require more cases explaining how to handle categories with and without indices. Which way to go is a notational question; my choice was fairly arbitrary.

#### 4.6.1 Example

Before I give the full formal definition I will present an example. Suppose we have a grammar in which the adjunct sets are defined as follows:

$$\text{Ad}(N)=\{\text{Adj}, P, C\}, \text{Ad}(\text{Adj})=\{\text{Adv}, \text{Int}\}, \text{Ad}(\text{Adv})=\{\text{Int}\}, \text{Ad}(V)=\{\text{Adv}\}$$

We can derive *Apparently, John very often sang* as in figure 4.13. *very* adjoins to *often* since *often* is at phrase level 0 and *very* has adjunct level 3, and  $3 \geq 0$ . The whole phrase adjoins to *sang* since it has adjunct level 18 and *sang* has phrase level 0. T Merges to the VP, yielding a phrase at level 25. *Apparently* has adjunct level 26, so it can adjoin. (*Very* is arbitrarily given level 3, just for concreteness.)

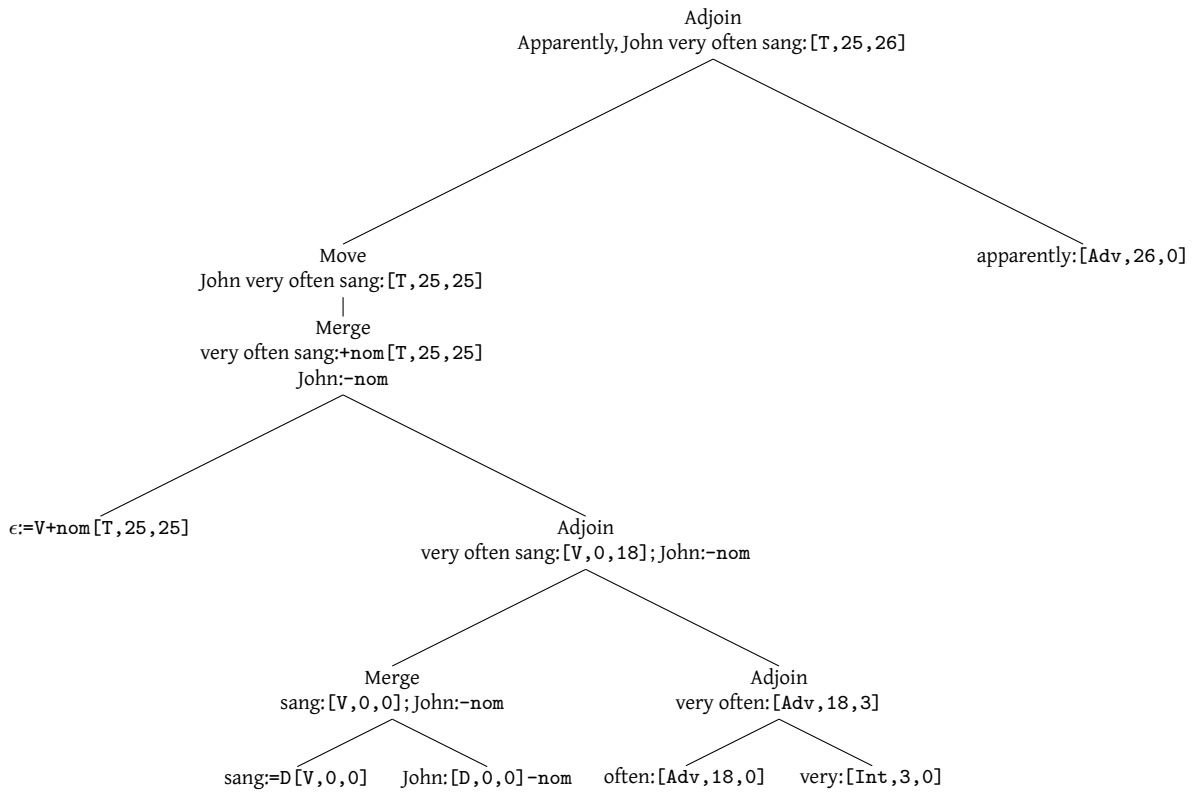


Figure 4.13: Adjunct of adjunct; functional head merge; adjunction after functional head merge

To get order, we require that the adjunct level (first number) of the adjunct be at least as high as the phrase level (second number) of the adjoined-to phrase. For example, in Figure 4.14, the derivation of *the big bad wolf* works because  $\text{Adj} \in \text{Ad}(N)$ , and  $6 > 4 > 0$ . The derivation of *\*the bad big wolf* fails because the category of *big wolf* is  $[N, 0, 6]$ .  $\text{bad}::[\text{Adj}, 4, 0]$  can't adjoin to it because *bad* is a level-4 adjunct, but *big wolf* is already at phrase level 6, and  $4 < 6$ .

#### 4.6.2 Definition: MGAs

**Merge** must be trivially redefined for categories as triples. **Merge** only cares about category, so it looks to match the positive selectional feature with the first element of the

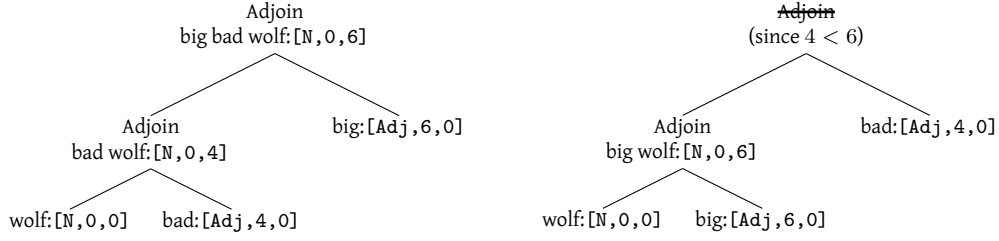


Figure 4.14: Adjunct ordering: valid and invalid derivations

triple. (**Move** is unchanged.)

**Definition 4.6.1** (Merge). For  $\alpha, \beta \in F^*$ ;  $s, t$  terms of the algebra,  $f$  the combining operation of the algebra,  $mvr_s, mvr_t$  expressions:

$$\text{Merge}(\langle s, =X\alpha \rangle :: mvr_s, \langle t, [X, i, j]\beta \rangle :: mvr_t) = \begin{cases} \langle f(s, t), \alpha \rangle :: mvr_s \cdot mvr_t & \text{if } \beta = \epsilon \\ \langle f(s, \epsilon), \alpha \rangle :: \langle t, \beta \rangle :: mvr_s \cdot mvr_t & \text{if } \beta \neq \epsilon \end{cases}$$

**Adjoin** applies when the category of the adjunct is an adjunct of the category it is adjoining to, and if the adjunct is a  $k$ -level adjunct then the level of the phrase it is adjoining to is no higher than  $k$ . **Move** works as expected: if the adjunct has negative licensing features left after it has had its category feature checked by **Adjoin**, it is added to the list of movers.

**Definition 4.6.2** (Adjoin). Let  $s, t \in \Sigma$  be terms of the algebra,  $f$  the combining operation of the algebra,  $Y, X \in \text{sel}$  be categories,  $i, j, n, m \in \mathbb{N}$ ,  $mvr$  be an expression (specifically, a mover list), and  $\alpha, \beta \in F^*$ .

$$\text{Adjoin}(\langle s, [X, i, j]\alpha \rangle :: mvr, \langle t, [Y, n, m]\beta \rangle) = \begin{cases} \langle f(t, s), [X, i, n]\alpha \rangle :: mvr & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \beta = \epsilon \\ \langle f(s, \epsilon), [X, i, n]\alpha \rangle :: \langle t, \beta \rangle :: mvr & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \beta \neq \epsilon \end{cases}$$

**Definition 4.6.3** (MGA). A *Minimalist Grammar with Adjunction* is a seven-tuple

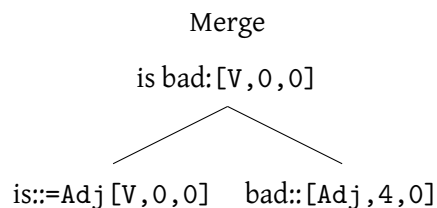


$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, \mathbf{Ad}, Lex, M, f \rangle$ .  $\Sigma$  is the *alphabet*.  $\mathbf{sel} \cup \mathbf{lic}$  are the *base features*. Let  $F = \{+f, -f, =X, [X, n, m] \mid f \in \mathbf{lic}; X, Y \in \mathbf{sel}; m, n \in \mathbb{N}\}$ .  $\mathbf{Ad} : \mathbf{sel} \rightarrow \mathcal{P}(\mathbf{sel})$  maps categories to their adjuncts.  $Lex \subseteq_{\text{fin}} \Sigma \times F^*$ ,  $M$  is the set of operations **Merge**, **Move**, and **Adjoin**, and  $f$  is a combining function for a closure algebra over  $\Sigma$ . The language  $L_G$  is the closure of  $Lex$  under  $M$ . A set  $C \subseteq \mathbf{sel}$  of designated features can be added;  $\{[c, i, j] \mid c \in C; i, j \in \mathbb{N}\}$  are the types of complete sentences.

### 4.6.3 Properties

Let us consider the desiderata laid out in section 4.4.

1. **Optionality:** ✓ the original category is kept as the first element of the category triple
2. **Transparency to selection:** ✓ the original category is kept
3. **Order: (Most)** The phrase level is the level of the last adjunct adjoined. The Adjoin rule requires that the adjunct be at least as high in the order as the phrase level. However, if low intervening functional heads Merge, correct adverb order can be violated. This problem is discussed in section 4.11.1.1 below, with a solution proposed in section 4.12.
4. **Selectability** ✓ Adjuncts have regular categories.



This is of particular importance for functional categories that are in fact required in a given sentence. It is important that we have the mechanism available to treat some functional categories in some cases as optional, and others, for example T in

English, as required. Required categories are modelled as per usual, with Merge. For example, in figure 4.15, the requirement of T is enforced by C's selecting it. If it were optional, C could select V directly, with or without T intervening. Since T selects V, rather than adjoining to it, the category of the phrase changes from V to T, allowing C to require T's presence.

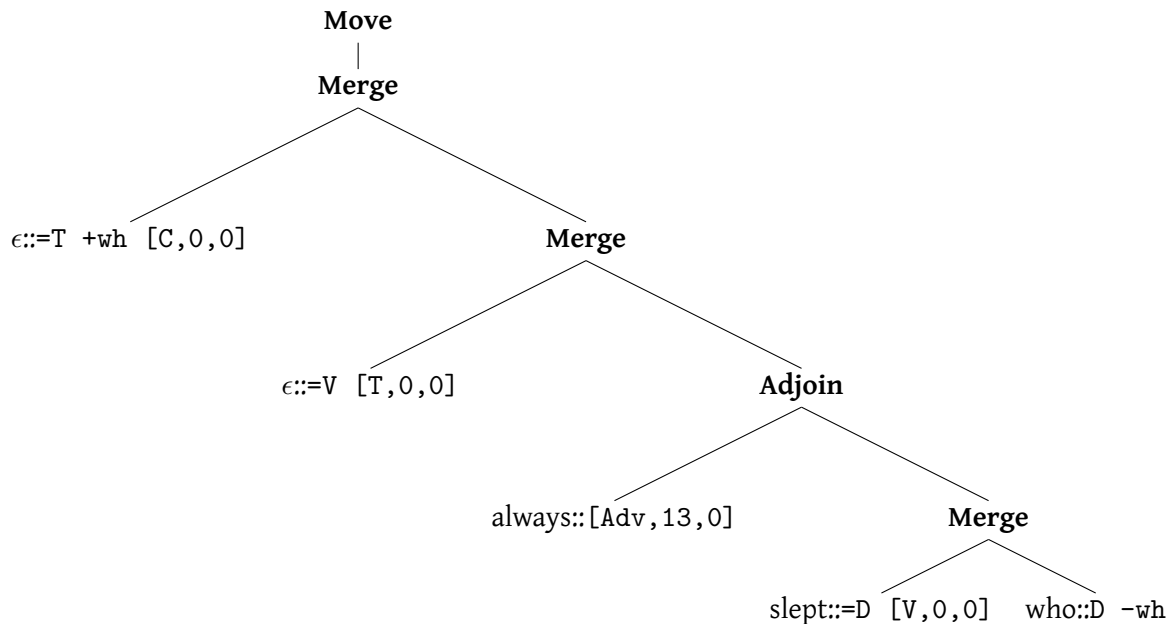


Figure 4.15: *Who always slept?*: required functional categories

- (a) **Efficiency:** ✓ Many adjuncts have the same category, so they have the same adjuncts. For example,  $\text{Ad}(\text{Adj}) = \{\text{Adv}, \text{Int}\}$
5. **Adjuncts of adjuncts** ✓ Adjunct categories are ordinary categories so they can have adjuncts too (Figure 4.13).
- (a) **Efficiency:** ✓ Many adjuncts have the same category, so they are selected by the same LI. For example, in the derivation of *is bad* above, *is* selects anything of category Adj.
6. **Unordered** ✓ See section 4.7.1.1 below.

7. **Obligatory adjuncts: Maybe.** See Section 4.7.2

## 4.7 Discussion and Extensions

This model captures both the strict ordering of the merge-only models and the optionality and transparency to selection of the categorial approaches. Cinque's observation that there is a hierarchy of functional heads and adverbs is modelled directly by defining a hierarchy in the grammar itself. The strict linear order falls out of the order imposed on the selectional features and the definition of **Adjoin**: adjunction is only defined when the hierarchy is respected. Optionality is the result of the transitivity of orders: intervening adjuncts are not necessary for a higher one to be adjoined. Transparency to selection is modelled by the pairing of the selectional features: the original category of the modified element is preserved, and **Merge** can see only that feature. The adjuncts are literally ignored.

The cross-linguistic consistency of the orders is accounted for by the claim that all human languages assign the same order to the same semantic classes of adjuncts. As such, it does not have to be learned, but rather comes with the grammar.<sup>15</sup>

Computationally, this approach has an advantage over the merge-only model with homophony as the latter increases the size of the lexicon by a polynomial function in the depths of the hierarchies of adjuncts, but the former does not.

---

<sup>15</sup>In minimalist grammars, most or all of language-specific grammars is encoded in the lexicon: what features does each LI have? I claim here that adjuncts are assigned their semantic class/hierarchy level based on their meaning; however, I give no mechanism here to do so, just as I give no account of why verbs generally select DPs, or how it is that an LI whose meaning is an  $n$ -place predicate selects  $n$  things of the right semantic type. The construction of the lexicon is a separate matter.

### 4.7.1 Pre- and post-head adjunction

The MG I am using as a basis for MGAs defines Merge always to put the selector on the left and the selectee on the right. As discussed in Chapter 3, a Kayne-like structure can be enforced by adding a third element in the (word, features) pair that says whether the item is lexical or derived. Then (nonfinal) Merge can be split into two cases: when the selector is lexical, ie a head, the selectee goes on the right, and when it is not, the selectee goes on the left. This arrangement yields spec-head-complement ordering.

Another option to vary Merge order is instead to have left-merge and right-merge selector features, written  $=_L X$  and  $=_R X$ , that say explicitly which side the selectee merges on. For example, German verbs might have a feature  $=_L D$  and English verbs  $=_R D$ . A parallel modification can be done to an MGA for Adjoin. For example, instead of a set of adjuncts for each category, the grammar could have a list of left adjuncts and a list of right adjuncts for a category. However, this is perhaps too powerful. In English, for example, it is exactly the post-nominal adjuncts that are also the unordered adjuncts.

#### 4.7.1.1 Unordered Adjuncts

As it stands, adjuncts such as PPs can be modelled as adjuncts, but they must all adjoin at the same level of the hierarchy, or else be cross-classified for each level of the hierarchy you want them to adjoin at. The former allows them to be freely ordered with respect to each other; the latter gives them freedom with respect to all adjuncts.

An expansion of this model<sup>16</sup> could add a non-number to the set of possible indices, call it  $\emptyset$ , and Adjoin could be defined to disregard the hierarchy and asymmetrically check the features for  $\emptyset$ -indexed adjuncts. Any distinct index also opens the door to adjoining on a different side of the head than other adjuncts; the definition I will give here models English PPs, which are post-head, unlike adjectives and many adverbs.

---

<sup>16</sup>I thank an anonymous reviewer for this suggestion.

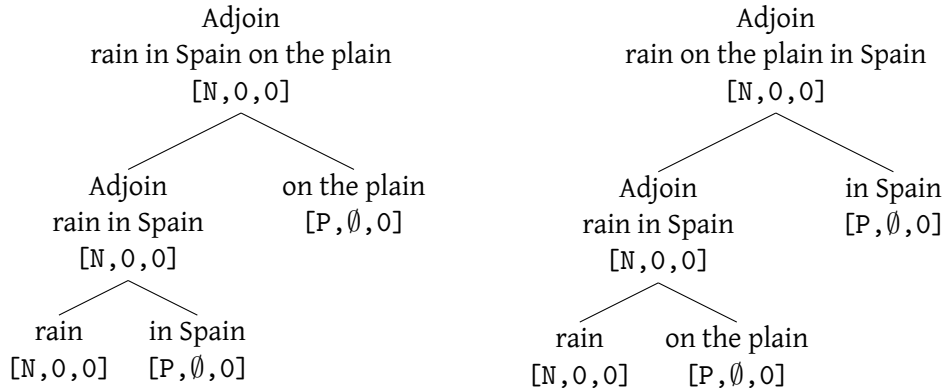


Figure 4.16: Unordered English PPs

In definition 4.7.1, the first and third cases are for adjuncts with number indices, and the second and fourth are for adjuncts with  $\emptyset$  indices.

**Definition 4.7.1** (Adjoin 2). Let  $s, t \in \Sigma$  be strings,  $Y, X \in \mathbf{sel}$  be categories,  $i, j, n, m \in \mathbb{N}$ ,  $mvs$  be a mover list, and  $\alpha, \beta \in F^*$ .

$$\begin{aligned}
 & \mathbf{Adjoin}(\langle s, [X, i, j]\alpha \rangle, \langle t, [Y, m, n]\beta \rangle :: mvs) \\
 = & \begin{cases} \langle f(t, s), [X, i, m]\alpha \rangle :: mvs & \text{if } m \geq j \ \& \ Y \in \mathbf{Ad}(X) \ \& \ \beta = \epsilon \\ \langle f(s, t), [X, i, j]\alpha \rangle :: mvs & \text{if } m = \emptyset \ \& \ Y \in \mathbf{Ad}(X) \ \& \ \beta = \epsilon \\ \langle s, [X, i, m]\alpha \rangle :: \langle t, \beta \rangle :: mvs & \text{if } m \geq j \ \& \ Y \in \mathbf{Ad}(X) \ \& \ \beta \neq \epsilon \\ \langle s, [X, i, j]\alpha \rangle :: \langle t, \beta \rangle :: mvs & \text{if } m = \emptyset \ \& \ Y \in \mathbf{Ad}(X) \ \& \ \beta \neq \epsilon \end{cases}
 \end{aligned}$$

#### 4.7.2 Obligatory Adjuncts

Recall that some elements which really seem to be adjuncts are not optional, for example *He worded the letter \*(carefully)* and *He makes a \*(good) father*. In MGAs there is a featural difference between nouns that have been modified and nouns that have not. For example, *father* is of category  $[N, 0, 0]$  and *good father* has category  $[N, 0, 4]$ . **Merge** is defined to ignore everything but the first element,  $N$ . However, the architecture is available to let

**Merge** look at the whole category triple, by way of a positive selectional feature of the form  $=[N, \_, 1]$ , which selects anything of category  $[N, i, j]$  with  $j \geq 1$ .

**Definition 4.7.2 (Merge 2).** For  $\alpha, \beta$  sequences of features,  $s, t$  elements of the algebra;  $X \in \mathbf{sel}; i, j, m \in \mathbb{N}; C = X$  or  $C = [X, \_, m]$  &  $j \geq m$ :

$$\mathbf{Merge}(\langle s, =C\alpha \rangle :: \mathbf{mvr}_s, \langle t, [X, i, j]\beta \rangle :: \mathbf{mvr}_t) = \begin{cases} \langle f(t, s), \alpha \rangle :: \mathbf{mvr}_s \cdot \mathbf{mvr}_t & \text{if } \beta = \epsilon \\ \langle s, \alpha \rangle :: \langle t, \beta \rangle :: \mathbf{mvr}_s \cdot \mathbf{mvr}_t & \text{if } \beta \neq \epsilon \end{cases}$$

For example, we can get *worded the letter carefully* with a little-v (or other functional head) requiring a modified verb phrase.

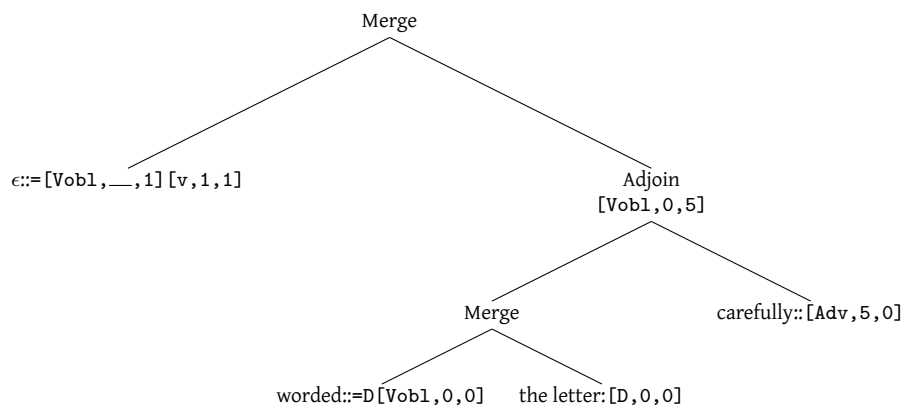


Figure 4.17: Obligatory adjunct extention

However, such an expansion of the definition of Merge is not of immediate help in all cases. In the case of *He makes a good father*, the NP *good father* is selected by D before the resulting DP is selected by *makes*, which is the verb that cares about whether the noun is modified. One solution is to cross-list *a* with a new determiner category only for modified NPs, and let *makes* select that category, as in Figure 4.18.

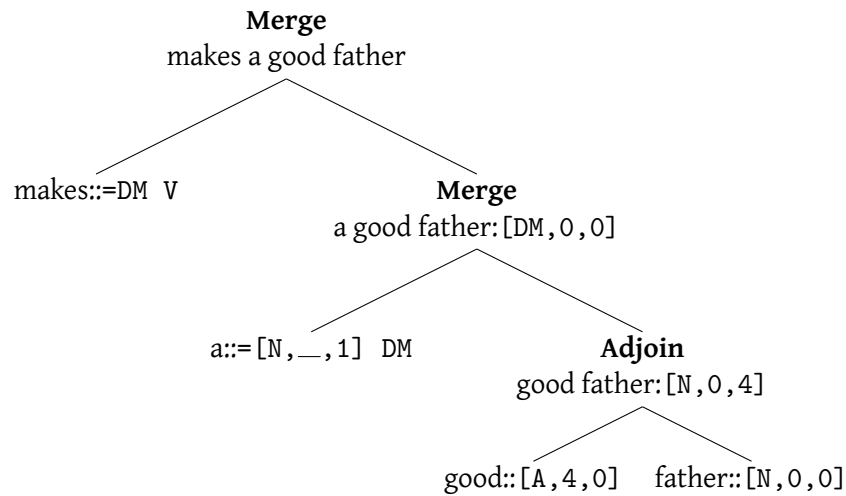


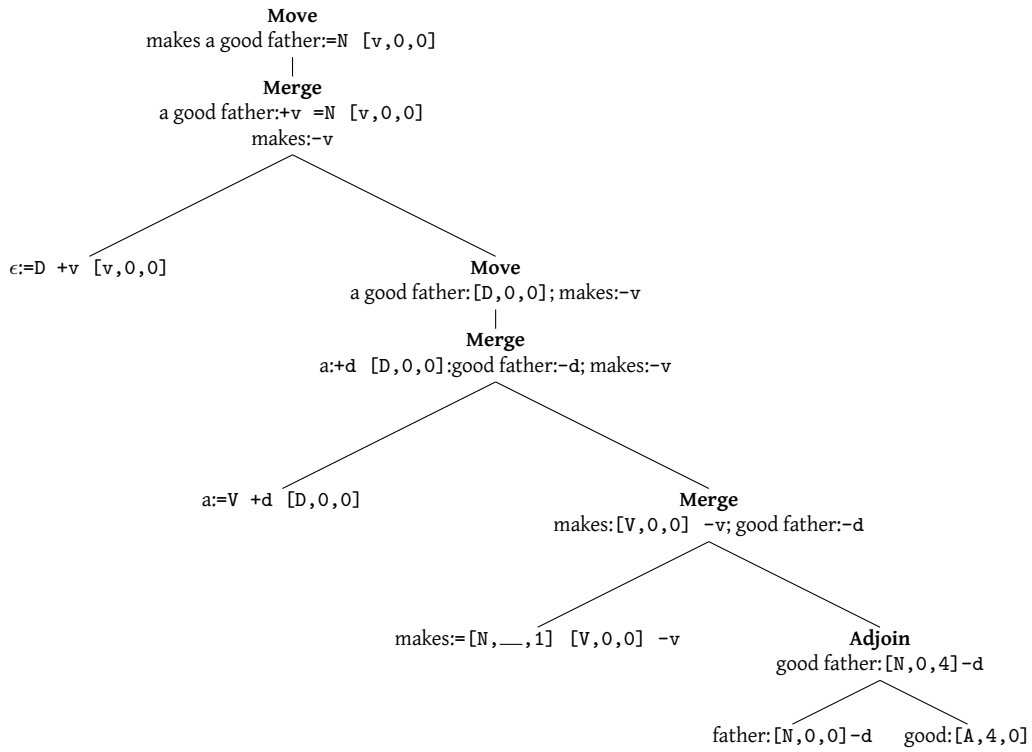
Figure 4.18: Determiners of modified NPs could have their own category DM

Obligatory adjuncts are not the only reason to suspect that the tighter relationship is between the verb and the noun, not the verb and the determiner; i.e. that V should perhaps select N, not D. When a head is incorporated into a verb, normally it is the head that the noun selects that is incorporated, as in (17).

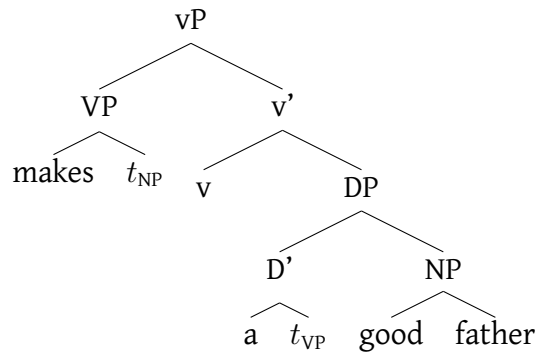
- (17) a. He [**stabbed** me [<sub>PP</sub> in [<sub>DP</sub> the [<sub>N</sub> **back**]]]]  
 b. back-stabbing  
 c. \*back-in, \*back-the, \*back-the-stabbing, \*back-in-the-stabbing

*Sportiche* (2005) proposes that verbs select NPs, and the NPs move to their Ds, which are functional heads on the spine.

For example we might have something like the partial derivation in Fig 4.19a. A derived tree is given in figure 4.19b. (Note that this derivation allows movement to the right (NP up to D); better would be head movement of the D *a* up to something above it. Since head movement explodes the number of cases in MGs I'm not using it here.)



(a) Directly selecting N; moving NP up to functional projection D



(b) Derived tree

Figure 4.19: Derivation and derived trees for *vP makes a good father*



## 4.8 Formal Properties

MGAs are clearly not strongly equivalent to traditional MGs, if we take *strong equivalence* to mean that the set of derivation trees are isomorphic. This is of course impossible since MGAs have an extra function, Adjoin. MGAs are, on the other hand, weakly equivalent to MGs, meaning that for every MGA, an MG can be defined that generates the same strings, and vice versa.

For this proof, we need to define the notion of a *suffix* of a lexical item. A suffix of a lexical item  $\langle s, \alpha \rangle$  is the features  $\alpha$  or any suffix of  $\alpha$ . We need to refer to suffixes because as a minimalist grammar acts on an expression, it removes features from the beginning of a feature stack, and that feature stack came originally from the lexicon.

**Definition 4.8.1** (The suffixes of the lexicon,  $\text{suffix}(Lex)$ ). For a minimalist grammar lexicon  $Lex$ ,

$$\text{suffix}(Lex) = \{\alpha \in F^* \mid \exists s \in \Sigma^*, \beta \in F^* \text{ s.t. } \langle s, \beta\alpha \rangle \in Lex\}$$

To show that traditional MGs and MGAs generate the same string sets (languages), first we show that any language generated by an MG can also be generated by an MGA (Lemma 4.8.2) and then that any language generated by an MGA can also be generated by a traditional MG. The second proof runs intermediately through a Multiple Context Free Grammar, which are already known to be weakly equivalent to MGs (Harkema, 2001).

**Lemma 4.8.2.**  $L(MG) \subseteq L(MGA)$

*Proof.* MGAs also include Merge and Move, and place no additional restrictions on their action. Any MGA language could have Adjoin stripped away and what remained would be an MG. □

For the next lemma, we use Multiple Context Free Grammar Seki et al. (1991). MCFGs

are like context free grammars, except tuples of strings, not just single strings, can be generated by the application of a rule.

**Definition 4.8.3** (MCFG). An  $2,m$ -MCFG is a 4-tuple  $G = \langle N, T, P, S \rangle$  such that:

- $N$  is a finite ranked alphabet of non-terminals of maximal rank  $m$
- $T$  is a finite alphabet of terminals (rank 0)
- $P$  is a set of rules of the form

$$A(s_1, \dots, s_k) : -B(x_1, \dots, x_i) C(y_1, \dots, y_j)$$

where:

- $A, B, C \in N$  with ranks  $k, i, j$  respectively
- the strings  $s_i$  consist only of words from the  $T$  and  $x_1, \dots, x_i, y_1, \dots, y_j$ .
- Each variable  $x_1, \dots, x_i, y_1, \dots, y_j$  appears at most once in  $s_1, \dots, s_k$
- $S$  is a non-terminal of rank 1 (the start category)

The language generated by an MCFG is the set of all terminal strings derivable by the rules  $P$  from the start category(s)  $S$ .

**Lemma 4.8.4.**  $L(MGA) \subseteq L(MG)$

*Proof.* MGs are weakly (and indeed strongly) equivalent to Multiple Context Free Grammars (MCFGs) so it suffices to show that  $L(MGA) \subseteq L(MCFG)$ .

We translate an MGA into an MCFG in the normal way, following [Harkema \(2001\)](#): the nonterminals of the MCFG are sequences of feature sequences from the MGA. This translation is based on the basic grammar given in [Definition 4.6.3](#), but it is easy to see how it could be expanded to include the extensions suggested in later sections.

Given MGA  $G = \langle \Sigma, F = \mathbf{sel} \cup \mathbf{lic}, Lex, M, S, Ad \rangle$ , define an MCFG  $MCFG(G) = \langle \Sigma, N, P, S \rangle$  defining the language

$$N = \{ \langle \delta_0, \delta_1, \dots, \delta_j \rangle \mid 0 \leq j \leq |\mathbf{lic}|, \text{ all } \delta_i \in \text{suffix}(Lex) \}$$

$$\text{Let } h = \text{Max}(\{i \mid \exists X \in \mathbf{sel} : i = |\mathbf{Ad}(X)|\})$$

The rules  $P$  are defined as follows,  $\forall \alpha, \beta, \delta_0, \dots, \delta_i, \gamma_0, \dots, \gamma_j \in \text{suffix}(Lex)$ .

$s_0, \dots, s_i, t_0, \dots, t_j$  are variables over strings.

**Lexical rules:**  $\alpha(s) \qquad \forall \langle s, \alpha \rangle \in Lex$

**Merge-and-stay rules:** Here is the first case of the Merge rule for MGAs.

$$\text{Merge}(\langle s, =X\alpha \rangle :: \text{mvr}_s, \langle t, [X, m, n] \rangle :: \text{mvr}_t) = \langle st, \alpha \rangle :: \text{mvr}_s \cdot \text{mvr}_t$$

It becomes a set of MCFG rules as follows. In the rule set below,  $s = s_0, t = t_0$ , the tree parts of  $\text{mvr}_s$  and  $\text{mvr}_t$  are  $s_1, \dots, s_i$  and  $t_1, \dots, t_j$  respectively, and their features become  $\delta_1, \dots, \delta_i$  and  $\gamma_1, \dots, \gamma_j$ . One rule is made for each index less than the maximum possible index  $h$  for the grammar. (Any rule indices that fall outside the set of indices for that particular category simply go unused in practice.)

Here is the description of the MCFG rules corresponding to this Merge rule:

$$\langle \alpha, \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0 t_0, s_1, \dots, s_i, t_1, \dots, t_j)$$

$$:- \langle = X\alpha, \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [X, m, n], \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$$

$$\forall X \in \mathbf{sel}, \forall n, m \leq h$$

The rest of the MCFG rules are formed similarly.

**Merge-and-move rules:**  $\forall X \in \mathbf{sel}, \forall n, m \leq h$

$$\langle \alpha, \beta, \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0, t_0, s_1, \dots, s_i, t_1, \dots, t_j)$$

$$:- \langle \text{X}\alpha, \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [\text{X}, \text{m}, \text{n}]\beta, \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$$

**Adjoin-and-stay rules:**  $\forall X, Y \in \mathbf{sel} \text{ s.t. } Y \in \mathbf{Ad}(X), \forall k, l, n, m \leq h \text{ s.t. } n \geq k$

$$\langle [\text{X}, \text{m}, \text{n}], \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0 t_0, s_1, \dots, s_i, t_1, \dots, t_j)$$

$$:- \langle [\text{X}, \text{m}, \text{k}], \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [\text{Y}, \text{n}, \text{l}], \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$$

**Adjoin-and-move rules:**  $\forall X, Y \in \mathbf{sel} \text{ s.t. } Y \in \mathbf{Ad}(X), \forall k, l, n, m \leq h \text{ s.t. } n \geq k$

$$\langle [\text{X}, \text{m}, \text{n}], \beta, \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0, t_0, s_1, \dots, s_i, t_1, \dots, t_j)$$

$$:- \langle [\text{X}, \text{m}, \text{k}], \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [\text{Y}, \text{n}, \text{l}]\beta, \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$$

**Move-and-stop rules:**  $\forall f \in \mathbf{lic}$

$$\langle \alpha, \delta_1, \dots, \delta_{i-1}, \delta_{i+1}, \dots, \delta_j \rangle (s_i s_0, s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_j)$$

$$:- \langle +f\alpha, \delta_1, \dots, \delta_{i-1}, -f, \delta_{i+1}, \dots, \delta_j \rangle (s_0, \dots, s_j)$$

**Move-and-keep-moving rules:**  $\forall f \in \mathbf{lic}$

$$\langle \alpha, \delta_1, \dots, \delta_{i-1}, \beta, \delta_{i+1}, \dots, \delta_j \rangle (s_0, \dots, s_j)$$

$$:- \langle +f\alpha, \delta_1, \dots, \delta_{i-1}, -f\beta, \delta_{i+1}, \dots, \delta_j \rangle (s_0, \dots, s_j)$$

These rule sets are finite since MGAs never add anything to feature sequences, but only either remove features or change just the indices of  $[X, i, j]$  features. As such, the suffixes  $\alpha, \beta, \delta, \gamma$  are limited in number. Since any given grammar has a maximal hierarchy depth  $h$ , the indices  $k, l, m, n$  in the rules are defined to be limited by  $h$ . The  $\delta$ 's and  $\gamma$ 's are limited by the SMC. They are the movers, and there can never be more than one mover in the mover list for a given licensing feature, and  $\mathbf{lic}$  is finite.

The MCFG defined as above derives the same strings as the MGA. The proof is by induction on the depth of the derivation tree. A derivation tree of depth  $k$  has a Merge, Move, or Adjoin node dominating two sisters of depth less than  $k$ , deriving expressions with MCFG equivalents. The construction of the grammar provides corresponding Merge, Move, or Adjoin rule equivalencies to build the equivalent expression.

A sentence  $s$  is in an MGA language if there is a derivation deriving an expression  $\langle\langle s, [C, i, j] \rangle\rangle$  for some final category  $C$  and any  $i, j \in \mathbb{N}$ . A sentence  $s$  is in an MCFG language if it can derive expression  $\langle C \rangle(s)$  for some final category  $C$ . The final categories of this MCFG are  $[C, i, j]$  for all MGA final categories  $C$  and all  $i, j \leq h$ .

□

**Theorem 4.8.5** (Weak equivalence of MGAs and MGs). *For any MGA*

$G = \langle \Sigma, \mathbf{sel}, \mathbf{lic}, \mathbf{Ad}, Lex, \{\mathbf{Merge}, \mathbf{Move}, \mathbf{Adjoin}\} \rangle$ , *there is a weakly equivalent MG*  $G' = \langle \Sigma, \mathbf{sel}_{MG}, \mathbf{lic}, Lex_{MG}, \{\mathbf{Merge}, \mathbf{Move}\} \rangle$ .

*Proof.* By lemmas 4.8.2 and 4.8.4

□

## 4.9 Interim Summary

We have seen that to account for both the “looseness” of adjuncts – their optionality and transparency – and their “strictness” – their ordering properties is difficult. Previous models account for one or the other, but not both. MGAs are able to account for both, by the simple expedience of splitting their categories into triples. The first element is the basic category name; its adjunct relation with other categories accounts for the optionality and transparency of adjunction. The second and third elements add a hierarchy level, and Adjoin is defined to forbid adjunction to a phrase that already has a higher adjunct adjoined to it.

The following table summarises the models we’ve seen so far.

	Trad 4.5.1	F&G 4.5.2	Syn. glue 4.5.3.1	Homoph 4.5.3.2	Fowlie13 4.5.4	MGA 4.6
Optional	✓	✓	✓	✓	✓	✓
Eff (opt)	✓	✓	✗	✗✗	✓	✓
Transparent	?	✓	✗	✗	✓	✓
Order	✗	✗	✓	✓	(✓)	(✓)
Select	✓	✓	✓	✓	✓	✓
Eff (Sel)	✗	✓	✗	✗	✗	✓
Adj of adj	✓	?	✓	✓	✓	✓
Eff (Adj)	✗	✓?	✗	✗	✗	✓
Unordered	✓	✓	✓	✓	✓	✓
Oblig	✗	✗	✗	✗	?	✓

Table 4.4: Summary of models re: desiderata

For a minimalist analysis of adjunction with feature-driven Merge, in which adjunct ordering is part of the syntax, a mechanism along the lines of MGA hierarchies of semantic classes is required. The adjuncts are optional and they belong to what appears syntactically to be only a few categories (perhaps just Adjectives, Adverbs, Prepositions, CPs, and Intensifiers), yet they are ordered. As such, they behave simultaneously as though they belong to the same categories and to distinct categories. The apparent difference in categories accounts for ordering; everything else about them is accounted for by their simply being adjuncts, not arguments.

I therefore claim that a function like Adjoin that operates on two complete phrases (in the features calculus, this means they are displaying their category features) according to what may adjoin to what, and which generates a phrase of the adjoined-to category, most naturally accounts for the non-order-related behaviour of adjuncts. If order is to be accounted for in the syntax, something like these indices is needed, separating the categories (so that the adjuncts may continue to behave as a small number of unified classes) from the hierarchy levels. (Two indices are needed to distinguish the hierarchy level of potential adjuncts-of-adjuncts from the hierarchy level of the adjunct itself.)

## 4.10 What should move?

This grammar models adjuncts as part of the phrase projected by the adjoined-to; for example, adjectives are part of the NP. This means that while adjectives can move out of the NP, the noun can't move independently, except possibly in the case of head-movement. I argue that we do want the adjective to be able to move to derive the special cases of non-canonical adjunct ordering. Non-canonical ordering is characterised by a change in intonation and semantics, so the syntax ought to change as well. For example, in (18), something like focus moves *ugly* forward.

(18) Wear the ugly enormous green hat

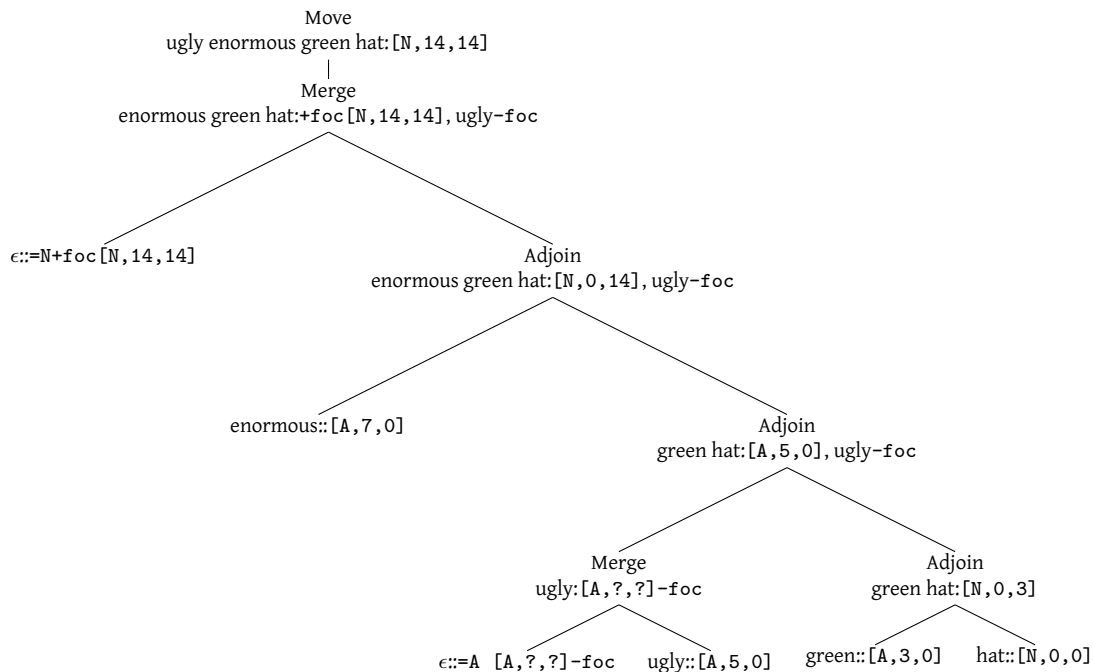


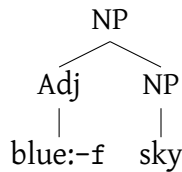
Figure 4.20: Non-canonical adjective order

Adjoin could be defined differently, with both adjunct and adjoinee moveable.

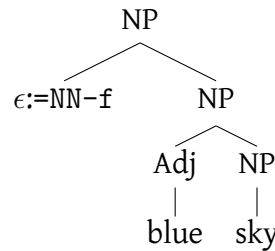
**Definition 4.10.1** ( $\text{Adjoin}_3$ ). Let  $s, t \in \Sigma$  be strings,  $Y, X \in \text{sel}$  be categories,  $i, j, n, m \in \mathbb{N}$ ,  $mvs \in (\Sigma^* \times F^*)^*$  be a mover list, and  $\alpha, \beta \in F^*$ .

$$\text{Adjoin}_3(\langle s, [X, i, j] \alpha :: mvs \rangle, \langle t, [Y, n, m] \beta \rangle) = \begin{cases} \langle ts, [X, i, n] \alpha \rangle :: mvs & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \alpha = \beta = \epsilon \\ \langle s, [X, i, n] \alpha \rangle :: \langle t, \beta \rangle :: mvs & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \alpha = \epsilon \ \& \ \beta \neq \epsilon \\ \langle t, [X, i, n] \rangle :: \langle s, \alpha \rangle :: mvs & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \beta = \epsilon \ \& \ \alpha \neq \epsilon \\ \langle \epsilon, [X, i, n] \rangle :: \langle s, \alpha \rangle :: \langle t, \beta \rangle :: mvs & \text{if } n \geq j \ \& \ Y \in \text{Ad}(X) \ \& \ \beta \neq \epsilon \ \& \ \alpha \neq \epsilon \end{cases}$$

Such a grammar would preclude move features from being introduced by the adjoinee and yet moving the whole phrase; to move the whole phrase we would need to Merge a silent element that introduces a move feature.



(a) Using  $\text{Adjoin}_3$  defined in Def 4.10.1, *blue* introduces the move feature and moves independently of *sky*: the moving phrase here is *blue*.

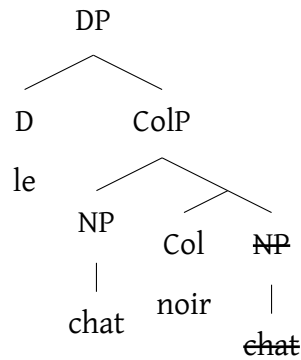


(b) To make *blue sky* move, a silent head Merges.

Figure 4.21: Adjoinee can move independently using  $\text{Adjoin}_3$

Whether this grammar or something like the original in Definition 4.6.2 is correct is an empirical question. Graf (2014) argues against and Koopman (2015) argues in favour of such a grammar, accounting for right adjuncts in French with phrasal movement of the noun phrase, excluding the adjective's projection. Notice that in her grammar, adjuncts are Merged, as in the model in Section 4.5.3.2, but the notion is the same.





#### 4.11 Graf 2014

Graf (2014) analyses the proposals given here from a model-theoretic perspective, and describes how the proposals alter some basic mathematical properties of MGs, as well as evaluating them for their ability to account for empirical facts. He finds that of the models analysed, only MGAs can account for all linguistic phenomena under consideration, but they do so at the expense of several nice mathematical properties, including locality of Merge. He argues, though, that this is not a result of the model itself so much as the phenomena to be modelled.

In a traditional MG, Merge is strictly  $k$ -local. What this means is that if you had an MG that never employed Move, the derivation trees would form a strictly  $k$ -local ( $SL_k$ ) tree language. An  $SL_k$  tree language is one that can be defined by a set of legal  $k$ -factors, which are building block subtrees of depth  $k$ .

A lexical item  $l$  controls a *slice*, a subpart of a derivation tree. The slice controlled by  $l$  is  $l$  itself plus any Merge and Move nodes dominating it that occur because of positive features in  $l$ 's feature stack. For example, in the derivation tree in Figure 4.22, the slice controlled by *man* is just *man* (blue), that for *the* is *the* and the Merge node above it (red), and *slept* controls itself and the root node (green).

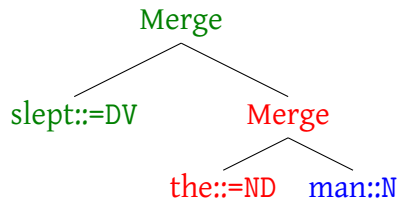


Figure 4.22: Slices by colour

The depth of an LI’s slice depends of the number of features preceding the category feature. Since Lex is finite, the largest possible slice is fixed for each grammar. One way of thinking about it is this: suppose you have a Merge node in a derivation tree, and you want to know if it is valid case of Merge. The farthest you have to look to find out if a given Merge node is licit is the maximal number of positive features on any item in Lex, plus one. For example, in Fig. 4.23, the farthest we need to look to determine the validity of the root node is 3 nodes, but we can imagine that if the node labelled =A=BC had more features before C, we would need to look that much farther. =A=BC has two features before the category feature C, but we also need to include the Merge node that makes it sister to =CD.

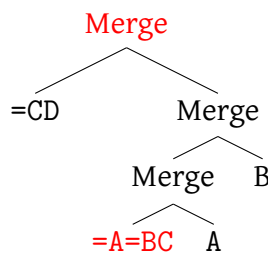


Figure 4.23: To determine if the root node is a valid application of Merge, we need to look 3 nodes away, to =A=BC.

Move interrupts this locality, because a mover’s first Merge site (which is where it occurs in the derivation tree) can be indefinitely far away from its final landing site.

(19) Who did River think that Simon said that Jayne would shoot \_\_\_?

We can't know if a given subtree is legal if the mover hasn't landed yet, and there is no way of knowing, locally, when that will be. For example, we can't know whether the (sub)tree in in Fig.4.24, is valid until *who* Moves. Perhaps a lexical item with feature +wh will never be Merged.

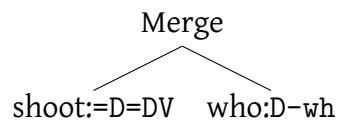


Figure 4.24: The validity of this derivation tree cannot be determined.

The question Graf raises is what happens when Adjoin is added to the grammar. The answer is that even without Move, the derivation tree language loses its  $SL_k$  status. Slices are no longer bounded, since Adjoin is defined by a partial order. For instance, in (20), an indefinite number of *big*s can intervene between *the* and *ship*, which translates into an indefinite number of Adjoin nodes intervening between the root node and *ship*, which it needs to see in order to determine whether *the* can safely Merge with *big ship*.

(20) the big big ... big ship

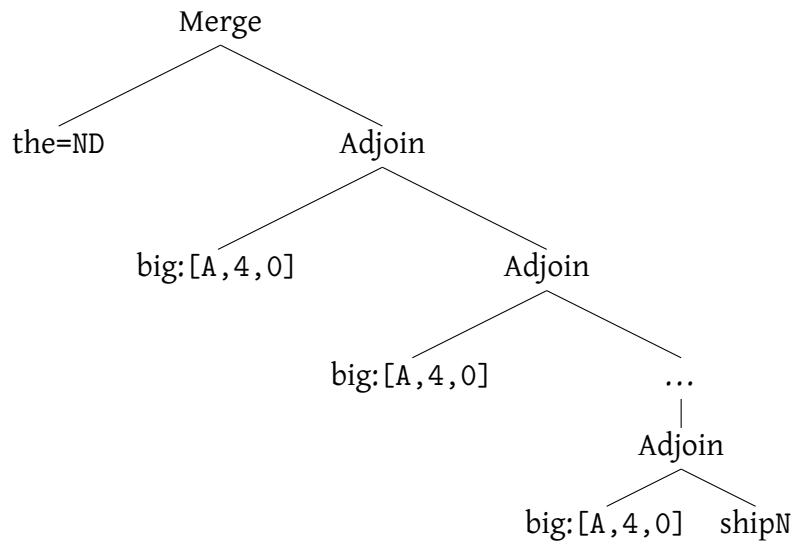


Figure 4.25: Adjoin interrupts the locality of Merge

Thus MGAs are not  $SL_k$  for any  $k$ .

Other properties of MGs lost in MGAs are local threshold testability (a kind of locality) and recognisability by a type of deterministic top-down tree automaton.

#### 4.11.1 Adverbs and Functional Heads

Contra Cinque (1999), I model adverbs as separate from functional heads. Adverbs and adjectives differ from functional heads in that functional heads are sometimes required and sometimes optional. For example, English requires T, but not, perhaps,  $Mod_{epistemic}$  in every sentence. To model this, I give adjectives and adverbs category triples with their second number set to 0. This allows adjuncts to adjoin to them, starting at the bottom of that hierarchy. Functional heads, on the other hand, will start with their phrase level already set to their adjunct level. This means that when they Merge, the resulting phrase is at the right level in the hierarchy, preventing low adjuncts from adjoining after the

merger of a high functional head.<sup>17</sup>

For example, in Figure 4.13, *very* adjoins to *often*, which is possible since the phrase level of *often* is 0. Later, functional head T Merges to the VP. Its phrase level is 25. This is important because we want to say that *apparently* can only adjoin here because its adjunct level is 26, which is higher than 25. A low adverb such as *again*::[Adv, 3, 0] cannot adjoin to T.

Cinque's adverbs and functional heads, on the other hand, are connected very directly: functional heads optionally select adverbs in their specifiers. In MGAs, since all adverbs can have the same category, adverbs as specifiers are not possible if we want to preserve the notion that adverbs all have the category *adverb*. However, this is not a bad thing: Cinque observes that a language, or even a sentence in a language, tends to present only one of the functional head and the adverb overtly. In other words, he observes that generally either the functional head or the adverb is (at least phonetically) null. Arguably, since the semantics of the heads and the adverbs match, the phonetically null one could also be semantically null. If something is both phonetically and semantically null, it would be more parsimonious if we could say it was not there at all. While MGAs do not preclude both being present, they also do not require that both be present.

Under the MGA approach, a functional head and adverb that Cinque pairs are at the same place in the hierarchy, but neither would select the other. For example, we might have (given as triples of *string::meaning::features* since the functional heads in English are phonetically null): *briefly*::BRIEFLY::[Adv, 10, 0] and  $\epsilon$ ::ASP-DURATIVE::[F, 10, 0], and *usually*::USUALLY::[Adv, 21, 0] and  $\epsilon$ ::ASP-HABITUAL::[F, 21, 0]. If we tried to model the first pair instead as a head-specifier, we would have  $\epsilon$ ::ASP-DURATIVE::=Adv [F, 10, 0]. Note, however, that  $\epsilon$ ::ASP-DURATIVE::=Adv [F, 10, 0] can also erroneously select *usually* in its specifier since it too is of category Adv. We would need a separate category for each

---

<sup>17</sup>This partly solves the problem; see section 4.11.1.1 for the remaining problem, which is solved in the final model presented in section 4.12.

adverb, since they are being selected via Merge by functional heads.

Thus efficient MGAs do not model adverbials as specifiers of functional heads. Functional heads should be Merged, as they are not adjuncts, but adjuncts should not be Merged.

#### 4.11.1.1 Problem

A shortcoming of the present model is that while Merge of a high functional head will prevent later adjunction of a low adverb, nothing prevents a low functional head that selects, say, V, from merging after the adjunction of a high adverb. For example, in Figure 4.26, the merger of the low aspectual head resets the phrase level to 2, allowing the adjunction of *soon*, even though it is lower in the hierarchy than *perhaps*.

This problem will be addressed in the next section (4.12.)

### 4.12 Proposal 2: Minimalist Grammars with Hierarchies

Adger (2008) proposes a related model of functional heads. He proposes a splitting of Merge into two functions: Sel-Merge (selectional Merge), which is the traditional Merge, and HoP-Merge (Hierarchy of Precedence Merge), which is not entirely unlike Adjoin.

An order is defined on the functional heads, and HoP-Merge is only defined if the Merging head is higher in the hierarchy than the most recently merged functional head.

The difference between his HoP-Merge and my Adjoin is that HoP-Merge is otherwise just like regular Merge: The features are symmetrically checked, which begins a new phrase. With Adjoin, the old category is kept; only the hierarchy level changes.

I have transformed his grammar into an MG, with HoP-Merge added. HoP-Merge is valid when the categories of its arguments are in the same hierarchy (defined in the grammar) and the left daughter is higher than the right.

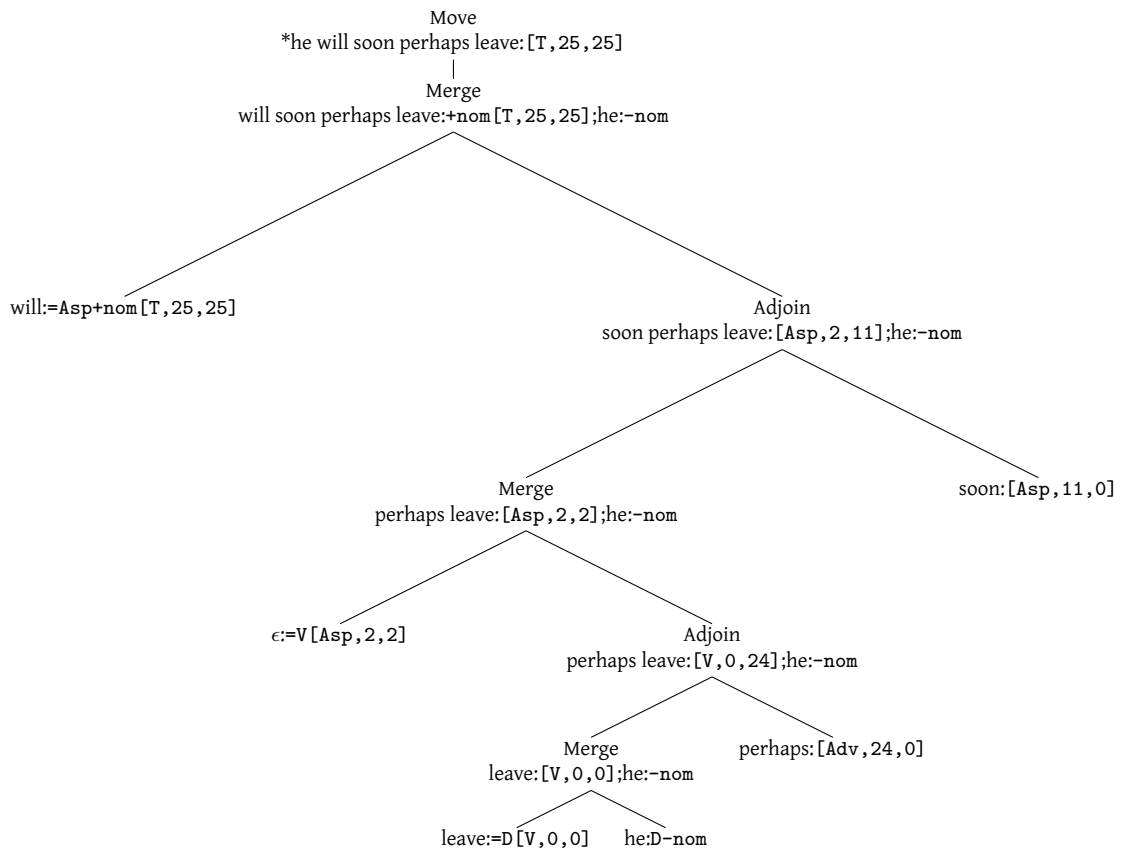


Figure 4.26: Derivation of ungrammatical sentence due to Merge resetting the hierarchy level

Suppose we have the following grammar:

$$Lex = \{the:=D, 0, three:Num, 4, kings:N, 0, sang:=D, V, 0\}$$

$$H_1 = D, 5 > Num, 4 > Poss, 3 > n, 2 > N, 1$$

$$H_2 = C, 3 > T, 2 > V, 1$$

We can derive *the three kings sang* as in Figure 4.27.

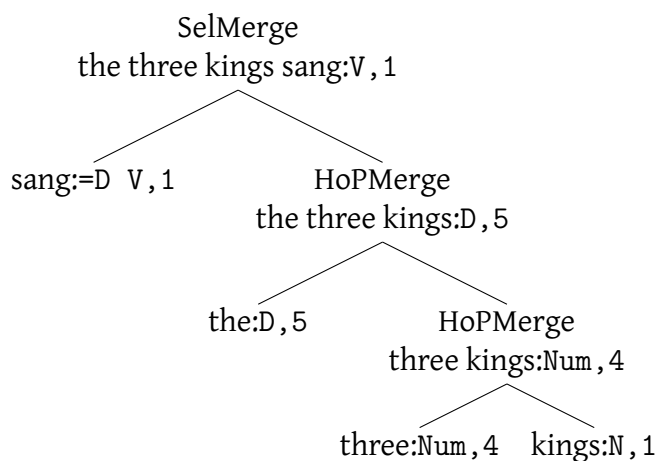


Figure 4.27: Derivation tree of *the three kings sang* using HopMerge

The interaction of the two Merges can force the presence of a functional head by having SelMerge select for it directly. Without SelMerge, the verb must be defined instead to select any nominal projection, by making it part of  $H_1$ :

$$H'_1 = V, 6 > D, 5 > Num, 4 > Poss, 3 > n, 2 > N, 1$$

This, however, is not only the wrong result (verb are not part of the functional projections of nouns) but makes it impossible for a verb to select a CP under normal assumptions. The reason is that we cannot get a partial order on the functional heads this way, because  $V$  selects  $C$  (for verbs like *think*) but in hierarchy 2, we have  $H_2 = C, 3 > T, 2 > V, 1$ . We require therefore that  $V$  be both above and below  $C$ , and clearly  $V \neq C$ , so we have a contradiction.

Therefore, if we are to have HoPMerge, we must also have SelMerge.



#### 4.12.1 A combined approach: Minimalist Grammars with Hierarchies (MGH)

To solve the problem laid out in section 4.11.1.1, I propose an alternate approach, call it Minimalist Grammars with Hierarchies (MGH), that incorporates both Adjoin and HoP-Merge.

The traditional MG is enriched with two partial functions,  $Ad$  and  $Fn$ , that map categories to their adjuncts and functional projections, respectively.<sup>18</sup>

Categories are themselves enriched with two numbers, the *adjunct level* which indicates the hierarchy level of the item, and the *phrase level* which indicates the hierarchy level of the whole phrase. This is just as in MGAs.

Merge, Move, and Adjoin are defined just as in MGAs. HoPMerge is defined as follows. Note that we have to dig down into the category feature, rather than always taking features off the top of the stack.

**Definition 4.12.1** (HoPMerge). Let  $s, t \in CL(\Sigma, f)$ ,  $Y, X \in \mathbf{sel}$  be categories,  $i, j, n, m \in \mathbb{N}$ ,  $mvr_s, mvr_t$  be mover lists, and  $\alpha, \beta, \gamma \in F^*$ . Suppose  $\exists Z \in \mathbf{sel}$  such that  $X, Y \in Fn(Z)$  and suppose  $i \geq j$ . Then:

$$\begin{aligned} & \mathbf{HopMerge}(\langle s, \gamma[X, i, m]\alpha \rangle :: mvr_s, \langle t, [Y, n, j]\beta \rangle :: mvr_t) \\ = & \begin{cases} \langle f(s, t), \gamma[X, i, i]\alpha \rangle :: mvr_s \cdot mvr_t & \text{if } \beta = \epsilon \\ \langle s, \gamma[X, i, i]\alpha \rangle :: \langle t, \beta \rangle :: mvr_s \cdot mvr_t & \text{if } \beta \neq \epsilon \end{cases} \end{aligned}$$

Figure 4.28 shows a derivation of *the three old kings*, with  $A \in \mathbf{Ad}(\mathbb{N})$  and  $D, \mathbf{Num}, \mathbb{N} \in Fn(\mathbb{N})$ .

---

<sup>18</sup>We might want to require that the values of  $Fn$  be disjoint. This would avoid a projection from “changing its mind” partway through about what kind of projection it is. Whether this is the right result is an empirical question.

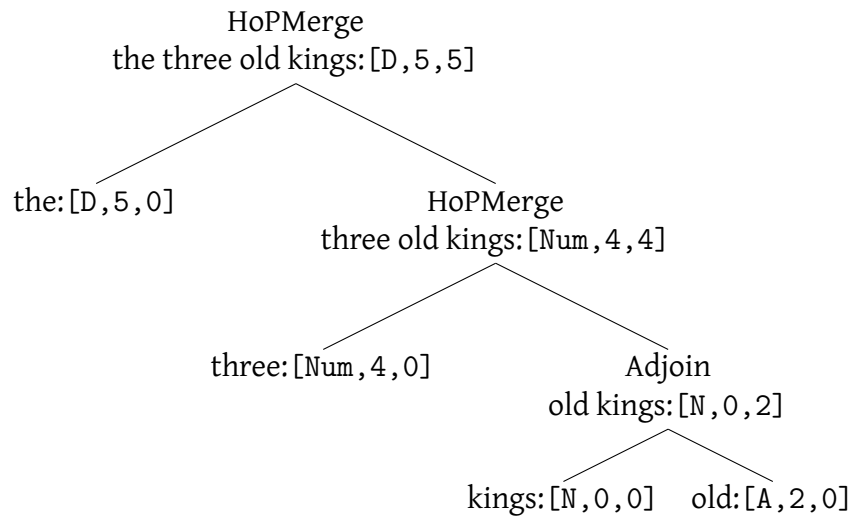


Figure 4.28: The three old kings derived by an MGH

The left argument of HoPMerge must be at a higher level than any previously Adjoined or HoPMerged item. The new category feature takes on the category of the functional head. Both numbers are set to the hierarchy level of the functional head. This ensures that even if the functional head has been adjoined to, changing its phrase level, the result is something that can only be Adjoined to or HoPMerged to by something at a higher level than the functional head.<sup>19</sup>

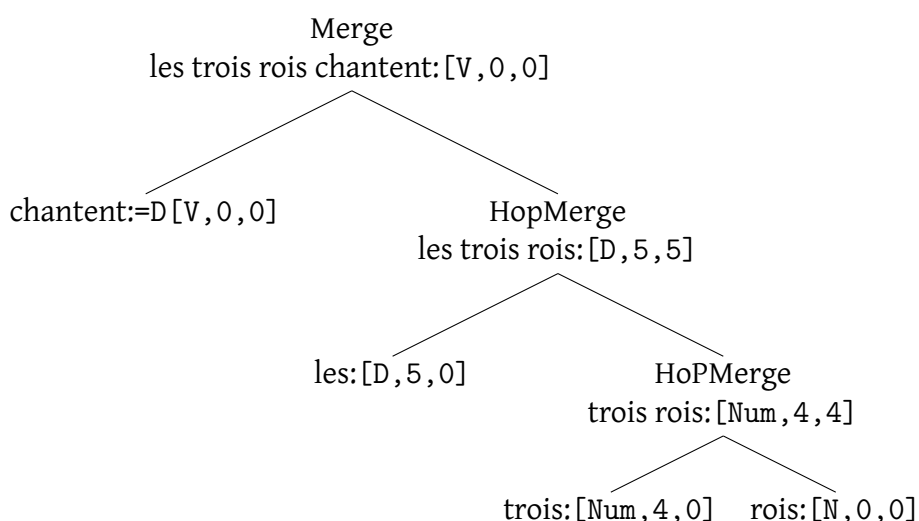
#### 4.12.2 Discussion

This approach solves the problem of low functional heads Merging after high adjuncts. We use the same numbering system to track both adjuncts and functional heads, but functional heads (HoP)merge, while adjuncts Adjoin. Both have the possibility of being

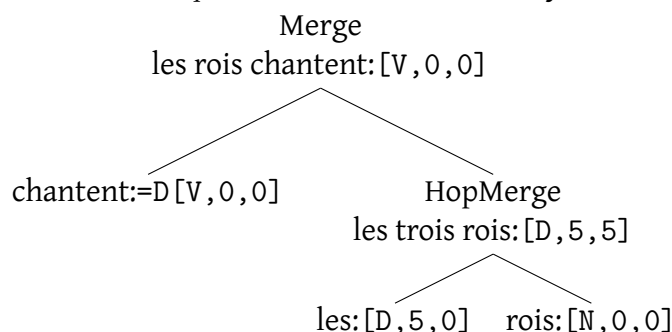
---

<sup>19</sup>Here too we may want to constrain the grammar, requiring that the functional head be new to the derivation – truly a functional head, not a complex structure. To do this we need a way to track whether an element is new to the derivation. For a tree-generating grammar this is straightforward: it will be a trivial tree. For a string-generating grammar we take inspiration from [Stabler and Keenan \(2003\)](#) and add a boolean element that tells us whether the item is new. In [Stabler and Keenan \(2003\)](#) this is used to put the head linearly between the specifier and the complement.

optional. Adjuncts are optional as they are in MGAs, and functional heads are optional when they have another functional head HopMerged right above them. That higher functional head may not be optional – it may itself be properly selected – but the lower one is. Whether this is the correct result is an empirical question. It seems to work well for French DPs. For example, in *Les trois rois chantent* ‘the three kings sing’, *trois* ‘three’ is optional, while *les* ‘the’ is not. The obligatoriness of the determiner is enforced by (Sel)Merge, since *chantent* ‘sing’ requires a D (Fig.4.29).



(a) The D is required since it is selected for by the verb



(b) The Num is optional since it's only HoPMerged to by D. HoPMerge only requires that the right argument be lower in the hierarchy than the left.

Figure 4.29: Optionality of internal functional head in French DPs

One might ask why the functional heads' order is not just given directly by a partial order in the grammar instead of explicitly by numbers. After all, functional heads do

project their own phrases, and they have their own category names. The reason is that Cinque’s hierarchy (Fig. 4.30) has a plethora of heads of the same sort: about a dozen Aspect heads, all ordered with respect to one another and interwoven with other categories, and Mod and T heads scattered throughout as well. Given the choice between giving them separate categories and ordering the categories, and giving them the same category and listing the order in the lexical items, I chose the second. The second solution fits much better in the existing MGA model – since the numbers are right there in the category triples – and allows similar heads to have the same category.

We can say that  $F(V) = \{\text{Asp}, \text{Voice}, \text{T}, \text{Mod}, \text{Mood}\}$  rather than listing all 30, and include such lexical items as  $\text{will} := V + \text{nom}[\text{T}, 25, 0]$  and  $\text{-ed} := V + \text{nom}[\text{T}, 26, 0]$ .

Still, if a partial order on **sel** is preferred to a separate index set, a different MGH with HoPMerge 2 can be used (Def 4.12.3).

**Definition 4.12.2.** Let  $\geq \subseteq \mathbf{sel} \times \mathbf{sel}$  be a partial order on **sel**.

**Definition 4.12.3** (HoPMerge 2). Let  $s, t \in CL(\Sigma, f)$ ,  $Y, X \in \mathbf{sel}$  be categories,  $I, J \in \mathbf{sel}$ ,  $mvr$ s be a mover list, and  $\alpha, \beta, \gamma \in F^*$ . Suppose  $\exists Z \in \mathbf{sel}$  such that  $X, Y \in F(Z)$  and suppose  $X \geq J$ . Then:

$$\begin{aligned} & \mathbf{HopMerge}_2(\langle s, \gamma[X, X, X]\alpha :: mvr \rangle, \langle t, [Y, I, J]\beta \rangle) \\ = & \begin{cases} \langle f(s, t), \gamma[X, X, X]\alpha \rangle :: mvr & \text{if } \beta = \epsilon \\ \langle s, \gamma[X, X, X]\alpha \rangle :: \langle t, \beta \rangle :: mvr & \text{if } \beta \neq \epsilon \end{cases} \end{aligned}$$

In Adjoin, we still need both indices, but we index with the corresponding functional head rather than an arbitrary indexing set like  $\mathbb{N}$ . Adjuncts are listed in the lexicon with their corresponding functional head in their triple.

**Definition 4.12.4** (Adjoin (for HoPMerge 2 model)). Let  $s, t \in CL(\Sigma, f)$ ,  $Y, X \in \mathbf{sel}$  be categories,  $I, J, N, M \in \mathbf{sel}$ ,  $mvr$ s be a mover list, and  $\alpha, \beta \in F^*$ .

Mood<sub>speech-act</sub> *frankly*  
 Mood<sub>evaluative</sub> *fortunately*  
 Mood<sub>evidential</sub> *allegedly*  
 Mod<sub>epistemic</sub> *probably*  
 T<sub>past</sub> *once*  
 T<sub>future</sub> *then*  
 Mod<sub>irrealis</sub> *perhaps*  
 Mod<sub>necessity</sub> *necessarily*  
 Mod<sub>possibility</sub> *possibly*  
 Asp<sub>habitual</sub> *usually*  
 Asp<sub>repetitive</sub> *again*  
 Asp<sub>frequentative(I)</sub> *often*  
 Mod<sub>volitional</sub> *intentionally*  
 Asp<sub>celerative(I)</sub> *quickly*  
 T<sub>anterior</sub> *already*  
 Asp<sub>terminative</sub> *no longer*  
 Asp<sub>continuative</sub> *still*  
 Asp<sub>perfect</sub> *always*  
 Asp<sub>retrospective</sub> *just*  
 Asp<sub>proximative</sub> *soon*  
 Asp<sub>durative</sub> *briefly*  
 Asp<sub>generic/progressive</sub> *characteristically*  
 Asp<sub>prospective</sub> *almost*  
 Asp<sub>sg.completive(I)</sub> *completely*  
 Asp<sub>pl.completive</sub> *tutto*  
 Voice *well*  
 Asp<sub>celerative(II)</sub> *fast/early*  
 Asp<sub>repetitive(II)</sub> *again*  
 Asp<sub>frequentative(II)</sub> *often*  
 Asp<sub>sg.completive(II)</sub> *completely*  
 Verb

Figure 4.30: Complete Cinque hierarchy

$$\begin{aligned}
& \mathbf{Adjoin}(\langle s, [\mathbf{X}, \mathbf{I}, \mathbf{J}] \alpha :: mvs \rangle, \langle t, [\mathbf{Y}, \mathbf{N}, \mathbf{M}] \beta \rangle) \\
= & \begin{cases} \langle f(t, s), [\mathbf{X}, \mathbf{I}, \mathbf{N}] \alpha \rangle :: mvs & \text{if } \mathbf{N} \geq \mathbf{J} \ \& \ \mathbf{Y} \in \mathbf{Ad}(\mathbf{X}) \ \& \ \beta = \epsilon \\ \langle s, [\mathbf{X}, \mathbf{I}, \mathbf{N}] \alpha \rangle :: \langle t, \beta \rangle :: mvs & \text{if } \mathbf{N} \geq \mathbf{J} \ \& \ \mathbf{Y} \in \mathbf{Ad}(\mathbf{X}) \ \& \ \beta \neq \epsilon \end{cases}
\end{aligned}$$

However, this model will only work if there are indeed functional heads corresponding to every adjunct level. At least for adjectives, it is not clear that this is really the case. As a technical fix, dummy category names can be added to **sel** to mimic the functional hierarchies that lack functional heads.

A potential inefficiency to be addressed in MGHs as well as MGAs is in how adjuncts interact with functional heads. Since functional heads can Merge, Adv needs to be in  $\mathbf{Ad}(\mathbf{V})$ ,  $\mathbf{Ad}(\mathbf{Mod})$ ,  $\mathbf{Ad}(\mathbf{Asp})$ ,  $\mathbf{Ad}(\mathbf{T})$ ,  $\mathbf{Ad}(\mathbf{Voi})$ , and  $\mathbf{Ad}(\mathbf{Mood})$ . It is not a coincidence that adjuncts of the verbal spine may also be adjuncts of the verb itself. Do we perhaps want to say that  $\mathbf{Ad}(X) = \mathbf{Ad}(Y)$  if  $Y \in F(X)$ ? In this case we need only specify some adjuncts in the grammar, indeed perhaps only for lexical categories (as opposed to functional). Whether adjuncts of a lexical category are also adjuncts of all its functional projections is an empirical question.

MGHs look like a better model of human language than MGAs, as they have the same properties in terms of our desiderata as MGAs, and they solve the problem of low functional heads Merging above high adjuncts. However, they do so at the expense of an additional operation, bringing the total to 4, double that of traditional MGs.

### 4.12.3 Formal Properties

MGHs are weakly equivalent to MGs. To the MCFG laid out in Lemma 4.8.4 we add two more sets of rules to match the two HoPMerge cases.

$$\begin{aligned}
& \forall \mathbf{X}, \mathbf{Y} \in \mathbf{sel} \ s.t. \ \exists \mathbf{Z} \ s.t. \ \mathbf{X}, \mathbf{Y} \in F_n(\mathbf{Z}); \\
& \forall \mathbf{k}, \mathbf{l}, \mathbf{n}, \mathbf{m} \leq h \ s.t. \ \mathbf{k} \geq \mathbf{l}; \\
& \forall i, j \leq |\mathbf{lic}|, \forall \alpha[\mathbf{X}, \mathbf{k}, \mathbf{m}] \alpha', \beta, \delta_1, \dots, \delta_i, \gamma_1, \gamma_j \in \mathbf{suffix}(Lex),
\end{aligned}$$

Form a rule:

**HoP-and-stay rules:**  $\langle \alpha[X, k, k]\alpha', \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0 t_0, s_1, \dots, s_i, t_1, \dots, t_j)$   
:-  $\langle [X, k, m], \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [Y, n, l], \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$

And a rule:

**HoP-and-move rules:**  $\langle [X, k, k], \beta, \delta_1, \dots, \delta_i, \gamma_1, \dots, \gamma_j \rangle (s_0, t_0, s_1, \dots, s_i, t_1, \dots, t_j)$   
:-  $\langle [X, k, m], \delta_1, \dots, \delta_i \rangle (s_0, \dots, s_i) \langle [Y, n, l], \beta, \gamma_1, \dots, \gamma_j \rangle (t_0, \dots, t_j)$

Because the lexicon is finite and HoPMerge never adds anything to the feature sequences, the number of suffixes ( $\alpha's$ ,  $\beta's$ ,  $\delta's$ ,  $\gamma's$ ) is finite. The finite depth  $h$  of the hierarchies constrains the indices  $k, l, m, n$  and the SMC constrains the numbers of  $\delta's$  and  $\gamma's$  to  $|\text{lic}|$ . Thus this grammar is indeed an MCFG, and an extension of the inductive proof on the depth of derivations sketched in 4.8.4 shows that this is the right MCFG to correspond to the minimalist grammar.

### 4.13 Conclusion

We have seen several models of adjuncts and functional heads. Their properties in terms of the desiderata in figure 4.3 are summarised in Table 4.5

	Trad 4.5.1	F&G 4.5.2	Syn. glue 4.5.3.1	Homoph 4.5.3.2	Fowlie13 4.5.4	MGA 4.6	MGH 4.12
Optional	✓	✓	✓	✓	✓	✓	✓
Eff (opt)	✓	✓	✗	✗✗	✓	✓	✓
Transparent	?	✓	✗	✗	✓	✓	✓
Order	✗	✗	✓	✓	✓	(✓)	✓
Select	✓	✓	✓	✓	✓	✓	✓
Eff (Sel)	✗	✓	✗	✗	✗	✓	✓
Adj of adj	✓	?	✓	✓	✓	✓	✓
Eff (Adj)	✗	✓?	✗	✗	✗	✓	✓
Unordered	✓	✓	✓	✓	✓	✓	✓
Oblig	✗	✗	✗	✗	?	✓	✓

Table 4.5: Summary of models re: desiderata

My proposals, MGAs (Minimalist Grammars with Adjunction) and MGHs (Minimalist Grammars with Hierarchies) account for the empirical facts laid out in [Cinque \(1999\)](#) as well as eight more desiderata, unlike the other proposal considered. They do so at the expense of some of the simplicity of MGs. MGAs and MGHs add 1 and 2 operations respectively, require categories to be tuples, and add to the grammar some additional functions (Ad and F). The derivation trees of MGAs and MGHs are not locally threshold testable, recognisable by a (certain type of) deterministic top-down tree automaton, or strictly  $k$ -local, unlike those of traditional MGs. However, it is the properties of adjunction facts, not merely of the models, that force these losses of simplicity ([Graf, 2014](#)).



## CHAPTER 5

### Learnability of adjuncts

#### 5.1 Introduction

At the heart of linguistic study is the question of how such a complex system can be learned by people so young and unformed that they cannot even survive on their own. In describing existent human languages we often hope that the phenomena we encounter will provide some insight into this puzzle: perhaps we have found a universal feature, meaning it could be somehow “built in”, sidestepping the necessity for children to learn it; or perhaps we have found a clearcut parameter on which human languages may differ, pointing out a specific fact that children might automatically watch for. Language acquisition studies children’s language learning directly, while formal language theory aims to discover what sorts of grammars are required to describe human language. Learnability theory is the study of mathematical models of language acquisition.

This chapter will explore learning models applied to two specific phenomena: optionality and repetition. Adjuncts are traditionally defined as optional in that although a sentence will have a different meaning without the adjunct, it is still perfectly grammatical, and the meanings of the sentences differ systematically. For example, in (1) we see that the adjective *red* is optional, and (1-a) entails (1-b).

- (1) a. My love is like a **red** rose.
- b. My love is like a rose.

In reality, some adjuncts are argued to be obligatory; see section 2.5 of chapter 2 for discussion. However, here I will consider learnability of optional adjuncts and leave the possibility of obligatory ones aside for now.

Many human languages allow optional repetition of adjuncts, as for example English, Farsi, Korean, Spanish, Russian, and Tagalog:

- (2) a. My love is like a **(red red)** rose.  
 b. She's **(really really really really really)** nice.  
 c. Don't you **(ever ever ... ever)** do that!
- (3) a. Man Kheili Kheili Kheili Kheili shokolat doost daram  
 I really really really really chocolate have friend  
 'I really really really really like chocolate' Persian<sup>1</sup>  
 b. Gole rose ghermeze ghermeze ghermeze ghermez  
 The rose red red red red  
 'the red red red red rose'
- (4) naneun chokhollis-eul maeu maeu maeu maeu joh-ahanda  
 I chocolate-by very very very very like  
 'I really really really really like chocolate' Korean<sup>2</sup>
- (5) Realmente es muy, muy, muy difícil  
 really is very very very difficult  
 It really is very very very difficult Spanish<sup>3</sup>
- (6) a. on bezhal ochen' ochen' ochen' ochen' bystro  
 he ran very very very very fast  
 'He ran very very very very fast' Russian<sup>4</sup>

---

<sup>1</sup>Data from Setareh Safavi, p.c.

<sup>2</sup>Data from Yun Jung Kim, p.c.

<sup>3</sup>Data from José Maria Lahoz Bengoechea, p.c.

<sup>4</sup>Data from Alexandra Grabarchuk and Natasha Korotkova, p.c.

- b. krasnyy krasnyy krasnyy krasnyy yabloko  
 red red red red apple  
 ‘red red red red apple’
- (7) natulog ang (malaking malaking ... malaking) babae  
 sleep D (big big ... big) woman  
 ‘The big big ... big woman is sleeping/slept’ Tagalog<sup>5</sup>

In this chapter I will look at several formal learning models and ask the following 2 questions:

1. Are optionality and repetition possible in the class of languages learnable by this model?<sup>6</sup>
2. What kinds of sentences would the learner need to encounter in order to conclude that a given element can be repeated indefinitely? Omitted?

The idea is to begin to answer the question *What kinds of language learners are babies?* by asking *If a baby were such-and-such a sort of learner, how would we predict them to behave with regard to these phenomena?* Formal definitions of repetition and optionality are given in definitions 5.3.4 and 5.3.3.

Repetition creates patterns without finite bound, and is relatively easy to study formally, making it a particularly useful property to examine to learn about how people might acquire language.

Optionality and repetition have a conceptual relationship which, as we will see, is reflected in some learners. First, repetition is a form of optionality: since an element is repeatable in a context if all sentences with one or more instances of the element in that context are grammatical, all instances of that element after the first one are optional.

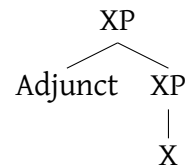
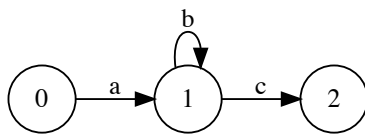
---

<sup>5</sup>Data from Seth Ronquillo, p.c.

<sup>6</sup>That is, is there some learnable language  $L$  with some sentence  $s$  containing an occurrence of a non-empty string  $a$  such that, in that position,  $a$  can occur any number of times, including 0?

Second, a common way to model both optionality and repetition in a grammar is to put the derivation into the same state whether the optional/repeated element has occurred or not. *State* intuitively means something like “situation”. In a phrase structure grammar, in which the formation of trees depends entirely on the labelling of root nodes of sister constituents, *state* comes down to that label. For example, in Fig. 5.1b, the X-bar rules  $XP \rightarrow X$  and  $XP \rightarrow \text{Adjunct } XP$  together mean that if you have an XP, you can add an adjunct to the left of it and the result is another XP: you’re back in the same state as you were in before you added the adjunct. This allows indefinite repetition of the adjunct, but also allows the adjunct to be absent.

Similarly, in a finite state automaton, states are the circles connected by lines. In Figure 5.1a, once we get to state 1 we remain in state 1 no matter how many times *b* occurs, until *c* occurs. This gives us an optional and repeatable *b*.



(a) Whether we hear *b* or not, we’re in state 1

(b) Whether we have an adjunct or not, we have an XP.

Figure 5.1: Repetition is optionality

In some of the learners I consider, if the learner is given examples that indicate optionality, it will generalise to repetition, and vice versa.

Such a result is not correct for human languages in general, although it is correct in many situations. For example, in English, adverbs are optional but not usually repeatable (8), but many adjectives are both optional and repeatable (2-a).

- (8) a. River laughed
- b. River laughed maniacally
- c. \*River laughed maniacally maniacally

This chapter considers five learners of five language classes, ordered to be progressively more suitable for human language. While the learners we explore are not suitable for human language, they can be considered stepping stones to future work on multiple context-free languages. For example, the substitutable context-free learner Section 5.6 is a very simple generalisation of the subregular 0-reversible learner in Section 5.5. Similarly, new learners of multiple context-free grammars such as Clark and Yoshinaka (2014) generalise the learner in Section 5.8 to parallel multiple context free languages.

Section 5.2 introduces the general concept of learnability and formalises optionality and repetition. Section 5.4 looks at  $n$ -gram languages, which are non-hierarchical and depend entirely on words that are at most  $n$  words away for some  $n$ . Section 5.5 is about 0-reversible regular languages, which treat sentence beginnings that can end the same way as equivalent. Section 5.6 treats a related class of context-free languages that treat constituents that can occur in the same context as equivalent. In both 0-reversible substitutable context-free languages, optionality and repetition entail each other. Section 5.7 takes a step sideways to look at a probabilistic learner for a class of regular languages in which sentence beginnings that can end “enough” of the same ways are treated as equivalent. These learners can learn both optionality and repetition, but there is no special entailment relationship. Finally, Section 5.8 examines a learner and language family that approaches human language. Although it is still context-free, it does not assume that all intersubstitutability entails complete intersubstitutability. This learner need only hear an optional repeatable element 0, 1, and 2 times in context, and need only parse and analyse the sentence in which it occurs once, in order to conclude that it is both optional and repeatable. However, if the situation is slightly more complex, and

there is a “related” context in which the optional repeated element cannot occur, more evidence is needed. For example, in English *really* is optional and repeatable in many, but not all, grammatical contexts of the form (*x, like y*).

- (9) a. I really\* like you
- b. This chapter is exactly like the last one
- c. \*This chapter is exactly really like the last one

The learner will have to parse and analyse more than just *I really like you* to learn that *really* is optional and repeatable since it cannot occur in the “related” context (*This chapter is exactly, like the last one*).

## 5.2 Learnability

Mathematically, a learner is a function from a sample of the language to a grammar. A new hypothesis grammar can be generated after each new sample sentence.

input sentence 1	→	hypothesis grammar 1
+ input sentence 2	→	hypothesis grammar 2
+ input sentence 3	→	hypothesis grammar 3
+ input sentence 4	→	hypothesis grammar 4
+ input sentence 5	→	hypothesis grammar 5
...		

Figure 5.2: Learning

For example, a very simple learner could simply remember every sentence it’s heard. The grammar is then simply the list of sentences. No novel sentences would be generated by this grammar.

input	Hypothesis grammar
$s_1$	$\rightarrow \{s_1\}$
$s_2$	$\rightarrow \{s_1, s_2\}$
$s_3$	$\rightarrow \{s_1, s_2, s_3\}$
$s_4$	$\rightarrow \{s_1, s_2, s_3, s_4\}$
...	

Figure 5.3: Finite language learner

Such a simple learner cannot learn infinite languages and cannot generalise, so although it is a valid learning model, in that it is a function from a sample to a grammar, it is definitely not a model of how babies learn language.

The kinds of patterns in the input that the learner is sensitive to depend on the assumptions that the particular learner makes about the nature of the language.

In current learnability theory, there are two broad categories of learners: those which are guaranteed to be successful eventually, and those that are not. The latter class consists of such algorithms as the inside-outside algorithm. Such learners can get the right grammar, but they can also get stuck in local minima, which essentially gives the learner the impression that it's right, since every time it tries to change its mind things get worse, but if only it would look even farther away, it could find the answer.

The learners that are guaranteed to find the correct grammar belong to three categories: Gold learners, PAC learners, and MAT learners. The first two differ according to what it means for a learner to have succeeded in learning a language, and the latter adds a source of information to the learner.

Gold learning, or learning in the limit from positive data, is achieved when the learner eventually converges on exactly the right language. Such learning is very hard; for example there is no one learner that can learn all finite languages plus even one infinite language (Gold, 1967). However, there are some classes of languages that are known to be Gold-learnable. None of these are human-like languages as of yet.

The formal definitions of learning rely on the notion of a *text*, which is a “good” sample: one that (eventually) includes all sentences of the language.

**Definition 5.2.1** (Text<sup>7</sup>). For a language  $L$ , a *text*  $T$  is a sequence of sentences of  $L$  such that every sentence of  $L$  occurs at least once in  $T$ .  $T[i]$  is the first  $i$  sentences in text  $T$ .<sup>8</sup>

**Definition 5.2.2** (Gold learning). Given a language class  $\mathcal{L}$  and a learner  $F$ , we say  $F$  is a learner for  $\mathcal{L}$  if, for every  $L \in \mathcal{L}$  and for every text  $T$  of  $L$ , there is an  $i \in \mathbb{N}$  such that for all  $j > i$ ,  $F(T[j]) = F(T[i])$  and  $L(F(T[j])) = L$ . That is, there is a point after which the learner always guesses the same grammar, and that grammar generates  $L$ . We say that the learner *converges* on that grammar or language.

Probably Approximately Correct (PAC) learning is a weaker requirement, and is achieved when the probability that the language is “close enough” to being correct is “high enough”. *Close enough* and *high enough* are determined by thresholds set in advance. For any such thresholds, there is a (finite) text size that will work, no matter how the strings are distributed within that text.

**Definition 5.2.3** (PAC learning).  $\forall 0 < \epsilon, \delta < 0.5$ , it is possible to specify a finite text size such that, for any distribution over the strings in  $T$ , the learner outputs a hypothesis grammar that is, with probability  $1 - \delta$ ,  $\epsilon$ -close to correct. (Valiant, 1984)

Learners that add a Minimally Adequate Teacher (MAT) are useful to study, if only as a stating point, because their learning problem is slightly easier. The teacher can answer

---

<sup>7</sup>Notation:  $\epsilon$  is the empty string (the sequence of no words).  $x^*$  means  $x$  repeated 0 or more times;  $x^+$  means  $x$  repeated 1 or more times;  $x^k$  means  $x$  repeated  $k$  times;  $x^{k+}$  means  $x$  repeated  $k$  or more times. These superscripts extend to sets, eg  $\Sigma^k$  is the set of all strings of length  $k$  where the words are drawn from  $\Sigma$ .  $\in$  means *is a member of*. For a function  $f$ ,  $f(x)$  is the result of applying  $f$  to  $x$ .  $\forall$  means *for all* and  $\exists$  means *there exists* or *for some*.  $\subseteq$  means *is a subset of*, so  $A \subseteq B$  means all members of  $A$  are also members of  $B$ .

<sup>8</sup>A text can also be defined to include non-sentences. They are labelled as such, so that the learner knows to ignore them or, in the case of probabilistic grammars, to assign them zero (or very low, for smoothing) probability.



certain questions about the language, such as membership inquiries and for some models even whether the learner's current hypothesis is correct. MAT learners, while being better learners than Gold and PAC learners, are also less natural in terms of a model of what a child experiences and pays attention to. In this chapter I will consider one learner with a MAT in section 5.8. This teacher can respond only to the membership queries, which the authors suggest could stand in for a model in which the child hears a lot more language than she actually analyses. If the child takes a certain subset of her language input as sentences to actually use as input to her learning algorithm, then in lieu of actual membership queries she might be able to search her memory of sentences that she's heard but not paid much attention to.

Learners of Regular languages are much better understood than those for languages higher on the Chomsky hierarchy (Chomsky, 1959). Since human languages are known to be Mildly Context-Sensitive (Joshi, 1985), learning algorithms for such languages are clearly more relevant to actual human language learning; however, as research into such learners is still in its infancy, and since our understanding of Regular learners has driven higher-level learners (see for example Clark, Eyraud, & Habrard (2008)'s substitutable CF learner and Yoshinaka (2008)'s  $k,l$ -substitutable CF learner, which can be regarded as, in a sense, generalisations of Angluin (1982)'s 0- and  $k$ -reversible learners to the context free level), I will look at learners low on the Chomsky hierarchy as well.

### 5.3 Repetition and Optionality

Here I formalise the notions of optionality and repetition in language  $L$ . Note that it only makes sense to define optionality and repetition for strings in a context.

**Definition 5.3.1** (Language). Given a finite set  $\Sigma$ ,  $\Sigma^*$  is the set of all finite sequences of elements of  $\Sigma$ .  $L$  is a language over  $\Sigma$  iff  $L \subseteq \Sigma^*$

That is, a language is just a (possibly infinite) set of sentences using words/morphemes<sup>9</sup> drawn from a finite lexicon.

**Definition 5.3.2** (Context). A *context* is a pair  $(u, v)$  where  $u, v \in \Sigma^*$

In other words, a context is a sentence with a gap in it where you could put some sequence of words. For example,  $(the, left\ in\ a\ hurry)$  is a context. Contexts are used primarily to describe where sequences of words may fall. For example, if we were talking about English, we might note that the context above is one in which any of  $\{man, clever\ woman, congregation\}$  can appear, as in *the congregation left in a hurry*.

**Definition 5.3.3** (Optional).  $x \in \Sigma^*$  is *optional in context*  $(u, v)$  iff  $uv \in L$  and  $uxv \in L$

We need to include the context in which something might be optional for it to have any sense. If we said that, say, Adjectives are optional in English, full stop, we would be wrong: *tall* is not optional in context  $(She\ seems, \epsilon)$  since *\*She seems* is not a sentence of English.

**Definition 5.3.4** (Repeatable).  $x \in \Sigma^*$  is *repeatable in context*  $(u, v)$  iff  $ux^+v \subseteq L$

This means that the language includes all sentences that have one or more  $x$ 's between  $u$  and  $v$ . For example, *really* is repeatable in context  $(She's, nice)$  in English since *She's really nice, She's really really nice, She's really really really nice* etc are all sentences of English.<sup>10</sup>

---

<sup>9</sup>In this chapter, I will call the elements of the lexicon “words” but they could be morphemes or even sounds, and the notion would be the same.

<sup>10</sup>Languages may also have other sorts of repeated elements, as for example in (i-a). Notice though that this does not follow our definition of repetition: (i-a) requires *exactly* two *fly*'s. One or three are not grammatical.

- (i)
  - a. Why won't the **fly fly** away?
  - b. \*Why won't the **fly** away?
  - c. \*Why won't the **fly fly fly** away?

The difference between (2) above and (i) is that the repeated element (*fly*) in (i) has two-way dependencies, while the repeated elements in (2) (*red* and *really*) only have one-way dependencies. The first *fly* is

It is not obvious what exactly it means to ask whether optionality and repetition are learnable. A simple definition of *is repetition learnable?* is *are there any learnable classes  $\mathcal{L}$  such that for some  $L \in \mathcal{L}, \exists u, x, v \in \Sigma^*$  such that  $ux^*v \subseteq L$ ?* That is, do any learnable classes have even one incidence of repetition in even one language? This question is readily answered: yes, in both senses. For example, the class containing only  $a^*$  is trivially learnable in that you could define a learner that just guesses  $a^*$  whatever its input.  $a^*$  has optionality since  $\epsilon, a \in a^*$ , so  $a$  is optional in the empty context  $(\epsilon, \epsilon)$ .  $a$  is repeatable in  $(\epsilon, \epsilon)$  since  $a^+ \subset a^*$ .

Slightly more interesting is to ask whether a certain learner can learn repetition. This amounts to asking whether the class of languages it can learn includes some  $L$  such that  $\exists u, x, v \in \Sigma^*, x \neq \epsilon$ , such that  $ux^+v \subseteq L$ . This question is also easily answered; it suffices to find an example, and they abound. All the language classes I will look at in this chapter contain a language with a repeating substring.

Of more interest is to ask what a learner must encounter in order to generalise to indefinite repetition or optionality. For example, is hearing an element repeated once in a context enough? (i.e. if the sample contains  $uxv$  and  $uxxv$  does the learner guess a language that includes  $ux^+v$ ?) What other conclusions will it draw about  $x$ ?

I will now consider five learners for five languages classes.

## 5.4 N-gram learners

**Definition 5.4.1** (n-gram). For  $n \in \mathbb{N}$ , an  $n$ -gram is a string of length  $n$ .  $g$  is an  $n$ -gram of string  $s$  just in case  $g$  is a substring of  $s$ . A *substring* is a contiguous subsequence. Often

---

dependent on the determiner *the*, and *the* is also dependent on it: without *fly*, the phrase becomes ungrammatical. The second *fly* is a verb, and is thus dependent among other things on its subject (*the fly*), and the subject is dependent on it. Conversely, in (2-a), *red* is dependent on *rose*, but *rose* is not dependent on *red*: the sentences is just fine without either *red*. Similarly, in (2), *really* is dependent on *nice*, but not vice-versa.

left and right boundary markers  $\{ \times, \times \}$  are included, so that  $s$  is written as  $\times s \times$  and the boundary markers are included in the  $n$ -grams. A bigram is a 2-gram and a trigram is a 3-gram.

An  $n$ -gram learner (Garcia et al., 1990), for some  $n \in \mathbb{N}$ , learns languages defined entirely by good substrings of length  $n$ . It simply memorises all  $n$ -grams it encounters, and accepts/generates strings that contain only  $n$ -grams from the list it memorised. This is a Gold learner, and could also be described as a PAC learner.

Human language syntax does not belong to the classes of  $n$ -gram languages for any  $n$ . This is due to unbounded dependencies such as that between *she* and *herself* in (10-a).

- (10) a. **She** really really ... really doesn't want to hurt **herself**  
b. \***She** really really ... really doesn't want to hurt **himself**  
c. **He** really really ... really doesn't want to hurt **himself**

Suppose we wanted to make a set of 7-grams that could account for the grammaticality of (10-a) and (10-c) and the ungrammaticality of (10-b). Our set of good 7-grams would include *she really really really really really really* (from (10-a)), *really really doesn't want to hurt himself* (from (10-c)), and *really really really really really really really doesn't want to hurt* from both. These are the 3 7-grams in (11), which should be ungrammatical.

- (11) \**She really really really really really really really doesn't want to hurt himself*

No matter how big we make our  $n$ , we will always be able to split a sentence into  $n$ -grams that include only *she* and  $n$ -grams that include only *herself*, making the dependency impossible to maintain. However, much of phonology is definable by  $n$ -grams and generalisations thereof (Heinz, 2010).

As an example of how an  $n$ -gram learner works, suppose a bigram learner encoun-

ters  $ac$ ,  $abc$ . Let  $\times$  and  $\times$  mark word boundaries. The learner generates the following grammar. A checkmark in a cell means that a bigram starting with the row symbol and ending with the column symbol is legal. For example, the checkmark in the  $c$  row under  $\times$  means that  $c\times$  is a legal bigram, meaning words can end with  $c$ .

	$\times$	a	b	c	$\times$
$\times$		✓			
a			✓	✓	
b				✓	
c					✓
$\times$					

Table 5.1: Grammar 1

A word is a valid word of the language if it begins with  $\times$ , ends with  $\times$ , and contains only bigrams from this grammar. This grammar does not generalise beyond the input strings: only  $ac$  and  $abc$  are valid strings.

We can see this by trying to build words. We must begin with  $\times$ . According to the grammar, only  $a$  can follow  $\times$ , so any word we build will start with  $a$ .  $a$  can be followed by either  $b$  or  $c$ , so we split into two words, beginning  $\times ab$  and  $\times ac$ .  $b$  can only be followed by  $c$ , so word 1 is now  $\times abc$ .  $c$  can only be followed by  $\times$ , so both words must end now, giving us  $\times abc\times$  and  $\times ac\times$ .

Suppose now the learner hears a third string,  $abbc$ . We update the bigram chart, adding  $bb$ . (The other bigrams in this string are already present.)

	$\times$	a	b	c	$\times$
$\times$		✓			
a			✓	✓	
b			✓	✓	
c					✓
$\times$					

Table 5.2: Grammar 2

Now we have grammar that generates an infinite language  $ab^*c$ , which is  $ac$  with zero or more  $bs$  in the middle. For example, the bigrams of  $abbbbbc$  are  $\{\times a, ab, bb, bc, c \times\}$ , just like those of  $abbc$ . The learner has generalised to indefinite repeatability from a single repetition.

Generalising this observation to repeating element  $x$  longer than one symbol, we have Theorem 5.4.2 which gives a sufficient sample for learning  $ux^+v$ .

**Theorem 5.4.2.** *Let  $u, x, v \in \Sigma^*$  and take  $n \leq |uxv|$ . For an  $n$ -gram learner to learn a language containing  $ux^+v$ , it suffices for the sample to include*

$$\{ux^d v \mid d \in \mathbb{N}, 1 < d < (n/|x| + 2)\}$$

*Proof.* Let  $k = n/|x| + 1$  be the number of  $x$ s required to completely fill an  $n$ -gram with  $x$ s. If the sample includes  $ux^k v$  then clearly we have enough  $x$ s to have an  $n$ -gram with only  $x$ s in it. This is not enough; in order to generate sentence with more than  $k$   $x$ s in that context, we also need  $n$ -grams  $x_2 \dots x_1, x_3 \dots x_2$ , etc (generally  $x_i \dots x_{i-1}$ ) to loop around and permit additional  $x$ s. We do not have that: the  $n$ -gram with the most  $x$ s that starts with  $x_2$  is  $x_2 \dots v_1$ . We therefore need one more  $x$ : if our sample includes  $ux^{k+1}v$  then we can generate a language that includes  $ux^{k+1}v$ .

We can not yet generate sentences with fewer than  $k + 1$   $x$ s, since we don't have an  $n$ -gram that includes both the end of  $u$  and the beginning of  $v$ . We need a sample with every number of  $x$ s up to  $k + 1$  to cover the remaining.

□

For the learner to learn  $ux^*v$ , i.e. as above but including  $uv$ , the sample need merely also include  $uv$ .

The gist of it is that once the sample includes sentences that contain the repeated phrase enough times to fill up an  $n$ -gram, and every number less than that, the learner

will generalise to indefinite repetition. Note that this is a sufficient condition, not a necessary one. Some situations may need less.

In the bigram example above, the repeated element  $b$ , the  $x$ , is of length 1, so the learner may need as many as 3  $bs$ ; as it happens it only needs 2.

Suppose we have a trigram learner and the sample so far is  $\{abcdab\}$ , and suppose the string that will repeat is  $cd$ , which has length 2. The number of  $cd$ 's needed to fill a trigram is 2. (This is  $k$  in the proof.) This sample should not suffice to generate  $ab(cd)^+ab$ ; we may need as many as 3  $cds$ .

The learner stores trigrams  $\{ \times ab, abc, bcd, cda, dab, ab \times \}$ .

This grammar does not generate  $abcdcdab$  because it contains trigram  $cdc$  ( $abcdcdcd$ ) and the trigram list does not. This is not surprising, since Theorem 5.4.2 says we may need our sample to contain  $abcdcdcdab = ab(cd)^{2+1}ab$ . In our case, however, just two  $cd$ 's suffices. It adds trigrams  $cdc$  and  $dcd$ , which opens the door to indefinite repetition of  $cd$ .

Sample 2 =  $\{abcdab, abcdcdab\}$ .

The learner adds trigrams  $cdc, dcd$ , yielding trigram list

$\{ \times ab, abc, bcd, cda, dab, ab \times, cdc, dcd \}$ .

These trigrams include a trigram for entering a sequence of  $cd$ 's from the left ( $bcd$ ), a trigram for repeating  $cd$  starting at  $c$  ( $cdc$ ), a trigram for repeating  $cd$  starting at  $d$  ( $dcd$ ), and a trigram for exiting a sequence of  $cd$ 's ( $cda$ ).

If instead we had a bigram learner, the repeating bigram would fit exactly into an  $n$ -gram. For a bigram learner learning this language,  $k = 1$ . However,  $\{abab, abcdab\}$  is not a sufficient sample to learn the repeatability of  $cd$ , since the learner does not get bigram  $dc$ . Here we need the full  $k + 1 = 2$   $cds$ :  $\{abab, abcdab, abcdcdab\}$

## 5.5 0-reversible learner

A 0-reversible language (Angluin, 1982) is a regular language with the property that any two prefixes of a valid sentence that share one suffix share all suffixes. A learner for a 0-reversible language builds a grammar based on this assumption. The procedure is simple: we start with a prefix tree of the input text. First final states are merged. Then we work from the end of the automaton, merging states that share a suffix.

**Definition 5.5.1** (Prefix, Suffix). A *prefix* of string  $s$  is a string  $x$  such that  $\exists y \in \Sigma^*$  such that  $xy = s$ . A *suffix* of string  $s$  is a string  $x$  such that  $\exists y \in \Sigma^*$  such that  $yx = s$ .

That is, a prefix of a sentence is any subpart that starts at the beginning, and a suffix is an subpart that ends at the end. For example, *Patience shot Mal* has four prefixes:  $\epsilon$  (nothing), *Patience*, *Patience shot*, and *Patience shot Mal* (the whole thing). It has suffixes *Patience shot Mal*, *shot Mal*, *Mal*, and  $\epsilon$ .

The set of prefixes of a set of strings  $S$  is  $Pre(S) = \{x \mid \exists s \in S \text{ s.t. } x \text{ is a prefix of } S\}$

**Definition 5.5.2** (0-reversible language).  $L \in \mathcal{L}_{0\text{-rev}}$  iff  $\forall s, t, u, v \in \Sigma^*$  if  $su, sv, tu \in L$  then  $tv \in L$

A 0-reversible language, then, is one in which, if we have two sentences that start differently but end the same way, such as *John left* and *Mary left*, and we also have the sentence *Mary slept*, then *John slept* is also a valid sentence.

The learner starts with a prefix tree, which is a special case of a finite state automaton. The 0-reversible grammar we then build is also an FSA.

**Definition 5.5.3** (Deterministic Finite State Automaton). A deterministic finite state automaton (DFSA) is a five-tuple

$$\langle \Sigma, Q, q_0, F, \delta \rangle$$

where:



$\Sigma$  is an alphabet

$Q$  is a finite set (*states*)

$q_0 \in Q$  is the designated *start state*

$F \subseteq Q$  is the set of *final states*

$\delta : Q \times \Sigma \rightarrow Q$  is the *transition function*

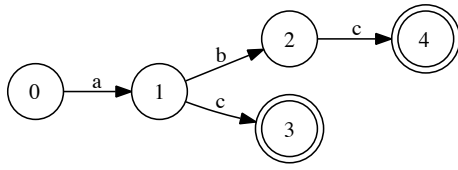
A prefix tree can be defined by the following algorithm, where  $\text{Tree} \subseteq (Q \times \Sigma) \times Q$ . For a sequence  $s$ ,  $s[i]$  is the  $i^{\text{th}}$  element, counting from 0, and  $s[i:]$  is the  $i^{\text{th}}$  and subsequent elements.

**Definition 5.5.4** (Prefix Tree (Angluin, 1982)). A *prefix tree* for a set of strings  $S$  is a finite state automaton  $P(S) = \langle Q, I, F, \delta \rangle$  where

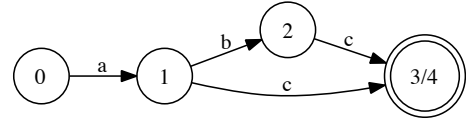
- $Q = \text{Pre}(S)$
- $I = \{\epsilon\}$  if  $S \neq \emptyset$ , otherwise  $I = \emptyset$
- $F = S$
- $\delta(u, a) = ua$  if  $u, ua \in Q$

Angluin (1982)'s 0-reversible learner generalises directly from optionality to indefinite repeatability. The learner starts with a prefix tree (Figure 5.4a) and then merges states with any suffix in common. For example, suppose we have two input strings  $ac$  and  $abc$ . (Here  $b$  is optional.) First the final states are merged since they share the suffix  $\epsilon$  (the empty string) (Figure 5.4b). Next, states 1 and 2 are merged, forming a loop, since the prefixes  $a$  and  $ab$  share the suffix  $c$  (Figure 5.4c).

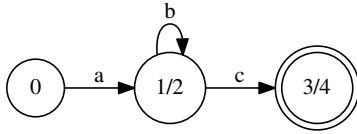
In 0-reversible languages, optionality and repetition co-occur.  $x \in \Sigma^*$  is optional in context  $C$  if and only if it is repeatable in  $C$ . A pair of simple inductive proofs is enough to show this.



(a) Prefix tree



(b) Grammar after states 3 and 4 are merged



(c) Grammar after states 1 and 2 are merged

Figure 5.4: A run of the 0-reversible learner

**Lemma 5.5.5** (0-reversible language: Optionality  $\therefore$  Repetition).

Let  $u, v, x \in \Sigma^*$  and let  $uv, uxv \in L \in \mathcal{L}_{0\text{-rev}}$ . Then  $ux^*v \subseteq L$

*Proof by induction on number of  $x$ 's.*

**Base case:**  $ux$  and  $u$  share suffix  $v$ .

$u$  also has suffix  $xv$

$\therefore ux$  also has suffix  $xv$ ,

$\therefore uxv \in L$ .

**Inductive step:** Suppose  $ux^k v, ux^{k+1} v \in L$ . Then  $ux^{k+1}$  and  $ux^k$  share suffix  $v$ .  $ux^k$  also has suffix  $xv$ , so so does  $ux^{k+1}$ . Then  $ux^{k+2} v \in L$  □

**Lemma 5.5.6** (0-reversible language: Repetition  $\rightarrow$  Optionality).

Let  $ux^k v, ux^{k+1} v \in L$  for some  $k \geq 0$ . Then  $uv, uxv \in L$ .

*Proof.* We subtract  $i \leq k$  from  $k$  and proceed by induction on  $i$ .

**Base case** Let  $i = 0$ . Assume  $ux^k v, ux^{k+1} v \in L$ . Then  $ux^{k-0} v = ux^k v \in L$  by assumption.

**Inductive step** Let  $i > 0$ . Show that if  $ux^{k-i} v, ux^{k+1-i} v \in L$  then  $ux^{k-(i+1)} v \in L$ .

Suppose  $ux^{k-i} v, ux^{k+1-i} v \in L$ .  $ux^{k+1-i} v = ux^{k-i} x v$  so  $ux^{k-i}$  has suffixes  $xv$  and  $v$ .

$ux^{k-i} v = ux^{k-i-1} x v$  so  $ux^{k-i-1}$  also has suffix  $xv$ .

Since  $ux^{k-i-1}$  and  $ux^{k-i}$  share suffix  $xv$  they share all suffixes, including  $v$ , so  $ux^{k-i-1} v \in L$ .

□

**Theorem 5.5.7.** Let  $L$  be a 0-reversible language over  $\Sigma$ . Then  $\forall k \in \mathbb{N}, \forall u, v, x \in \Sigma^*$  if  $(ux^k v, ux^{k+1} v \in L)$  then  $(ux^* v \subseteq L)$

*Proof.* By lemma 5.5.6, if  $ux^k v, ux^{k+1} v \in L$  then  $uv, uxv \in L$ . By Lemma 5.5.5, if  $uv, uxv \in L$  then  $ux^* v \in L$ . □

This means that any learner for a 0-reversible language given a sample that contains  $ux^k v$  and  $ux^{k+1} v$  **for some**  $k \geq 0$  will correctly hypothesise a grammar that generates  $ux^k v$  **for all**  $k \geq 0$ .

0-reversible languages treat optionality and repetition as the same thing, which, as we have seen, is reasonable for some but not all adjuncts. However, Human language is supra-regular, so such a learner will not suffice anyway. Substitutable context free languages are closer to human language, and behave very similarly to zero-reversible languages.

## 5.6 Substitutable context free languages

Substitutable CF languages (Clark and Eyraud, 2007) are the context-free equivalent of 0-reversible languages. 0-reversible languages are defined by common suffix generalisation: if two prefixes share one suffix, they share all suffixes. Substitutable CF languages are defined by common *context* generalisation: if two substrings share one context, they share all contexts. This is a Gold-learnable class. Briefly, the learner tries all partitions of each input sentences and hypothesises CF rules of the forms  $A \rightarrow b$  and  $A \rightarrow BC$  where  $A, B, C$  are sets of contexts and  $b \in \Sigma$ . In the sample, the right hand sides of the rules have appeared in at least one of the contexts on the left hand sides of the rules.<sup>11</sup>

Like with 0-reversible languages, optionality and repetition co-occur.

**Lemma 5.6.1** (Substitutable CF: Optionality  $\rightarrow$  Repetition). *Let  $u, v, x \in \Sigma^*$  and  $uv, uxv \in L \in \mathcal{L}_{subCF}$  Then  $ux^*v \subseteq L(G_i)$ .*

*Proof.* By induction on the number of  $x$ s.

**Base case:**  $u, ux$  share context  $(\epsilon, v)$ .

$u$  also has context  $(\epsilon, xv)$  so  $ux$  also must have this context.

Therefore  $uxxv \in L(G_i)$ .

**Inductive step:** Suppose  $ux^k v, ux^{k+1} \in L$  for some  $k \geq 0$ . Then  $ux^{k+1}, ux^k$  share context  $(\epsilon, v)$ .

$ux^k$  also has context  $(\epsilon, xv)$  so  $u^{k+1}x$  also must have this context.

Therefore  $ux^{k+2}v \in L(G_i)$ .

□

**Lemma 5.6.2** (Substitutable CF: Repetition  $\rightarrow$  Optionality).

*Let  $ux^k v, ux^{k+1}v \in L$  for some  $k \geq 0$ . Then  $uv, uxv \in L$ .*

<sup>11</sup>See section 5.6.1 below for more details on both the learning algorithm and the learnability of the class.

*Proof.* We subtract  $i \leq k$  from  $k$  and proceed by induction on  $i$ .

**Base case** Let  $i = 0$ . Assume  $ux^k v, ux^{k+1} v \in L$ . Then  $ux^{k-0} v = ux^k v \in L$  by assumption.

**Inductive step** Let  $i > 0$ . Show that if  $ux^{k-i} v, ux^{k+1-i} v \in L$  then  $ux^{k-(i+1)} v \in L$ .

Suppose  $ux^{k-i} v, ux^{k+1-i} v \in L$ .  $ux^{k+1-i} v = ux^{k-i} xv$  so  $ux^{k-i}$  has contexts  $(\epsilon, xv)$  and  $(\epsilon, v)$ .

$ux^{k-i} v = ux^{k-i-1} xv$  so  $ux^{k-i-1}$  also has context  $(\epsilon, xv)$ .

Since  $ux^{k-i-1}$  and  $ux^{k-i}$  share context  $(\epsilon, xv)$  they share all contexts, including  $(\epsilon, v)$ , so  $ux^{k-i-1} v \in L$ .

□

**Theorem 5.6.3.** Let  $L$  be a substitutable CF language over  $\Sigma$ . Then  $\forall k \in \mathbb{N}, \forall u, v, x \in \Sigma^*$  if  $(ux^k v, ux^{k+1} v \in L)$  then  $(ux^* v \subseteq L)$

*Proof.* By lemma 5.6.2, if  $ux^k v, ux^{k+1} v \in L$  then  $uv, uxv \in L$ . By Lemma 5.6.1, if  $uv, uxv \in L$  then  $ux^* v \in L$ . □

This means that any learner for a substitutable CF language given a sample that contains  $ux^k v$  and  $ux^{k+1} v$  **for some**  $k \geq 0$  will correctly hypothesise a grammar that generates  $ux^k v$  **for all**  $k \geq 0$ .

### 5.6.1 Learnability of substitutable CF languages

The learning algorithm works by first creating a graph of substitution classes, and then a grammar from that graph.

**Definition 5.6.4** (Substitution Graph). Given a finite sample  $S$ , define the substitution

graph  $SG(S) = \langle V, E \rangle$  as follows:

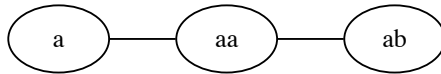
$$V = \{u \in \Sigma^+ | \exists l, r \in \Sigma^*, lur \in S\}$$

$$E = \{(u, v) \in \Sigma^+ \times \Sigma^+ | u \text{ and } v \text{ share a context in } S\}$$

That is, the nodes are the substrings of the sample and the edges connect substrings that have a common context.

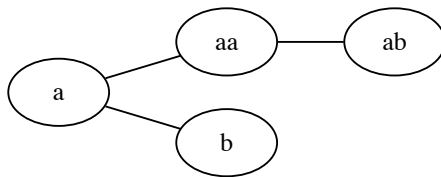
**Example** This small example results in a graph with only one component.

Sample= $\{a, aa, ab\}$ .  $V = \{a, b, aa, ab\}$ .  $a, aa, ab$  share context  $(\epsilon, \epsilon)$  so we have



In fact, we should take the reflexive, transitive closure, since intersubstitutability is reflexive, symmetrical, and transitive but for readability I will leave the graph as is and simply declare it transitive and reflexive.

$a, b$  share  $(a, \epsilon)$ , so we extend the graph:



The algorithm for making a grammar from this graph is as follows. Each component of the graph (each set of connected nodes) is a non-terminal category. They are equivalence classes so we can notate them with an exemplar in square brackets. For example,

if we have a component consisting of  $a$  and  $b$ , we can call the component  $[a]$  or  $[b]$ . We break the substrings of the sample (the nodes,  $V$ ) into all their possible partitions. If the substring is just one symbol  $a$ , we make a lexical rule  $[a] \rightarrow a$ . Otherwise, we split them every way possible, for example  $abc = (ab, c)$  and  $abc = (a, bc)$ . For each split we make a rule, forming  $[abc] \rightarrow [ab] [c]$  and  $[abc] \rightarrow [a] [bc]$ .

Formally, we use Algorithm 1.

<p><b>Data:</b> Substitution graph <math>SG = \langle V, E \rangle</math></p> <p><b>Result:</b> CFG <math>\hat{G} = \langle \Sigma, \hat{V}, \hat{S}, \hat{P} \rangle</math></p> <p>Let <math>\Sigma</math> be the set of all symbols used in <math>V</math>;</p> <p>Compute <math>\hat{V}</math> the set of graph components of <math>\hat{G}</math>;</p> <p>Let <math>\hat{S}</math> be the component corresponding to context <math>(\epsilon, \epsilon)</math>;</p> <p><math>\hat{P} \leftarrow \emptyset</math>;</p> <p><b>for</b> <math>u \in V</math> <b>do</b></p> <table style="border-left: 1px solid black; border-right: 1px solid black; padding-left: 20px;"> <tr> <td style="border-right: 1px solid black; padding-right: 5px;"><b>if</b> <math> u  &gt; 1</math> <b>then</b></td> <td style="padding-left: 5px;"><b>for</b> <math>v, w</math> s.t. <math>u = vw</math> <b>do</b></td> </tr> <tr> <td style="border-right: 1px solid black; padding-right: 5px;"> </td> <td style="padding-left: 5px;"><math>\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow [v] [w])</math></td> </tr> <tr> <td style="border-right: 1px solid black; padding-right: 5px;"> </td> <td style="padding-left: 5px;"><b>end</b></td> </tr> <tr> <td style="border-right: 1px solid black; padding-right: 5px;"><b>else</b></td> <td style="padding-left: 5px;">  <math>\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow u)</math></td> </tr> <tr> <td style="border-right: 1px solid black; padding-right: 5px;"> </td> <td style="padding-left: 5px;"><b>end</b></td> </tr> </table> <p><b>end</b></p> <p>output <math>\hat{G} = \langle \Sigma, \hat{V}, \hat{S}, \hat{P} \rangle</math></p>	<b>if</b> $ u  > 1$ <b>then</b>	<b>for</b> $v, w$ s.t. $u = vw$ <b>do</b>		$\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow [v] [w])$		<b>end</b>	<b>else</b>	$\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow u)$		<b>end</b>
<b>if</b> $ u  > 1$ <b>then</b>	<b>for</b> $v, w$ s.t. $u = vw$ <b>do</b>									
	$\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow [v] [w])$									
	<b>end</b>									
<b>else</b>	$\hat{P} \leftarrow \hat{P} \cup ([u] \rightarrow u)$									
	<b>end</b>									

**Algorithm 1:** Generates a grammar from a substitution graph

This method can result in a very large and/or ambiguous grammar, for a variety of reasons. For example, suppose the target grammar is

$$\langle \{a\}, \{S, A\}, \{S \rightarrow a, A \rightarrow a, S \rightarrow AS\}, \{S\} \rangle$$

and the sample is  $(a, aa)$ .

According to the target grammar the derivations are as follows



Figure 5.5: Derivations of  $a, aa$  in the target grammar

Applying the learner, we get a graph with one component consisting of  $\{a, aa\}$ . We use each of the nodes,  $a, aa$  to make rules.  $a$  is unary so we get a lexical rule  $[a] \rightarrow a$ .  $aa$  splits once, into  $a$  and  $a$ , so we get  $[a] \rightarrow [a][a]$ .  $[a]$  fits in  $(\epsilon, \epsilon)$ , so it's the start category. The resulting grammar is

$$\langle \{a\}, \{S\}, \{S \rightarrow a, S \rightarrow SS\}, \{S\} \rangle$$

$aaa$  is in the target language, and is generated by the hypothesis grammar. However, in the target grammar it is unambiguous, but in the hypothesis grammar it is ambiguous.

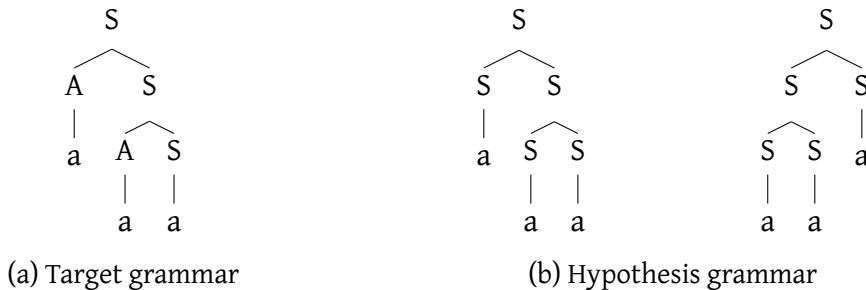


Figure 5.6: Ambiguity arises in the hypothesis grammar

The proof that this learner works – that, given enough representative data, it will converge on a grammar that generates the target language – follows fairly readily from the construction of the grammar. The hypothesis grammar does not overgenerate, no matter the sample, because the grammar is constructed so directly from the sample and



because it only takes one common context to make two substrings equivalent. The authors show this in two lemmas, the first showing that the grammar can indeed derive any substring  $w$  from its category  $[w]$  (Lemma 10 in Clark and Eyraud (2007)). I repeat the proof here, because all of the results use similar mechanisms.

**Notation**  $A \Rightarrow_G^* u$  means that from category  $A$ , grammar  $G$  can derive string  $u$

$A \Rightarrow_G^k u$  means that from category  $A$ , grammar  $G$  can derive string  $u$  in  $k$  steps.

**Lemma 5.6.5** (Clark and Eyraud (2007)). *For any sample  $C$ , if  $w$  is a substring in the sample then  $[w] \Rightarrow_G^* w$*

*Proof.* By induction on the length of  $w$ . If  $|w| = 1$  then by the construction of the grammar there is a lexical rule  $[w] \rightarrow w$ .

If  $w = k + 1$  then  $w = ua$  for some  $|u| = k$  and  $a \in \Sigma$ . By the inductive hypothesis,  $[u] \Rightarrow_G^* u$  and we already saw that there is a lexical rule in the grammar  $[a] \rightarrow a$ . By the construction of the grammar, there is a production  $[w] \rightarrow [u] [a]$ .

□

Lemma 11 in Clark and Eyraud (2007) shows that the hypothesis grammar maintains the equivalence classes of substrings: if the hypothesis grammar can derive  $x$  from category  $[u]$ , then  $x$  and  $u$  do indeed share all the same contexts in the target language. This can be seen by an induction on the length of derivations, and the inductive step again depends on the fact that the grammar is built inductively with rules  $[u] \rightarrow [v][w]$  for each way of splitting up  $u$  into substrings, and by the inductive hypothesis,  $v$  and  $w$  are equivalent to some way of splitting up  $x$  into  $x_1$  and  $x_2$ .

Thus there is nothing the grammar will put in the same contexts that should not be in the same contexts, so the hypothesis grammar cannot overgenerate.

To see that the learner does not undergenerate, Clark and Eyraud show that there is

a sufficient sample, called a characteristic sample, that will guarantee the learner has enough information to generate the whole language.

A characteristic sample for a target grammar follows all of the smallest examples for each expansion in the grammar. “Smallest” means fewest symbols, and an order is defined on  $\Sigma$  to resolve disputes when two sentences are of the same length.  $CS(G) = \{lwr | \exists \alpha \in (\Sigma \times V) \text{ s.t. } (N \rightarrow \alpha) \in P, (l, r) \text{ is the smallest pair of substrings such that } S \Rightarrow_G^* lNr, \text{ and } w \text{ is the smallest word generated by } \alpha\}$

The characteristic sample is bounded in size by the size of the target ruleset  $P$ , so it is indeed finite. That the characteristic sample suffices to learn the target grammar follows by induction on the length of the derivation and the way CS is built.

Thus the learner laid out in Clark and Eyraud (2007) is indeed a Gold learner.

### 5.6.2 Suitability of substitutable CF languages for human language

Human language is not substitutable. In example (12), *he* and *she* both occur in context  $(\epsilon, \textit{left})$  but only *he* can occur in  $(\epsilon, \textit{shot himself in the foot})$

- (12)
- a. He left
  - b. She left
  - c. He shot himself in the foot
  - d. \*She shot himself in the foot

However, this is not to say that a substitutable learner is useless for our purposes. Human language may not be substitutable, but intersubstitutability plays a major role in word classes. Indeed, substitution classes were the basis for what is arguably the first generative grammar, Zellig Harris’s substitution grammars (Harris, 1954; Chomsky, 1953). Learners for classes of languages closer to human language, including CF languages with finite kernel and context properties (Clark et al., 2010) discussed in section 5.8, take this

learner as their starting point.

## 5.7 Clark & Thollard

Clark and Thollard (2004) describe a PAC (Probably Approximately Correct) learner of probabilistic finite state languages. The learner is similar to Angluin’s 0-reversible learner, except that the criterion for merging states is stricter: the similarity of the suffix sets of two states must be within a pre-determined margin for them to be merged. One suffix in common is not enough.

This learner can learn repeatably if the input is representative of the probabilities in the generating grammar; i.e. under normal circumstances. Unlike the 0-reversible and substitutable CF learners, there is no “short cut”.

For example, suppose the language to be learned is  $ab^*c$ , with the distribution  $p(ab^n c) = (1/2)^{n+1}$ . This PDFA generates  $L$ :

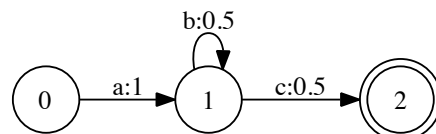
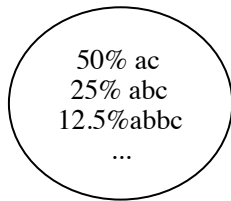
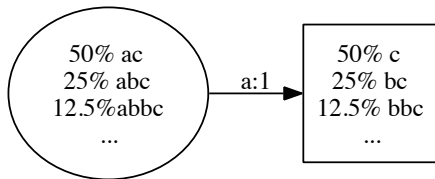


Figure 5.7:  $L = ab^*c$

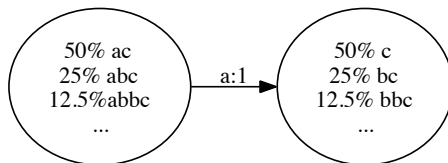
The learner hypothesises a DFA such that the states are the suffix sets in the sample. It then considers possible “candidate nodes” to follow each node in the hypothesis grammar. If a candidate node is similar enough to an existing node, they are merged (i.e. the existing node is given a new transition). After the first iteration of the learner trained on  $L$ , the hypothesis grammar will look like this:



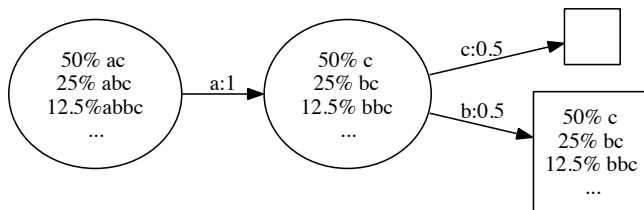
Next candidate nodes are considered. Since all strings start with *a* only one candidate is proposed (shown as a square):



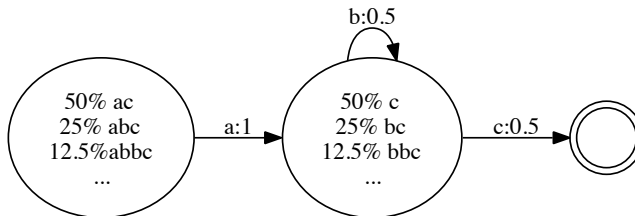
Since these two nodes do not have similar suffix sets, they are not merged, and the candidate becomes a real node:



Next, more samples are drawn and candidates are proposed, one following *b* and the other following *c*. Nothing follows *c* so the latter suffix set is empty.



Since the candidate node with transition  $b$  and the real node before it have similar suffix sets, the two are merged, yielding a loop.



Clark & Thollard's learner can learn any PDFFA with an upper bound on the expected length of strings and size of machine, along with  $\mu$ -distinguishability, which means that any pair of states is such that there is at least one suffix on which they differ by at least the threshold  $\mu$ .

**Definition 5.7.1.** For  $\mu > 0$  two states  $q_1, q_2$  are  $\mu$ -distinguishable if there is a string  $s$  such that the differences in the probabilities of  $s$  as a suffix of  $q_1$  and  $s$  as a suffix of  $q_2$  is at least  $\mu$ .

Unlike for n-gram learners, 0-reversible learners, and substitutable CF learners, this learner needs a representative sample to learn repetition: there are no short cuts. Repetition is, however, perfectly learnable.

## 5.8 Context free languages with finite kernel and context properties

Context free languages with the finite kernel and finite context properties (Clark et al., 2010) are a MAT-learnable class of context free languages. Loosely, these are CF languages such that you can make a context-free grammar using just sets of contexts substrings can appear in. CFGs have a finite set of category names. If a CFL has the finite kernel and context properties, a grammar built out of just a finite subset of the infinite possible substrings and contexts generates the right language.

The learner laid out in Clark et al. (2010) works by decomposing sentences into substrings and their contexts, and building phrase structure rules in which the categories are sets of contexts. Lexical rules have as their left-hand side sets of contexts a word can appear in. Non-lexical rules are entirely sets of contexts. These grammars are called *contextual binary feature grammars*.

For example, suppose we want to build a contextual binary feature grammar that generates just  $abc$ . We make two lists, one of contexts of and one of non-empty substrings of  $abc$ . For example,  $a$  is a substring of  $abc$  and since when you put  $a$  before  $bc$  you get the original sentence back,  $(\epsilon, bc)$  is a context in  $abc$ .

Substrings	Contexts
$abc$	$(\epsilon, \epsilon)$
$ab$	$(\epsilon, c)$
$bc$	$(a, \epsilon)$
$a$	$(\epsilon, bc)$
$b$	$(a, c)$
$c$	$(ab, \epsilon)$

Table 5.3: Substrings and contexts in  $abc$

To get the lexical rules, we take all the contexts in which each of  $a$ ,  $b$ , and  $c$  can occur. In this case, it's just one context per word. The category names are these sets of contexts.

- $\{(\epsilon, bc)\} \rightarrow a$
- $\{(a, c)\} \rightarrow b$
- $\{(ab, \epsilon)\} \rightarrow c$

To get the non-lexical rules we see what contexts longer substrings can appear in and then split the substrings in two and see what contexts the two halves can appear in. For example,  $ab$  can occur in one context,  $(\epsilon, c)$ . Therefore we have a rule with left-hand side (LHS)  $\{(\epsilon, c)\}$ .

- $\{(\epsilon, c)\} \rightarrow$

To get the right-hand side (RHS) we look at contexts of  $a$  and  $b$ . Those we already know:  $\{(\epsilon, bc)\}$  and  $\{(a, c)\}$  respectively. Those sets become the two daughters.

- $\{(\epsilon, c)\} \rightarrow \{(\epsilon, bc)\} \{(a, c)\}$

Similarly, for  $bc$  we get the rule

- $\{(a, \epsilon)\} \rightarrow \{(a, c)\} \{(ab, \epsilon)\}$

Finally, we look at  $abc$ , which can be split in two in two ways:  $ab, c$  and  $a, bc$ . We do the same as we did for  $a, b$  to these two pairings. The LHS is just  $\{(\epsilon, \epsilon)\}$  for both rules.

- $\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, bc)\} \{(a, \epsilon)\}$  *things that can go before  $bc$  (ie  $a$ ) followed by things that can go after  $a$  (ie  $bc$ )*
- $\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, c)\} \{(ab, \epsilon)\}$  *things that can go before  $c$  (ie  $ab$ ) followed by things that can go after  $ab$  (ie  $c$ )*

Thus we get the following grammar:

- $\{(\epsilon, bc)\} \rightarrow a$
- $\{(a, c)\} \rightarrow b$
- $\{(ab, \epsilon)\} \rightarrow c$
- $\{(\epsilon, c)\} \rightarrow \{(\epsilon, bc)\} \{(a, c)\}$
- $\{(a, \epsilon)\} \rightarrow \{(a, c)\} \{(ab, \epsilon)\}$
- $\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, bc)\} \{(a, \epsilon)\}$
- $\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, c)\} \{(ab, \epsilon)\}$

Figure 5.8: Grammar

To make this grammar more readable, we can rename the categories.

Old name	New name
$\{(\epsilon, bc)\}$	A
$\{(a, c)\}$	B
$\{(ab, \epsilon)\}$	C
$\{(\epsilon, c)\}$	AB
$\{(a, \epsilon)\}$	BC
$\{(\epsilon, \epsilon)\}$	S

Table 5.4: Old and new category names

This gives us the following grammar:



- $A \rightarrow a$
- $B \rightarrow b$
- $C \rightarrow c$
- $AB \rightarrow A B$
- $BC \rightarrow B C$
- $S \rightarrow A BC$
- $S \rightarrow AB C$

Figure 5.9: Grammar

$abc$  has two parses.



Figure 5.10: Two parses of  $abc$

Notice that all we have done here is restate the sentence. This grammar generates exactly  $\{abc\}$ . The learner does more than what we have done here. It also has an *oracle*: the learner can ask *is this sentence ok?* The oracle is a recogniser using the target grammar: a function that can determine the grammaticality of a sentence given a grammar.

The way the learner works is, instead of looking only at the contexts in which a substring has been seen, it also tries the substring in every context it has seen in the whole sample and asking the oracle if the result is grammatical. Then the category is built not only from the contexts in which the substring has been seen, but also the known contexts in which the substring could be seen. This allows for generalisation.

The oracle may sound far too powerful, almost as though the learner need merely ask the oracle for the grammar and be done. Two crucial points belie this notion. First, notice that the oracle is only ever asked about sentences built from substrings and contexts that the learner has seen. It is not free to make general inquiries. Second, Clark et al conjecture that the oracle could be replaced by a probability distribution on the input. This means essentially that the oracle is arguably just a computationally convenient stand-in for the fact that the child has been surrounded by huge amounts of language, and even if she is not actively learning from all the input (i.e. if each sentence is not actual input to the learning algorithm), she still remembers a lot of it and can consult her memory: thinking “*Have I ever heard this sentence?*” rather than asking “*Mummy, is this grammatical?*”

The learning algorithm works as follows: Let  $D$  be the sample of the language seen so far. We build a kernel  $K$  of substrings of  $D$  and a set  $F$  of contexts. We consult the oracle to see which substrings can go in which contexts in  $D$ , and create rules accordingly. Definition 5.8.1 defines the set of contexts of a substring, given a particular set of possible contexts.

**Definition 5.8.1** (The contexts of a substring). Let  $L$  be a language,  $u$  be a string, and  $F$  be a set of contexts. Then  $F(u) = \{(x, y) | (x, y) \in F \ \& \ xuy \in L\}$

The context free grammar built from a kernel  $K$  and set of contexts  $F$  is defined as follows. The oracle is the source of information for the question of whether a sentence is in the grammar, which is needed to build the sets  $F(u)$ ,  $F(v)$ , and  $F(uv)$ .

**Definition 5.8.2** ( $G_0(K, F, O)$ ). Let  $K$  be a set of strings,  $F$  be a set of contexts, and  $O$  be a membership oracle.  $G_0(K, F, O)$  is the context free grammar defined as follows:

$P_L$  is the set of lexical rules  $\{F(u) \rightarrow u | u \in K \ \& \ |u| = 1\}$

$P$  is the set of non-lexical rules  $\{F(uv) \rightarrow F(u)F(v) | u, v, uv \in K\}$

$(\epsilon, \epsilon)$  is the start category

In other words, the grammar is built by trying all combinations of substrings and

contexts, and checking the oracle to see if the sentence is grammatical. The rules look just like those in the example above.

**Definition 5.8.3.** A finite set  $K \subseteq \Sigma^*$  is a *kernel* for a language  $L$  if, for any set of contexts  $F$ , and  $O$  an oracle for  $L$ ,  $L \subseteq L(G_0(K, F, O))$ . This means that a set of substrings  $K$  is a kernel for  $L$  if the language generated by the grammar you get when you put the substrings in  $K$  together with any set of contexts in the way described in definition 5.8.2 above contains  $L$ .

We need some notation to understand the algorithm.

**Notation (Con)**

For  $u \in \Sigma^+, U \subseteq \Sigma^+$ :

$$\text{Con}(u) = \{(x, y) | \exists z \in \Sigma^+ \text{ s.t. } xzy = u\}$$

$$\text{Con}(U) = \bigcup_{u \in U} \text{Con}(u)$$

That is, *Con* gives us all the contexts in a sentence or set of sentences.

**Notation (Sub)**

For  $u \in \Sigma^+, U \subseteq \Sigma^+$ :

*Sub*( $u$ ) is the set of non-empty substrings of  $u$

$$\text{Sub}(U) = \bigcup_{u \in U} \text{Sub}(u)$$

That is, *Sub* gives us all the substrings in a sentence or set of sentences.

**Notation ( $\odot$ )**

Let  $F$  be a set of contexts and let  $K$  be a set of substrings.

$$F \odot K = \{xuy | x, y \in F \ \& \ u \in K\}$$

That is,  $\odot$  builds all sentences from a set of contexts and substrings.

Figure 5.11 gives the learning algorithm. We go one sentence at a time, adding the current sentence  $w_i$  to  $D$ , the sample seen so far. We find all contexts and all substrings of  $D$ , and put them together in every possible way, forming  $T$ .

The two *if...then* statements capture over- and under-generalisation, respectively. In the case of over-generalisation, more contexts are needed in the grammar to more specifically constrain the distributions of substrings. In the case of under-generalisation, more information is needed in general, so more contexts and substrings are added.

---

**Algorithm for learning  $\text{CFG}_{F,K}$**

---

**Data:** A sequence of strings  $S = \langle w_1, w_2, \dots \rangle$ , membership oracle  $O$ .

**Result:** Sequence of grammars  $\langle G_1, G_2, \dots \rangle$

$K \leftarrow \emptyset; D \leftarrow \emptyset; G_0 = G_0(K, F, O);$

**for**  $w_i \in S$  **do:**

$D \leftarrow D \cup \{w_i\}$
$T \leftarrow \text{Con}(D) \odot \text{Sub}(D)$
<b>if</b> $\exists w \in T$ such that $w \in L(G_{i-1})$ but $w \notin L$ <b>then</b> $F \leftarrow \text{Con}(D)$
<b>if</b> $\exists w \in T$ such that $w \notin L(G_{i-1})$ but $w \in L$ <b>then</b> $F \leftarrow \text{Con}(D); K \leftarrow \text{Sub}(D)$
<b>Output</b> $G_i = G_0(K, F, O)$

**end**

Figure 5.11: Algorithm for learning  $\text{CFG}_{F,K}$

In practice, the algorithm looks something like this:

1. Get all the contexts in the sample (=F)
2. Get all the substrings in the sample (=K)
3. For each substring s, stick s into each context in F
4. Ask the oracle if you just built a grammatical sentence
5. if YES: make a rule:
  - (a) Using the oracle, make a list of all the contexts from F in which s can appear.  
That's your LHS
  - (b) if s is just one word, that's your RHS
  - (c) if s is longer, for each way of splitting s into 2 substrings,  $s_1, s_2$ :

- (d) Try sticking  $s_1$  into all the contexts in  $F$  and asking the oracle if you made a grammatical sentence. The list of all the usable contexts is the left daughter
- (e) Try sticking  $s_2$  into all the contexts in  $F$  and asking the oracle if you made a grammatical sentence. The list of all the usable contexts is the right daughter

Let us see what happens when we feed the learner  $abc$ , given the grammar  $G_{ab*c}$ :

non-lexical rules	lexical rules
$S \rightarrow A C$	$A \rightarrow a$
$C \rightarrow B C$	$B \rightarrow b$
	$C \rightarrow c$

Figure 5.12:  $G_{ab*c}$

Let  $O$  be the oracle for  $G_{ab*c}$ .

- $G_0(\emptyset, \emptyset, O)$  Start with an empty grammar, generating  $\emptyset$
- Input:  $abc$   $D = \{abc\}$
- Contexts:  $\{(\epsilon, \epsilon), (\epsilon, bc), (\epsilon, c), (a, c), (ab, \epsilon), (a, \epsilon)\}$
- Substrings:  $\{abc, ab, bc, a, b, c\}$
- Instead of building  $T$  all at once, let's build it one sentence at a time
- Try  $abc$  in  $(\epsilon, \epsilon)$ . Ask oracle  $abc \in L?$   $\rightarrow$  **True** under-generalised ( $abc \notin L(G_0) = \emptyset$ )
- We need to add all contexts to  $F$  and all substrings to  $K$  to make  $G_1$ . Build  $G_1 = G_0(F, K, O)$  using the oracle:
- Rule set 1:
  - Try  $abc$  in  $(\epsilon, \epsilon)$ . Ask oracle  $abc \in L?$   $\rightarrow$  **True**

- Try abc in all other contexts → **False**
- Split abc into a, bc:
  - \* Try a in  $(\epsilon, \epsilon)$  (F),  $(\epsilon, bc)$  (T),  $(\epsilon, c)$  (T),  $(a, c)$  (F),  $(ab, \epsilon)$  (F),  $(a, \epsilon)$  (F)
  - \* Try bc in  $(\epsilon, \epsilon)$  (F),  $(\epsilon, bc)$  (F),  $(\epsilon, c)$  (F),  $(a, c)$  (F),  $(ab, \epsilon)$  (T),  $(a, \epsilon)$  (T)
  - \* New rule:  $\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, bc), (\epsilon, c)\} \{(ab, \epsilon), (a, \epsilon)\}$
- Split abc into ab and c, do it again (gets the same rule.)
- Rule 2: Try ab in all contexts. Fits in  $(\epsilon, bc)$  and  $(\epsilon, c)$
- Split ab into a and b
  - Try a in all contexts. Fits in  $(\epsilon, bc)$  and  $(\epsilon, c)$
  - Try b in all contexts. Fits in  $(a, c)$ .
  - New rule:  $\{(\epsilon, bc), (\epsilon, c)\} \rightarrow \{(\epsilon, bc), (\epsilon, c)\} \{(a, c)\}$
- Rule 3: Similarly for bc, we get new rule  $\{(a, \epsilon), (ab, \epsilon)\} \rightarrow \{(a, c)\} \{(a, \epsilon), (ab, \epsilon)\}$
- Rule 4: a occurs in  $(\epsilon, c)$  and  $(\epsilon, bc) \rightarrow \{(\epsilon, c), (\epsilon, bc)\} \rightarrow a$
- Rule 5: b occurs in  $(a, c) \rightarrow \{(a, c)\} \rightarrow b$
- Rule 6: c occurs in  $(ab, \epsilon)$  and  $(a, \epsilon) \rightarrow \{(a, \epsilon), (ab, \epsilon)\} \rightarrow c$

$G_1 =$

non-lexical rules	lexical rules
$\{(\epsilon, \epsilon)\} \rightarrow \{(\epsilon, bc), (\epsilon, c)\} \{(ab, \epsilon), (a, \epsilon)\}$	$\{(\epsilon, c), (\epsilon, bc)\} \rightarrow a$
$\{(a, \epsilon), (ab, \epsilon)\} \rightarrow \{(a, c)\} \{(a, \epsilon), (ab, \epsilon)\}$	$\{(a, c)\} \rightarrow b$
$\{(\epsilon, c), (\epsilon, bc)\} \rightarrow \{(\epsilon, c), (\epsilon, bc)\} \{(a, c)\}$	$\{(a, \epsilon), (ab, \epsilon)\} \rightarrow c$

Figure 5.13:  $G_1$

Or, in readable form,

non-lexical rules	lexical rules
$S \rightarrow A C$	$A \rightarrow a$
$C \rightarrow B C$	$B \rightarrow b$
$A \rightarrow A B$	$C \rightarrow c$

Figure 5.14: Human-readable  $G_1$

Notice that  $G_1$  is not exactly the same as  $G_{ab^*c}$ : there is an extra rule  $A \rightarrow A B$ . This makes  $abc$  structurally ambiguous, as seen in Figure 5.15.



Figure 5.15:  $abc$  is structurally ambiguous

Now that we are using the learning algorithm, we have generalisation. This new grammar generates not just  $\{abc\}$  but  $ab^*c$ . This is because of the oracle. When the algorithm tried putting  $bc$  into context  $(ab, \epsilon)$ , forming the sentence  $abbc$ , the oracle was able to tell it that although  $abbc$  hadn't been seen yet, it was indeed grammatical. Similarly, substring  $a$  and context  $(\epsilon, c)$  had been encountered, so the algorithm asked the oracle for the grammaticality of new sentence  $ac$ .

In  $ab^*c$ ,  $b$  is both optional and repeatable. To determine this, the learner need only encounter the optional/repeatable element  $b$  in the context in which it is optional and repeatable  $(a, c)$ .

Assuming the oracle is a stand-in for memory, in real life, this means that the baby must have heard  $ac$  and  $abbc$  from time to time. Notice, however, that we never asked the oracle “is  $abbbbbbcb$  ok?”, yet we still built a grammar that generates it. That is, the

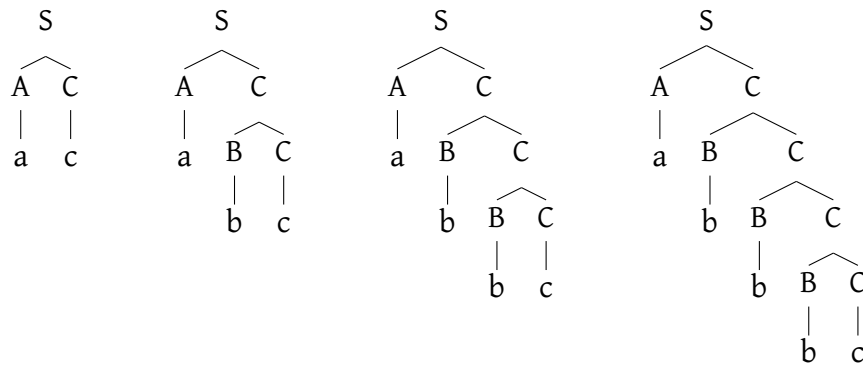
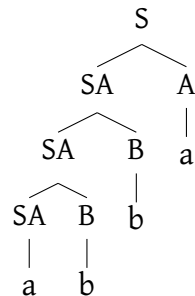


Figure 5.16:  $b$  is optional and repeatable in  $G_1$

learner need only encounter  $ac$ ,  $abc$ , and  $abbc$ , and need only analyse  $abc$ , in order to learn the optionality and indefinite repeatability of  $b$  in context  $(a,c)$ .

This fact does not extend to the general case, where any  $CFG_{F,K}$  language that includes  $ux^*v$ ,  $u, x, v \in \Sigma^*$ , has the property that a single instance of  $uxv$  in the input suffices to learn that  $ux^*v \subseteq L$ . For example, take the following grammar  $G$ , designed to distinguish cases where  $a$  can be followed by a (repeating)  $b$  and cases where it cannot be followed by  $b$  at all.

$S \rightarrow SA \ A \ A \rightarrow a$   
 $SA \rightarrow SA \ B \ B \rightarrow b$   
 $SA \rightarrow a$



$L(G) = ab^*a$ , so  $b$  is optional and repeatable in context  $(a, a)$ . Suppose we start with input  $aba$ , putting the optional/repeatable element in the context in which it is optional and repeatable. In this case, the learner does not immediately generalise to repeatability.

The reason is that one of the  $as$  can be followed by  $bs$ , but the other cannot be. The



result is that categories A and SA are kept separate, which prevents recursive rules like  $SA \rightarrow SA$  A from forming immediately.

Contexts:  $\{(\epsilon, \epsilon), (\epsilon, ba), (\epsilon, a), (a, a), (ab, \epsilon), (a, \epsilon)\}$

Contexts for  $a$ :  $\{(\epsilon, ba), (\epsilon, a), (ab, \epsilon), (a, \epsilon)\}$

Contexts for  $ab$ :  $\{(\epsilon, ba), (\epsilon, a)\}$

Since the contexts for  $a$  and  $ab$  are not the same, we make a rule

$$\{(\epsilon, ba), (\epsilon, a)\} \rightarrow \{(\epsilon, ba), (\epsilon, a), (ab, \epsilon), (\epsilon)\} \{(a, a)\} (= SA \rightarrow A B)$$

but not a rule

$$\{(\epsilon, ba), (\epsilon, a)\} \rightarrow \{(\epsilon, ba), (\epsilon, a)\} \{(a, a)\} (= SA \rightarrow SA B)$$

What we need is a situation in which one more  $x$  is not going to make for a different set of contexts. In the example above,  $aba$  does not suffice to learn repetition because  $a$  occurs not to the left of  $b$ . If the sample instead is  $abba$ , we are set. Although there will still correctly be a rule that keeps A separate from SA, we now get a recursive rule  $SA \rightarrow SA$  A as follows:

$abb \in K$  and  $\{(\epsilon, a), (\epsilon, ba), (\epsilon, bba)\} \subseteq F$ . The learner tries to make a rule with the contexts of  $abb$  on the LHS and those of  $ab$  and  $b$  on the RHS. The contexts of  $abb$  and  $ab$  are both  $\{(\epsilon, a), (\epsilon, ba), (\epsilon, bba)\}$ , yielding the recursive rule

$$\{(\epsilon, a), (\epsilon, ba), (\epsilon, bba)\} \rightarrow \{(\epsilon, a), (\epsilon, ba), (\epsilon, bba)\} \{(a, a), (a, ba), (ab, a)\}$$

This is a linguistically natural problem to have. Generally, in human language, if a substring is optional and repeatable in a context, then it is also optional and repeatable in “related” contexts. For example, in English, *very* is optional and repeatable in context

(*she's, funny*). It is also optional and repeatable in (*he's, funny*). In fact, when *funny* is to its right, it's almost always optional and repeatable. The result is that *very funny* shares all its contexts with *funny*, and, simplifying slightly, *funny* shares all its contexts with *very funny*.<sup>12</sup> This means that the learner would create a recursive rule something like:

$$\{(She's, \epsilon), (She's \text{ very}, \epsilon)\} \rightarrow \{(She's, funny)\} \{(She's, \epsilon), (She's \text{ very}, \epsilon)\}$$

However, if we find places where *funny* can occur but *very\** may not accompany it, then the contexts for *funny* will not be the same as those for *very funny*. Suppose, for example, we assume that in the compound *funny bone*, *funny* and *bone* are separate words. Since (13-a) is grammatical but (13-b) is ungrammatical, *funny* has context (*I hit my, bone*) but *very funny* does not.

- (13) a. I hit my funny bone  
 b. \*I hit my very funny bone

This fact yields non-recursive rules like:

$$\{(She's, \epsilon), (She's \text{ very}, \epsilon)\} \rightarrow \{(She's, funny)\} \{(She's, \epsilon), (She's \text{ very}, \epsilon), (I \text{ hit my}, \text{ bone})\}$$

A grammar with a rule like the above instead of the earlier recursive one cannot generate *She's very very very funny*.

This is not to say that a word playing two different roles – one accompanied by a repeating modifier and one not – are unlearnable by this learner. They do, however, require more input: enough repetitions to distinguish the repeating context from the non-(indefinitely)-repeating context. Theorem 5.8.4 shows that the input needed to hypothesise indefinite repetition is a sentence with the repeating phrase occurring more times than it occurs anywhere else.

**Notation** Contexts of a string drawn from a sample: Let  $S \subseteq L$  and let  $s \in S$ . Define

---

<sup>12</sup>Putting aside for now compounds like *funnybone* (*\*I hit my very funnybone!*)

$C_S(s) := \{(a, b) \in \text{Con}(S) \mid asb \in L\}$ . That is, given a sample of  $L$  and a sentence  $s$ ,  $C_S(s)$  is the contexts *from the sample* where  $s$  can occur. If the sample is clear from context, the subscript may be omitted.

**Theorem 5.8.4.** *Let  $u, y \in \Sigma^*$ .*

*Let  $c = \{(x, z) \mid xu^*yz \subseteq L\}$  be the contexts of  $u^*y$ .*

*Let  $d = \{(x, z) \mid \exists n \in \mathbb{N}[xu^n yz \in L]\}$  –  $c$  be the contexts where  $u$  (followed by  $y$ ) may occur and even repeat, but not indefinitely.*

*Let  $m = \max(\{n \in \mathbb{N} \mid xu^n yz \in L \ \& \ (x, z) \in d\})$  be the maximal number of times  $u$  (followed by  $y$ ) can repeat without it being able to repeat indefinitely in a context in  $d$ .*

*Then, for any  $(x, z) \in c$  and any  $n > m + 1$ , if  $xu^n yz$  occurs in a sample, then the learner will hypothesise a language  $L_i$  such that  $xu^{m+1+}yz \subseteq L_i$ , where  $u^{m+1+}$  is the set of all strings consisting of  $m+1$  or more  $u$ 's.*

*Proof.* Let  $xu^n yz$  be in the sample  $S$  for some  $n > m + 1$ .

We show that the learner will hypothesise a grammar that includes a rule:

$$C(u^{m+1}y) \rightarrow C(u) \ C(u^{m+1}y)$$

The learner will try all substrings including  $u^n y$  with all splits including  $(u, u^{n-1}y)$ . It tries to make a rule

$$C(u^n y) \rightarrow C(u) \ C(u^{n-1}y)$$

We need to show that  $C(u^n y) = C(u^{n-1}y) = C(u^{m+1}y)$ .

To find  $C(u^n y)$ , for all  $(a, b) \in \text{Con}(S)$ , we ask the oracle whether  $au^n yb \in L$ . If the oracle answers yes, this context is in  $C(u^n y)$ .

$$n > m \text{ so } (a, b) \notin d$$

$$\therefore (a, b) \in c$$

$$\therefore au^*yb \subseteq L$$

$$\therefore au^{m+1}yb \in L$$

$$\therefore (a, b) \in \text{Con}(u^{m+1}y)$$

$$\therefore \text{Con}(u^n y) \subseteq \text{Con}(u^{m+1}y)$$

Similarly,  $n > m + 1 \therefore n - 1 > m$  so by the same reasoning,  $\text{Con}(u^{n-1}y) \subseteq \text{Con}(u^{m+1}y)$ .

To see  $\text{Con}(u^{m+1}y) \subseteq \text{Con}(u^n y)$ :

$$m + 1 > m \text{ so } (a, b) \notin d$$

$$\therefore (a, b) \in c$$

$$\therefore au^*yb \subseteq L$$

$$\therefore au^n yb \in L$$

$$\therefore (a, b) \in \text{Con}(u^n y)$$

$$\therefore \text{Con}(u^{m+1}y) \subseteq \text{Con}(u^n y)$$

Similarly,  $\text{Con}(u^{m+1}y) \subseteq \text{Con}(u^{n-1}y)$ .

Therefore,  $\text{Con}(u^{m+1}y) = \text{Con}(u^{n-1}y) = \text{Con}(u^n y)$ .

Thus we have the rule:

$$C(u^{m+1}y) \rightarrow C(u) C(u^{m+1}y)$$

which is recursive and generates, possibly among other things,  $u^{m+1}y$  of category  $C(u^{m+1}y)$ .

Since the sample includes  $xu^n yz$ , we make rules

$$C(xu^n yz) \rightarrow C(x) C(u^n yz)$$

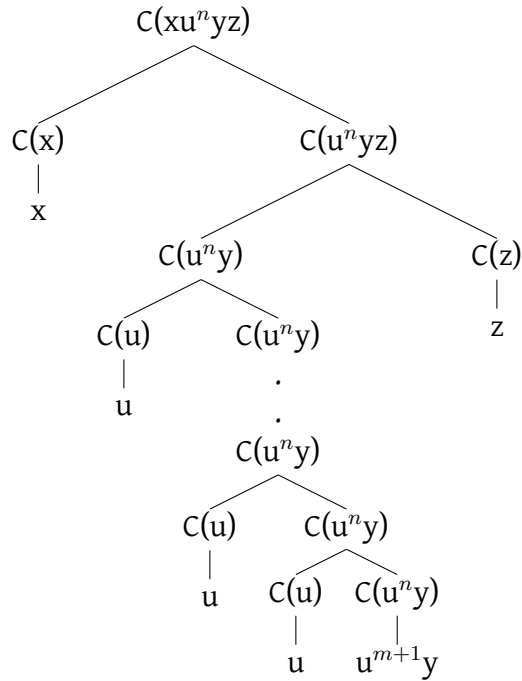
$$C(u^n yz) \rightarrow C(u^n y) C(z)$$

$$C(u^n y) \rightarrow C(u) C(u^{n-1} y)$$

However, we just showed that  $C(u^n) = C(u^{n-1})$ , so the third rule is equivalent to

$$C(u^n y) \rightarrow C(u) C(u^n y)$$

$xu^n y \in L$  so  $(\epsilon, \epsilon) \in C(xu^n yz)$ ; ie  $C(xu^n yz)$  is a start category. Since the categories are contexts, we know that for any string  $s$ ,  $s$  can occur in  $C(s)$ . We also showed that  $C(u^n) = C(u^{m>+1})$ , so  $u^{m+1}y$  can occur in the same context as  $u^n y$ . As such, we can generate  $xu^k yz$  for some  $k > m$ .



□

Depending on just what the contexts are where  $uy$  can occur without  $u$  repeating indefinitely, we may need more specific input to get rules for fewer than  $m + 1$   $u$ 's, and thus a grammar that generates  $xu^* yz$ .

Ultimately, what this means is that with this learner, if repeatable things in sub-contexts (like before  $y$ , wherever it is that  $u^*y$  can occur) are always repeatable in that context, then they are very easy to learn with a  $CFG_{F,K}$  learner. If, however, the repeating phrase isn't always a repeating phrase, it requires much more input for the learner to hypothesise a grammar that gets that indefinite repetition.

## 5.9 Conclusion

In summary, if babies are  $n$ -gram learners, they will learn repetition given a sample that includes an  $n$ -gram consisting only of the repeated matter, (plus  $n$ -grams that cover both sides of the context and all repetitions that can fit, if necessary.) If they are 0-reversible or substitutable context free learners, they will learn both repetition and optionality in a context given any two examples with  $k$  and  $k + 1$  occurrences, for any  $k$ . This means in particular that anything optional becomes repeatable, and anything repeatable becomes optional. If babies are  $PDFA_\mu$  learners, a representative sample will lead them to learn optionality and repetition if present, but there is no special relationship. If babies are  $CFL_{F,K}$  learners, we expect them to learn repetition and optionality from very limited data if the adjunct is always repeatable and optional, but to require much more evidence if it can also be required or non-repeating.

	n-gram	0-rev	sub CFL	$PDFA_\mu$	$CFL_{F,K}$
opt $\rightarrow$ rep	<b>X</b>	✓	✓	<b>X</b>	sometimes
rep $\rightarrow$ opt	<b>X</b>	✓	✓	<b>X</b>	sometimes
ac,abc,abbc $\rightarrow$ ab*c	sometimes	✓	✓	<b>X</b>	✓
HL-like	no	no	somewhat	no	closer

Table 5.5: Summary of learners

Optionality and repeatability are closely linked both conceptually – repetition is a form of optionality – and in natural language – adjuncts tend to be optional and repeatable. This chapter surveyed some formal learners and found all to be capable of learning

repetition and optionality, and for 0-reversible and substitutable CF learners, optionality and repetition in a context always co-occur.

None of these language classes suffice to describe human syntax. However, substitutable CF languages are not a bad place to start. A major reason to conclude that two phrases have the same category is if they are intersubstitutable. For example, here *the man they call Jayne* and *he* are of the same category and are intersubstitutable. The fact that they are intersubstitutable in the context ( $\epsilon$ , *stole money*) is usable as evidence that they are also intersubstitutable in the context ( $\epsilon$ , *shot Mal*).

- (14) a. **The man they call Jayne** stole money.  $\in$  English  
b. **He** stole money.  $\in$  English  
c. **The man they call Jayne** shot Mal.  $\in$  English  
d.  $\therefore$  **He** shot Mal.  $\in$  English

However, they are not always intersubstitutable, for example in the context (*They built of statue of*,  $\epsilon$ ).

- (15) a. They built of statue of **the man they call Jayne**.  
b. \*They built of statue of **he**.

Surely humans bring more to learning than intersubstitutability: they have access to context and meaning if nothing else. Substitutability is still a very useful tool, and human learners may well use it.

With  $CFL_{F,K}$  learners, intersubstitutability is not nearly so absolute, but is still a major factor in the grammar: it is just those phrases that the oracle can tell us occur in all the same contexts that are given the same category.

While the learners considered here are not yet sufficient for human language, they give us useful insight into how humans might learn adjuncts: The intersubstitutability

of phrases with and without adjuncts yields recursive rules that allow for indefinite repetition of adjuncts, and this intersubstitutability may well be part of the strategies that real babies employ in language acquisition.

### 5.9.1 A note on learners not analysed here

More recent work on learnability approaches closer to human-language like grammars; indeed some are even contenders for actual human language.

Clark (2010) define a learner for congruential context free languages using a Minimally Adequate Teacher (MAT).<sup>13</sup> A congruential grammar is one in which substrings with all the same contexts necessarily form a unique category. In congruential CFLs, repetition does not necessarily entail optionality. The sample run they present learns the following grammar:

- $S \rightarrow SS \mid AB \mid \epsilon$
- $A \rightarrow AS \mid SA \mid a$
- $B \rightarrow BS \mid SB \mid b$

This grammar generates a language with the subset  $a^+b$ , as demonstrated in figure 5.17. This means that  $a$  is repeatable in the context  $(\epsilon, b)$ . However,  $a$  is not optional in this context: there is no way to generate the sentence  $b$  using this grammar since the first daughter of  $S$  must be either  $A$  or  $S$ .  $A$  can give us  $a$  or another  $A$  or  $S$  as its left daughter. There is therefore no way to generate a sentence that starts with  $b$ .

---

<sup>13</sup>MAT learning is a weaker learning criterion than Gold learning, so there are language classes that are MAT-learnable but not Gold-learnable. A MAT learner can ask questions of a teacher, such as membership queries.



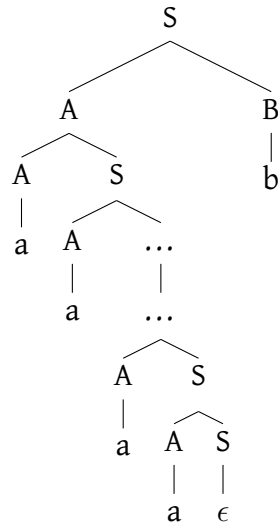


Figure 5.17: Generating  $a^+b$

Optionality also does not entail repetition in these languages. For example, the following grammar is clearly congruential, but generates the finite language  $\{b, ab\}$  with  $a$  optional but not repeatable in context  $(\epsilon, b)$ .

- $S \rightarrow A B$
- $A \rightarrow a \mid \epsilon$
- $B \rightarrow b$

Yoshinaka and Clark (2012) define an extension of Clark (2010) to congruential MCFGs. Since a CFG can be defined as an MCFG where the maximal tuple on the LHS is 1, the CFGs considered above belong to the class of congruential MCFGs and therefore also constitute counter-examples to the possibility of entailment between optionality and repetition.

Clark and Yoshinaka (2014) define an extension of the MAT learner described in Clark et al. (2010) (Section 5.8) to parallel multiple context free languages with the finite context and kernel properties. This class of languages is by far the best candidate for human

language we have seen, and may in fact be correct. The learner works in polynomial time but requires membership queries. I would predict that this learner treats repetition and optionality similarly to that of Clark (2010): no special relationship is derived between them, but repetition is readily predicted.

## CHAPTER 6

### Artificial language learning experiment

*This project was a group effort. Ed Stabler, Carson Schutze, and Martin Monti advised; Seth Ronquillo recorded stimuli; Shayna Lurya served as a research assistant, and Floris van Vugt provided extensive technical help with both the experimental interface and the statistical analysis.*

#### 6.1 Introduction

To get a picture of how people really learn adjuncts, I conducted an artificial language learning experiment. This is the first in a planned series of four to explore the first question we wanted to answer: how do people learn iterability of adjuncts?

Answering the question of learnability of adjuncts begins with the definition of adjuncts: what is an adjunct, and how would a learner recognise it as such? Chapter 2 covers the question of what an adjunct is, exploring such central properties as optionality, iterability, transparency, and ordering restrictions. Ordering restrictions are a property of non-adjuncts as well, and even optionality sometimes applies to non-adjuncts. Iterability, while not a property of all adjuncts, is at least arguably never a property of non-adjuncts.<sup>1</sup>

By iterability, or repetition, we mean that the element can occur unboundedly many times in a context. A context is just the strings before and after the repeating element.

---

<sup>1</sup>An exception is in some more poetic and story-telling language, such as “run run away” or “row row row your boat”.

If  $u$ ,  $v$ , and  $x$  are sequences of words from the lexicon, then we say  $x$  is repeatable, or iterable, in context  $(u, v)$  if  $uxv$ ,  $uxxv$ ,  $uxxxv$ ,  $uxxxxv$  etc are all sentences in the language.

Notice that although we know a lot about adjuncts from a syntactic analysis point of view, the descriptions of this property is entirely on the surface. It doesn't assume that  $u$  and  $v$  are constituents, or have any particular properties at all, but rather just that they are any sequence of words from the lexicon. This is deliberate: when an experimental participant encounters a sentence in a new language, we have no idea what kind of structure they might be assigning it. We must therefore ask experimental questions that assume nothing about the structure, and ask only about the string-level behaviour.

This experiment tests the ability of participants to learn word-level iterability. This means that the repetition in the grammar is repetition of words, not just categories (as in *red red rose* rather than *big red balloon*). Participants listen to sentences with an optional element repeated up to three times, and are tested on sentences with the element repeated up to five times. If participants accept sentences with more repetitions than they have ever heard before, we suggest they have learned a grammar with indefinite repetition.

It is not obvious that making such a generalisation is beneficial to learners. If people generalise directly from optionality to repetition – that is, if they are given evidence that a word can appear or not appear in a particular context, and from this they conclude that the word is repeatable – learners will run into trouble with many phenomena. For example in English, the complementiser *that* is usually optional, but it is not repeatable. Even some adjuncts have this problem: adverbs and PPs are optional but not iterable.

- (1) a. I think (that) River should flee.  
b. \*I think that that that River should flee.
- (2) a. Jayne sleeps (with all his guns).  
b. \*Jayne sleeps with all his guns with all his guns with all his guns.

- c. Kaylee (quickly) fixed Serenity.
- d. \*Kaylee quickly quickly fixed Serenity.

However, generalising even from minimal repetition – perhaps especially when the element is also optional – may be very beneficial. First, a learner of a language with unbounded repetition must somehow hypothesise that the repetition is unbounded even though she will never hear a sentence with infinitely many adjuncts. For whatever maximal level of repetition  $m$  she encounters, she needs to know that  $m + 1$  is also grammatical, or else she has not learned the language. Second, I am aware of very few possible mistakes she could make, making this generalisation. The only situation I have been able to discover in which repetition is possible but limited is with reduplication. However, I will assume that reduplication is a morphological process, not a syntactic one, so the learner may be safe here, as she will presumably be learning how to make words differently from how she is learning to make sentences.

Other instances of repetition that may come to mind are not repetition according to my definition. Repetition does not mean merely that a word (or phrase) occurs more than once in a row in a sentence. Rather, crucially, the repetition occurs *in a context*. Let us use the term *limited repetition* for cases where  $ux^k v \in L$  for some  $k > 1$ . For example, (3-a) is not repetition, but is rather limited repetition. If it were repetition proper, then (3-c) would also be grammatical.

- (3) a. I think that that that Simon foresaw did come to pass.<sup>2</sup>
- b. I think that that Simon foresaw did come to pass.
- c. \*I think that Simon foresaw did come to pass.
- d. \*I think Simon foresaw did come to pass.

---

<sup>2</sup>Meaning “I think that it is the case that the thing that Simon foresaw did come to pass”

Similarly, (4-c) is not repetition, or else sentences such as (4-d) and (4-e) would also be grammatical. If *oysters* is repeatable in the context (*oysters, eat eat eat*) then (4-d) is grammatical, and if *eat* is repeatable in context (*oysters oysters oysters, eat*), then (4-e) is grammatical.

- (4)
- a. Oysters eat
  - b. Oyster oyster eat eat
  - c. Oysters oysters oysters eat eat eat
  - d. \*Oysters oysters oysters oysters eat eat eat
  - e. \*Oysters oysters oysters eat eat eat eat

Therefore, we know that learners can run into problems if they assume that any evidence of repetition should be generalised. However, Example (3-d) shows that *that* is not optional in this context. I know of no situation in which a learner would run into trouble assuming repetition plus limited optionality implies unlimited repetition. To be on the safe side, the experiment also includes up to three, not just two, adjuncts in the training stimuli. This choice also precludes the possibility that the learner will treat Tagalog as a language like Chinese in which only two stacked adjectives are possible.

To give us something of a baseline, we also conducted a follow-up survey of English speakers and their acceptance of English word-level repetition.

The experiment described centres around intransitive sentences with adjectives optionally modifying the noun. We find that people are willing to generalise from limited to indefinite word repetition, but, surprisingly, surface facts about the language affect this generalisation. In particular, learners of a repeating-adjective language in which the adjective appears sentence-medially were much more willing to generalise repetition than learners of an adjective-final language.

In an artificial language learning experiment we try to partially replicate the circum-

stances under which babies learn language. The intention is that the learning tap into the unconscious – and possibly language-specific – learning mechanism enough to tell us something about real language learning. In ALL studies, evidence of learning tends to be subtle. Participants are thought to have learned a grammar when they perform better than chance; perfect or near-perfect scores are not required to indicate learning. See section 1.2.4 for some background on ALL research.

## 6.2 Design

The artificial language learning paradigm has two phases, a training phase and a testing phase.

**Training phase** Participants are exposed to grammatical items from the target language

**Testing phase** Participants are tested on (usually new) items to see what they learned.

Data like reaction time and grammaticality judgments are gathered to infer what the participants learned

This study is of the style Culbertson (2012) calls a *Poverty of the Stimulus*, in which participants are trained on a controlled subset of the data and the measure of interest is their willingness to generalise.

In many ALL studies (eg Reber (1967), Opitz and Friederici (2007), Rohrmeier et al. (2012)), participants have no access to meaning at all: they are not given any context or idea about what the sentences are about; they just read them or listen to a speaker read them. The present study is in this tradition.

In this study, participants listened to sentences of Tagalog which contained optional and repeating adjectives. There were two grammars, and participants were assigned to one or the other grammar. All stimuli conformed to the templates given below. Tagalog is verb-initial; all sentences in this experiment are intransitive, so there is only one DP

following the verb. To shift the linear position in the sentence in which the repetition takes place, some sentences have an adverb directly following the verb. This was done to reduce possible reliance on counting. Adj\* means 0 or more of the same adjective.

<p>Grammar 1: V (Adv) D Adj* N          Grammar 2: V (Adv) D N Adj*</p>
---

Figure 6.1: Grammatical sentence templates

Some example stimuli:

**Grammar 1 (Adj-N):**

- |     |    |   |                       |
|-----|----|---|-----------------------|
| (5) | a. | natulog ang babae<br>sleep D woman<br>‘The woman is sleeping/slept’   | No adjective          |
|     | b. | natulog ang malaki babae<br>sleep D big woman<br>‘The big woman is sleeping/slept’  | One adjective         |
|     | c. | natulog ang malaki malaki babae<br>sleep D big big woman<br>‘The big big woman is sleeping/slept’                                   | 2 adjectives          |
|     | d. | natulog ang malaki malaki malaki babae<br>sleep D big big big woman<br>‘The big big big woman is sleeping/slept’                    | 3 adjectives          |
|     | e. | natulog siguro ang babae<br>sleep maybe D woman<br>‘Maybe the woman is sleeping/slept’  | Adverb                |
|     | f. | natulog siguro ang malaki malaki malaki babae<br>sleep maybe D big big big woman<br>‘Maybe the big big big woman is sleeping/slept’ | 3 adjectives + adverb |

**Grammar 2 (N-Adj):**



- |     |  |                       |
|-----|--|-----------------------|
| (6) | a. natulog ang babae<br>sleep D woman<br>'The woman is sleeping/slept'   | No adjective          |
|     | b. natulog ang babae malaki<br>sleep D woman big<br>'The big woman is sleeping/slept'  | One adjective         |
|     | c. natulog ang babae malaki malaki<br>sleep D woman big big<br>'The big big woman is sleeping/slept'                                   | 2 adjectives          |
|     | d. natulog ang babae malaki malaki malaki<br>sleep D woman big big big<br>'The big big big woman is sleeping/slept'                    | 3 adjectives          |
|     | e. natulog siguro ang babae<br>sleep maybe D woman<br>'Maybe the woman is sleeping/slept'  | Adverb                |
|     | f. natulog siguro ang babae malaki malaki malaki<br>sleep maybe D woman big big big<br>'Maybe the big big big woman is sleeping/slept' | 3 adjectives + adverb |

The question to be answered is, in learning language, do people generalise from limited to indefinite repetition?

To probe this question we exposed people to sentences with a maximum of 3 adjectives, but the testing phase included sentences with 4 and 5 adjectives. People's acceptance of these so-called *generalised* stimuli suggests they are willing to generalise from limited to indefinite repetition.<sup>3</sup>

**Training stimulus design:** sentences with up to 3 of the same adjective  
**Testing stimulus design:** sentences with up to 5 of the same adjective plus ungrammatical stimuli

Figure 6.2: Design of Stimuli

<sup>3</sup>Of course, it could also mean that for some reason they conclude that up to five, or six, or eight-three, repetitions is grammatical but no more.

To answer the research question, we compare responses to ungrammatical and generalised stimuli, since both are unfamiliar classes of stimuli. If participants accept generalised stimuli more than ungrammatical stimuli, we infer that they've generalised repetition. Ideally, we would like to see them accept the generalised stimuli just as much as the familiar stimuli, but we expect a surface familiarity effect and a length effect will reduce the acceptance level of generalised stimuli somewhat.

### 6.2.1 Stimuli

Tagalog allows adjectives to appear on the left or the right of the noun. We created two grammars which are fragments of Tagalog, one of which has leftward adjectives (Grammar 1), and the other right (Grammar 2).

We generated the stimuli with an OCaml program for generating sentences with a finite state automaton. Although Tagalog is not finite state, these fragments of it are. The automata are in Figure C.1 in Appendix C.

The vocabulary is given in Table 6.1<sup>4</sup>. (The sentences were randomly generated rather than counter-balanced for sentence type.)

---

<sup>4</sup>*ang* marks the argument that the verb is voice-marked for. In these examples, the verb is in actor voice, so *ang* marks the actor (the sleeper/leaver). In real Tagalog, this is the only argument that can be extracted and placed before the verb.

#	category	words
2	V	natulog 'sleep' umalis 'leave'
1	Adv	siguro 'maybe'
1	D	ang
4	Adj	malaki 'big' masaya 'happy' matanda 'old' mapula 'red'
4	N	babae 'woman' pusa 'cat' susi 'key' kotse 'car'

Table 6.1: Vocabulary

Training stimuli were 180 grammatical sentences with no more than 3 adjectives. The adjective repetition was always word-level repetition: if more than one adjective occurred, it was always the same adjective. All grammatical sentences conform to the templates in figure 6.1. Table 6.2 gives the distribution of sentence types in the training stimuli.

	no adv	adv
no adj	24	33
1 adj	34	46
2 adj	11	15
3 adj	8	9

Table 6.2: Training stimuli

The stimuli were recorded by a Tagalog native speaker, Seth Ronquillo, in a sound booth at UCLA. Examples can be heard at <http://meaghanfowlie.com/research/tagalog/stimuli>, and a full stimulus list is in appendix D.1.

### 6.2.2 Testing stimuli

Testing stimuli were also generated with a finite state automaton.

There were three basic kinds of testing stimuli. **Familiar** stimuli were grammatical sentences generated by the same finite state machine as the training sentences, most of which were not in the training session. See section 6.3.2 below for a discussion of those that were. **Generalised** stimuli are also grammatical, but they have 4 or 5 instances of an adjective, while the participant had only ever heard up to 3 before. **Ungrammatical** stimuli were of two varieties. **Scrambled** stimuli were created by randomising the words in a grammatical sentence. **Noun-repetition** stimuli had one adjective and a repeating noun (2 to 5 times) instead of a repeating adjective.

For recording the ungrammatical stimuli, we went through the ungrammatical sentences with the speaker to see if any were grammatical in real Tagalog. We found that 6/20 scrambled items were grammatical. For those six, the speaker recorded them naturally as grammatical sentences. For the remaining scrambled items, we simply asked him to say them as naturally as possible, and if there were any he just found too difficult we would remove them from the list. None were rejected. Upon listening to the scrambled stimuli, we found that they sounded quite natural, and we couldn't hear the difference between the real Tagalog sentences and the truly ungrammatical. Moreover, in the exit survey none of our participants commented on unnatural prosody in some sentences.

For the noun-repetition stimuli, we instructed the speaker to imagine the nouns were adjectives and the adjectives nouns. Since Tagalog allows nouns and adjectives in either order, this worked well. We recorded them two ways. In Tagalog, a *linker* /-ŋ/ occurs between adjectives and between the adjective and the noun. If Tagalog were to repeat a noun as if it were an adjective, there would be a linker. The linker is a suffix, so it does not appear on the final word in a sentence. We hypothesised that the linker would make the recording of the sentences sound more natural – and indeed, that was our impression of

the recordings – but also that the linker might make the nouns sound like different words entirely; for example *susi'* vs *susing*.

In Grammar 1 (Adj-N), adjectives were always pronounced with a linker, and in Grammar 2 (N-Adj), adjectives were pronounced with the linker except the last one.

- (7) a. Natulog ang matanda-ng matanda-ng pusa' **G1**  
 sleep D old-LK old-LK cat
- b. Natulog ang pusa' matanda-ng matanda **G2**  
 sleep D cat old-LK old

In the ungrammatical noun-repetition stimuli, it was Grammar 1 that had a repeating item with no linker:

- (8) a. \*Natulog ang matanda-ng pusa-ng pusa' **G1**  
 sleep D old-LK cat-LK cat
- b. \*Natulog ang pusa-ng pusa-ng matanda **G2**  
 sleep D cat-LK cat-LK old

As such, we recorded the noun-repetition stimuli both ways and included all in the testing stimuli. No significant differences were found in acceptance rates; see the results section for details.

The testing stimuli were distributed as follows:

#	Type	Description
62	Familiar	no more than 3 adjectives
38	Generalised	
20	(4)	4 adjectives
18	(5)	5 adjectives
78	Ungrammatical	
58	Noun repeated	Noun repeated instead of the adjective
20	Scrambled	Randomised grammatical sentences

Table 6.3: Testing stimuli

In future studies we intend to add a preliminary study in which untrained participants guess the grammaticality of the testing stimuli.

### **6.2.3 Procedure**

After giving informed consent, participants were shown the computer and given the following instructions:

This experiment has two parts. In the first part you'll be listening to a native speaker of Tagalog say Tagalog sentences. At the same time, a red dot will appear on either the left or the right of the screen. If it's on the left, push the left arrow, and the right arrow if it's on the right. If you see a question mark appear on the screen that just means it's waiting for your answer.

In the second part you'll be listening to new sentences, most of which you've never heard before, but this time, some of them are real and some are fake. Your job is just to guess which are real. You'll use the arrow keys for that too. It'll show on the screen which is for yes and which is for no. If you don't know, don't worry, just go with your gut!

Do you have any questions?

When you're done, or if you have any questions during, I'll be [right out here].

#### **6.2.3.1 Training Phase**

Participants listened to randomised training stimuli over headphones in an experiment room at UCLA. Because the repetition is so salient, we wanted to reduce the chance that participants would do something extralinguistic like counting. As such, we gave them a small distractor task. While they listened they saw a red dot on the screen, on either the right or left. They respond by pressing the right arrow key if it was on the right, and the

left arrow key if it was on the left. The dot disappeared when the participant pressed the button, or after 500ms, after which a question mark would appear until the participant responded. There was one distractor task per sentence, and participants had to respond before the next sentence would play. The distractor started a randomised time after the start of the stimulus soundfile, between 200ms and 800ms.

Sentences came in batches of 20, and participants could break as long as they liked between batches.

### **6.2.3.2 Testing Phase**

After the training ended, instructions appeared on the screen for the testing phase. Participants could break here for as long as they liked.

Participants listened to the testing stimuli over headphones in the same room. For each testing stimulus, they responded with a keypress whether they thought it was a “real” (left arrow) or “fake” (right arrow) sentence of Tagalog. This was a forced choice task: if they didn’t know, they were instructed to “go with their gut.”

Sentences came in batches of 10, between which participants could break for as long as they liked.

After the experiment was over, participants were given an optional survey with basic demographic and language background data and the question “Did you consciously notice any patterns in the Tagalog, or make up any rules for yourself in how you answered the questions in the second part? What were they?”

The whole session took about 45 minutes.

### **6.2.4 Participants**

51 UCLA undergraduates participated. They were given course credit for their time. 29 learned Grammar 1 and 22 learned Grammar 2.

### 6.2.5 Analysis

For each participant, we computed the total number of sentences rated as acceptable in each of the categories (Familiar, Generalised, Ungrammatical).

We proceeded to discard participants who showed insufficient learning. This was done as follows. For each participant, we calculated the (arithmetic) difference between the participant's mean acceptance of familiar items (out of 1) and her mean acceptance of ungrammatical items. If this difference was smaller than 0.15 then we discarded the participant from further analysis. This was the case for 6 participants (12% of our sample). We chose 0.15 fairly arbitrarily; impressionistically there was a cluster division there. However, we also ran the analysis with the cut-off at 0.05, 0.10, and 0.20, and the results were similar.

We ran a 2-way analysis of variance (ANOVA) with within-subjects factor *category* (Familiar, Generalised, Ungrammatical) and between-subjects factor *group* (Grammar 1, Grammar 2), and random variable *participant*. The dependent variable was number of accepted sentences out of the total number of sentences in the category. Since each sentence was rated either as acceptable or not, we used a binomial response model with the logit link function<sup>5</sup>.

To see if participants favoured ungrammatical stimuli with or without linkers, we ran a 2-way ANOVA with within-subjects factor *subtype* (linker, no linker) and between-subjects factor *group* (Grammar 1, Grammar 2). The dependent variable was number of accepted sentences out of the total number of sentences in the subtype.

---

<sup>5</sup>We also ran a traditional ANOVA with Gaussian response assumptions instead of the binomial response, and these two analyses agreed on all the critical comparisons for our purposes.



### 6.3 Results

We found that the main effect of category was significant ( $\chi^2(2) = 1429.89, p \ll 0.0001$ ), meaning that acceptance rates of least two of the three categories were different from each other. The main effect of group was also significant ( $\chi^2(1) = 6.37, p < 0.05$ ), so one group had a higher overall acceptance rate. The interaction between category and group was significant ( $\chi^2(2) = 32.96, p \ll 0.0001$ ), so the differences between the acceptance rates of different stimulus categories varied depending on whether participants were trained on the adjective-medial grammar (group 1) or the adjective-final grammar (group 2).

Since both main effects and the interaction were significant, we ran Tukey contrasts with correction for multiple comparisons of each of the three contrasts for the two groups separately. Participants learning grammar 1 (adjective-medial) accepted familiar items ( $M = 0.777, SD = 0.134$ ) more than ungrammatical ( $M = 0.262, SD = 0.167$ ) ( $z = -28.87, p \ll 0.0001$ ), meaning participants learned the grammar. Familiar stimuli were also much more acceptable than generalised ( $M = 0.412, SD = 0.318$ ) ( $z = -18.32, p \ll 0.0001$ ), which means that generalised stimuli were not simply accepted as though they were just like the training stimuli. Generalised items were accepted more than ungrammatical items ( $z = 8.305, p \ll 0.0001$ ), which we take to mean that participants in this group did generalise from limited to indefinite repetition.

Participants learning grammar 2 (adjective-final) accepted familiar items ( $M = 0.865, SD = 0.103$ ) more than ungrammatical ( $M = 0.393, SD = 0.152$ ) ( $z = -23.86, p \ll 0.0001$ ), meaning participants learned the grammar. Familiar stimuli were also much more acceptable than generalised ( $M = 0.401, SD = 0.234$ ) ( $z = -20.852, p \ll 0.0001$ ), which means that generalised stimuli were not simply accepted as though they were just like the training stimuli. However, in this group, generalised items were *not* accepted more than ungrammatical items ( $z = 0.409, p > 0.1$ ), leaving us with no evidence that partic-

ipants trained on the adjective-final grammar generalised repetition.

Figure 6.3 summarises our results.<sup>6</sup> Acceptance rates for each category (Familiar, Generalised, Ungrammatical) are divided by group (grammar 1 (adjective-medial), grammar 2 (adjective-final)). The bars in the boxes indicate mean acceptance rates. Stars indicate significance, and are colour-coded by grammar, with the blue markers at the top for grammar 2, and the red ones at the bottom for grammar 1. We can see that acceptance of familiar stimuli was very high, and acceptance of ungrammatical fairly low (indicating learning). Acceptance of generalised stimuli is highly variable, but falls in between familiar and ungrammatical for grammar 1. Grammar 2 learners gave almost the same mean ratings to all unfamiliar (ungrammatical and generalised) stimuli.

---

<sup>6</sup>This is Tukey-style [box-and-whisker plot](#). Boxes cover the central 50% of the participants. The length of the box is the Inter-Quartile Range, or IQR. Whiskers extend to the most distal data points within  $1.5 \times$  IQR; dots are outliers.

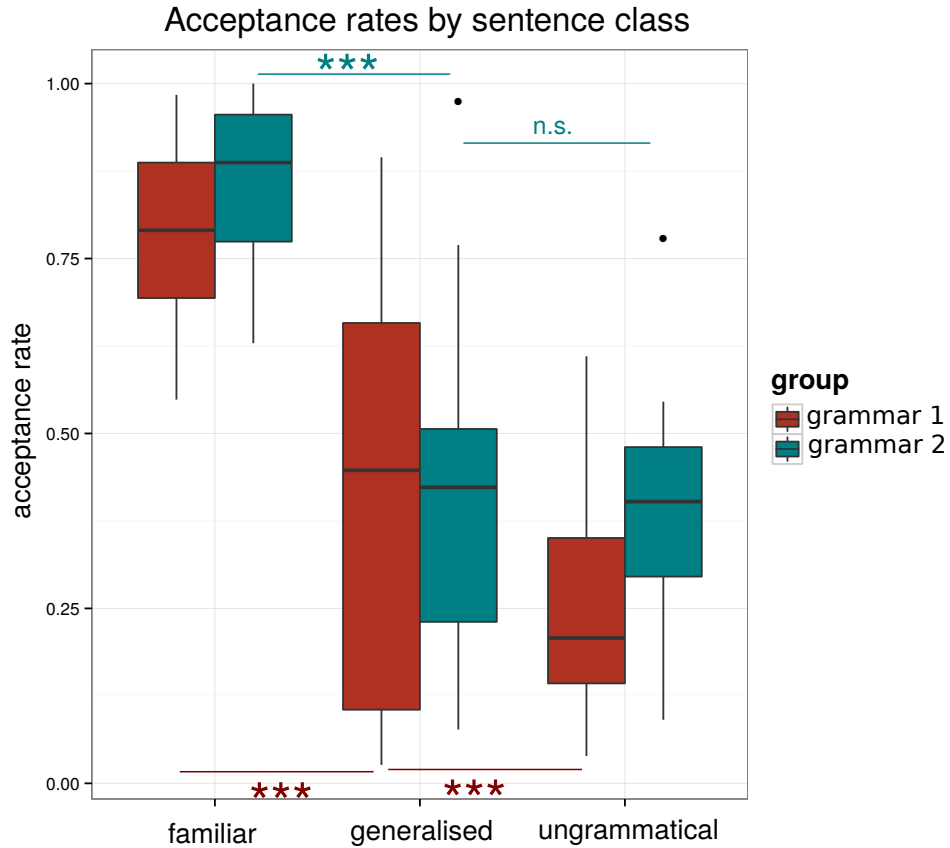


Figure 6.3: Results by category and group

We found no evidence for a preference for ungrammatical stimuli with or without linkers. There was no significant main effect of group ( $F(1, 49) = 2.06, p > 0.1$ ) or subtype ( $F(1, 49) = 3.13, p > 0.05$ ), and no interaction between the two ( $F(1.49) = 0.68, p > 0.4$ ).

### 6.3.1 Ungrammatical subtypes

Our design included two kinds of ungrammatical stimuli: scrambled and noun-repetition. Scrambled stimuli were created by randomising the words of grammatical stimuli, while noun-repetition stimuli were created by taking grammatical sentences with  $n$  adjectives

and one noun and changing them so that they had one adjective and  $n$  nouns, in their usual positions. The scrambled stimuli we expected would be roundly rejected, as they violated the grammar badly and contained many unfamiliar  $n$ -grams as well. The noun-repetition stimuli were included because learners might have learned the basic word-order as the grammar, with repetition as a general rule on top of the grammar. If participants learned in this way, we would expect them to accept noun-repetition stimuli at a similar rate to grammatical stimuli.

Arguably, only the noun-repetition stimuli are important. We expected these to be harder for the learners to correctly recognise as ungrammatical, as everything but the repeated element was correct. Only if participants learned repetition as an integral part of the grammar, rather than an extra-grammatical rule “repetition is allowed” would we expect them to correctly reject noun-repetition stimuli.

To see whether participants accepted ungrammatical subtypes at different rates, we ran a 2-way analysis of variance (ANOVA) with within-subjects factor *type* (scrambled, noun-repetition) and, since group had been important in the main analysis, a between-subjects factor of *group* (grammar 1, grammar 2). The dependent variable was number of accepted sentences out of the total number of sentences in the type.

We found a main effect of *type* ( $F(1, 49) = 8.42, p < 0.01$ ), but not of *group* ( $F(1, 49) = 3.16, p > 0.05$ ), and no interaction ( $F(1, 49) = 0.01, p > 0.9$ ). This means the two groups accepted ungrammatical stimuli at about the same rate, but one ungrammatical type was preferred over the other. Unexpectedly, participants preferred scrambled stimuli ( $M = 0.43, SD = 0.24$ ) to noun-repetition stimuli ( $M = 0.27, SD = 0.20$ ).

Given the arguments that noun-repetition stimuli are the only relevant ungrammatical stimuli, we planned a second analysis just like our main analysis but using only noun-repetition stimuli as ungrammatical stimuli. Given our discovery that noun-repetition stimuli were overall actually more accepted than scrambled stimuli, this second analysis loses some of its conviction: rather than compare familiar and generalised stimuli

with the most difficult ungrammatical stimuli, instead we compare them with the easiest. However, we include this analysis anyway, addressing the possibility that indeed only noun-repetition stimuli are important ungrammatical stimuli.

We ran a 2-way analysis of variance (ANOVA) with within-subjects factor *category* (Familiar, Generalised, (Noun-repetition) ungrammatical) and between-subjects factor *group* (Grammar 1, Grammar 2), and random variable *participant*. The dependent variable was number of accepted sentences out of the total number of sentences in the category. Since each sentence was rated either as acceptable or not, we used a binomial response model with the logit link function.

We found that the main effect of category was significant ( $\chi^2(2) = 1299.27, p \ll 0.0001$ ), meaning that acceptance rates of least two of the three categories were different from each other. The main effect of group was also significant ( $\chi^2(1) = 5.22, p < 0.05$ ), so one group had a higher overall acceptance rate. The interaction between category and group was significant ( $\chi^2(2) = 25.11, p < 0.00001$ ), so the differences between the acceptance rates of different stimulus categories varied depending on whether participants were trained on the adjective-medial grammar (group 1) or the adjective-final grammar (group 2).

Since the interaction between group and category was significant we ran Tukey contrasts with correction for multiple comparisons of each of the three contrasts for the two groups separately. All contrasts were found to be significant.

In group 1, participants accepted familiar items ( $M = 0.77, SD = 0.14$ ) more than ungrammatical ( $M = 0.20, SD = 0.16$ ) ( $z = -26.86, p < 0.0001$ ), meaning participants learned the grammar. Familiar stimuli were also much more acceptable than generalised ( $M = 0.40, SD = 0.31$ ) ( $z = -17.21, p < 0.0001$ ), which means that generalised stimuli were not simply accepted as though they were just like the training stimuli. Generalised items were accepted more than ungrammatical items ( $z = 9.75, p < 0.0001$ ), which we take to mean that participants did generalise from limited to indefinite repetition.

In group 2, participants accepted familiar items ( $M = 0.0.86$ ,  $SD = 0.10$ ) more than ungrammatical ( $M = 0.31$ ,  $SD = 0.13$ ) ( $z = -22.78$ ,  $p < 0.0001$ ), meaning participants learned the grammar. Familiar stimuli were also much more acceptable than generalised ( $M = 0.38$ ,  $SD = 0.19$ ) ( $z = -19.02$ ,  $p < 0.0001$ ), which means that generalised stimuli were not simply accepted as though they were just like the training stimuli. Generalised items were accepted more than ungrammatical items ( $z = 2.77$ ,  $p < 0.05$ ), which we take to mean that participants did generalise from limited to indefinite repetition.

Figure 6.4 summarises our results.<sup>7</sup> Acceptance rates for each category (Familiar, Generalised, Ungrammatical) are divided by group (grammar 1 (adjective-medial), grammar 2 (adjective-final)). The bars in the boxes indicate mean acceptance rates. Stars indicate significance, and are colour-coded by grammar, with the blue markers for grammar 2, and the red ones for grammar 1. We can see that acceptance of familiar stimuli was very high, and acceptance of ungrammatical fairly low (indicating learning). Acceptance of generalised stimuli is highly variable, but falls in between familiar and ungrammatical. The spread of the middle 50% of the data for grammar 2 generalised and ungrammatical has significant overlap, but the mean acceptance rate of ungrammatical is significantly lower, unlike the results when all ungrammatical stimuli are considered.

---

<sup>7</sup>This is Tukey-style [box-and-whisker plot](#). Boxes cover the central 50% of the data. The length of the box is the Inter-Quartile Range, or IQR. Whiskers extend to the most distal data points within  $1.5 \times$  IQR; dots are outliers.

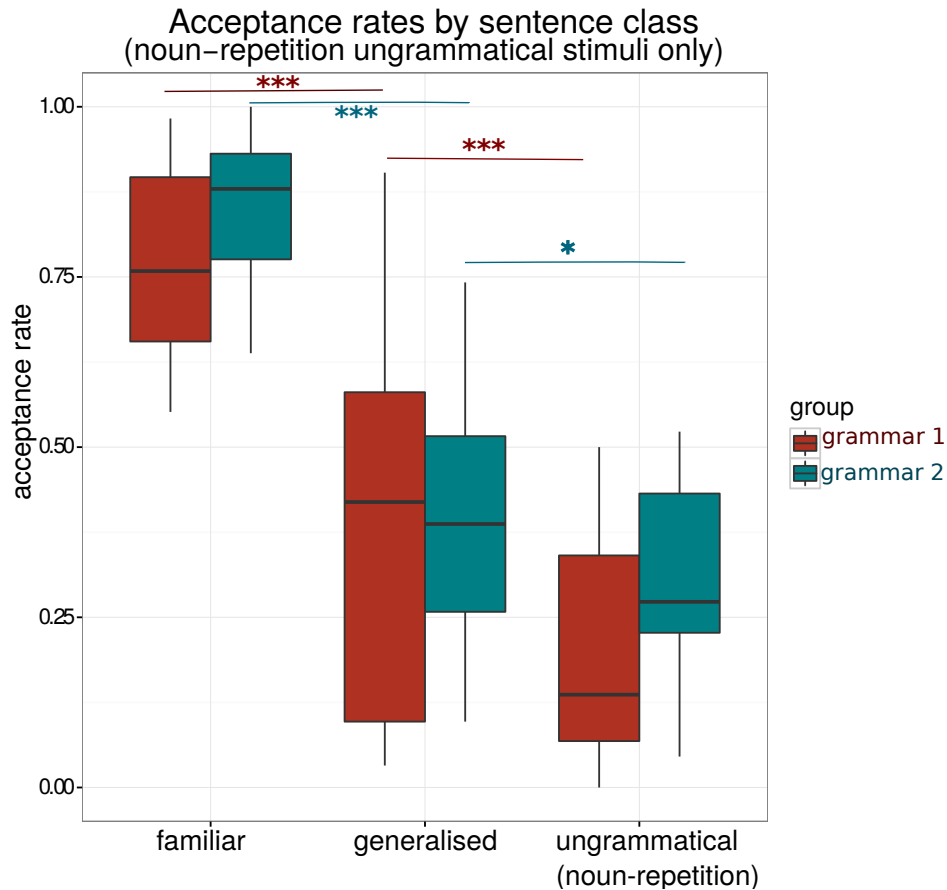


Figure 6.4: Results by category and group

### 6.3.2 Duplicate items

In designing the stimuli I erred: the intention was that all but the shortest testing stimuli would be new to the participant. This was not the case: 33 of 89 test stimuli were also in the training stimuli. Of these, 22 contained adjectives. All testing stimuli with no adjectives were also present in the training stimuli; this was intentional as there are very few such stimuli.

Because of the unintended presence of training stimuli in the testing stimuli, an alternative analysis presents itself: perhaps participants merely memorised the sentences

they heard in the training session, and most or all of their high acceptance rates of the familiar grammatical stimuli was driven by the duplicate stimuli from the training session.

To test the hypothesis that participants prefer sentences they've heard before (*duplicates*) to those they have not (*non-duplicates*), we ran a 2-way ANOVA with within-subjects factor *duplicate* (duplicate, non-duplicate). The dependent variable was number of accepted sentences out of the total number of familiar sentences. However, knowing that many of the duplicate stimuli were very short, lacking adjectives, and might therefore be preferred in general, we also ran a 2-way ANOVA with within-subjects factors *duplicate* (duplicate, non-duplicate) and *adjectives* (1,2,3). The dependent variable was number of accepted sentences out of the total number of familiar sentences containing adjectives. (Those without were excluded from the latter analysis as they had no non-duplicates to be compared with.)

In the simpler model, disregarding number of adjectives, we found a significant main effect of duplicate ( $F(1, 50) = 7.73, p < 0.01$ ). Participants preferred duplicate items ( $M = 0.819, SD = 0.144$ ) to non-duplicates ( $M = 0.767, SD = 0.157$ ). However, once number of adjectives was taken into account, no difference was found between duplicates and non-duplicates. There was a main effect of adjectives ( $F(2, 100) = 12.84, p < 0.0001, p_{GG} < 0.0001$  with sphericity correction) but no effect of duplicate ( $F(1, 50) = 0.326, p > 0.5$ ), and no interaction ( $F(2, 100) = 2.34, p > 0.1, p_{GG} > 0.1$  with sphericity correction). Figure 6.5 summarises the results.



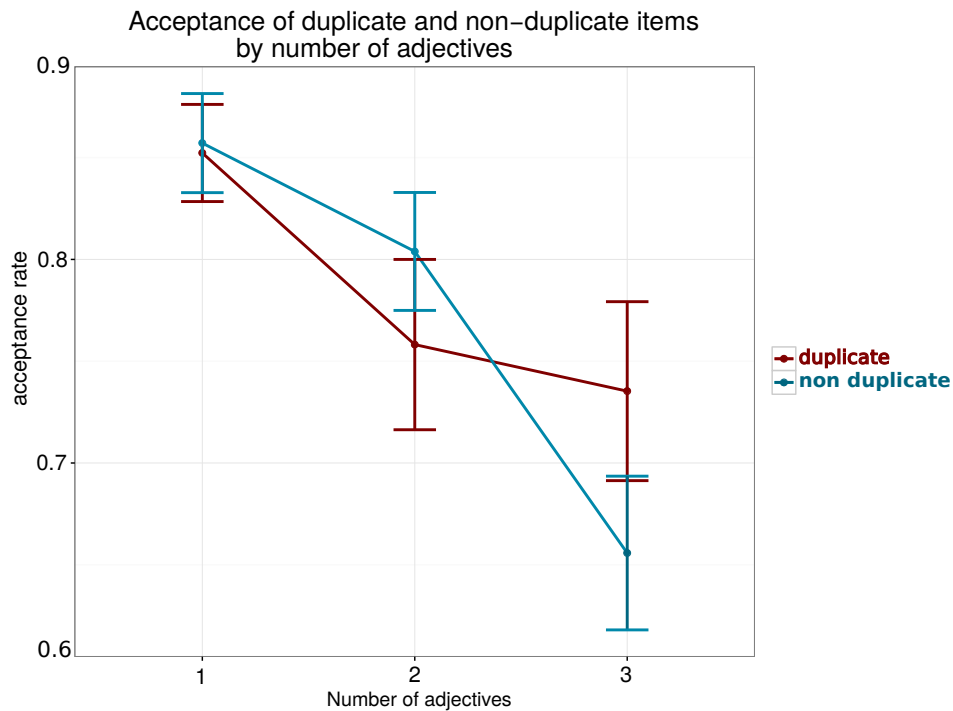


Figure 6.5: Testing stimuli repeated from the training phase

## 6.4 Discussion

The two major findings are (1) people do not completely generalise repetition – if they did, we would expect them to treat the familiar and generalised stimuli equally – and (2) people accept generalised repetition as more grammatical than the ungrammatical stimuli: one of the two groups preferred generalised stimuli to outright ungrammatical, and both groups preferred generalised stimuli to ungrammatical stimuli with the wrong element repeated.

On the other hand, the preference for generalised over ungrammatical stimuli could also be a reflection less on the acceptance of generalised stimuli and more of a reflection of how terrible ungrammatical stimuli were. This remains a possibility; our efforts to avoid this included using the same vocabulary as grammatical stimuli, and either the

same distribution of words for the scrambled stimuli or essentially the same word order, in the case of the noun-repetition stimuli. We also were able to use a number of “ungrammatical” sentences which, while ungrammatical in our mini-Tagalog, were grammatical, but with a different meaning, in real Tagalog. This ensured that for these sentences at least, the native speaker recording the sentences gave them a completely natural intonation. Sentences that were not grammatical in real Tagalog we could be less sure of, but impressionistically we found the recordings very natural-sounding, and no participants made comments that we were able to interpret as meaning that some of the sentences had unnatural prosody. Sentence length of ungrammatical sentences was kept the same as those of grammatical stimuli. Finally, comments from participants did not support this alternative hypothesis. People who accepted generalised stimuli and commented on it did not say things like “Some of the sentences had too many of the same word in a row, but at least they didn’t sound totally mixed up!” Though we cannot be certain, we are reasonably confident that the preference for generalised stimuli over ungrammatical is a reflection of their perceived grammaticality, not the dire ungrammaticality of ungrammatical stimuli.

#### **6.4.1 Scrambled stimuli**

We were surprised to find that Group 2’s failure to generalise was primarily a reflection of their high acceptance of scrambled stimuli. They rejected ungrammatical stimuli with noun-repetition similarly to Group 1, and accepted generalised stimuli similarly as well. Scrambled stimuli were the same for both groups; perhaps the scrambled stimuli were in a surface sense less scrambled relative to grammar 2 than relative to grammar 1.<sup>8</sup> To test this we assigned each scrambled sentence a pair of difference measures, one for each grammar. The difference measure was calculated by summing the distance between the position of each word in the scrambled sentence and the position it would have in the

---

<sup>8</sup>Suggested by Carson Schutze, pc 2016

unscrambled sentence.

For example, scrambled item 5, *matanda umalis susi' ang*, in both grammars, the verb *umalis* is one position away from its proper position and the determiner *ang* is two away. The adjective *matanda* is 2 away from its grammar 1 position and 3 away from its grammar 2 position. The noun *susi'* is 1 away in grammar 1 and in its proper position in grammar 2. In this case, the scrambled sentence is measured as equally different from its grammatical partner in both grammars.

Scrambled:	matanda	umalis	susi'	ang
G1 grammatical:	umalis	ang	matanda	susi'
Total: 6	1	2	2	1
G2 grammatical:	umalis	ang	susi'	matanda
Total: 6	1	2	0	3

We ran an independent-samples t-test to compare the difference measure in grammar 1 and grammar 2. There was a marginal difference in the difference measure for grammar 1 ( $M=7.68$ ,  $SD=4.07$ ) and grammar 2 ( $M=9.00$ ,  $SD=5.49$ );  $t(1)=2.00$ ,  $p = 0.06$ .

#### 6.4.1.1 Discussion of scrambled stimuli

The hypothesis tested here was that scrambled stimuli were more similar to grammar 2 grammatical items than to grammar 1 grammatical items, leading Group 2 participants to accept scrambled stimuli at a higher rate. The results do not support this hypothesis; indeed if anything, scrambled stimuli were *more* different from grammatical grammar 2 stimuli.

#### 6.4.2 Surface effects

The formal learning models discussed in chapter 5 all predict generalisation, based on the input participants were given.

We know that humans are not learners of those exact types – since those learners lack the ability to learn human-language-level syntax – but general properties of human language suggest that successful learners of human-like languages may still predict similar behaviour to some of the more successful learners considered. In particular, intersubstitutability as a criterion for generalisation can easily lead to generalisation of repetition: essentially, a phrase is treated the same way whether additional instances of the repeating item are present or not, in terms of substitution. As such, although humans are not (only) learners of any of the types explored in chapter 5, they may well be learners of a type that would also predict generalisation of repetition.

Supposing this were the case, what might explain participants' reluctance to accept 4 and 5 adjectives in a row at the same rate they accept 2 and 3? We suggest there are extra-linguistic factors at play. To begin with, there is the basic familiarity. Participants should in general be expected to be more accepting of more familiar-sounding stimuli, as they bring to bear on the task not only their language-learning capacity but also their general powers of observation. Second, there is length: only 9/180 of the training stimuli and 2/62 of the familiar testing stimuli were as long as the 15 shortest generalised stimuli at 7 words, and none were as long as the 16 longest of them at 8 words. Finally, there is the high salience and strangeness of the repeating word. A word being repeated, even if grammatical, arguably sounds strange quite quickly even to native speakers, and its presence is consciously very obvious.

In support of the third factor, we have comments from the exit survey. For example, one participant, who accepted only 12% of generalised stimuli, wrote *I remembered specific words that were repeated in the first portion. If they were repeated too many times, I thought it sounded grammatically incorrect.* Many participants commented on the number of repetitions, and said they rejected sentences with more than 3, or “too many”.

To test the hypothesis that even grammatical repetition of words is in some sense “strange”, we conducted a web survey of English sentences.

## 6.5 English survey

We conducted an online survey of English speakers to see if acceptance of sentences goes down as repetition increases, even in English, where indefinite repetition of some adjectives and intensifiers is grammatical. [Survey Monkey](#) participants answered 30 questions about English sentences. Survey Monkey is an online survey tool in which participation is by unpaid anonymous volunteers. The target sentences were grammatical sentences with repeating words, either adjectives or intensifiers (*really, so, ever*) such as *What a stupid stupid stupid idea!* and *I really really really like her*.

For each sentence, participants were asked to rate the grammaticality with one of the following 5 choices:

1. Doesn't sound like English
2. Sounds pretty weird
3. Sounds a bit weird
4. I wouldn't say it, but it does sound like English
5. Definitely English

### 6.5.0.1 Stimuli

There were 30 sentences, 19 of which were target sentences. Target sentences were all grammatical English sentences with either a repeating adjective or a repeating intensifier (*so, really, ever*). They were distributed as in Table 6.4. “number of adjuncts” is the number of times the adjunct occurred. For example, the sentence with 1 intensifier was *I really like her*.

Fillers included 2 non-English sentences (one French, one Q'anjob'al), 2 very ungrammatical strings of English words, 2 violations of the Cinque hierarchy, and 5 grammatical

Number of adjuncts	1	2	3	4	5	6
Adjective	0	4	2	3	2	0
Intensifier	1	1	1	2	1	2

Table 6.4: English survey stimuli

sentences without repeating words.

A full stimulus list is in Appendix D.3.

### 6.5.0.2 Participants

109 people responded to at least part of the survey. Of those, 88 completed the survey and of them, 85 reported their English proficiency as “*I’m a native speaker (I’ve been speaking English my whole life or close, and I don’t feel like I have an accent or anything)*”. The results from participants who did not meet these criteria were not included in the analysis. Two of the remaining participants gave perfect scores of *Definitely English* to all sentences, including ungrammatical ones, that contained only English words. These results were also discarded, leaving us with 83 participants, or 76% of the total.

### 6.5.0.3 Results

We divided the target sentences into types of repetition. Since we hypothesised that intensifier repetition may be more acceptable than adjective repetition, we divided the sentences into two sets, adjective-repetition and intensifier-repetition. Within those two sets we divided sentences into subgroups according to how many adjectives or intensifiers occurred. Adjectives had 2-5 occurrences, and intensifiers had 1-6 (see Table 6.4).

First, combining all responses for all subjects, we looked for a correlation between the number of repetitions and the grammaticality judgement. We used Kendall’s rank correlation, rather than Spearman’s, since the response values were non-parametric (it’s not clear if *Definitely not English* and *Sounds pretty weird* are the same distance apart in ac-

ceptability as *Sounds pretty weird* and *Sounds a bit weird*, for example.) We found a negative correlation (Kendall  $\tau = -0.2756566$ ,  $p \ll 0.0001$ ) between the variables, meaning the more repetition there was, the lower participants rated the sentences as English.

This correlation held true when we looked at adjectives (Kendall  $\tau = -0.6881024$ ,  $p \ll 0.0001$ ) and intensifiers (Kendall  $\tau = -0.396297$ ,  $p \ll 0.0001$ ) separately.

Figures 6.6 and 6.7 show the correlation between amount of repetition and grammaticality score. For each sentence category, we calculated the proportion of responses given to each acceptability level, represented both in the size of the dot and the number within it. Notice that the dots on the top row get smaller as you move from left to right; this means the more adjuncts there were, the fewer people gave it a perfect grammaticality rating.

These correlations were measured with all participants' results analysed together, with no regard for the fact that in actuality, some of the responses came from the same person and some did not. Ignoring this clustering of our data may lead to erroneous conclusions because the overall correlation found could have resulted from between-subject differences in rating behaviour rather than within-subject rating correlations. Therefore, we proceeded to check that this correlation held true for individual participants as well. For each participant and type of adjunct (adjective or intensifier) we computed the Kendall rank correlation ( $\tau$ ) using the Woodbury tie-breaking method,<sup>9</sup> as implemented in the `rankor` function from the `pvrnk` R package. (Note that we discarded the  $p$  values produced in this analyses because they are not valid due to multiple comparisons.) If the correlation between amount of repetition and acceptance did not hold on the individual level, the Kendall  $\tau$  values should show a zero-mean distribution.

Shapiro-Wilk normality tests showed that neither distribution was normal, neither adjectives ( $W = 0.9518$ ,  $p < 0.01$ ) nor intensifiers ( $W = 0.9064$ ,  $p \ll 0.00001$ ), so we

---

<sup>9</sup>We didn't use Woodbury for the whole group correlations above, as Woodbury only works for  $N < 60$ .

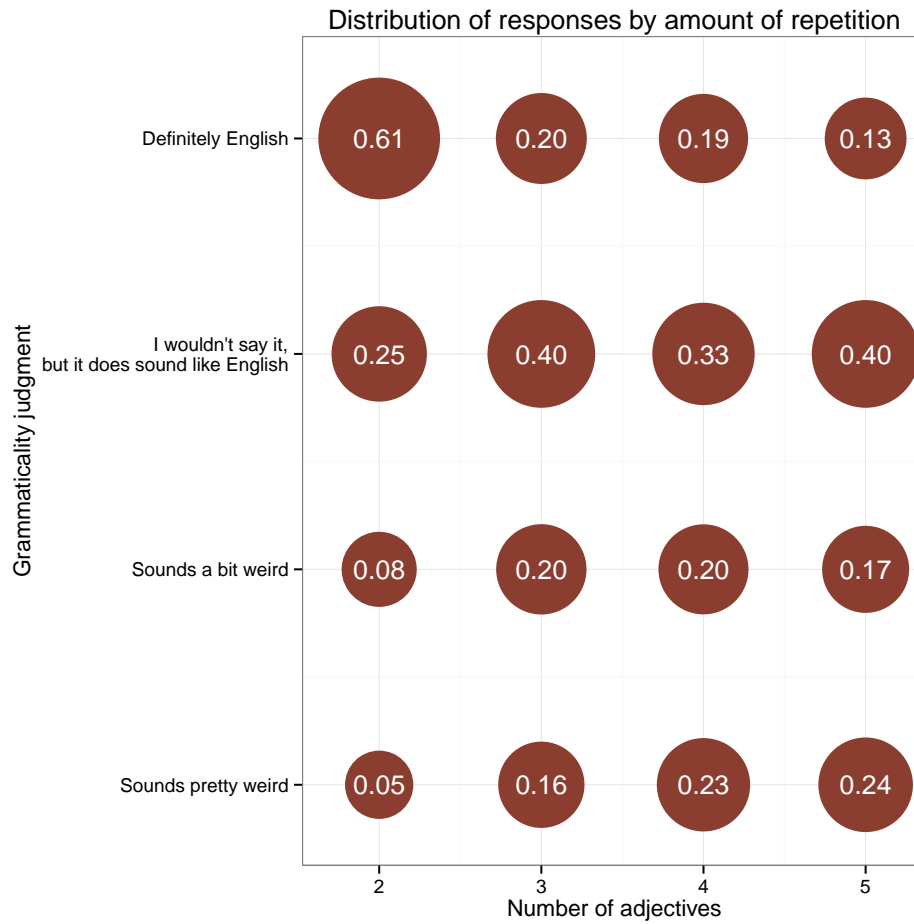


Figure 6.6: Correlation between amount of repetition of adjectives and grammaticality judgement. The row for *definitely not English* is empty



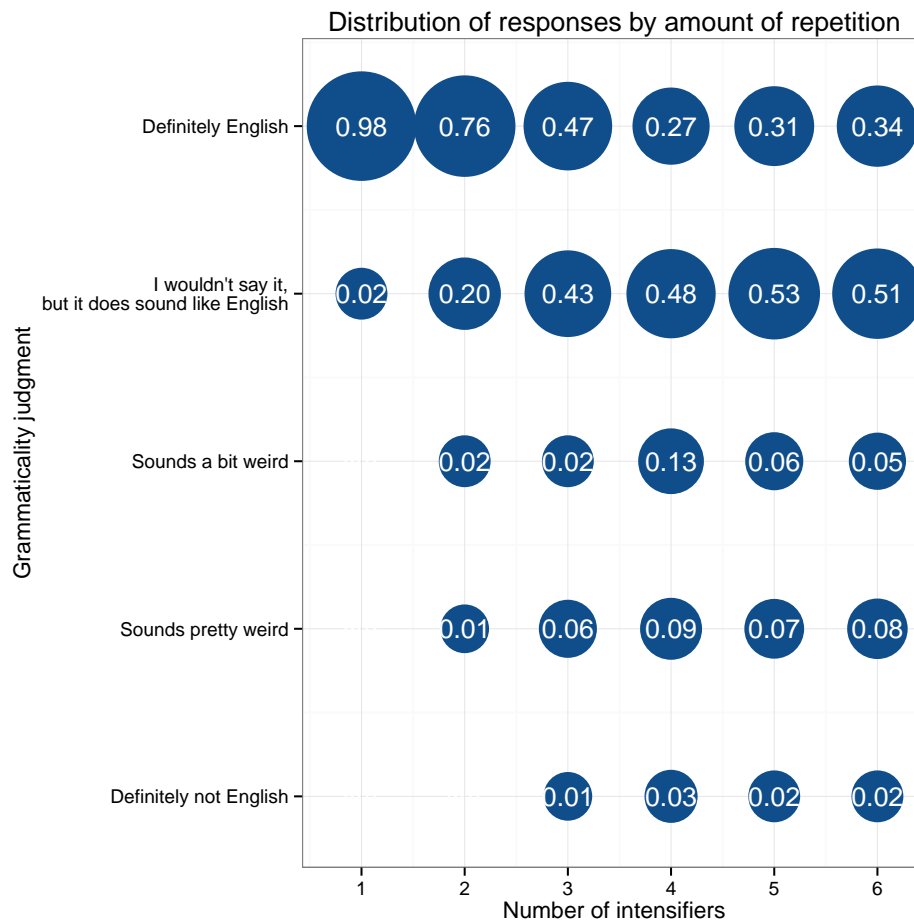


Figure 6.7: Correlation between amount of repetition of intensifiers and grammaticality judgement

performed Wilcoxon signed rank tests, rather than t-tests, to see if the  $\tau$  means were different from 0. Both were significantly different. For adjectives, the mean  $\tau$  was  $-0.3205$  ( $V = 6, p \ll 0.0001$ ), and for intensifiers the mean  $\tau$  was  $-0.2745$  ( $V = 0, p \ll 0.0001$ ). Correlation between acceptance and amount of repetition does therefore hold even on the individual level.

#### 6.5.0.4 Discussion

Despite there being no cut-off for grammaticality of English adjective and intensifier repetition, participants' acceptance gradually decreased as the number of repetitions increased. We propose that this extra-linguistic factor brought ratings of Tagalog generalised stimuli down from the grammar many participants may well have learned.

## 6.6 Limitations

Our fragment of Tagalog is infinite because of the recursion on the adjectives, but naturally of this fragment we used only a subset. There are  $2 \times 2 \times 4 \times (1 + 4 \times 3) = 208$  sentences in the training sub-language and  $2 \times 2 \times 4 \times (1 + 4 \times 5) = 336$  in the testing sub-language (a superset of the training sub-language). Because the languages are so small it would in principle have been possible to take all of the sentences and divide them up into testing and training stimuli with the types (number of adjectives, presence of adverb, word choice) counterbalanced. Instead we generated the sentences randomly, sacrificing counterbalancing for randomness. Future studies using such small sub-languages would be better served by counterbalanced stimuli.

Similarly, the English survey should have been carried out with equal numbers of different stimuli types.

In Grammar 1 (Adj-N), adjectives were always pronounced with a linker, and in Grammar 2 (N-Adj), adjectives were pronounced with the linker except the last one.

- (9) a. Natulog ang matanda-ng matanda-ng pusa' **G1**  
 sleep D old-LK old-LK cat
- b. Natulog ang pusa' matanda-ng matanda **G2**  
 sleep D cat old-LK old

In the ungrammatical noun-repetition stimuli, it was Grammar 1 that had a repeating item with no linker:

- (10) a. \*Natulog ang matanda-ng pusa-ng pusa' **G1**  
 sleep D old-LK cat-LK cat
- b. \*Natulog ang pusa-ng pusa-ng matanda **G1**  
 sleep D cat-LK cat-LK old

This could have made it more difficult for Grammar 2 learners to learn the repetition, as they may have interpreted the linked and unlinked adjectives (eg *matanda* and *matandang*) as separate words. It may also have led Grammar 1 learners to interpret noun-repetition as adding a new word to the language they had never heard before (eg they knew *pusa'* but now they are hearing *pusang*).

However, given that we found no difference in acceptance rates for stimuli with and without linkers, and that the difference in results between the two groups was driven primarily by Group 2's high acceptance of scrambled stimuli, it is unlikely linkers made a large difference.

## 6.7 Conclusion

We used an artificial language learning experiment to see if people generalise from limited to indefinite word-level repetition. We found that people do not find sentences with more repetition than they were trained on to be as good as those with the same amount of repetition as they were trained on. However, we did find that when the repeating adjective was sentence-medial, participants preferred the sentences with more repetition

than they were trained on to ungrammatical sentences. When scrambled ungrammatical stimuli were taken out of the picture, participants in both grammars preferred generalised to ungrammatical stimuli.

In order to further understand the basis of this result, we also conducted an English survey, and found that people prefer English sentences with less repetition to sentences with more. This fact may explain the difference in acceptance between familiar (up to 3 adjectives) and generalised (4 or 5 adjectives) stimuli.

We conclude that people can and often do generalise word-level repetition, but that the high salience of the pattern may interfere with implicit learning, leading participants to learn much more explicitly than is desirable in an ALL experiment. However, we expect this to be less of an issue in the next planned studies. The Tagalog experiment is the first in a planned series of four:

**Exp 1** Do people generalise from limited to indefinite word repetition? (This chapter)

**Exp 2** Do people generalise from optionality to word repetition?

**Exp 3** Do people generalise from limited to indefinite category repetition?

**Exp 4** What about when you embed the repetition in a highly complex grammar?

Experiment 2 asks whether people accept repetition of stimuli that they have learned are optional. Several formal learners predict this to be the case, but we find the possibility unlikely, given the amount of non-repeatable optional material in human language, for example English adverbs and complementiser *that*.

- (11)
- a. John left (suddenly).
  - b. \*John left suddenly suddenly
  - c. I think (that) you should go.
  - d. \*I think that that you should go.

The third experiment, category-level repetition generalisation, we hypothesise will show people do generalise from limited to indefinite repetition. The strangeness/salience of word-level repetition gone, we predict people will be more accepting of repeating categories. However, it may be that acceptance of generalised stimuli will still be lower than that for sentences with fewer adjectives. Not only would they be unfamiliar and longer, but an early study by [Danks and Glucksberg \(1971\)](#) showed that even in English, people ranked sentences with three adjectives below grammatical sentences without such category repetition.

Finally, in an attempt to keep things simple, we have chosen a language that is so simple as to be generable by a bigram grammar (see chapter 5). Human syntax is much more complex than bigrams, yet we do know that people can use knowledge of transitional probabilities (probabilities on bigrams) in artificial language learning tasks ([Hunt and Aslin, 2001](#)). If participants generalise the same ways in a more complex language, we can be more confident they are bringing their language faculty to bear on the subject, not only their ability to remember transitions.

## CHAPTER 7

### Conclusion

Throughout this dissertation, I argued that adjuncts should be treated separately from arguments due to their distinct properties, particularly their optionality and iterability. I showed that the Cinque hierarchies of adjuncts and functional heads can be incorporated into minimalist grammars while still treating adjuncts as distinct from arguments. Finally I explored the learnability of optionality and iterability, distinct properties of adjuncts, finding that many learning models predict a close relationship between iterability and optionality, and that human learners can generalise from limited to indefinite repetition.

#### 7.1 Adjuncts vs Arguments

From the perspective of weak generative capacity, it is not strictly necessary to treat adjuncts as distinct from arguments. Sentences that are traditionally thought to contain adjuncts can be generated by grammars that don't include a distinct adjunction operation; theorem 4.8.5 shows that Minimalist Grammars with Adjunction generate the same languages, in terms of the surface strings, as traditional MGs with only Merge and Move. However, weak generative capacity is a bare minimum criterion by which to judge a model. The structure of the derivation should reflect our understanding of how language works: the types of syntactic and semantic relationships between parts of the sentence, which substrings form constituents, and so forth. Treating adjunction as distinct reflects decades of literature that finds them to be distinct. It allows adjuncts to be ignored when

it comes to selection; it allows adjuncts to iterate. It can do all of this while still retaining the capacity to require an order on adjuncts.

Ernst (2002) argues convincingly that at least most of the ordering on adverbs can be accounted for by semantic types. If an adverb is of the wrong type to combine with its sister, it cannot occur there; otherwise it can. If this works, then whether this is enough to handle adverbs is a matter of whether the syntax should independently enforce selection, via Merge being driven by selectional features, or whether only Move should be driven by features, and Merge can be entirely semantic. An MG of course has a feature-driven Merge, but this is not necessarily correct for human language. The question of whether syntactic grammaticality and semantic interpretability ought to be considered independent is an old one. Chomsky (1957) famously distinguished (1-a) and (1-b) to illustrate the difference between syntax and semantics.

- (1) a. Colourless green ideas sleep furiously
- b. \*Furiously sleep ideas green colourless

However, (1-a) might be nonsensical, but it is not necessarily semantically uninterpretable.

It means

$$\text{colourless} \cap \text{green} \cap \text{ideas} \subseteq \text{sleep} \cap \text{furiously}$$

which is true, since  $\text{colourless} \cap \text{green} \cap \text{ideas} = \emptyset$ . Arguably it contains a presupposition failure, if it presupposes that  $\text{colourless} \cap \text{green} \cap \text{ideas} \neq \emptyset$ , and is certainly infelicitous, but its syntactic grammaticality can still be explained, if any sentence can be, by the semantic types, which work out just fine.

I find it plausible that Ernst's analysis is correct for adverbs and functional heads. However, adjectives also have ordering, but there are no obvious differences in semantic type. Notice also that adjective orders are far less strict than adverb orders. For example, (2-a) sounds strange; (2-b) is word salad.

- (2) a. ?the Norwegian green great beautiful dragon  
b. \*Usually, Zoe early frankly once fought probably

We have also seen (Definition 2.8.1) that the worst misorderings of adjectives cross a semantic boundary: non-predicative adjectives precede predicative adjectives. It could be argued that this boundary reflects semantic types, while the orderings within these types are purely syntactic.

If we can farm as much as possible out to the semantics, we are still left with most of the adjective ordering to be taken care of by the syntax. It is the ordering that is least important: out-of-order adjectives might make you sound like a “maniac” (Fig.7.1), but they are comprehensible, judgments are not always clear, and some languages do not enforce these orders at all. Some speakers even claim that many if not all out-of-order adjectives are not out of order at all. In such a case we still want something like MGAs (section 4.6). We might not need the categories themselves – the semantics might be enough – but we would still need an order on the adjectives and some mechanism to keep track of the hierarchy level of the last one adjoined. However, this ordering would perhaps not be absolute: whatever it is in the grammar that keeps grammaticality from always being strictly true or false is what should be affected by out-of-order adjectives: the sentence should be degraded.

All of this discussion aside, my primary claim here is not that adjunction must work as in MGAs, but rather that any model that seeks to model adjunct ordering in the syntax will need to look something like an MGA. Whatever it is that allows Merge and controls what can adjoin to what should be untouched by adjunction, and something should track the hierarchy level of the last adjunct adjoined.

My current research includes linking minimalist grammars with *abstract meaning representations*, a graphical representation of semantics. A project with Jonas Groschwitz, Mark Johnson, and Alexander Koller (2017, in submission) describes a restrictive algo-





**Eben Marks** @EbenMarks · Sep 4  
 This word order thing is really interesting but as I've already read it, I tried it with my partner who hadn't

Figure 7.1: A BBC tweet that was making the rounds a few months ago which quotes Forsyth (2013), and a reader’s response – his partner chose the pictured order for the adjectives given. This violates the order claimed.

bra for AMRs that distinguishes arguments and adjuncts; my current project is in linking this algebra to an MGA via an *interpreted regular tree grammar*.

Also ongoing is research into the subregular properties of Adjoin. We have seen that an MG with only Merge is strictly  $k$ -local. An MGA with only Merge and Adjoin is not: an unbounded number of adjoin nodes can intervene between one Merge and the next.

I am looking at MGAs from the perspective of *tier-based* locality (Heinz et al., 2011): if we consider only the Merge nodes or only the Adjoin nodes, what is the locality?

## 7.2 Learnability

Chapter 5 examines some formal learners, and Chapter 6 describes an experiment. Future experiments are planned to explore more aspects of repetition; these are described in section 6.7. However, what is still missing here is the link between the learning models and the behaviour of human learners. The language in the experiment is a bigram language. If the humans were bigram learners, they had enough information to generalise according to Theorem 5.4.2, which states they may need as much as  $\{uA^i v \mid i \leq 3\}$  (for  $u, v$  the material preceding and following the adjective(s)), and that is exactly what they are trained on. They also receive enough information if they are 0-reversible or substitutable CF learners: they have 0,1,2, and 3 adjectives in the training data, and all they needed was any two in a row of those. This experiment therefore does not distinguish n-gram and substitutable learners, but the second planned experiment, which tests whether optionality is enough to predict iterability, would. (I predict it is not enough.)

Because the structure participants assign to a sentence is so crucial in what sort of learner they might be, artificial language learning of syntax is very difficult to design and interpret. While the behaviour of my participants can be explained if they are bigram learners who think a lot of repetition is a bit silly, everything we know about human language speaks against their being ngram learners. Human language syntax is very clearly not strictly local; if participants are not bringing their syntax but rather just their statistical sense of transitions to bear on this task, then ALL studies don't teach us anything about human language acquisition. It is for this reason that the best ALL experiments account for bigrams, and the reason my future experiments are designed to test learnability of a language that is not strictly local, so that I can control for bigrams. These

well-designed studies, such as [Rohrmeier et al. \(2012\)](#), indicate that learners in ALL syntax experiments are using their real capacity to learn syntax, not just bigrams; I consider this to be good evidence that ALL syntax experiments are worth doing.

Learnability theory continues apace, and there are new models I have not yet examined for their iterability and optionality learning capacities. Other learning methods, such as neural networks, provide further, and entirely mysterious, avenues of exploration for adjunct learnability. I am just beginning a new project that explores the workings of neural network language learning.

### **7.3 Final Thoughts**

In an interview with a company that makes use of NLP for document interpretation, I was asked whether my work on adjunction is applicable to any real-world problems in NLP. I had to admit that at this juncture it probably is not: the subtleties of adjunct ordering judgments are far beyond the needs of basic information retrieval from a document, and the kind of learning companies like Google are currently interested in is not of the Gold learning variety and certainly cannot incorporate an oracle. However, what I see in the not-so-distant future is a coming together of linguistic modelling with NLP. Companies such as Nuance are already working in this direction. Deep neural networks are performing astonishing linguistic tasks, but no one knows how or why. What I see, then, is a near future in which learning of linguistically adequate grammars, and yes even subtleties of adjunct ordering, are in fact exactly what computational linguistics is all about.

# Appendix A

## Mathematical definitions and symbols

**Definition A.0.1.** *iff*: if and only if: the two statements are always true or false at the same time. Also written  $\iff$ .

**Definition A.0.2.**  $\exists!$  means "there is a unique".

**Definition A.0.3.** *Set*: unordered collection of things, notated in braces, e.g.  $\{a,b,c\}$

**Definition A.0.4.** *Natural numbers*:  $\mathbb{N} = \{0, 1, 2, 3, \dots\}$

**Definition A.0.5.** *Element/member*: things in sets are often called *elements*.  $a \in B$  means  $a$  is an element (or member) of the set  $B$ ;  $a \notin B$  means  $a$  is not an element of  $B$ .

**Definition A.0.6.** *Subset, superset*:  $A$  is a subset of  $B$  (written  $A \subseteq B$ ) iff every element of  $A$  is also an element of  $B$ .  $A$  is a superset of  $B$  (written  $A \supseteq B$ ) iff every element of  $B$  is also an element of  $A$ .

**Notation** *Defining sets*: a common way to define a set is as follows:  $\{\text{variables in sets} \mid \text{some statement}\}$ .  $\mid$  means *such that*; sometimes it's written with a colon instead. For example  $\{a \in \mathbb{N} \mid a < 3 \ \& \ a > 10\}$  is the set of all natural numbers greater than 3 and less than ten:  $\{4, 5, 6, 7, 8, 9\}$ .

**Definition A.0.7.** *set intersection*:  $A \cap B = \{a \mid a \in A \ \& \ a \in B\}$

*set union*:  $A \cup B = \{a \mid a \in A \ \text{or} \ a \in B\}$

*set difference*:  $A - B = \{a \in A \mid a \notin B\}$ .

**Definition A.0.8.** *Sequence:* ordered collection of things. Notation varies:

$$abc = a \ b \ c = \langle a, b, c \rangle = (a, b, c) = a :: (b :: (c :: \epsilon))$$

**Definition A.0.9.** *Subsequence:*  $s$  is a subsequence of  $t$  if and only if all items in  $s$  are also in  $t$  and they occur in the same order

**Definition A.0.10.** *String:* sequence of words or letters. Often notated without commas.

**Definition A.0.11.** *Substring:* Contiguous (all in a row) subsequence of a string

**Definition A.0.12.** *Empty set:* the empty set has no members. Written  $\emptyset$  or  $\{\}$

**Definition A.0.13.** *Empty sequence/string:* a sequence with no members. I write it  $\epsilon$ . Another common symbol is  $\lambda$ .

**Notation** *concatenation:* forming a sequence. Can be written  $\frown$ .

*Cons:* When a new element is added to a sequence we can write  $::$  eg:  $a :: \langle b, c, d \rangle = \langle a, b, c, d \rangle$ .

If  $\alpha$  is a sequence of, say, features, and  $f$  is a feature then  $f\alpha$  is shorthand for  $f :: \alpha$ .

The symbol  $::$  is also used in some MGs to indicate that a grammatical object is a lexical item.

**Note:** Note that sets and sequences can contain sets and sequences. For example, lexical items in a minimalist grammar are in their simplest form a two-element sequence made up of a word – or sequence of words – and a sequence of features. The features themselves are really pairs of an arity and a basic feature. I write them as follows:

$$\langle \text{see}, =D=DV \rangle \text{ or } \text{see}::=D=DV$$

If we get really careful and consistent we might write  $\langle \langle \text{see} \rangle, \langle \langle =, D \rangle, \langle =, D \rangle, \langle , V \rangle \rangle \rangle$  but that's really hard to read.

Technically, the  $::$  in the second notation is a third element in the sequence indicating whether the item is lexical ( $::$ ) or derived ( $:$ )

$\langle\langle\text{see}\rangle, ::, \langle\langle=, D\rangle, \langle=, D\rangle, \langle, V\rangle\rangle\rangle$

**Definition A.0.14.** *Binary relation:* a binary relation is a set of ordered pairs (two-element sequences) in which the first element is chosen from one set,  $A$ , and the second is from set  $B$ . We write  $A \times B$  and say  $A$  cross  $B$  for the full set of ordered pairs from  $A$  and  $B$ . A general binary relation over  $A \times B$  is then some subset of  $A \times B$ .

**Definition A.0.15.** *Function:* a binary relation in which a given element occurs at most once as a left-hand element. A *total function* has all of the elements of the first set appearing exactly once as the first element of a pair; a *partial function* does not require that all elements appear. This dissertation focuses primarily on partial functions such as those governing syntactic operations. Syntactic operations are usually restricted, so they are partial functions.

A function  $f$  can be notated in several ways:

term notation:  $f(x) = 2x$

Term notation with cases:  $f(x) = \begin{cases} 2x & \text{if } x > 5 \\ -2x & \text{otherwise} \end{cases}$

arrow notation:

	$f$	
	$1 \rightarrow 2$	
	$2 \rightarrow 4$	
	$3 \rightarrow 6$	
	$4 \rightarrow 8$	
	$\dots$	

To say what the two sets are we usually write  $f : A \rightarrow B$  and say  $f$  maps from  $A$  to  $B$  or a function  $f$  from  $A$  to  $B$ .

**Definition A.0.16** (Algebra).

An algebra has a syntax and a semantics. Syntactically, it has a *signature* of function symbols with set *arities/ranks*: numbers associated with them. If  $f$  has rank  $i$  we can write  $f^{(i)}$ .

$$S = \langle \Sigma^{(0)}, F_1^{(1)}, \dots, F_k^{(k)} \rangle$$

is a signature with maximal rank  $k$ .

*Terms* over the signature  $T_S$  are defined as follows:

1. if  $w \in \Sigma^{(0)}$  then  $w$  is a term
2. if  $f \in F_i^{(i)}$  and  $t_1, \dots, t_i$  are terms then  $f(t_1, \dots, t_i)$  is a term

Terms are *interpreted* in the *domain*  $D$  of the algebra, where symbols of rank 0 are interpreted as elements of the domain, and those of rank  $i > 0$  as functions of arity  $i$  on  $D$ . The interpretation  $\llbracket t \rrbracket$  of a term  $t$  is the value of the functions applied to their arguments, and the symbols of rank 0 are interpreted as themselves.

For example, let  $A$  be string-concatenation algebra with signature  $\langle \Sigma^{(0)}, \bullet^{(2)} \rangle$ , and domain  $\Sigma^*$ .  $\bullet$  is interpreted as string concatenation, call it  $\frown: \Sigma^* \rightarrow \Sigma^*$ . Then we can interpret for example  $\llbracket \bullet(\bullet(a, \bullet(b, b)), c) \rrbracket = \frown (\llbracket \bullet(a, \bullet(b, b)) \rrbracket, \llbracket c \rrbracket) = \frown (\frown (\llbracket a \rrbracket, \llbracket \bullet(b, b) \rrbracket), c) = \frown (\frown (a, \frown (b, b)), c) = abbc$

## **Appendix B**

### **Adjective ordering**

The following tables are the various adjective orders in the literature discussed in Chapter 2, put together so that they can be compared. I have guessed where the authors of different papers intended the same set of adjectives; this should be considered an approximation only.



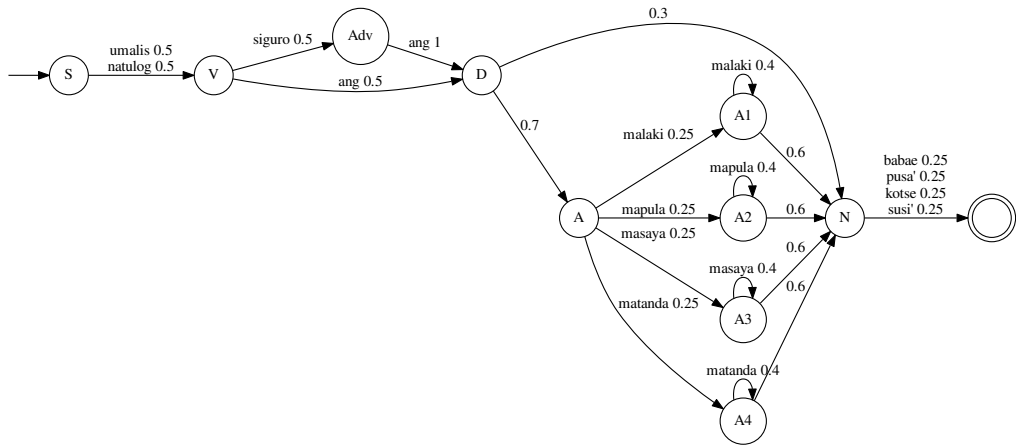


## Appendix C

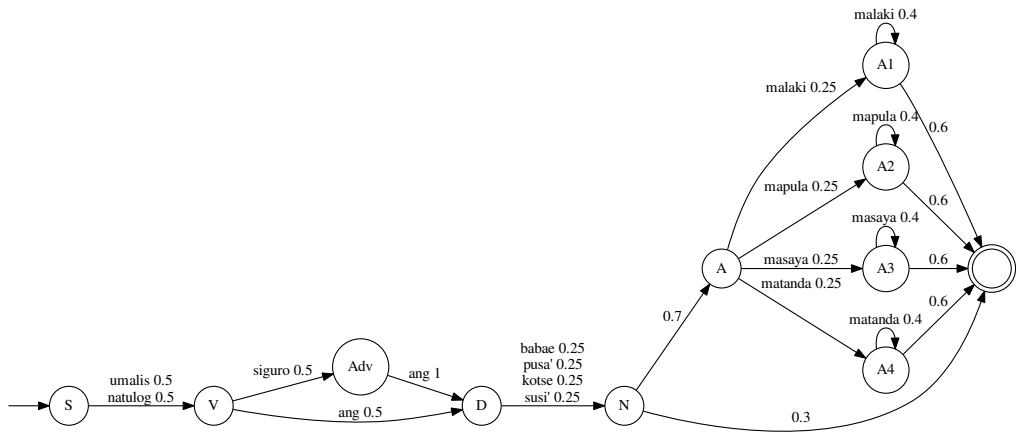
### Tagalog Experiment: Probabilistic Finite State Automata

The following PFSA's generated the training stimuli for the experiment in Chapter 6. Since training stimuli have no more than 3 adjectives, we simply removed the few stimuli with more than three adjectives. We generated the stimuli for Grammar 1 with the machine in Figure C.1a. To keep all stimuli the same between grammars, we created the stimuli for Grammar 2 by moving the adjectives to the end, rather than generate new stimuli using the machine in C.1b. The machine in C.1b therefore just represents the grammar of Grammar 2.

For the testing stimuli we changed the probabilities slightly to increase the probability of repeated adjectives. In the training stimuli, the probability of another adjective when one adjective has already been generated is 0.4. In the testing stimuli it is 0.6. To generate testing stimuli we generated sentences using this PFSA until we had 90 unique sentences, none of which had appeared in the training stimuli, save the eight sentences of the form VDN.

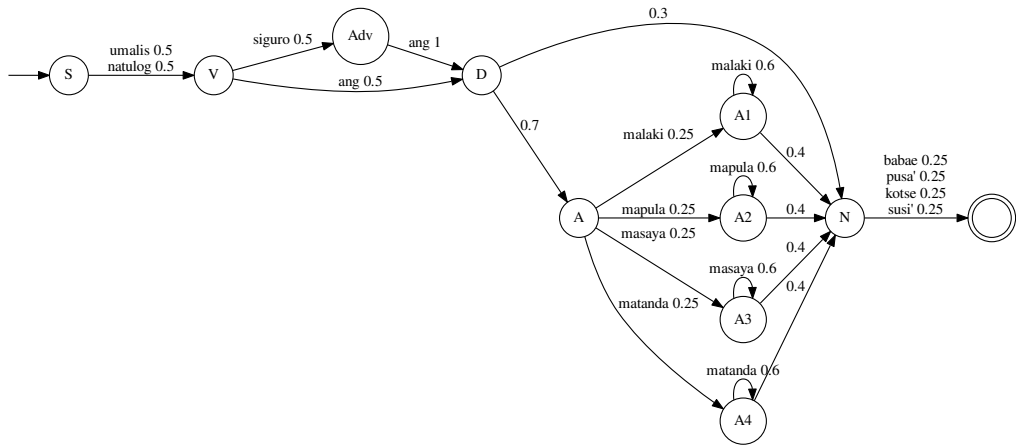


(a) Grammar 1: adjective-medial

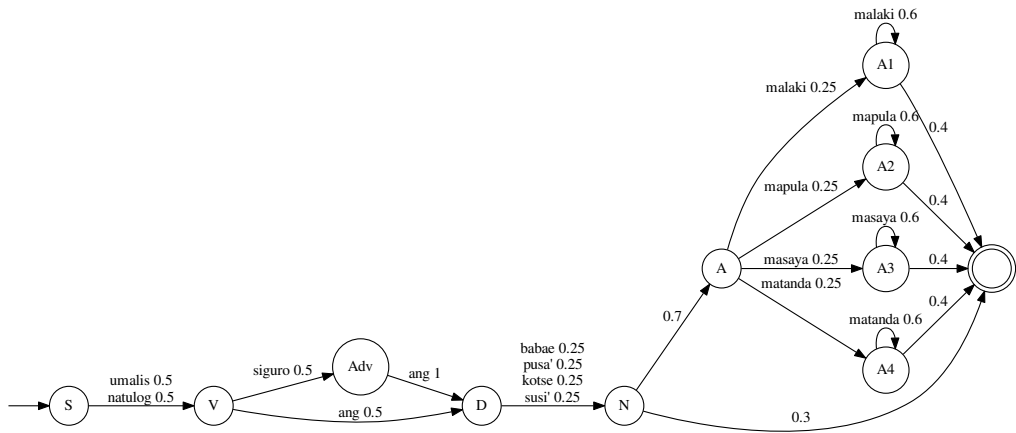


(b) Grammar 2: adjective-final

Figure C.1: Finite State Automata that generated the training stimuli



(a) Grammar 1: adjective-medial



(b) Grammar 2: adjective-final

Figure C.2: Finite State Automata that generated the testing stimuli

## Appendix D

### Tagalog Stimuli

#### D.1 Training stimuli

1. natulog ang pusa'
2. natulog siguro ang matanda matanda pusa'
3. umalis siguro ang babae
4. natulog siguro ang mapula kotse
5. umalis ang malaki pusa'
6. umalis ang kotse
7. natulog siguro ang masaya masaya masaya kotse
8. umalis siguro ang malaki malaki malaki pusa'
9. natulog ang masaya masaya masaya kotse
10. natulog siguro ang malaki pusa'
11. natulog siguro ang mapula pusa'
12. umalis siguro ang kotse
13. umalis siguro ang matanda kotse
14. natulog ang babae
15. umalis ang masaya masaya susi'
16. natulog siguro ang babae
17. umalis siguro ang malaki pusa'
18. umalis ang kotse
19. umalis siguro ang susi'
20. umalis siguro ang kotse
21. natulog siguro ang matanda susi'
22. natulog siguro ang malaki babae
23. umalis ang matanda pusa'
24. natulog siguro ang susi'
25. natulog ang babae
26. umalis ang mapula pusa'
27. umalis ang malaki susi'
28. natulog ang pusa'

29. umalis ang masaya kotse
30. natulog ang matanda matanda matanda susi'
31. natulog siguro ang masaya masaya masaya pusa'
32. natulog ang matanda pusa'
33. natulog ang kotse
34. umalis siguro ang matanda matanda matanda babae
35. umalis siguro ang kotse
36. natulog siguro ang masaya susi'
37. natulog siguro ang susi'
38. natulog siguro ang masaya pusa'
39. natulog siguro ang masaya babae
40. natulog siguro ang susi'
41. umalis ang masaya masaya pusa'
42. umalis siguro ang masaya susi'
43. umalis siguro ang susi'
44. umalis ang malaki babae
45. natulog ang pusa'
46. natulog ang malaki malaki malaki kotse
47. umalis siguro ang mapula babae
48. natulog siguro ang mapula kotse
49. umalis siguro ang babae
50. natulog siguro ang masaya babae
51. natulog siguro ang masaya susi'
52. umalis siguro ang mapula mapula pusa'
53. umalis siguro ang mapula babae
54. umalis ang matanda babae
55. umalis ang mapula pusa'
56. natulog ang pusa'
57. umalis ang matanda matanda matanda kotse
58. umalis ang masaya masaya kotse
59. natulog siguro ang mapula mapula pusa'
60. natulog ang malaki babae
61. umalis siguro ang matanda babae
62. natulog siguro ang malaki babae
63. umalis ang matanda matanda matanda susi'
64. umalis ang mapula kotse
65. umalis siguro ang matanda susi'
66. natulog ang mapula pusa'
67. natulog siguro ang matanda matanda matanda pusa'
68. natulog ang masaya masaya masaya pusa'
69. umalis siguro ang masaya masaya masaya babae
70. natulog siguro ang babae

71. umalis siguro ang mapula mapula kotse
72. umalis siguro ang masaya susi'
73. umalis siguro ang masaya kotse
74. umalis siguro ang masaya babae
75. umalis siguro ang masaya masaya kotse
76. natulog ang malaki malaki babae
77. natulog siguro ang kotse
78. umalis siguro ang mapula susi'
79. natulog siguro ang mapula mapula mapula babae
80. natulog siguro ang matanda matanda babae
81. natulog ang matanda susi'
82. umalis ang mapula mapula mapula pusa'
83. natulog siguro ang mapula susi'
84. natulog ang babae
85. natulog siguro ang matanda matanda susi'
86. umalis ang matanda kotse
87. natulog siguro ang babae
88. natulog ang masaya masaya babae
89. umalis siguro ang babae
90. umalis ang mapula susi'
91. natulog ang kotse
92. umalis ang pusa'
93. umalis siguro ang matanda pusa'
94. natulog siguro ang susi'
95. natulog siguro ang matanda babae
96. umalis ang susi'
97. umalis ang masaya babae
98. umalis siguro ang susi'
99. umalis siguro ang masaya babae
100. natulog siguro ang susi'
101. umalis siguro ang susi'
102. natulog siguro ang mapula pusa'
103. umalis siguro ang kotse
104. natulog siguro ang masaya susi'
105. umalis siguro ang susi'
106. umalis ang kotse
107. natulog ang mapula kotse
108. natulog siguro ang malaki pusa'
109. natulog ang malaki malaki babae
110. natulog siguro ang masaya masaya masaya babae
111. natulog ang matanda susi'
112. natulog ang masaya kotse

113. umalis ang matanda matanda babae
114. umalis ang mapula susi'
115. umalis ang babae
116. natulog ang mapula mapula kotse
117. natulog siguro ang malaki malaki pusa'
118. umalis ang malaki kotse
119. umalis ang malaki pusa'
120. umalis ang kotse
121. umalis siguro ang masaya susi'
122. natulog ang malaki malaki susi'
123. umalis siguro ang mapula susi'
124. natulog siguro ang malaki malaki pusa'
125. umalis siguro ang mapula kotse
126. umalis ang matanda susi'
127. natulog siguro ang mapula susi'
128. umalis ang susi'
129. umalis ang kotse
130. natulog siguro ang susi'
131. umalis siguro ang matanda babae
132. natulog siguro ang kotse
133. natulog siguro ang masaya susi'
134. natulog siguro ang kotse
135. natulog siguro ang malaki pusa'
136. umalis ang babae
137. umalis siguro ang pusa'
138. umalis siguro ang susi'
139. natulog siguro ang babae
140. natulog ang pusa'
141. umalis ang malaki pusa'
142. natulog siguro ang pusa'
143. natulog siguro ang malaki malaki malaki kotse
144. umalis ang mapula pusa'
145. umalis ang mapula babae
146. umalis siguro ang kotse
147. umalis ang mapula kotse
148. umalis ang masaya kotse
149. umalis siguro ang mapula babae
150. umalis siguro ang matanda babae
151. natulog siguro ang kotse
152. umalis siguro ang pusa'
153. natulog siguro ang masaya masaya susi'
154. natulog siguro ang mapula babae



155. natulog ang mapula mapula susi'
156. natulog ang mapula kotse
157. umalis siguro ang masaya susi'
158. natulog ang malaki kotse
159. umalis siguro ang pusa'
160. natulog ang malaki malaki susi'
161. umalis siguro ang malaki babae
162. umalis ang susi'
163. natulog siguro ang malaki malaki susi'
164. natulog siguro ang babae
165. natulog siguro ang kotse
166. natulog siguro ang kotse
167. natulog siguro ang malaki pusa'
168. umalis siguro ang mapula mapula kotse
169. natulog siguro ang mapula babae
170. natulog siguro ang matanda matanda pusa'
171. umalis siguro ang malaki kotse
172. natulog ang susi'
173. natulog ang malaki pusa'
174. umalis ang masaya masaya kotse
175. natulog siguro ang kotse
176. natulog ang mapula mapula mapula kotse
177. umalis siguro ang malaki malaki babae
178. natulog ang kotse
179. umalis ang matanda babae
180. umalis siguro ang masaya susi'

## **D.2 Testing stimuli**

### **D.2.1 Grammatical**

#### **D.2.1.1 Familiar**

1. umalis siguro ang babae
2. natulog ang malaki malaki babae
3. umalis ang malaki malaki malaki babae
4. natulog siguro ang kotse
5. natulog ang masaya masaya susi'
6. umalis ang malaki pusa'

7. natulog ang malaki susi'
8. umalis siguro ang mapula mapula mapula kotse
9. umalis ang matanda susi'
10. natulog siguro ang malaki malaki susi'
11. natulog siguro ang babae
12. natulog ang malaki babae
13. umalis siguro ang masaya pusa'
14. natulog ang malaki kotse
15. umalis siguro ang masaya masaya susi'
16. natulog ang matanda matanda matanda babae
17. natulog siguro ang masaya kotse
18. umalis ang malaki susi'
19. umalis ang matanda matanda matanda kotse
20. umalis siguro ang susi'
21. umalis ang kotse
22. umalis ang pusa'
23. natulog ang mapula mapula pusa'
24. umalis ang malaki babae
25. natulog siguro ang mapula babae
26. umalis ang mapula susi'
27. natulog ang mapula mapula mapula pusa'
28. natulog ang mapula susi'
29. umalis siguro ang mapula mapula mapula babae
30. umalis ang masaya masaya masaya babae
31. umalis ang masaya susi'
32. umalis ang matanda pusa'
33. natulog ang masaya kotse
34. natulog ang matanda susi'
35. umalis ang malaki malaki pusa'
36. natulog ang matanda matanda matanda susi'
37. natulog ang mapula babae
38. natulog ang pusa'
39. natulog ang matanda matanda susi'
40. umalis ang matanda matanda susi'
41. natulog siguro ang matanda matanda kotse
42. umalis ang matanda matanda matanda babae
43. umalis ang malaki kotse
44. natulog ang malaki malaki susi'
45. natulog ang masaya masaya masaya pusa'
46. natulog ang matanda matanda matanda pusa'
47. umalis ang mapula mapula susi'
48. umalis siguro ang kotse

49. natulog ang matanda matanda pusa'
50. natulog siguro ang matanda babae
51. natulog ang susi'
52. natulog ang mapula mapula mapula kotse
53. umalis ang masaya masaya masaya susi'
54. natulog ang babae
55. natulog ang masaya susi'
56. natulog siguro ang mapula susi'
57. umalis ang mapula mapula babae
58. natulog ang malaki pusa'

#### D.2.1.2 Generalised

1. natulog siguro ang masaya masaya masaya masaya susi'
2. umalis siguro ang malaki malaki malaki malaki babae
3. umalis ang malaki malaki malaki malaki malaki kotse
4. natulog ang matanda matanda matanda matanda matanda pusa'
5. umalis ang malaki malaki malaki malaki kotse
6. natulog ang matanda matanda matanda matanda pusa'
7. umalis ang malaki malaki malaki malaki malaki babae
8. natulog ang mapula mapula mapula mapula mapula susi'
9. umalis ang malaki malaki malaki malaki malaki malaki kotse
10. natulog ang masaya masaya masaya masaya pusa'
11. umalis ang matanda matanda matanda matanda matanda kotse
12. umalis ang matanda matanda matanda matanda kotse
13. umalis ang malaki malaki malaki malaki pusa'
14. umalis ang malaki malaki malaki malaki malaki babae
15. natulog ang mapula mapula mapula mapula mapula babae
16. umalis siguro ang masaya masaya masaya masaya babae
17. umalis ang matanda matanda matanda matanda babae
18. natulog ang matanda matanda matanda matanda matanda susi'
19. umalis ang masaya masaya masaya masaya masaya susi'
20. natulog ang mapula mapula mapula mapula babae
21. umalis ang masaya masaya masaya masaya masaya kotse
22. umalis ang mapula mapula mapula mapula pusa'
23. umalis siguro ang masaya masaya masaya masaya pusa'
24. umalis ang malaki malaki malaki malaki malaki babae
25. natulog ang matanda matanda matanda matanda babae
26. natulog ang masaya masaya masaya masaya kotse
27. natulog ang malaki malaki malaki malaki malaki susi'

28. umalis siguro ang mapula mapula mapula mapula babae
29. umalis ang masaya masaya masaya masaya masaya babae
30. natulog ang mapula mapula mapula mapula mapula babae
31. umalis ang malaki malaki malaki malaki babae
32. natulog ang mapula mapula mapula mapula susi'

## **D.2.2 Ungrammatical**

### **D.2.2.1 Scrambled**

1. siguro matanda matanda babae ang natulog
2. natulog masaya masaya babae ang
3. babae ang umalis
4. ang siguro natulog susi'
5. matanda umalis susi' ang
6. babae matanda siguro ang natulog matanda
7. natulog susi' ang siguro
8. natulog masaya ang masaya pusa' siguro
9. babae ang masaya umalis
10. ang mapula umalis mapula pusa' mapula mapula
11. umalis ang masaya pusa' siguro
12. ang malaki malaki natulog malaki malaki babae
13. umalis ang susi' mapula
14. matanda matanda umalis matanda ang susi'
15. siguro kotse ang natulog malaki
16. masaya masaya masaya masaya ang umalis susi'
17. natulog ang mapula siguro mapula kotse
18. ang matanda matanda umalis matanda siguro susi'
19. ang babae umalis malaki
20. natulog pusa' ang

### **D.2.2.2 Noun-repetition**

1. natulog siguro ang matanda kotse kotse
2. umalis ang masaya babae babae babae babae babae
3. natulog ang malaki babae babae
4. umalis ang malaki babae babae babae
5. natulog ang matanda susi' susi' susi' susi' susi'

6. natulog ang masaya susi' susi'
7. umalis ang malaki babae babae babae babae babae
8. umalis siguro ang mapula kotse kotse kotse
9. umalis ang matanda kotse kotse kotse
10. natulog siguro ang malaki susi' susi'
11. umalis ang matanda kotse kotse kotse kotse kotse
12. umalis ang masaya susi' susi' susi'
13. natulog ang masaya pusa' pusa' pusa' pusa'
14. umalis ang malaki babae babae babae
15. umalis siguro ang masaya susi' susi'
16. natulog ang matanda babae babae babae
17. umalis ang malaki kotse kotse kotse kotse kotse
18. natulog ang matanda pusa' pusa' pusa' pusa' pusa'
19. umalis ang malaki kotse kotse kotse kotse
20. umalis ang masaya susi' susi' susi' susi' susi'
21. umalis ang matanda kotse kotse kotse
22. umalis ang masaya kotse kotse kotse kotse kotse
23. natulog ang mapula pusa' pusa'
24. natulog ang mapula pusa' pusa' pusa'
25. umalis siguro ang mapula babae babae babae
26. umalis ang masaya babae babae babae
27. natulog siguro ang masaya susi' susi' susi' susi'
28. umalis ang malaki pusa' pusa'
29. natulog ang matanda susi' susi' susi'

### **D.3 English stimuli**

1. My love is like a red red rose
2. What a tiny tiny ant
3. Can you help me tear out this itchy itchy tag?
4. Are you afraid of the hungry hungry hippos?
5. Who saw that funny funny funny cat?
6. What a stupid stupid stupid idea!
7. I haven't seen her in a long long long long time
8. The happy happy happy happy dog wagged its tail
9. Did you ever see such red red red red hair
10. The big big big big big elephant stomped
11. Who saw that funny funny funny funny cat?
12. I really like you
13. I really really like you

14. I really really really like you
15. I really really really really like you
16. I really really really really really like you
17. Marie ate so so so so much food
18. I ll never ever ever ever ever ever ever leave you
19. I really really really really really really like you
20. Je suis un pizza
21. John left suddenly
22. Who left this great big ugly metal sculpture on my lawn?
23. Ch'ochlay naq Xhun yuj ix Malin
24. The stupid ugly big monster roared
25. My doll silly
26. Elephant big big big big stomped
27. I like her; I don't like like her
28. Silly old bear
29. Look out for the great green dragon
30. Look out for the green great dragon

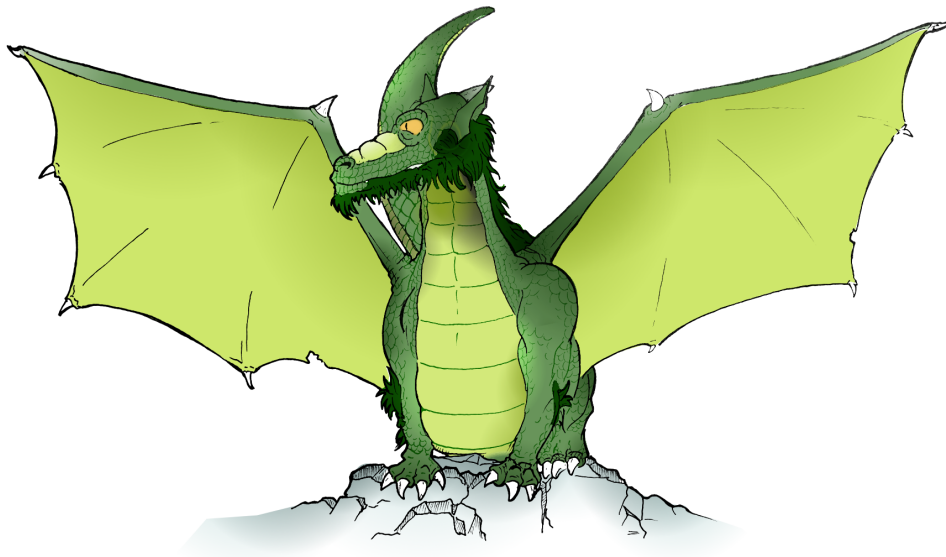
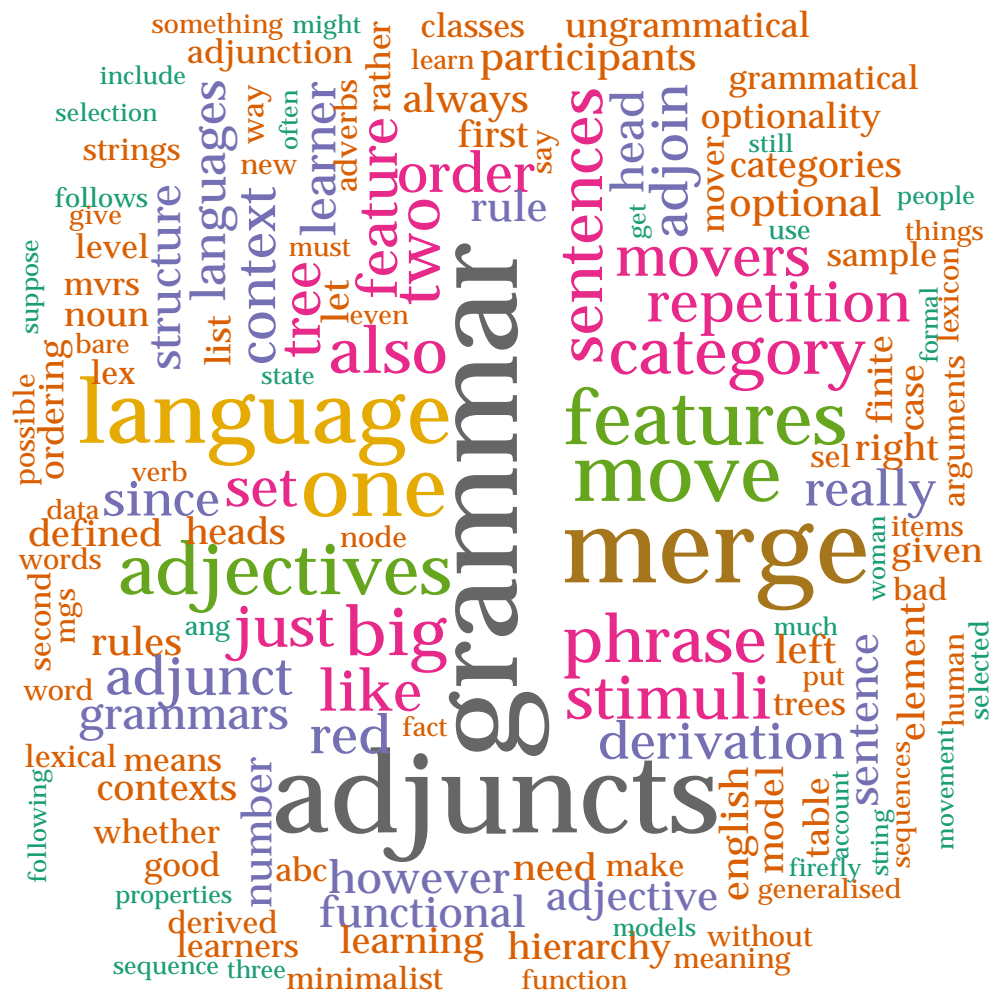


Figure D.1: Illustration by Martín Villalba



## Bibliography

- Ades, Anthony E, and Mark J Steedman. 1982. On the order of words. *Linguistics and philosophy* 4:517–558.
- Adger, David. 2008. A minimalist theory of feature structure. MSS, May 2008.
- Ajdukiewicz, Kazimierz. 1935. O spójności syntaktycznej. *Język i poznanie* 1:222–242.
- Angluin, Dana. 1982. Inference of reversible languages. *Journal of the ACM (JACM)* 29:741–765.
- Bar-Hillel, Yehoshua. 1953. A quasi-arithmetical notation for syntactic description. *Language* 29:47–58.
- Bervoets, Mel, Niki Foster, Meaghan Fowlie, Laura Kalin, Jianjing Kuang, Laura McPherson, Pamela Munro, Kathleen O’Flynn, Denis Paperno, Michael Tseng, Kaeli Ward, David Wemhaner, and Craig Sailor. 2011. Notes on Q’anjob’al as spoken by Alejandra Francisco. Database.
- Brown, Roger. 1965. *Social psychology*. New York: The Free Press.
- Carnie, Andrew. 2010. *Constituent structure*, volume 5. Oxford University Press.
- Carnie, Andrew. 2013. *Syntax: A generative introduction*. John Wiley & Sons.
- Carpenter, Humphrey. 1981. *The letters of JRR Tolkien*, volume 144. Boston: Houghton Mifflin.
- Chomsky, Noam. 1953. Systems of syntactic analysis. *The Journal of Symbolic Logic* 18:242–256.
- Chomsky, Noam. 1957. *Syntactic structures*. The Hague: Mouton.



- Chomsky, Noam. 1959. On certain formal properties of grammars. *Information and control* 2:137–167.
- Chomsky, Noam. 1970. Remarks on nominalization. *Readings in English transformational grammar* .
- Chomsky, Noam. 1981. Lectures on government and binding. *Dordrecht: Foris* .
- Chomsky, Noam. 1993. A minimalist program for linguistic theory. In *The view from building 20: Essays in linguistics in honor of sylvain bromberger*, ed. Kenneth Hale and Samuel Jay Keyser, 1–52. Cambridge, MA: MIT Press.
- Chomsky, Noam. 1995. *The minimalist program*. Cambridge, MA: MIT Press.
- Chomsky, Noam. 2004. Beyond explanatory adequacy. In *Structures and beyond: The cartography of syntactic structures*, ed. Adriana Belletti, volume 3, 104–131. Oxford University Press.
- Chomsky, Noam, and Chris Collins. 2001. *Beyond explanatory adequacy*, volume 20. MITWPL.
- Cinque, Guglielmo. 1994. On the evidence for partial n-movement in the romance dp. In *Paths towards universal grammar: Studies in honour of Richard S. Kayne*, 85–110. Georgetown University Press Washington, DC.
- Cinque, Guglielmo. 1999. *Adverbs and functional heads: a cross-linguistic perspective*. Oxford studies in comparative syntax. Oxford: Oxford University Press.
- Cinque, Guglielmo. 2010. *The syntax of adjectives: a comparative study*. Linguistic Inquiry monographs. Cambridge, MA: MIT Press.
- Clark, Alexander. 2010. Distributional learning of some context-free languages with a minimally adequate teacher. In *International Colloquium on Grammatical Inference*, 24–37. Springer.

- Clark, Alexander, and Rémi Eyraud. 2007. Polynomial identification in the limit of context-free substitutable languages. *Journal of Machine Learning Research* 8:1725–1745.
- Clark, Alexander, Rémi Eyraud, and Amaury Habrard. 2008. A polynomial algorithm for the inference of context free languages. In *Grammatical inference: Algorithms and applications*, 29–42. Springer.
- Clark, Alexander, Rémi Eyraud, and Amaury Habrard. 2010. Using contextual representations to efficiently learn context-free languages. *The Journal of Machine Learning Research* 11:2707–2744.
- Clark, Alexander, and Franck Thollard. 2004. PAC-learnability of probabilistic deterministic finite state automata. *The Journal of Machine Learning Research* 5:473–497.
- Clark, Alexander, and Ryo Yoshinaka. 2014. Distributional learning of parallel multiple context-free grammars. *Machine Learning* 96:5–31.
- Collins, Chris, and Edward Stabler. 2015. A formalization of minimalist syntax. *Syntax* 19:43–78.
- Culbertson, Jennifer. 2012. Typological universals as reflections of biased learning: Evidence from artificial language learning. *Language and Linguistics Compass* 6:310–329.
- Danks, Joseph H., and Sam Glucksberg. 1971. Psychological scaling of adjective orders. *Journal of Verbal Learning and Verbal Behavior* 10:63 – 67. URL <http://www.sciencedirect.com/science/article/pii/S0022537171800956>.
- De Vries, Meinou H, Padraic Monaghan, Stefan Knecht, and Pienie Zwitserlood. 2008. Syntactic structure and artificial grammar learning: The learnability of embedded hierarchical structures. *Cognition* 107:763–774.
- Ernst, Thomas. 2002. *The syntax of adjuncts*, volume 96. Cambridge University Press.

- Etchemendy, John. 1990. *The concept of logical consequence*. Harvard University Press.
- Finley, Sara. 2012. Testing the limits of long-distance learning: Learning beyond a three-segment window. *Cognitive science* 36:740–756.
- Fitch, W Tecumseh, and Marc D Hauser. 2004. Computational constraints on syntactic processing in a nonhuman primate. *Science* 303:377–380.
- Forsyth, Mark. 2013. *The elements of eloquence: How to turn the perfect english phrase*. Icon Books Ltd.
- Fowlie, Meaghan. 2011. Multidominant minimalist grammars. Master's thesis, University of California, Los Angeles.
- Fowlie, Meaghan. 2013. Order and optionality: Minimalist grammars with adjunction. In *The 13th Meeting on the Mathematics of Language*, 12.
- Fowlie, Meaghan. 2014. Adjuncts and minimalist grammars. In *Proceedings of Formal Grammar 2014*, ed. Glyn Morrill, Reinhard Muskens, Rainer Osswald, and Frank Richter, volume 8612 of *Lecture Notes in Computer Science*.
- Fox, Danny. 2000. *Economy and semantic interpretation*. 35. MIT Press.
- Frey, Werner, and Hans-Martin Gärtner. 2002. On the treatment of scrambling and adjunction in minimalist grammars. In *Proceedings of the Conference on Formal Grammar (FGTrento)*, 41–52. Trento.
- Friederici, Angela, and Bertram Opitz. 2002. Brain signatures of artificial language processing” evidence challenging the critical period hypothesis. *Proceedings of the National Academy of Sciences* 99:529–534.
- Fromkin, Victoria, Bruce Hayes, Susan Curtiss, Anna Szabolcsi, Tim Stowell, Edward Stabler, Dominique Sportiche, Hilda Koopman, Patricia Keating, Pamela Munro, and Nina Hyams. 2000. *Linguistics: An introduction to linguistic theory*. John Wiley & Sons.

- Garcia, Pedro, Enrique Vidal, and José Oncina. 1990. Learning locally testable languages in the strict sense. In *ALT*, 325–338.
- Giusti, Giuliana. 1996. Is there a focusp and a topicp in the noun phrase structure? *Working Papers in Linguistics*, 6.2, 1996, pp. 105-128 .
- Gold, E Mark. 1967. Language identification in the limit. *Information and Control* 10:447 – 474. URL <http://www.sciencedirect.com/science/article/pii/S0019995867911655>.
- Goldberg, Adele E, and Farrell Ackerman. 2001. The pragmatics of obligatory adjuncts. *Language* 798–814.
- Graf, Thomas. 2013. Local and transderivational constraints in syntax and semantics. Doctoral Dissertation, UNIVERSITY OF CALIFORNIA Los Angeles.
- Graf, Thomas. 2014. Models of adjunction in minimalist grammars. In *Formal Grammar*, 52–68. Springer.
- Grimshaw, Jane, and Sten Vikner. 1993. Obligatory adjuncts and the structure of events. In *Knowledge and language*, 143–155. Springer.
- Groschwitz, Jonas, Meaghan Fowlie, Mark Johnson, and Alexander Koller. 2017. A constrained graph algebra for semantic parsing with amrs. *12th International Conference on Computational Semantics (IWCS)* .
- Gross, Maurice. 1979. On the failure of generative grammar. *Language* 859–885.
- Harkema, Henk. 2001. A characterization of minimalist languages. In *Logical aspects of computational linguistics*, 193–211. Springer.
- Harris, Zellig S. 1951. *Methods in structural linguistics*.. University of Chicago Press.
- Harris, Zellig S. 1954. Distributional structure. *Word* 10:146–162.

- Heinz, Jeffrey. 2010. Learning long-distance phonotactics. *Linguistic Inquiry* 41:623–661.
- Heinz, Jeffrey, Chetan Rawal, and Herbert Tanner. 2011. Tier-based strictly local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. Portland, Oregon, USA: Association for Computational Linguistics.
- Hetzron, Robert. 1978. On the relative order of adjectives. In *Language universals*, ed. H. Seiler, 165–184. Tübingen.
- Hill, Archibald A. 1958. *Introduction to linguistic structures: from sound to sentence in English*. Harcourt, Brace.
- Hornstein, Norbert, Jairo Nunes, and Kleanthes K Grohmann. 2005. *Understanding minimalism*. Cambridge University Press.
- Hunt, Ruskin H, and Richard N Aslin. 2001. Statistical learning in a serial reaction time task: access to separable statistical cues by individual learners. *Journal of Experimental Psychology: General* 130:658.
- Huybregts, Riny. 1984. The weak inadequacy of context-free phrase structure grammars. In *Van periferie naar kern*, ed. G.J. de Haan, M. Trommelen, and W. Zonnenveld. Foris.
- Jackendoff, Ray. 1977. *X' syntax: A study of phrase structure*. 2. Linguistic Inquiry Monographs Cambridge, Mass.
- Jäger, Gerhard, and James Rogers. 2012. Formal language theory: refining the chomsky hierarchy. *Philosophical Transactions of the Royal Society B: Biological Sciences* 367:1956–1970.
- Joshi, Aravind. 1985. How much context-sensitivity is necessary for characterizing structural descriptions. In *Natural language processing: Theoretical, computational and psycho-*

- logical perspectives*, ed. D. Dowty, L. Karttunen, and A. Zwicky, 206–250. New York: Cambridge University Press.
- Joshi, Aravind, K. Vijay-Shankar, and David Jeremy Weir. 1991. The convergence of mildly context sensitive grammar formalisms. In *Processing of linguistic structure*, ed. P. Sells, S. Sheiber, and T. Waslow. MIT Press.
- Kamp, J.A.W. 1975. Two theories about adjectives. *formal semantics of natural language*, ed. by Edward Keenan, 123–55.
- Kayne, Richard. 1994. *The antisymmetry of syntax*, volume 25 of *Linguistic Inquiry Monographs*. Cambridge, MA: MIT Press.
- Keenan, Edward L., and Edward P. Stabler. 2003. *Bare grammar*. Stanford: CSLI Publications.
- Kobele, Gregory M. 2010. Inverse linking via function composition. *Natural Language Semantics* 18:183–196.
- Kobele, Gregory M., and Jens Michaelis. 2009. Two type 0-variants of minimalist grammars. In *Proceedings of the 10th conference on Formal Grammar and the 9th Meeting on Mathematics of Language*, ed. Gerhard Jäger, Paola Monachesi, Gerald Penn, James Rogers, and Shuly Wintner. Stanford: CSLI Online Publications.
- Kobele, Gregory M., Christian Retoré, and Sylvain Salvati. 2007. An automata-theoretic approach to minimalism. In *Model Theoretic Syntax at ESSLLI '07*, ed. J. Rogers and S. Kepser. ESSLLI.
- Koopman, Hilda. 2015. Violations of the final-over-final constraint (Nawdem). UCLA presentation handout.
- Lai, Regine. 2011. Learnable vs unlearnable harmony patterns. University of Delaware MSS.

- Lambek, Joachim. 1958. The mathematics of sentence structure. *The American Mathematical Monthly* 65:154–170.
- Martin, J.E. 1969. Semantic determinants of preferred adjective order. *Journal of Verbal Learning and Verbal Behavior* 8:697 – 704. URL <http://www.sciencedirect.com/science/article/pii/S0022537169800320>.
- Opitz, Bertram, and Angela Friederici. 2007. Neural basis of processing sequential and hierarchical syntactic structures. *Human Brain Mapping* 585–592.
- Pallier, Christophe, Képa Erdocia, and Stanislas Dehaene. 2012. Artificial grammar learning. In prep.
- Perruchet, Pierre, and Chantal Pacteau. 1990. Synthetic grammar learning: Implicit rule abstraction or explicit fragmentary knowledge? *Journal of Experimental Psychology: General* 119:264–275.
- Peters, P Stanley, and Robert W Ritchie. 1973. On the generative power of transformational grammars. *Information Sciences* 6:49–83.
- Pollock, Jean-Yves. 1989. Verb movement, universal grammar, and the structure of ip. *Linguistic inquiry* 365–424.
- Putnam, Hilary. 1970. Is semantics possible? *Metaphilosophy* 1:187–201.
- Reber, Arthur S. 1967. Implicit learning of artificial grammars. *Journal of Verbal learning and Verbal Behaviour* 6:855–863.
- Rizzi, Luigi. 2004. Locality and left periphery. In *Structures and beyond: The cartography of syntactic structures*, ed. Adriana Belletti, volume 3, 223–251. Oxford: Oxford University Press.
- Rohrmeier, Martin, Qiufang Fu, and Zoltan Dienes. 2012. Implicit learning of recursive context-free grammars. *PloS one* 7:e45885.

- Rowling, J.K. 2008. *Harry Potter and the Deathly Hallows*. Bloomsbury.
- Schieber, Stuart. 1985. Evidence against the context-freeness of natural language. *Linguistics and Philosophy* 333–343.
- Scott, Gary-John. 2002. Stacked adjectival modification and the structure of nominal phrases. *Functional structure in DP and IP. The cartography of syntactic structures* 1:91–120.
- Seki, Hiroyuki, Takashi Matsumura, Mamoru Fujii, and Tadao Kasami. 1991. On multiple context-free grammars. *Theoretical Computer Science* 88:191–229.
- Sportiche, Dominique. 2005. Division of labor between merge and move: Strict locality of selection and apparent reconstruction paradoxes. *LingBuzz* .
- Sproat, Richard, and Chilin Shih. 1988. Prenominal adjectival ordering in english and mandarin. In *Proceedings of NELS*, volume 18:2, 465–489.
- Stabler, Edward. 1997. Derivational minimalism. *Logical Aspects of Computational Linguistics* 68–95.
- Stabler, Edward. 2013. Two models of minimalist, incremental syntactic analysis. *Topics in cognitive science* 5:611–633.
- Stabler, Edward P. 2004. Varieties of crossing dependencies: structure dependence and mild context sensitivity. *Cognitive Science* 28:699–720.
- Stabler, Edward P, and Edward L Keenan. 2003. Structural similarity within and among languages. *Theoretical Computer Science* 293:345–363.
- Steedman, Mark. 2001. *The syntactic process*. MIT press.
- Stowell, Timothy Angus. 1981. Origins of phrase structure. Doctoral Dissertation, Massachusetts Institute of Technology.
- Sweet, Henry. 1898. *A new English grammar, logical and historical*. Oxford: Clarendon Press.



- Travis, Lisa DeMena. 1984. *Parameters and effects of word order variation*. MIT Press.
- Valiant, L. G. 1984. A theory of the learnable. *Commun. ACM* 27:1134–1142. URL <http://doi.acm.org/10.1145/1968.1972>.
- Vendler, Zeno. 1961. Order of adjectives. *Transformations and discourse analysis papers, University of Pennsylvania* 31:1–16.
- Vendler, Zeno. 1968. *Adjectives and nominalizations*. 5. Walter De Gruyter Inc.
- Weir, David Jeremy. 1988. Characterizing mildly context-sensitive grammar formalisms. Doctoral Dissertation, University of Pennsylvania.
- Yoshinaka, Ryo. 2008. Identification in the limit of  $k$ ,  $l$ -substitutable context-free languages. In *Grammatical inference: Algorithms and applications*, 266–279. Springer.
- Yoshinaka, Ryo, and Alexander Clark. 2012. Polynomial time learning of some multiple context-free languages with a minimally adequate teacher. In *Formal Grammar*, 192–207. Springer.
- Ziff, Paul. 1960. *Semantic analysis*. Cornell University Press.