Productivity and Reuse in Language

A dissertation presented
by
Timothy J. O’Donnell
to
The Department of Psychology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of
Psychology

Harvard University
Cambridge, Massachusetts
May 2011
Productivity and Reuse in Language

Abstract

A much-celebrated aspect of language is the way in which it allows us to make “infinite use of finite means” (von Humboldt, 1836). This property is made possible because language is fundamentally a productive computational system: Novel expressions can be composed from a large inventory of stored, reusable parts.

For any given language, however, there are many more potential ways of forming novel expressions than can actually be used in practice. For example, English contains suffixes that are highly productive (e.g., -ness; Lady-Gagaseness; pine-scentedness), but also contains suffixes which can only be reused in specific, existing words (e.g., -th; truth, width, warmth). How are such differences in productivity and reusability represented? How can the child acquire this system of knowledge?

This thesis presents a formal model of productivity and reuse which treats the problem as a structure-by-structure inference in a Bayesian framework. The model—Fragment Grammars, a generalization of Adaptor Grammars (Johnson et al., 2007a)—is built around two proposals. The first is that anything that can be computed can be stored. The specific computational mechanism by which this is accomplished, stochastic memoization, is inherited from Adaptor Grammars (Goodman et al., 2008; Johnson et al., 2007a). The second proposal is that any stored item can include subparts which must be computed productively. This is made possible by the computational mechanism of stochastically lazy evaluation, introduced in the thesis.

Throughout the thesis, Fragment Grammars are systematically compared to four other probabilistic models of productivity and reuse which formalize historical proposals from the linguistics and psycholinguistics literatures. The five models are evaluated on two very different sub-systems of English morphology: the English past tense, which is characterized by a sharp dichotomy in productivity between regular (i.e., +ed) and irregular (e.g., sing/sang) forms, and English derivational morphology, which is characterized by a graded cline from very productive (e.g., -ness) to very unproductive (e.g., -th). The thesis examines many aspects of these two domains including: performance on past–tense inflection, past–tense processing phenomena, developmental overregularization, the productivity of derivational suffixes, ordering restrictions on derivational suffixes, and base–driven selectional restrictions on suffix combinations.
## Contents

Title Page ........................................... i
Abstract ......................................... iii
Table of Contents ................................... iv
Acknowledgments ................................... ix
Dedication ......................................... x

1 Introduction ...................................... 1
   1.1 The Problem of Productivity ..................... 1
   1.2 Sketch of the Proposal .......................... 3
      1.2.1 Productivity and Reuse as a Tradeoff: The Learner as Scientist .......... 3
      1.2.2 The Computational Framework ............... 4
   1.3 The Study of Productivity and Reuse .......... 7
      1.3.1 Productivity in Early Generative Theories of Morphology ............. 11
      1.3.2 A Radical Critique: Di Sciullo and Williams (1987) ................. 13
      1.3.3 Converging Evidence from Other Methods: Psycholinguistic Studies of Word Structure .......... 13
      1.3.4 Converging Evidence from Another Domain: Syntax ................... 17
      1.3.5 Productivity and Reuse as an Inference .............. 18
   1.4 The Proposal .................................. 19
      1.4.1 A Model of Linguistic Computation .............. 19
      1.4.2 Partial Computations in PCFGs ................. 21
      1.4.3 Productivity and Reuse as a Tradeoff ............ 22
      1.4.4 Four Approaches to Productivity and Reuse ........... 25
         Full–parsing .................................. 26
         Full-listing .................................. 26
         Exemplar–based Productivity and Reuse ........ 28
         Inference–based Productivity and Reuse .......... 29
   1.5 Scope of the Thesis ......................... 31
      1.5.1 Assumptions about the Starting State of Learning ................. 31
      1.5.2 Context–Free Grammars as Models of Linguistic Computation ....... 34
      1.5.3 Relation to Processing and Neuroscience Theories ............... 34
      1.5.4 Resource Tradeoffs versus Prediction Tradeoffs .............. 35
2 The Framework

2.1 Structured Bayesian Generative Modeling

2.1.1 Bayesian Statistics

Bayes' Rule

2.1.2 Hierarchical Generative Modeling

2.1.3 Structured Models

2.2 Probability Theory, Optimality and Cognitive Science

2.3 The Modeling Framework from a Probabilistic Programming Perspective

2.3.1 Distributions as Programs and Church

2.3.2 Structure Building with unfold

2.3.3 Remembering Earlier Work with Memoization

Stochastic Memoization

Balancing Novelty and Reuse with Pitman–Yor Processes

Properties of the Pitman–Yor Process

Relationship between Pitman–Yor Processes and Activation–Based Models in Psycholinguistics

2.3.4 Stochastic Memoization of unfold: Adaptor Grammars

2.3.5 Lazy and Eager Evaluation Strategies

Stochastic Lazy unfold

2.3.6 Fragment Grammars

2.3.7 A Generalization: fragment-lambda

2.4 The Models Examined in the Thesis and Related Work

2.4.1 Full–Parsing: Multinomial–Dirichlet PCFG (MDPCFG)

2.4.2 Full–Listing: Adaptor Grammars (AG)

2.4.3 Exemplar–Based Productivity and Reuse: Data Oriented Parsing (DOP)

2.4.4 Inference–Based Productivity and Reuse: Fragment Grammars (FG)

Other Inference–Based Models of Reuse

2.4.5 Categorial Refinement Approaches

3 Formalization of the Models and Inference

3.1 Formal Details of the Models

3.1.1 Context–free Grammars

3.1.2 Multinomial Probabilistic Context–Free Grammars

3.1.3 Multinomial–Dirichlet Distributions

3.1.4 Multinomial–Dirichlet Probabilistic Context–Free Grammars

3.1.5 Data Oriented Parsing

PCFG Reductions of DOP Grammars

3.1.6 The Pitman–Yor Process

3.1.7 Adaptor Grammars

3.1.8 Fragment Grammars

3.2 Inference for Fragment Grammar

3.2.1 A Metropolis–Hastings Sampler

3.2.2 The Approximating PCFG

3.2.3 Implementation

3.3 Inference for Other Models

3.4 Computation of Conditional Probabilities

3.5 Practicalities of Inference for FG

3.5.1 Initialization
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5.2</td>
<td>Selective Model Averaging</td>
<td>83</td>
</tr>
<tr>
<td>3.5.3</td>
<td>Type/Token Binning</td>
<td>84</td>
</tr>
<tr>
<td>3.5.4</td>
<td>Conditioning on Constituent Information</td>
<td>84</td>
</tr>
<tr>
<td>3.5.5</td>
<td>Auxiliary Variable Slice Sampling</td>
<td>85</td>
</tr>
<tr>
<td>3.5.6</td>
<td>Hyperparameters</td>
<td>85</td>
</tr>
<tr>
<td>4</td>
<td>The English Past Tense: Abstraction and Competition</td>
<td>86</td>
</tr>
<tr>
<td>4.1</td>
<td>Introduction</td>
<td>86</td>
</tr>
<tr>
<td>4.2</td>
<td>The English Verbal Paradigm</td>
<td>86</td>
</tr>
<tr>
<td>4.3</td>
<td>Empirical Review of the English Past Tense</td>
<td>88</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Transparency and Systematicity</td>
<td>89</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Productivity</td>
<td>89</td>
</tr>
<tr>
<td>4.3.3</td>
<td>Frequency</td>
<td>90</td>
</tr>
<tr>
<td>4.3.4</td>
<td>Processing</td>
<td>91</td>
</tr>
<tr>
<td>4.3.5</td>
<td>Development</td>
<td>92</td>
</tr>
<tr>
<td>4.4</td>
<td>Theories of the Past Tense</td>
<td>93</td>
</tr>
<tr>
<td>4.4.1</td>
<td>The Necessity for Abstract Representations</td>
<td>94</td>
</tr>
<tr>
<td></td>
<td>Rumelhart and McClelland (1986) and Related Connectionist Theories</td>
<td>94</td>
</tr>
<tr>
<td></td>
<td>Feedforward Networks and Overregularization</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Related Approaches which Allow for Abstraction</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>Other Connectionist Approaches</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>Analogical Models</td>
<td>98</td>
</tr>
<tr>
<td>4.4.2</td>
<td>The Necessity for a Quantitative Account of Competition</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>Classical Deterministic Theories</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>The Blocking and Retrieval Failure Hypothesis of Marcus et al. (1992)</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>The Rule–Learning Component in BRF and Overregularization</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>The Dual–Mechanism Model of Taatgen and Anderson (2002)</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td>Probabilistic Rule–based Theories</td>
<td>104</td>
</tr>
<tr>
<td></td>
<td>Other Models of Rule Induction</td>
<td>105</td>
</tr>
<tr>
<td>4.4.3</td>
<td>How to Handle Less Productive Generalizations</td>
<td>107</td>
</tr>
<tr>
<td>4.5</td>
<td>Conclusion</td>
<td>110</td>
</tr>
<tr>
<td>5</td>
<td>The English Past Tense: Simulations</td>
<td>111</td>
</tr>
<tr>
<td>5.1</td>
<td>Introduction</td>
<td>111</td>
</tr>
<tr>
<td>5.2</td>
<td>The Simulations</td>
<td>112</td>
</tr>
<tr>
<td>5.2.1</td>
<td>The Starting CFG and Input Representation</td>
<td>112</td>
</tr>
<tr>
<td>5.2.2</td>
<td>Training</td>
<td>113</td>
</tr>
<tr>
<td>5.3</td>
<td>Adult Data: SWITCHBOARD Simulations</td>
<td>114</td>
</tr>
<tr>
<td>5.3.1</td>
<td>Data Preparation and Simulations</td>
<td>114</td>
</tr>
<tr>
<td>5.3.2</td>
<td>Simulation Results</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td>Overall Performance</td>
<td>115</td>
</tr>
<tr>
<td></td>
<td>Expected Number of Constituents</td>
<td>117</td>
</tr>
<tr>
<td></td>
<td>Whole–form Frequencies and Stem Frequencies</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td>Discussion</td>
<td>123</td>
</tr>
<tr>
<td>5.3.3</td>
<td>Other Analyses</td>
<td>125</td>
</tr>
<tr>
<td></td>
<td>Doublets</td>
<td>125</td>
</tr>
<tr>
<td></td>
<td>Anti–Frequency Effects</td>
<td>127</td>
</tr>
<tr>
<td>5.4</td>
<td>Developmental Data: CHILDES Simulations</td>
<td>129</td>
</tr>
<tr>
<td>5.4.1</td>
<td>Data Preparation and Simulations</td>
<td>129</td>
</tr>
</tbody>
</table>
5.4.2 Results .......................................................... 131
Trajectory of Overregularization Rates ......................... 131
Protection from Overregularization ............................. 132
Early Correct Performance and the Discovery of the Regular Rule 135
Discussion .......................................................... 137
5.5 Conclusion ...................................................... 137

6 English Derivational Morphology: Productivity and Conspiracies 138
6.1 Introduction ..................................................... 138
6.2 Derivational Morphology ..................................... 138
6.3 Empirical and Theoretical Background ...................... 140
  6.3.1 Selectivity .................................................. 141
  6.3.2 Transparency .............................................. 143
  6.3.3 Productivity .............................................. 144
    Measuring Productivity ..................................... 146
      Conditional Probabilities that Measure Productivity 146
      Measures of Productivity from the Literature .......... 147
      Baayen’s Measures of Productivity ...................... 149
  6.3.4 Parsability: Decomposition and Retrieval .............. 152
    Parsability and Frequency: Measures and Models ......... 152
    Information-Theoretic Models of Parsability ............ 154
    Measures of Parsability based on Models of Lexical Processing 154
    The Relative Frequency Hypothesis ....................... 155
    Parsability and Productivity ............................. 156
    Parsability and Transparency .......................... 157
  6.3.5 Ordering ................................................. 158
    Selectional-Restiction Based Theories .................... 159
    Complexity-Based Ordering ............................... 160
    The Mean Rank Statistic .................................. 162
    Complexity-Based Ordering and Selectional Restrictions 164
6.4 Conclusion ..................................................... 164

7 English Derivational Morphology: Simulations 165
7.1 Introduction ..................................................... 165
7.2 The Training Corpus and Simulations ....................... 166
  7.2.1 The Starting CFG and Input Representations .......... 166
  7.2.2 Input Corpus ............................................. 168
    Extending the CELEX Sample ............................... 168
    Problems with the CELEX Morphological Analyses ....... 169
    Hand Correction ............................................ 170
    Forms Included in Training ............................... 170
  7.2.3 Training ................................................... 170
7.3 Simulation Results .......................................... 170
  7.3.1 Productivity and Parsability .......................... 171
    The Distribution of Productivity Scores ................. 171
    Correlation with Productivity Scores from Hay and Baayen (2002) 174
    Examples of Individual Suffixes ........................ 174
    Parsability and the Relative Frequency Hypothesis .... 176
    Discussion of Productivity and Parsability ............ 180
## Fragment Grammars and the Relative Frequency Hypothesis

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3.2 Ordering and Base-Driven Selectional Restriction</td>
<td>182</td>
</tr>
<tr>
<td>Complexity-Based Ordering</td>
<td>182</td>
</tr>
<tr>
<td>Correlation with Mean Ranks from Plag and Baayen (2009)</td>
<td>183</td>
</tr>
<tr>
<td>Correlation with Raw Frequencies from Plag and Baayen (2009)</td>
<td>184</td>
</tr>
<tr>
<td>Base-Driven Selectional Restrictions</td>
<td>185</td>
</tr>
<tr>
<td>Comparison of -ity and -ness</td>
<td>185</td>
</tr>
<tr>
<td>Examples of Other Suffix Combinations</td>
<td>186</td>
</tr>
<tr>
<td>Discussion</td>
<td>188</td>
</tr>
<tr>
<td>Fragment Grammars and Complexity-Based Ordering</td>
<td>189</td>
</tr>
<tr>
<td>Base-Driven Selectional Restrictions</td>
<td>190</td>
</tr>
</tbody>
</table>

## 7.4 Conclusion

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>190</td>
</tr>
</tbody>
</table>

## 8 Conclusion

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Summary of the Thesis</td>
<td>192</td>
</tr>
<tr>
<td>8.2 Productivity, Irregularity, and Frequency</td>
<td>194</td>
</tr>
<tr>
<td>8.2.1 Earlier Accounts</td>
<td>194</td>
</tr>
<tr>
<td>8.2.2 The Prior Probability of Irregularity under Fragment Grammars</td>
<td>195</td>
</tr>
<tr>
<td>8.2.3 Discussion</td>
<td>197</td>
</tr>
<tr>
<td>Fully-Regular Rules</td>
<td>198</td>
</tr>
<tr>
<td>What Counts as Irregularity?</td>
<td>198</td>
</tr>
<tr>
<td>The Structure of Irregulars</td>
<td>199</td>
</tr>
<tr>
<td>Connections to the Cultural Evolution of Language</td>
<td>199</td>
</tr>
<tr>
<td>The Origins of Skewed Word-Frequency Distributions</td>
<td>200</td>
</tr>
<tr>
<td>8.3 The Emergence of Productive Generalizations during Development</td>
<td>200</td>
</tr>
<tr>
<td>8.4 Future Directions</td>
<td>201</td>
</tr>
<tr>
<td>8.5 Conclusion</td>
<td>202</td>
</tr>
</tbody>
</table>

## Bibliography

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A Integrating Phonology</td>
<td>229</td>
</tr>
<tr>
<td>B Past Tense Inflection Classes</td>
<td>230</td>
</tr>
<tr>
<td>C Derivational Suffixes</td>
<td>233</td>
</tr>
</tbody>
</table>
Acknowledgments

All PhD dissertations are the product of the efforts of numerous people besides the one whose name appears on the title page; however, I feel as though I have been unusually lucky to have so many remarkable people help me along the way. First, I would like to thank the members of my committee. Noah Goodman, with whom I developed the Fragment Grammars model, taught me to do Bayesian modeling and has been a source of constant encouragement, intellectual stimulation, and friendship. Mark Johnson, who developed the Adaptor Grammars model, upon which Fragment Grammars is based, generously provided advice about countless aspects of this work (and computational linguistics, in general) without which I would have never been able to implement the model. Josh Tenenbaum shares a large part of the responsibility for any clarity in the presentation of these ideas and their relationship to bigger questions of cognitive science. I continue to be inspired by the research program he has developed and thankful for the home away from home that he has provided at MIT. Steve Pinker suggested that I look at the past tense and asked crucial questions at critical times which—it is no exaggeration to say—have shaped the direction of this entire project. Finally, my advisor, Jesse Snedeker’s intellectual (and editorial) influence can be found on every page of this thesis. Her deep insight, rigor, and thoughtfulness have made the arguments in this thesis tighter, the ideas clearer, and the presentation more coherent. Her generosity, sense of humor, and humanity have made her a great advisor and her lab a wonderful place to work.

A number of colleagues and friends have gone above and beyond the call of duty in helping me with various aspects of this project, for which I am deeply grateful. Manizeh Khan, Josh Hartshorne, and Amy Geojo all read large parts of the thesis, and it is better for it. Dan Roy, Jelle Zuidema, and Andrew Nevins were my go-to experts on all things mathematical, all things DOP, and all things linguistic, respectively. I thank them for patiently answering my more naive or incoherent questions. Miriam Meyerhoff deserves special mention for supporting me in my intellectual pursuits for the years before I was able to get into graduate school and the years after, and for being a friend.

I would like to thank my friends in Snedlab, Cocosci lab, and elsewhere for making the easy parts fun and the hard parts bearable, and my parents and brothers for providing support and encouragement along the way. Other collaborators, friends and family to whom I am grateful for help and encouragement include: Asaf Bachrach, Erin Conwell, John Hale, Frank Jaekel, Roni Katzir, Vikash Mansingkha, Clemens Mayr, Charles O’Donnell, Sam Scarano, Kenny Smith, Mahesh Srinivasan, and Melissa Troyer.

Finally, I would like to thank Marjorie Freedman who tolerated the worst, shared in the best, and read everything twice, all with insight, patience, and good humor. This thesis really would not have been written without her. Thanks, Marj.
Dedicated to the word “coolth” because coolity in not trying.
Chapter 1

Introduction

1.1 The Problem of Productivity

Consider the following words: coolness, circuitousness, grandness, orderliness, pretentiousness, cheapness, and warmth. Most native speakers of English will immediately recognize that all of these words end in the suffix, -ness, and—upon a moment’s reflection—will be able to explain that this suffix carries a meaning like “the abstract quality of being X” where X is some property such as cool or circuitous. They might also observe that this suffix generally attaches to adjectives to form nouns, and that, when it does, it usually does not change the way that the adjective base is pronounced (although it introduces some standard spelling changes). There are a number of exceptions to these generalizations: For example, the word forgiveness ends in -ness, but forgive is not an adjective; Likewise, the word business is historically derived from busy and -ness, but, over time, the pronunciation of the base, busy, and the meaning have changed. Nevertheless, such exceptions are rare; -ness is overwhelmingly regular and transparent.

The suffix -ness has another very important property. It can be used effortlessly to construct new words. Take any adjective—common, rare, or even novel—and a corresponding abstract noun can be formed from it: pine-scented/pine-scentedness, sketchy/sketchiness, Lady Gagaesque/Lady Gagaesqueness. Many new words formed with -ness seem so natural that an English speaker may even have trouble knowing whether he or she has heard the word before. This property of -ness is what linguists and psychologists call productivity—the ease with which a linguistic process gives rise to new forms.

Now, consider another set of words: verticality, tractability, severity, seniority, inanity, and electricity. These words also all share a suffix: -ity. This suffix has much in common with -ness. It attaches to adjectives to form nouns, and its meaning is approximately the same: the abstract quality of possessing the property described by the adjective. However, closer examination reveals that -ity also differs from -ness in a number of important ways. While words such as curiosity, ability, and ethnicity can be synonymous with curiousness, ableness, and ethnicness, they can also mean something that the -ness words cannot. The words ending in -ity can be used to refer to particular specialized instances of the abstract quality which they describe: an albino rhinoceros may inspire curiousness because it is a curiosity; juggling reflects ableness because it is an ability; Italian is an example of ethnicness because it is an ethnicity. Unlike -ness, -ity can also attach to bases that are not words (e.g., hilarity, unity, charity). The suffix -ity also differs from -ness in that it often causes systematic changes in the pronunciation of the base to which it attaches. For example, when it attaches to words which end in -ic, the pronunciation of the final /k/ changes to /s/ (e.g., electric/electricity, historic/historicity). In some words, it causes a vowel in the base to be
Chapter 1: Introduction

pronounced differently (e.g., inane/inanity, sane/sanity).

Most importantly, -ity is far less productive than -ness. It cannot be used to turn arbitrary adjectives into abstract nouns: pine–scented/*pine–scentedity, sketchy/*sketchity, Lady Gagaesque/*Lady Gagaesquity. There are occasional exceptions: For example, an internet commentary exhorts us that “coolity is not trying.”1 However, this use of -ity is clearly intended to be humorous and this is precisely because we know that -ity usually cannot be generalized productively. To complicate matters, however, there are several contexts where -ity can be generalized productively: for example, after the suffixes -able, -ic, and -al. If one uses the term googleable, the associated abstract noun is googleability. The same is true for -ic and -ak; it appears that in certain restricted contexts -ity is fully productive.

Finally, consider the following words: warmth, width, truth, and depth. These words all end in the suffix -th, which also turns adjectives into abstract nouns, and means approximately the same thing as -ity and -ness. However, this suffix is very different from the first two. Most native speakers are not even aware of its existence until it is explicitly pointed out to them. Furthermore, many words ending in -th involve substantial changes to the pronunciation of their base adjectives: hale/health, dead/death, young/youth. In many cases, the changes result in a relationship between the base adjective and the abstract noun that is very obscure: vile/filth, slow/sloth. There are even a few words that appear to end in -th but don’t seem to have any corresponding adjective (at least in modern English): weal/wealth, wroth/wrath, merry/mirth.

The suffix -th is even less productive than -ity. While it may be possible to to interpret *pine–scentedity, *sketchity, and *Lady Gagaesquity; *pine–scent(e)th, *sketchth, and *Lady Gagaesquant are hardly more than gibberish. Nevertheless, -th can be used occasionally to form new words for humorous or poetic effect. George Elliot wrote of the “gleams and greenth of summer” in Daniel Deronda; Steve Allen wrote a book called Dumbth: The Lost Art of Thinking With 101 Ways to Reason Better & Improve Your Mind;2 and on the internet we are told: “Many enjoy the warmth, Vikings prefer the Coolth.”3

Patterns like -ness/-ity/-th—three ways of forming abstract nouns from adjectives that mean approximately the same thing, but differ greatly in their productivity and regularity—and many similar cases throughout English and other languages—raise several serious problems for linguistic and psycholinguistic theory.

First, what kind of representations can account for this mixture of productivity and reuse? What is it that speakers know that allows them to recognize that coolth is intended to be humorous while pine–scentedness is a perfectly well-formed word?

Second, how does the language learner arrive at this knowledge? The affix -th was productive at some point in the history of English. There is nothing cross-linguistically unusual about it: other languages have phonologically and semantically similar affixes which are used productively. Furthermore, -th appears in dozens of English words. How and why does the child learn that it (usually) cannot be used to form new words despite these facts? By contrast, how and why do they learn that -ness can? What sorts of evidence are available and how are they used?

Third, why is -ness so much more phonologically and semantically regular than -ity, and -ity so much more so than -th? More generally, why is there a correlation between phonological and semantic transparency and productivity in all human languages?


2Thanks to Steve Pinker for the example.

3http://www.coolth.ca/
Finally, why do several ways of expressing (roughly) the same meaning exist in English in the first place? How do speakers resolve competition between these alternate means of expression? Why does English continue to maintain multiple possibilities?

Together, these questions illustrate the problem of productivity: In this thesis, I propose a model which provides a partial solution to this problem. The model treats the problem of productivity as the target of inference in a Bayesian framework. Situating this question in a Bayesian framework allows us to formulate the problem in terms of probabilistic inference, and adopt the rational analysis methodology of Anderson (1990)—asking how an optimal agent would make use of patterns in the linguistic input to learn the distribution of productive processes and reusable forms in their language. In the next section, before examining the historical lines of research which have led to these proposals, I give an intuitive sketch of the main ideas behind the formal model.

1.2 Sketch of the Proposal

The model presented in this thesis, Fragment Grammars, is a generalization of a model known as Adaptor Grammars (Johnson et al., 2007a). Adaptor Grammars provide a probabilistic formalization of the idea that anything that can be computed can be stored. Fragment Grammars generalize Adaptor Grammars by allowing productivity and abstraction to occur at arbitrary points within individual stored structures. In Chapters 2 and 3, I will describe both of these models (and the other models which I outline below) from a variety of formal perspectives.

These models can be understood as special cases of a more general formal framework which, in turn, is built on two fundamental ideas. The first is a formal proposal for how storage can be integrated with (probabilistic) generative models in an elegant and straightforward way—thus simplifying traditional linguistic and psycholinguistic accounts of the interaction of memory and grammar. The second idea is that productivity and reuse should be treated as targets of inferences in a Bayesian framework. In this section, I provide a high–level, intuitive overview of these two ideas.

1.2.1 Productivity and Reuse as a Tradeoff: The Learner as Scientist

Consider two structure–building processes, such as the affixation of -ness and -th to an adjective. How is the learner to determine that of one these is productive and can be used to form new words, while the other is unproductive and can only be found in a (relatively) small set of existing words? What kinds of evidence are available?

The proposal advocated in this thesis is extremely simple: the only (distributional) evidence that counts in favor of the hypothesis that a word–formation process is productive (i.e., that it is like -ness) is the frequency with which it is used to form novel words; the only (distributional) evidence that counts in favor of the hypothesis that a word–formation process can only be reused in a restricted set of words (e.g., that it is like -th) is the repetition of those words.\footnote{As I will discuss more in later chapters, other kinds of evidence, such as semantic of phonological regularity, can play a role.}

The first kind of evidence—use in novel words—is the criterion used by language scientists to prove the productivity of word–formation processes. Confronted with the question of whether -ness or -th is productive, the linguist or psychologist will see whether they can construct an acceptable new word using that suffix: Do speakers judge pine–scentedness or to be acceptable? Thus, the first half of the proposal is just that the child learner is like the language scientist. The child learns that certain grammatical operations are productive by observing their productive use.
Chapter 1: Introduction

Note that there is a logically necessary tradeoff between novelty and reuse. A novel word cannot be a repetition and vice versa. Every word is novel exactly once and, therefore, each instance of a novel word using -ness or -ity cannot count towards the reusability of that affix in some specific previously observed word. Likewise, the repetition of some previously encountered word using one of these affixes cannot count as evidence that the affix generalizes productively. As I will show in subsequent chapters, this somewhat obvious observation has surprisingly deep consequences. In particular, it means that the learner must attend to the entire pattern of productivity and reuse over all forms to infer whether any particular form is actually productive.

Although the formal details of the models which I describe below are complex, the intuitions underlying most of the results in this thesis rest on the simple assumptions about evidence for productivity and reuse just outlined. For intuition, the reader should lean heavily on the idea that productivity is evidenced by novelty, while reuse is evidenced by repetition.

1.2.2 The Computational Framework

In the preceding section, I discussed the kinds of evidence that the Bayesian system presented in this thesis uses to infer productivity and reuse. In this section, I give an intuitive overview of how differences in the productivity and reusability of different structure-building processes can be represented by a (probabilistic) computational system. In order to illustrate these ideas, I begin with an overview of the modeling idiom adopted in the thesis.

In this dissertation, I adopt a probabilistic generative modeling framework, implementing generative models with programs for a natively probabilistic model of computation (i.e., a Probabilistic Programming perspective on cognitive modeling; see the next chapter and Erwig and Kollmannsberger, 2006; Goodman et al., 2008; Kiselyov and Shan, 2009; Mansinghka, 2009; Pfeffer, 2001; Roy, 2011). A generative model is a model that explicitly captures knowledge about the causal processes which give rise to the data observed in the world. Data, however, can be uncertain for two reasons. First, observations are often noisy. Second, and more importantly, there are usually many causal chains that could have led to the same observations. The Bayesian approach uses probability theory to quantify and reason about this uncertainty. In a Bayesian setting, probabilities are interpreted as degrees of belief about the origins of observations and the processes which give rise to them.

An example of a probabilistic generative model is shown in Figure 1.1. This figure shows a very simple generative process consisting of three consecutive coin flips. The diagrams in the top part of the figure show the possible outcomes of a sequence of three coin flips. The resulting distribution over observable sequences is shown in the lower half of the figure. Even in this simple model, there are 8 possible ways that an observation can be generated.

A generative model can be formalized as a program. Programs are defined in terms of functions or procedures which map from inputs to outputs. Probabilistic generative processes can be expressed as stochastic procedures. Whereas a deterministic procedure maps each input to a single output value, a stochastic procedure maps each input to a distribution over output values. Alternatively but equivalently, a stochastic procedure can be thought of as sampling outputs from a distribution over values given the inputs. Figure 1.2 shows a possible procedure implementing the generative process shown in Figure 1.1, in the Church language. The details of procedures such as this will be described in Section 2.3. Figure 1.2 is intended merely to give a broad sense of how.

---

5I would like to thank Vikash Mansinghka and Beau Cronin for allowing me to base these figures on one of their own.

6Careful usage distinguishes between functions which are abstract mathematical mappings between sets, and procedures which are particular algorithms for computing such mappings. However, I will use these terms interchangeably.
Figure 1.1: **Probabilistic Generative Model**: This figure shows a simple generative model consisting of three coin flips. The upper portion of the figure shows the different possible ways in which the three flips can be generated. The bottom half shows the resulting distribution over sequences of coin flips.

Generative processes can be expressed as programs.\(^7\)

Generative processes can be combined with data via *conditioning*. Conditioning is an operation that consists of two steps. First, all possible causal chains that are inconsistent with the data are thrown out. Second, the strengths of the remaining possibilities are readjusted so that the probability distribution again sums to 1. Conditioning is shown in Figure 1.3. The lower half of this figure shows a number of samples which have been drawn from the generative model. In the upper half, I have highlighted the computations that could have given rise to the observed data points.

The fundamental idea behind the Fragment Grammar model is that probabilistic generative processes can be extended by allowing for the storage and reuse of useful computations and subcomputations. This is illustrated in 1.4 which shows the same observations and causal paths as Figure

---

\(^7\)Church is *universal* in the space of probabilistic generative models; that is, any generative model which is computable can be expressed in Church in a relevant formal sense (Roy, 2011).
(define flip3
  (lambda ()
    (list (flip) (flip) (flip))))

Figure 1.2: Simple Generative Model: A Church procedure implementing the generative process shown in Figure 1.1.

1.3, but emphasizes two particular aspects of the system in red: First, it highlights a frequently reoccurring pattern in the data: Many of the observations share the sequence $HH$ as the outcomes of the last two coin flips. Second, it highlights (also in red) the subsequences of computations which give rise to this common pattern.

The model described in this dissertation provides a mechanism which allows these shared subsequences of computation to be stored and reused. Because these subcomputations are stored, they can be treated as single units—atomic random choices. This, in turn, changes the generative model, placing more probability on certain outcomes—the probability distribution is concentrated on those outcomes.

Figure 1.5 shows the resulting generative model after the common subcomputations have been stored and added as atomic choices. Now, after the first coin flip, the process has the option of choosing to reuse the subcomputation which samples two heads in a single step. However, by making this choice, it commits to a sequence ending in two heads. By adding this additional possibility, the probability of producing an outcome that ends in two heads is increased, and the probability of other possibilities is reduced. The resulting generative model is more complex than the original because there are now more possible causal paths which can generate data. However, although there are more choices, the actual uncertainty of the model has been reduced.

The ability to store commonly reused subcomputations amounts to the ability to make choices once, ahead of time, and then cheaply reuse those choices later. Since the observer believes that observations will continue to frequently end in two heads, it makes sense to automatize the subcomputations which lead to this state of affairs.

In Figure 1.3, although the many datapoints ended in two heads, the data still exhibited a certain amount of variability. For example, the first flip varied between heads and tails, and, occasionally, the final two flips varied. By preserving some of the productivity in the starting system, the concentrated generative model in 1.5 maintains the possibility of variability.

Thus, the question of balancing productivity and reuse can be reframed as a question of determining what (probabilistic) work can be done ahead of time—by storing frequently reused computations—and what work must be delayed until it is actually needed—thus preserving necessary variability and the potential to generate novel observations.

The generative processes underlying linguistic structures are a great deal more complex than the simple example discussed above. In particular, they often generate infinite distributions of structured objects such as trees or graphs. In the next chapter, I show how this proposal can be adapted to these more complex models. However, the intuition underlying these models is present even in this simple example. Linguistic expressions—such as words—are built by a probabilistic computational system. Frequent subsequences of computation—such as the combination of the stem true with the affix -th (i.e., the word truth)—can be compiled and stored so that they can produced or comprehended in a single step. By contrast, in contexts where the system expects novelty and variability in the future—such as the set of potential forms ending in -ness—the productive potential of the underlying computational system can be preserved and reinforced. Importantly, this approach
allow any subsequence of computations to be stored, including sequences which include the possibility of productive subcomputations.

In the next section, I turn to the history of models of productivity and reuse in generative linguistics and psycholinguistics, focusing on a number of alternative accounts, and why they ultimately fail.

### 1.3 The Study of Productivity and Reuse

The creative potential of natural language has been recognized and celebrated for centuries. The term *productivity*, in the current sense, appears to be due to the German grammarian Diez writing in the middle of the 19th century (Diez, 1838; Schultink, 1992), but several authors have traced an awareness of the concept back to the ancient Sanskrit grammarians (see Bauer, 2001; Schultink, 1992, for detailed discussions of the history of the concept). Similarly, Chomsky (1966) famously (and controversially) traced this focus back to enlightenment thinkers in the Cartesian tradition, highlighting, in particular, Wilhelm von Humboldt’s description of the linguistic system
Chapter 1: Introduction

Figure 1.4: Shared Subcomputations: This figure highlights a common pattern in the data in red: the final two flips resulting in heads. The figure also highlights (red) the common subcomputations that give rise to this pattern.

as making “infinite use of finite means” (von Humboldt, 1836).

However, linguistic productivity resisted formal treatment until the second half of the 20th century. During the first half of this century, concerns about the foundations of mathematics led a number of researchers such as Turing (1937), Church (1932, 1933), Post (1943, 1944), and others to develop formal systems which (with hindsight) laid the foundations for a general theory of computation. By the 1950s, several researchers had begun to adapt these tools to provide formal accounts of linguistic structure—and, especially, lawful explanations of linguistic productivity (e.g., Bar-Hillel, 1953a,b; Chomsky, 1956a; Hockett, 1955; Lambek, 1958).
Chapter 1: Introduction

Figure 1.5: Generative Model with Common Subsequences added as Primitive Random Choices: This figure shows the generative model after common shared subsequences have been added as atomic choices. The effect is to increase the probability of outcomes which include those subcomputations, and to decrease the probability of the other choices. The subcomputations which can now be reused as atomic choices, and the corresponding increased probabilities are highlighted in red.

These systems varied in their details, but demonstrated how a set of primitive elements together with a set of (recursive) structure-building operations could generate an unbounded number of linguistic structures; explaining, in principle, how a finite system can give rise to new, productive computations. However, unlike the mathematical problems which inspired research into the theory of computation, natural language is riddled with exceptions and irregularities, generalizations which apply only in restricted contexts, and structure-building processes which appear to differ in their degree of productivity. Classical models of computation were not developed to provide a natural account of how productivity could be controlled or constrained.

In this thesis, I argue that the problem of controlling the productivity of a generative system should be treated as an inference problem. However, there are two alternate perspectives on this problem which have (often implicitly) been behind much work in linguistics and psycholinguistics. The first is a view which can be called the representational approach to productivity and reuse:
**Representational Approach:** Differences in productivity and reuse between different linguistic structures or processes can be predicted from independently definable representational properties of those structures or processes.

Although every theory of linguistic structure must provide mechanisms for both storage and computation, the crucial idea behind the representational approach is that storage and computation can be predicted from other, independent properties of linguistic representations. For example, *words* can be defined using various phonological or syntactic properties by different linguistic theories (see Di Sciullo and Williams, 1987, for an extended discussion). Under one theory, a word might be defined as the domain under which certain phonological operations (e.g., vowel harmony) apply. Under another theory, a word might be defined as the minimal unit visible to syntactic operations (e.g., binding or movement). A representational account of productivity and reuse might hypothesize that storage coincides precisely with one of these definitions of words.

The representational approach to productivity originates from a simple empirical observation: certain kinds of linguistic structures tend to be produced productively and others tend to be stored. Natural language structures can be organized into a hierarchy of constituent types, where each level higher in the hierarchy is built using pieces from lower levels. One possibility for such an organization is illustrated in Figure 1.6. The arrows in this figure represent *consist--of* relations: For example, in the figure, words *consist--of* morphemes and other words. Note that this diagram is meant to be illustrative and does not necessarily correspond to any specific theoretical proposal.

![Hierarchy of Linguistic Constituents](image)

**Figure 1.6: Hierarchy of Linguistic Constituents:** This figure shows one possible hierarchy of linguistic constituents.

Higher levels in this hierarchy tend to demonstrate greater levels of productivity and computation while lower levels tend to demonstrate more idiosyncrasy and storage. For example, most words tend to be stored, while most syntactic phrases tend to be computed compositionally. A representational theory can provide an explanation for this fact: only and all (syntactic or phonological) words are stored; everything else is computed.
Representational theories of productivity and reuse have another potential advantage. If productive and non–productive processes can be segregated according to some independent criterion, then learning which processes are productive and which are not may be easier. For example, once learners identify the domain under which vowel harmony operates in their language, then, if only phonological words are stored, they have solved the learning problem.

A second approach which has been frequently invoked in linguistic and psycholinguistic theories is the Bloomfieldian approach (Bloomfield, 1933). This approach holds that only idiosyncratic structures are stored; all structures that are computable by regular rules are productive. Like the representational approach, Bloomfieldian theories are also based on an important empirical observation: purely idiosyncratic structures must be stored, while novel structures—produced by productive rules—can not be stored prior to the first time they were produced.

Note that these two approaches—representational and Bloomfieldian—are not mutually exclusive. There is a possible world in which only linguistic forms of a certain type—for example, words—are stored, and all words are idiosyncratic. Both ideas have played a large role in attempts to provide accounts of word structure, apart and together. The intuitions behind these attempts, and their ultimate failure, are best illustrated by example. In the next several sections, I illustrate these ideas with a brief historical overview of early models of morphology in generative grammar.

1.3.1 Productivity in Early Generative Theories of Morphology

Morphology was relatively neglected in early generative theory (Spencer, 1991), but became a focus of research in the wake of Chomsky’s influential paper Remarks on Nominalization (Chomsky, 1970). This paper concerns morphological processes which turn verbs into nouns (e.g., destroy → destroying/destruction). Earlier versions of generative theory had proposed that such nominalizations were the result of syntactic transformations whereby an underlying clause, such as Ghengis Khan destroyed the city, was transformed into a noun phrase such as Ghengis Khan’s destruction of the city or Ghengis Khan’s destroying the city. The transformational account of such structures allowed these theories to explain facts such as why the selectional restrictions of the underlying verb are preserved after nominalization.

In Remarks, however, Chomsky observed that there are important differences between between regular gerundive nominals (e.g., destroying) and derived nominals (e.g., destruction). In particular, the former can be freely constructed for any verb, enter into standard syntactic transformations, and are involved in predictable semantic relations, while the latter have a restricted distribution and display numerous idiosyncratic properties. He concluded that nominalization should not be handled by the transformational component of grammar, and went on to present an alternate theory which explained the commonalities between verbs and their nominalizations.

Remarks was widely interpreted as an argument that word formation processes and syntactic processes should take place in separate components of the grammar and be handled by different kinds of rules. This theoretical move became known as Lexicalism. This, in turn, stimulated the development of theories of this separate, lexical component of grammar.

An early, agenda–setting study, Halle (1973), proposed that morphology should be understood as the theory of possible words. Halle argued that word formation should be accounted for by a productive system in order to handle novel formations. However, as he pointed out, this

---

9 As Marantz (1997) points out, very little of Chomsky (1970) concerns arguments against the transformational analysis of nominalizations, and Chomsky presents no theory of (idiosyncratic and irregular) derived nominal forms. Instead, Chomsky’s main purpose is to provide a theory that explains the commonalities between clauses and noun phrases in the absence of a transformational account. Nevertheless—whether or not it was the intention of the original paper—Remarks has been widely understood as promoting the need for a separate lexical module to account for word structure.
Chapter 1: Introduction

raises the serious problem of the large number of possible, but non–occurring words. To deal with this problem, Halle proposed that non-occurring words are marked with a feature that prevents their insertion into syntactic structures. Although Halle did not provide an account of how such non–insertion features can be learned, he does note that it is possible that productive rules are not always used for existing complex words—rather existing words may be redundantly stored by the language user. Halle’s discussion highlights the fundamental tension in accounting for morphological structure: A generative system is needed to account for the ability to coin new words; however, this generative system inevitably licenses many impossible words and, therefore, mechanisms must be introduced to control its productivity.

Another influential early paper, Jackendoff (1975), more explicitly focuses on this question of storage, offering explicit arguments for a full–listing theory where all word forms are stored.\(^{10}\) For Jackendoff (1975), shared structure between word forms is accounted for via lexical redundancy rules (LRRs). These are rules which express the generalizations inherent in stored lexical items but differ from the productive rules of syntax. Unlike the latter, they always store their output; furthermore, they are used during learning, and in cases of analogy (e.g., coolth by analogy with warmth), but do not necessarily function during normal language use. Jackendoff also suggests that it may be necessary to posit such rules to account for commonalities between idiosyncratic syntactic structures—such as idioms like kick the bucket—and regular syntactic structures.

For present purposes, the most important aspect of the analyses in Jackendoff (1975) is the proposal that a certain kind of rule, the LRR, always stores its output, and that this is an inherent feature of being part of the lexicon. Jackendoff’s arguments explicitly link the concept of storage and the (non)productivity of derived word forms. However, as was the case with Halle’s theory, this again raises a serious learning problem (as Jackendoff himself later observed. Jackendoff, 2002): how can the learner know which forms are attributable to one kind of rule or another?

Although the problem of controlling the variable productivity of different kinds of word–formation rules was inherent in these earlier works, it was Aronoff (1976) who highlighted it most clearly and brought it to the forefront of morphological research. Aronoff (1976) discusses the disparity between formations in productive -ness and unproductive -ity and reaches the conclusion that -ity forms are stored in the lexicon, while -ness forms are not. Whereas for Jackendoff all rules in the lexicon (however defined) store their outputs, for Aronoff, even this component of grammar contains a mixture of storage and productive computation.

There are several points to take from this discussion. First, the move towards lexicalism in generative grammar was motivated, at least in part, by differences in productivity and regularity between syntactic structure–building processes on one hand and word formation processes on the other. This aspect of lexicalism is an example of a representational approach because holds that storage is a property which is associated with a certain kind of linguistic representation—words—while productivity is a property of syntax. This representational perspective continued (and continues) to have currency in much research subsequent to these early proposals. For example, Fabb (1984) explicitly argues that productivity should be taken as criterial for whether a structure is syntactic or lexical.\(^{11}\)

However, this discussion also emphasizes that the hypothesis that productivity can be accounted for by sorting linguistic structures into separate modules of grammar is not entirely satisfactory. Aronoff (1976) highlights the difficulty of completely purging productive rules from

---

10Jackendoff (1975) uses the term full–entry theory rather than full–listing theory.

11This representational perspective also lies behind many debates over the proper place of inflectional morphology in grammar (i.e., in the syntax or in the lexicon). For example, Anderson (1982) (syntax) versus Kiparsky (1982b); Williams (1981) (lexicon).
Chapter 1: Introduction

the lexical component of grammar, and Jackendoff (1975) points out that storage is necessary even for some large syntactic units. Nevertheless, one important idea which emerged from these early works is that there is a close relationship between productivity and whether grammatical rules store their output: the outputs of more productive processes tend not to be stored, while the outputs of less productive processes tend to be stored. I now turn to another proposal about productivity and reuse, which—driven by these considerations—completely rejects the representational approach.

1.3.2 A Radical Critique: Di Sciullo and Williams (1987)

Driven by the problems discussed above, Di Sciullo and Williams (1987, p. 2) offer the following remarks.

The distinction between words and phrases is a distinction in the theory of grammar. The listed/unlisted distinction has nothing to do with grammar. Syntax and morphology are both recursive definitions of sets of objects—but of different sets, with different atoms and different rules of combination. These are the only differences the grammarian need acknowledge.

The most immediate consequence of this view is that productivity and listedness are not grammatical concepts. [...] Di Sciullo and Williams (1987) give a number of arguments in favor of this perspective. They observe that the differences in productivity and storage between words and phrases are differences in proportion rather than kind. Even in the productive realm of syntax, there are structures, such as idioms, performatives, verb–particle combinations, etc., which must be stored. Conversely, in the idiosyncratic realm of morphology, there are fully productive word–formation processes, for example, the process whereby adverbs are formed by the addition of -ly. Furthermore, Di Sciullo and Williams (1987) point out that competition between productive and idiosyncratic structure–building processes crosses the word/phrase boundary. Based on these observations, Di Sciullo and Williams (1987) conclude that storage can vary from form to form, cutting across levels of linguistic structure and that productivity can vary from rule to rule, also cutting across different rule types. 12

Although Di Sciullo and Williams (1987) provide convincing arguments against the representational approach, the alternate theory of productivity and reuse which they advocate is Bloomfieldian: They argue that the property of listedness coincides with idiosyncrasy. In the following section, I will argue (following others) that, like the representational approach, this Bloomfieldian position is also empirically untenable.

1.3.3 Converging Evidence from Other Methods: Psycholinguistic Studies of Word Structure

Di Sciullo and Williams (1987) propose that productivity and reuse are properties which can vary independently across different linguistic modules, structures, and rules. Since the early 1970s, a long sequence of psycholinguistic studies of word processing have supported this view. However, these studies also provide evidence—contra Di Sciullo and Williams (1987)—that a Bloomfieldian

12Although Di Sciullo and Williams (1987) reject the representational approach to productivity and reuse, they explicitly argue in favor of lexicalism. The thrust of their argument is that while productivity and reuse cannot be used as evidence for, or against, a separate lexical module of grammar, other evidence does support a lexicalist view. Note that the issue of the number of generative components underlying language is also orthogonal to the work in this thesis.
theory must also be rejected. In this section, I give a brief overview of this work (see Chapters 4 and 5 for a more detailed discussion).

The first studies which I will review examine the question of whether words are retrieved whole from memory or composed on the fly during lexical access: they are studies of word parsability. The most common paradigm for asking such questions has been the lexical decision experiment. In such experiments, subjects are asked to classify strings of letters (or, occasionally, sequences of sounds) as words or non–words, and their response time is measured. Decomposition versus retrieval is assessed in one of two ways. First, some studies vary the degree of morphological overlap between the target items (words or non–words) and existing words. For example, Taft and Forster (1975) compared how long it took subjects to reject real stems (e.g., *juvenate* derived from *rejuvenate*) with how long it took them to reject pseudo stems versus (e.g., *luvenate* which does not occur in an existing English word). They found that the former took longer to reject, suggesting that the stem *juvenate* is represented independently of the prefix *re–*. Many similar experiments have found that shared substructure affects processing time, thus providing evidence for decomposition (e.g., Andrews, 1986; Caramazza et al., 1988; Taft and Forster, 1975, 1976). However, for existing words, the effect of shared structure is mediated by the frequency of the form (Colé et al., 1997); shared structure affects the processing time of low frequency words more than it affects the processing time of high frequency words.

The second approach to studying parsability in lexical decision paradigms exploits the fact that people respond more quickly to items that are more frequent. Such frequency effects are observed in wide variety of cognitive domains and at multiple levels of linguistic analysis (see Sections 4.3.3 and 6.3.4 for more detailed reviews), and are assumed to reflect some property of storage whereby more frequent stored representations are easier to access. A number of such studies have found whole–form frequency effects in morphologically complex words, strongly suggesting that (some of) these words are stored as wholes. Importantly, such whole–form frequency effects are often present even for words formed by perfectly regular morphological processes (Bertram et al., 2000b,c; Manelis and Tharp, 1977; Rubenstein et al., 1970; Sereno and Jongman, 1997), seriously calling into question a Bloomfieldian account where only idiosyncratic structures are stored.

However, frequency effect data also often provides evidence for decomposition. For example, in some studies the frequency of bases (e.g., *agree* in *agreeable*) predicts processing latency, even when whole–form frequencies are controlled. In other studies, whole–form frequency effects are present for high frequency words while base frequency effects are present for lower frequency words (see e.g., Alegre and Gordon, 1999a; Baayen et al., 1997b, 2003; Bradley, 1980; Burani and Caramazza, 1987; Burani et al., 1984; Colé et al., 1989; Gordon and Alegre, 1999; Moscoso del Prado Martín et al., 2005; Taft, 1979; Vannest and Boland, 1999).

Evidence for both decomposition and retrieval has also been found using primed lexical decision. In primed lexical decision experiments, the lexical decision task described above is manipulated by the inclusion of prime trials. Prime items are manipulated to share different amounts of structure with target items. For example, if the target item is *agreeability*, a prime trial might consist of a word that shared the same stem (e.g., *agreement*); a word that shared a suffix, or combination of suffixes (e.g., *curiosity*/drinkability*/watchable*); the word itself (known as identity priming); or a word that shared no structure with the target. If any of these primes facilitates the recognition of the target, this can be taken as evidence that whatever is shared between the prime and target is represented independently in the lexicon.

Some studies have found evidence for identity priming (e.g., Katz et al., 1991)—suggesting whole–form retrieval during processing. Other studies have found evidence that identity priming and priming by shared structures, such as bases, are equally strong—suggesting decomposition (Forster and Azuma, 2000; Fowler et al., 1985). In some studies, both kinds of priming are discovered, but identity priming is larger in magnitude than priming by shared structures (e.g., Stanners et al.,
1979b). Furthermore, the difference in the size of the priming effect can be modulated by other factors. For instance, more transparent complex words are primed by shared structures to a greater degree than less transparent complex words (Kielar et al., 2008; Marslen-Wilson et al., 1994; Sandra, 1990; Stanners et al., 1979a). Priming is also modulated by frequency: Higher frequency items show smaller effects of priming by shared structures than lower frequency items (de Vaan et al., 2007; Meunier and Segui, 1999). I will explore these mediating factors in greater detail in later chapters. For present purposes, these results emphasize that lexical processing is characterized by a complex mixture of decomposition and storage.

A final method used to assess decomposition and retrieval in the literature is speeded naming. Words are presented visually to participants who are required to say the word (or a morphologically related form of the word) while response latencies are tracked. Several speeded naming experiments have discovered that processing latency and whole-form frequency are correlated only for words formed by irregular word-formation processes; regular word-formation processes show no effect (Beck, 1997; Clahsen et al., 2004; Prasada et al., 1990). These results again emphasize that lexical processing involves a mixture of retrieval from memory and real-time composition.

The studies just discussed have focused on parsability; another line of research has sought to directly examine the ability of particular morphological processes to generalize to novel forms, that is, it has probed the productivity of morphological processes. A number of studies have used judgment or rating tasks, showing that there is variation in productivity between different affixation processes, and that this variation is correlated with regularity—more productive processes are also more regular (Alegre and Gordon, 1999b; Aronoff and Schaneveldt, 1978; Cutler, 1980; Hay, 2001; Huang and Pinker, 2010; Kim et al., 1991; Marcus et al., 1995; Prasada and Pinker, 1993; Wheeler and Schumsky, 1980).13 A number of experiments using an alternate task, which involves the free production of words using a given affix, have found that the number of novel words that speakers produce correlates with the productivity of the affixes (Anshen and Aronoff, 1988; Baayen, 1994).14 Some affixes generalize freely; some generalize not at all; and some fall in between.

A final set of studies has employed a more controlled production task: the wug test. Here speakers are given a novel form, such as a novel verb stem, and asked to produce an inflected variant of that form, such as the past tense. These most famous of these studies, Berko (1958), showed that children robustly generalize English inflectional morphology (as well as some derivational affixes). Other studies have generally replicated the pattern discussed above: Morphological processes vary in their productivity from completely generalizable, to completely frozen into existing forms (Albright and Hayes, 2003; Bybee and Moder, 1983; Dąbrowska, 2004, 2008; Köpcke, 1988; Ramscar, 2002).

To summarize, psycholinguistic evidence broadly supports the arguments of Di Sciullo and Williams (1987) that productivity and reuse are independent of other aspects of linguistic representations and that can vary from word to word and rule to rule. Lexical processing is a complex mixture of compositional and retrieval, and productivity varies across different affixes. There is one point, however, where the arguments of Di Sciullo and Williams (1987) are not supported: Processing studies have found evidence that even non-idiomatic structure can sometimes be stored, thereby

---

13 These studies have varied more than lexical decision experiments in both methods and hypotheses. The question of morphological productivity in derivational morphology is investigated in Aronoff and Schaneveldt (1978); Cutler (1980). These studies ask for yes/no judgments about possible and/or existing words. Wheeler and Schumsky (1980) ask specifically about the presence or absence of morpheme boundaries. Alegre and Gordon (1999b); Hay (2001); Huang and Pinker (2010); Kim et al. (1991); Marcus et al. (1995); Prasada and Pinker (1993) investigate the acceptability of forms created by inflectional and derivational processes in English and German. Huang and Pinker (2010); Kim et al. (1991); Marcus et al. (1995) specifically argue for the importance of morphological (as opposed to purely phonological and semantic) structure to account for patterns of generalization in language.

14 Note that in such studies novelty is defined with respect to some reference corpus or dictionary. Any words produced by the speaker which are not in the reference data set are considered novel.
making a Bloomfieldian account untenable.

Conspiracies

In the preceding section, my goal was to establish that productivity and reuse are properties that cut across different levels of linguistic structure and different kinds of representations, and, furthermore, that they cannot be identified with (non)idiosyncrasy in the Bloomfieldian sense. However, in the introduction, I pointed out that, in addition to differences in productivity, -ness, -ity, and -th also exhibit differences in phonological and semantic transparency. In the last section, I also mentioned that in several cases productivity, decomposition, and retrieval were mediated by factors such as frequency and regularity. These are all instances of a very general pattern which will be encountered throughout the thesis. Morphological structure exhibits a strong correlation between productivity, semantic and phonological transparency, and low (average) token frequency on one hand, and between lack of productivity, semantic and phonological irregularity, and high token frequency, on the other. In other words, productive affixes are more transparent and tend to occur in a greater variety of lower frequency forms; low productivity affixes are less transparent and tend to occur in a smaller number of higher frequency forms.

The correlation between these properties is an example of what might be called a learning conspiracy—several different kinds of evidence available to the language learner all conspire to make similar, redundant predictions about the productivity of word formation processes. I will review more evidence for such conspiracies in Chapters 4 and 6. In Chapter 8, I will provide the first steps of a theoretical account that can explain them.

Psycholinguistic Theories

In general, psychologists have developed their own theories of word representation and processing which are distinct from linguistic theories. Like in linguistics, some researchers have advocated a full–listing viewpoint, where all word forms are stored as wholes (e.g., Butterworth, 1983), or full–parsing viewpoint, where lexical processing is obligatorily decompositional (e.g., Taft, 1988). However, most researchers have converged on a so–called dual–route approach which allows for a mixture of decomposition and whole–form retrieval (e.g., Alegre and Gordon, 1999b; Baayen and Schreuder, 1999, 2000; Bybee, 1995a; Caramazza et al., 1988; Chialant and Caramazza, 1995; Clahsen, 1999; Frauenfelder and Schreuder, 1992; Hay, 2001, 2003; Pinker, 1991, 1999; Schreuder and Baayen, 1995). Although these models vary greatly in specifics, they share a commitment to the ideas that individual forms can vary in whether they are retrieved from memory or composed on the fly (see Chapters 4 and 6, for a review a large number of specific proposals).

15 Full–parsing models, however, often have levels of representation—for example, modality–specific access representations—in which words are represented as wholes. Very few researchers have proposed models in which all levels of representation require complete decomposition.

16 In this literature the terms dual–route and dual–mechanism are often used interchangeably for two different ideas. Sometimes the terms are used to refer specifically to the theory of Pinker and colleagues which post specific computational mechanisms underlying composition and retrieval—in particular, a traditional linguistic rule component for the former and an associative memory for the latter (see Chapter 4 for a detailed discussion). I will use the term dual–route to refer to the more general sense—any theory which permits arbitrary mixtures of composition and storage—and dual–mechanism to refer to the specific sense.
1.3.4 Converging Evidence from Another Domain: Syntax

My discussion so far has focused on the need for a theory that allows arbitrary mixtures of productivity and reuse in morphology. Converging evidence from syntax suggests that such a theory is necessary in that domain as well. The grammatical system must be able to store linguistic units larger than individual words in order to account for syntactic non-compositionality like that exhibited by idioms (e.g., *kick the bucket*). Many accounts of idiom structure have tried to preserve some version of the representational approach. For example, Emonds (1969) and Chomsky (1981) proposed that idioms like *kick the bucket* are not verb phrases, but rather, that they are a special kind of complex verb. This theoretical move preserves the distinction between a fully compositional syntactic component and a lexicon which is the repository for stored, idiosyncratic items. However, as Di Sciullo and Williams (1987) observe, these accounts suffer from the fact that they require a potential replication of theoretical machinery. *Kick the bucket* behaves like a fully compositional verb phrase in many respects: For example, it can be inflected for tense (e.g., *kicked the bucket, will kick the bucket*). If such an item is just a special kind of verb—no different from other simple verbs such as *eat* or *walk*—then accounting for the ways in which it continues to behave like a compositional syntactic phrase requires replication of the rules of syntax in the lexicon. Moreover, such an account poses a formidable learning problem for the child. How can the child recognize that this structure—which appears to be like any other verb phrase in every sense besides its non-compositional meaning—is actually a special kind of lexical verb?

The problem is pervasive. Idiomaticity, partial compositionality, and, in general, constructions which in some ways generalize productively and in other ways are idiosyncratic are frequent in syntax. Nunberg et al. (1994) examined a large range of idioms, and argued that these occupy a cline of compositionality and productivity. At one end are idioms, such as *shoot the breeze*, which are almost completely non-compositional, and share very little with more productive structures (e.g., they cannot be passivized *The breeze was shot by me and John*). On the other end are idioms, such as *leave no stone unturned*, which are semi-compositional and can enter into a variety of productive syntactic alternations and constructions (e.g., *no stone was left unturned, I left no legal stone unturned*). Recent experimental work also provides evidence that idiom processing has both compositional and non-compositional aspects (Peterson et al., 2001; Sprenger et al., 2006).

The need for a theory of variable productivity and reuse in syntax is not limited to non-compositional idiomatic structures. Fillmore et al. (1988) highlight the important phenomenon of productivity within the context of an idiomatic frame, showing that the non-compositional construction [let alone X] generalizes regularly to a variety of novel expressions (e.g., *let alone John, let alone leaving work early today, let alone too young to drink*). Culicover (1999) identifies a large number of problematic syntactic constructions which are compositional, but exhibit some exceptionality in their distribution. For example, he looks at a number of prepositions in English which vary in properties such as whether they can appear before or after an NP (e.g., *in two years v. two years ago*), arguing that the pervasive exceptionality in the system must be learned on a case-by-case basis. Although there have been attempts to deal with some of these issues by positing more abstract or complex underlying representations (e.g., Coppock, 2007), many of the cases have not yet been explained. In addition to this work, there is accumulating experimental evidence from psycholinguistics that even fully compositional and regular syntactic structures larger than words can be stored (e.g., Arnon and Snider, 2010; Bannard and Matthews, 2008; Berant et al., 2008; Bod, 2001; Conklin and Schmitt, 2008; Gurevich and Goldberg, 2009; Tremblay and Baayen, 2010).
1.3.5 Productivity and Reuse as an Inference

One result of converging evidence from morphology and syntax is that productivity and storage are now widely recognized as properties which can vary from form to form and rule to rule, cutting across levels of representation. Furthermore, even fully-compositional and regular structures are sometimes stored. Jackendoff has proposed the term heterogeneous lexicon to refer to the class of resulting theories which allow for a lexicon that mixes storage and computation of any grammatical units, regular or otherwise (e.g., Jackendoff, 2002). Heterogenous approaches to the lexicon have been widely adopted: for morphology, see Aronoff and Fudeman (2005); Bauer (2001); Pinker (1999); Plag (2003, 2004); for syntax, see Bybee and Hopper (2001); Croft (2001); Culicover and Jackendoff (2005); Goldberg (2005); Kay (2002); Kay and Fillmore (1999); Nooteboom et al. (2002); Poß and van der Wouden (2004); Sag et al. (2003); Wood (2010).

However, although pervasive storage and variable productivity have been widely recognized as empirical phenomena, very few theories have been offered that explain how such a system can be learned. As we have seen, the traditional solutions to the problem—linking storage to certain kinds of representations (i.e., the representational approach) or linking it to idiosyncrasy (i.e., Bloomfieldian storage)—are untenable. If any structure—large or small, corresponding to a word or phrase—can be stored; and if productivity can vary from process to process, and be localized to specific contexts (e.g., let alone X, V-ability), the child faces a serious learning problem: Which processes are productive, and which structures are stored?

The model proposed in this thesis, Fragment Grammars, is an attempt to provide an answer to this question. To do this, the model will address a number of related problems: What is the role of different kinds of frequency? Why are regularity and productivity correlated? How is competition resolved? Why do multiple ways of producing an expression exist in the first place? Although it cannot provide complete answers to these questions, the model makes specific, novel predictions for each of them. The theory also has potential consequences for linguistic theories: By moving the problem of controlling productivity to an independent (perhaps domain-general) set of mechanisms, theories of linguistic representation can be simplified.

There are two proposals behind the Fragment Grammars model. The first is that anything that can be computed can be stored. The specific computational mechanism by which this is accomplished, stochastic memoization, is inherited from the Adaptor Grammars model (Johnson et al., 2007a). Adaptor Grammars provide a probabilistic framework for learning the set of stored, reusable subcomputations in some linguistic system. The second proposal is that stored items themselves can allow points of abstraction and further productive computation. In Fragment Grammars, a stored item can include subcomputations which must be composed on the fly, and these subparts can vary on a structure-by-structure basis. This generalization of Adaptor Grammars is made possible by the computational mechanism of stochastically lazy evaluation. Crucially, it is this respect in which Fragment Grammars treat productivity as an inference. Although Adaptor Grammars provide a way to probabilistically learn and deploy a system of reusable subcomputations, Fragment Grammars also learn which subparts of structures should be computed productively.

In this thesis, I will examine two morphological systems: the English past tense and English derivational morphology. These two domains are of special interest in evaluating models of productivity and reuse because they exhibit very different distributions of these properties. Derivational morphology is characterized by a broad array of affixes of differing levels of productivity. This sort of gradience is exactly the kind of structure that one might expect probabilistic models, such as the one advocated in the thesis, to excel at capturing (see e.g., Hay and Baayen, 2005). By contrast, the English past tense’s regular -ed rule (e.g., walk/walked) is highly productive, while the various irregular form classes (e.g., sing/sang, sleep/slept, etc.) generalize only very rarely (e.g., Prasada and Pinker, 1993, see also Chapter 4). Thus the English past tense provides an important counterpoint
Chapter 1: Introduction

19

to the gradient structure of derivational morphology. A model of productivity and reuse must be able to handle both kinds of linguistic systems: those with widely varying mixtures of productive computation and reuse; and those with sharp, nearly deterministic, dichotomies between the two.

In the following section, I will delve into the proposal in more detail and also consider a number of alternative models that have been chosen to formalize some of the ideas just discussed (i.e., full-listing and full-parsing models), as well as some more recent theoretical proposals (i.e., exemplar-based models).

1.4 The Proposal

In this section, I give an overview of the model proposed in this thesis along with three alternative approaches. First, I discuss Probabilistic Context-Free Phrase Structure Grammars (PCFGs) which I use as the underlying model of linguistic computation.

1.4.1 A Model of Linguistic Computation

A crucial feature of linguistic representation is hierarchical constituency. Sentences are composed of phrases, phrases are composed of words, words are composed of morphemes, and so on. A generative model of such hierarchical structures consists of formulae which specify how smaller units can be combined into larger units. Context-free Phrase Structure Grammars (CFGs) are a simple, widely-known, and well-studied formalism for modeling hierarchically structured computation (see Autebert et al., 1997, for a review of formal results). CFGs are, in some sense, the simplest generative model which capture the idea of arbitrary (and possibly recursive) hierarchical structure.

PCFGs are a probabilistic variant of Context-free Phrase Structure Grammars. CFGs are inherently non-deterministic: There can be more than one way to build a constituent from smaller pieces. PCFGs replace this non-determinism with probability. During the computation of a form, whenever a choice must be made, a PCFG provides a probability distribution over the set of possible options.

A PCFG consists of several elements. The first element is the set of nonterminals. By convention, nonterminals are written with capital letters and represent categories of constituents such as “adjective” (Adj), or “noun” (N). One nonterminal is known as the start symbol, and represents the category of complete structures. In the examples below, it will represent the category of complete words (W). The second element of a PCFG is the set of terminals—written with lowercase letters—which represent the most primitive elements out of which larger constituent categories can be built. In what follows, terminals will be morphemes such as -ness or happy. Finally, there is the set of production rules (or just rules). The production rules are written $A \rightarrow \gamma$, where $\gamma$ is some sequence of terminals and nonterminals, and $A$ is a nonterminal. These rules define the set of possible consists-of relations between categories of constituents. For example the rule $N \rightarrow \text{Adj} -\text{ness}$ says that a noun (N) can be built by appending -ness to an adjective (Adj). The list of symbols to the right of the arrow is referred to as the right-hand side (RHS) of that rule. The nonterminal to the left of the arrow is the rule’s left-hand side (LHS).

Terminal symbols are atomic—they cannot consist of other constituent types. Starting with the start symbol W and applying the rules until only terminals remain, it is possible to derive sequences of terminals such as happy -ness (happiness).

As an example, consider the PCFG in Figure 1.7. The rules of the PCFG are on the right; the associated probabilities, on the left. Suppose that the generative process is building a noun (N). In this grammar, there are many ways of building a word of type N, five of the rules for doing so are shown in the figure. The probabilities of a PCFG provide a distribution over such choices, and
must, therefore, sum to 1: \( \sum_i p_{N_i} = 1 \). The generative process will use this distribution to make a particular choice by sampling one of the possible RHSs associated with the \( N \) category. Once a particular rule has been chosen, sampling continues recursively on each nonterminal of that rule’s RHS.

The particular set of choices associated with building a linguistic expression can be represented as a tree, called a derivation tree or parse tree. Figure 1.8 shows such a tree associated with the process of generating the word agreeability. In this figure, and in the derivation trees shown later, I have suppressed the top-level \( W \) category.

Like the simple coin-flip example discussed earlier, a PCFG can be thought of either in terms of a computational process which samples expressions, or in terms of a distribution over expressions.\(^\text{17}\) There is a distribution associated with each nonterminal and each rule in a PCFG. The distribution for the entire grammar is simply the distribution associated with the start symbol \( W \).

Since this distribution is infinite in general (due to recursion), it cannot be displayed in its entirety; however, a small part of it is shown schematically in Figure 1.9.

The probabilities associated with rules in a PCFG are interpreted as degrees of belief that the category of the LHS will be composed as dictated by the RHS. For example, the two rules \( p_{N_1}: N \rightarrow \text{Adj} \sim\text{ness} \) and \( p_{N_2}: N \rightarrow \text{Adj} \sim\text{ity} \) state that when a noun is generated, it is expected to

\(^\text{17}\)This is essentially a stochastic version of the intensional/extensional distinction often made in set theory.
1.4.2 Partial Computations in PCFGs

The formal technique explored in this thesis is the storage and reuse of partial computations. In a PCFG, such partial computations correspond to contiguous derivation tree fragments such as those shown in Figure 1.10.

Each of these tree fragments corresponds to a sub–computation of the original sampled expression in Figure 1.8. There are many such subcomputations: The number of subtrees of a PCFG derivation tree is exponential in the number of nodes of the tree in general. The first subtree in Figure 1.10 corresponds to a minimal computational step which results from choosing the underlying PCFG rule: $N \rightarrow Adj -ity$. The second subtree represents the entire computation of agreeability. The third and fourth fragments show intermediate subcomputations of the agreeability derivation. Each one corresponds to a partial fragment of the original computation, which is larger than a PCFG rule, but smaller than the full derivation tree. The right–most tree in Figure 1.10, for example, sets the suffixes -able and -ity as constant in expressions which use this tree, but leaves the choice of stem $V$ as a variable.

Like rules from the original PCFG, these fragments define distributions over expressions. Figure 1.11 shows the distribution (in blue) for the right–most subtree in Figure 1.10 (for comparison, the distribution of the full PCFG from Figure 1.9 is shown in black outline). The distribution associated with this subtree places probability mass in all forms which contain the sequence of suffixes -able -ity, and have verbs in their base position. Notice that the distribution associated with the $V$ -ability subtree places probability on fewer outcomes than the overall distribution associated with the full PCFG; it concentrates the probability mass on a subset of forms.

In section 1.2.2, I described how the proposal in this thesis can be understood as storing frequently occurring subcomputations so that they can be reused in a single step. A subcomputation like the one just discussed above is equivalent to a PCFG rule of the form $N \rightarrow V -able -ity$. Storing this subtree is equivalent to adding this new rule back into the original grammar, as is shown in Figure 1.12 (the new rule is highlighted in red).

By adding this new rule into the original PCFG, the overall distribution has been modified. This rule makes it more likely that words ending in -ability are sampled (the corresponding increases in probability are highlighted in red), and less likely that other kinds of words are sampled.

\footnote{This is only true if we are interested in the final distribution over expressions rather than the internal structure of derivations.}
Note that because every stored fragment must correspond to some combination of rules in the underlying PCFG, storage can never fundamentally change the set of structures that are possible to generate. However, storage can substantially change which structures are probable—by increasing the probability of certain combinations and (therefore) decreasing the probability of others.

### 1.4.3 Productivity and Reuse as a Tradeoff

Consider Figure 1.13 which shows a small set of word derivations. The highlighted subtrees in this figure represent computations that have been stored and reused—subtrees in the same color are shared across multiple derivations. In this figure, all the reused structures are small units of the same granularity as a PCFG. Small fragments, such as these, are very general, abstract, and highly reusable. For example, the subcomputation \( \text{Adj} \rightarrow \text{V-able} \) is reused in every form in the figure.

The distribution over forms which is associated with the rule \( \text{Adj} \rightarrow \text{V-able} \) is shown in blue in Figure 1.14 (for comparison, the distribution of the full PCFG from Figure 1.9 is shown in black outline). This figure illustrates that small, abstract subcomputations, like \( \text{Adj} \rightarrow \text{V-able} \), spread probability over many possible forms. Since many of these forms may never have been observed, such abstract fragments of structure also predict the possibility of novel computations.

There is a tight link between the (average) size of stored units, the (average) probability of individual forms, and the kind of distribution defined by the system. In generative models, the probability of an entire computation is given by the product of the probabilities of all choices made during that computation. Because probabilities are numbers between 0 and 1, the probability of
Chapter 1: Introduction

Figure 1.10: PCFG Sub–computations: Some sub–computations of the PCFG derivation tree shown in Figure 1.8.

A derivation declines geometrically as the number of choices increases. Computations using small, abstract units must make many random choices, and, therefore, each possible derivation must have low probability (on average). This is illustrated in Figure 1.15 which shows the calculation of the probability of a form consisting of minimal units. In a setting where most stored items are small and abstract, each computation will tend to have a probability given by a product with a large number of factors. Because the probability of all forms must sum to 1, if each form has low probability (on average), then there must be many of them. The consequence for a system like that in Figure 1.13 is that it must spread probability mass over many forms, including potentially novel ones.

The interpretation of probabilities as degrees of belief implies that the storage of small, abstract fragments of structure is a hypothesis or a prediction about future computation. Specifically, storing small, abstract structures predicts that, in the future, many different—possibly novel—forms will appear. A generative system that allows greater freedom of choice predicts that there will be variability and novelty in the data.

Figure 1.16 shows the opposite extreme. Here, what has been stored are entire word derivations. When complete derivations like this are stored, each one is highly specific. Choosing to reuse such a stored subcomputation is equivalent to sampling an entire form in a single step (in essence, such a stored item can be thought of as a single, complex terminal symbol). Furthermore, these stored units can only be reused when exactly the same form is desired, and, therefore, they allow only minimal sharing. However, under those circumstances where the exact same form is actually desired, it can be generated with high probability.

Figure 1.17 shows the distribution (in blue) over forms which corresponds to the stored fragment $N \rightarrow agree -able -ity$ in Figure 1.16 (for comparison, the distribution of the full PCFG from Figure 1.9 is shown in black outline). This distribution is a degenerate distribution which puts all of its probability mass on a single outcome.\(^\text{19}\)

The example in Figure 1.16 again illustrates the tight link between the size of stored units, the probability of individual forms, and the kind of distribution defined by the overall system. Figure 1.18 shows the calculation for the probability of the fragment $N \rightarrow agree -able -ity$. The product involves only one term—the probability of the entire structure. Because the word $agreeability$ can be generated in a single step using this large, specific fragment, the corresponding distribution is highly concentrated. Furthermore, because probability distributions sum to 1, storing entire subcomputations reduces the probability of other possibilities. In particular, when many large, specific forms are stored there will be little probability left over to be assigned to unobserved, novel

\(^{19}\)Such distributions are known as $\delta$–distributions in probability theory.
Figure 1.11: Distribution over Forms Ending in -ability. This figure shows (in blue) the distribution over forms which use the right-most subtree in Figure 1.10 in their computation. This distribution is more specific or concentrated than the original PCFG distribution. The original distribution is shown in black outline for comparison.

forms. Storage of such large, specific subcomputations is a hypothesis that the corresponding forms are expected to appear frequently in the future, and that other combinatorial possibilities in the underlying PCFG are unlikely to be observed.

Figure 1.19 shows one of the many possible intermediate storage scenarios. The distribution for one fragment, \( N \rightarrow V -able -ity \), is shown in Figure 1.20 (in blue; original PCFG distribution in black outline). On one hand, this distribution spreads its probability mass over more forms than the whole-form storage distribution (Figure 1.17). However, it allows for less variability and novelty than the distribution associated with the more abstract fragments (Figure 1.14).

Choosing to use this subcomputation reduces uncertainty to an intermediate degree. Overall, computations will require fewer choices than the underlying PCFG, but a greater number of choices than when whole-form computations are stored. The calculation of the probability of a form using the fragment \( \text{Adj} \rightarrow V -able -ity \) is shown in 1.21.

Storing a subcomputation like \( N \rightarrow V -able -ity \) corresponds to a hypothesis about future productivity and reuse with two parts: First, it predicts that the affixes -able and -ity are likely to appear together. Second, it predicts that this combination can generalize easily to novel forms by replacing the variable \( V \) with new verb stems.

The model presented in this thesis attempts to find the set of stored subcomputations which best explain the input data. On one hand, when some contexts in the data exhibit novelty (e.g., the \( V \) variable), the model will prefer to treat the computation for those contexts as productive. On the
other hand, when particular combinations of primitive computations tend to be reused together, the model will prefer to treat these as stored units which are reusable in a single step. The optimization is global: It will involve tradeoffs at multiple levels of structure over multiple different forms.

1.4.4 Four Approaches to Productivity and Reuse

Throughout the thesis, I will systematically consider four approaches to productivity and reuse: full–parsing, full–listing, exemplar–based, and inference–based. All of these approaches start from a underlying system which defines the space of possible computations—here I assume that this starting system provides the ability to generate a set of tree–shaped computations like those discussed above (i.e., it is a context–free grammar).

These approaches have been chosen to encode theoretical proposals in the literature, including several of the classical linguistic accounts discussed above (see also the theoretical discussions in e.g., Baayen et al., 1997b; Domínguez et al., 2000). Each of the approaches has been implemented as a probabilistic model; the models differ, however, in the strategies that they use to determine which subcomputations (subtrees) can be stored and reused. Although each approach captures the essential properties of a theoretical proposal from the literature, all of them are fully probabilistic, state–of–the–art models that have been applied successfully to problems in natural language processing and machine learning. In particular, unlike the classical non–deterministic linguistic theories, they all handle uncertainty and probabilistic inference (up to the limits allowed by their approach to storage). I will describe these models, and the literature that surrounds them in greater detail in

---

20As I will describe in the next chapter, the exemplar–based approach to productivity and reuse has been implemented in two variants.
Chapter 1: Introduction

26

Figure 1.13: **Storing Minimal Subcomputations**: This figure shows the consequences of storing minimal subcomputations that are equivalent to the rules of the underlying PCFG. Each subcomputation can be shared across many computations. However, each derivation of an entire word requires many probabilistic choices.

Chapters 2 and 3, and especially in Section 2.4. In this section, I give an informal overview framed in terms of the tradeoffs just discussed.

**Full–parsing** In this model, all structures are the result of computation using minimal–sized units. No larger items are stored in memory. This model corresponds to the scenario described in Figure 1.13. In such a setting, each primitive is highly reusable. However, any computation will involve choosing many small, abstract primitives. Such an approach to reuse will be most effective when there is large amount of combinatoriality, variability, and novelty in the data.

In this thesis, all of the empirical data considered is morphological in nature. In my discussion of the history of models of morphology, I emphasized that the problem of controlling the productivity of word formation rules was a major concern even in the earliest work on generative morphology. As a result, no researcher has ever proposed a model of word structure which is completely decompositional for all of morphology; thus this model is best seen as a **baseline** against which to compare the performance of the other models.

The full–parsing approach is formalized using a mathematical model known as Multinomial–Dirichlet Probabilistic Context–free Grammar. See Sections 2.4.1 and 3.1.4 for details and discussion.

**Full-listing** In this model, each structure is stored in its entirety after the first time it is built. Thus, while this system can account for productive generalization by using an underlying PCFG, it is nevertheless very conservative—it probabilistically prefers to reuse previously built structures in a way similar to the scenario described in Figure 1.16. Under such a storage strategy, each stored item is extremely specific, and therefore can only be reused in limited contexts. Such an approach to reuse

---

21The formalization of this model (MDPCFG—see below) differs from the simple PCFGs discussed above in that it is able to learn the weights of individual rules.

22This remark warrants qualification: Some models, such as Distribute Morphology (Harley and Noyer, 2003), are fully decompositional in a certain sense. However, because they contain various mechanisms which filter the output of rules, they do not predict that every logically possible combination of morphemes will be witnessed in the linguistic data—unlike the full–parsing model studied in this thesis.
Figure 1.14: Distribution over Forms Using Rule Adj $\rightarrow$ V -able : This histogram shows the distribution over forms which use the fragment Adj $\rightarrow$ V -able. Since this is a small, abstract fragment the distribution puts mass on many forms. This means that in order to choose a single form, more choices must be made. The original PCFG distribution is shown in outline for comparison.

will be most effective when the language consists of a small number of highly specific, frequently reused, forms.

This model can be seen as implementing a probabilistic version of classical lexicalist theories: Each computation is stored in full after it is first computed (e.g., Bresnan, 1982; Jackendoff, 1975; Lieber, 1980). Importantly, however, the variant of the full–listing approach used in this thesis differs from classical linguistic proposals in a number of ways. First, while the present model stores each computation deterministically, it reuses each computation probabilistically. Thus, unlike earlier proposals, it provides a learning model in which there is always some probability that a form (even one which has been previously stored) can be computed from scratch. Second, unlike classical approaches, this model provides a precise characterization of how the distribution of forms in the data gives rise to probabilistic reuse. Third, and finally, the full–listing model used in this thesis not only stores each complete top–level computation in its entirety, but also recursively stores all (complete) subcomputations of the top–level form, and these, themselves, can be reused by later computations.
Figure 1.15: **Probability of Fully Composed Derivation**: This figure shows the calculations of probabilities for a form under the assumption that stored subcomputations are maximally small.

Figure 1.16: **Storing Maximal Subcomputations**: This figure shows the consequences of storing maximal subcomputations that are equivalent to entire, complete computations of the underlying PCFG. These subcomputations cannot be easily shared because they are very specific. However, they are cheap to reuse—requiring only a single random choice.

The full-listing approach is formalized using a mathematical model known as Adaptor Grammar. See Sections 2.4.2 and 3.1.7 for details and discussion.

**Exemplar–based Productivity and Reuse**  This model stores all structures which are consistent with the data, both small and abstract, and large and specific (and all in between). This model differs from the earlier two in that it stores both abstract and specific fragments. However, it differs from the proposal in this thesis in that it does not commit to a single analysis of each data point, but rather hedges across many different levels of abstraction.

Because the storage strategy used by this model is different from those previously considered, I illustrate it in Figure 1.22. This figure again highlights subcomputations which can be shared across forms. However, unlike figures 1.13, 1.16, and 1.19, this model allows many different, overlapping derivations for each word, and distributes probability between all of them.

There are a large variety of different kinds of models in psychology and linguistics which are called *exemplar* theories (e.g., Duelemans and van den Bosch, 2005; Murphy, 2004; Pierrehumbert, 2001). However, one unifying characteristic of such models is that they operate by storing all of the data that they encounter. Generalizations to new data points are handled by computing the
Figure 1.17: **Distribution over Forms Using Rule** $N \rightarrow agree -able -ity$. This histogram shows the distribution over forms (blue) which use the fragment $N \rightarrow agree -able -ity$. Note that since there is no uncertainty left in such a fragment, this distribution puts all of its mass on a single outcome. Therefore, while reusing this particular form is cheap, there is no sharing with other forms. The original PCFG distribution is shown in black outline for comparison.

similarity of new observations with this database of stored exemplars. Exemplar models are often contrasted with theories which make use of summary representations of categories. In the case of the models used in this thesis, a derivation for a particular word can be considered a (complex) category for the word—a hypothesis about how the word was constructed. Unlike the other models, the exemplar–based model does not assign a single hypothesis to each datapoint, but, rather, represents all possible derivations.

The exemplar–based approach is formalized using two different versions of the Data Oriented Parsing estimator for tree–substitution grammars. See Sections 2.4.3 and 3.1.5 for details and discussion.

**Inference–based Productivity and Reuse** The model advocated in this thesis, Fragment Grammars, treats as an inference the problem of which subcomputations are productive and which
Figure 1.18: **Probability of Fully Stored Tree**: This figure shows the calculations of probabilities for a form under the assumption that stored subcomputations are maximally large. The tree is sampled in a single step.

![Figure 1.18](image)

Figure 1.19: **Storing Intermediate Sized Subcomputations**: This figure shows the consequences of storing intermediate sized subcomputations. This allows more sharing on average than storing maximal subcomputations with less uncertainty on average than storing minimal sized subcomputations.

![Figure 1.19](image)

should be stored for later reuse.\textsuperscript{23} Like the full–parsing approach, Fragment Grammars are able to store abstract structures. Like the full–listing approach, they can store specific structures, and furthermore, like the exemplar–based approach, they can store all intermediate structures. However, unlike the previous approaches, the particular solutions it finds to productivity and reuse are determined by an inference which balances simplicity against fit to the data. For eachdatapoint to which it is exposed, the model hypothesizes whether it is optimal to account for the structure using reusable stored items, productive computation, or some mixture of both (see Figure 1.19). I formalize and describe Fragment Grammars in more detail in the next two chapters.

\textsuperscript{23}Note that because all of these models are probabilistic, they are all inferential in a certain sense. However, only the Fragment Grammar model does inference both over the set of stored subcomputations and points of productivity within those stored units.
Figure 1.20: Distribution over Forms Using Rule $N \rightarrow V$ -able -ity : This histogram shows the distribution over forms which use the fragment $N \rightarrow V$ -able -ity (in blue). There is an intermediate amount of uncertainty in such a fragment compared to earlier examples. The original PCFG distribution is shown in black outline for comparison.

1.5 Scope of the Thesis

Before moving on to a detailed development of the models used in the thesis, I address several issues regarding their implementation and psychological interpretation.

1.5.1 Assumptions about the Starting State of Learning

All approaches to linguistic structure must satisfy a basic principle—called by Frauenfelder and Schreuder (1992) the productivity constraint—which states that any theory must provide a mechanism for generating novel expressions. As discussed in Section 1.3, productivity is made possible in classical linguistic and psycholinguistic models by the use of compositional, combinatorial computational systems. In this thesis, all of the models make use of an underlying model of computation (formalized as a context–free grammar) which defines the space of possible computations over which each model learns. Productivity is made possible by the compositional and combinatorial possibil-
Chapter 1: Introduction

Figure 1.21: Probability of Tree Built with Intermediate Sized Subcomputations: This figure shows the calculations of probabilities for a form built with intermediate sized stored fragments.

Figure 1.22: Exemplar–based Storage: This figure shows exemplar–based storage. Here every possible subtree consistent with the data is stored. Note that this leads to many overlapping analyses for each word.

ities inherent in this underlying context–free grammar—which is provided a priori to the models. Therefore, the theory presented in the thesis should be understood as an approach to controlling or constraining productivity. In other words, the models in this thesis inherit the solution to the problem of productivity from classic linguistic and psycholinguistic theories which, in turn, inherit this solution from classical theories of computation. What is added is a proposal about how the pattern of productivity and reuse can be learned from data.

For each empirical domain in the thesis, I use context–free grammars which are already quite specialized to the domain. Furthermore, morphological analyses of training data forms are provided as input to each model. For example, in Chapter 5, where I examine the English past tense, the underlying context–free grammar defines computations similar to those shown in Figure 1.23.

Thus, the input to these simulations provides a representation which is already quite abstract, and specialized to English verbal morphology: Stems are segmented from their inflections; abstract stems such as_GO_ are identified when they underlie non–transparent past forms such as_went_; the space of possible verb tenses is provided and tenses are labeled on individual forms; etc. At the same time, the system does not encode any information about the semantic or phonological selectional restrictions of various morphemes: It allows, for example, a rule like /I/ → /æ/ to apply to a stem like_EAT_—even though this is phonologically impossible. On one hand, when compared with the starting state of the child learner, these modeling assumptions provide too much information—in
the form of the segmentation, abstract identity, and categories of morphemes. On the other hand, they provide too little information, by not encoding any phonological or semantic structure.24

These assumptions are not meant to correspond to a particular stage of development or learning procedure used by children. Instead, the question addressed by all of the simulations is: What does the distribution of morphemes in each empirical domain imply for productivity and reuse under the assumptions encoded by each model? Providing segmentations, categorial information, and the abstract identity of morphemes allows us to focus on just the patterns of reuse and novelty in the morphological data. While a more complete model would simultaneously learn these other kinds of structure, it would also make the underlying representations themselves a moving target—complicating interpretation.25

By not modeling semantic and phonological selectional restrictions, just those aspects of productivity and reuse which depend only on the distribution of morphemes can be isolated. This choice is adversarial to learning the correct patterns of productivity and reuse. As I mentioned in Section 1.3.3, all of the empirical domains examined in this thesis exhibit conspiracies: Productivity is correlated with phonological and semantic transparency, while reuse is correlated with irregularity. If phonology and semantics were modeled, this would provide a valuable additional source of evidence about the productivity of various word–formation processes. For example, cases like the past tense allomorph /i/ → /æ/ applying to EAT would be ruled out a priori. As I will show, Fragment Grammars are able to learn to rule out incorrect combinations like this despite the lack of phonological and semantic structure. This strongly supports the idea that the distribution of morphemes alone provides robust evidence of the correct pattern of productivity. Furthermore, it suggests that the construction of a more complete model which integrated these other kinds of structure would lead to improved performance. For the interested reader, I give a sketch of how phonology and semantics might be integrated into the current framework in Appendix A.

Finally, note that all of the assumptions about the starting context–free grammar, set of abstract morphemes, and phonological and semantic selectional restrictions are shared identically by all of the models explored in the thesis. Therefore to the degree that they affect the performance of any of the models, they affect the performance of all of the models.

24In Section 5.2.1, all of these choices and others are discussed and justified in more detail. Similar choices made in modeling English derivational morphology are explained in Section 7.2.1.

25It would also significantly complicate inference.
Chapter 1: Introduction

1.5.2 Context–Free Grammars as Models of Linguistic Computation

There are several reasons CFGs were selected as the underlying model of linguistic computation in this thesis. First, they are simple, well–understood, widely–used in computational linguistics and have (relatively) easy–to–implement inference algorithms (Manning and Schütze, 1999). Second, the model which I directly generalize in this dissertation—Adaptor Grammars—is built on them. Third, in terms of the sets of strings that they define (i.e., their weak generative capacity), they subsume the machinery needed for English derivational and inflectional morphology. Fourth, and most importantly, (P)CFGs characterize tree–shaped computational processes. The reuse of fragments of tree–shaped computations is the main technique proposed in this thesis.

Nevertheless, the adequacy and appropriateness of (P)CFGs as models of linguistic structures has been widely questioned in the literature. There are two main issues. The first is whether CFGs can express the necessary linguistic dependencies to characterize the set of permissible strings for some domain—the property known as weak generative capacity of a formalism. The current consensus is that although CFGs are inadequate for syntax[26] they suffice for morphology and phonology (indeed, the weaker class of finite–state systems suffices for morphology and phonology; see e.g., Beesley and Kartutunen, 2003; Mohri, 1997; Roark and Sproat, 2007).

The second issue is whether CFGs can describe the correct kinds structural descriptions for linguistic expressions—the so–called strong generative capacity of the formalism (Miller, 2000). This issue is closely related to the question of whether CFGs provide the right kind of inductive biases for learning linguistic systems. These latter issues are much less well understood, and there is no general consensus. Nevertheless, many theorists have argued that words have hierarchical constituent structure (e.g., Halle and Marantz, 1993; Selkirk, 1982; Williams, 1981), and, therefore, CFGs are a reasonable system to adopt for the empirical problems studied in this thesis.

In summary, CFGs provide an adequate formalization of the underlying computational system for morphology, and their drawbacks are minimal for present purposes. Given the generalizations of the present framework which I describe in Section 2.3.7 and Appendix A, it should be straightforward to extend the models to richer representations in the future.

1.5.3 Relation to Processing and Neuroscience Theories

Most theories of morphological phenomena in psychology have been process–level theories, offering explicit representations and algorithms to model morphological learning and processing. By contrast, the work in this thesis is intended to be a rational analysis—an examination of how different modeling assumptions project patterns of productivity and reuse from the data. It is not a proposal for the specific algorithms and data structures used by speakers during learning or processing.

Because of this, the account is not necessarily inconsistent with various proposals already in the literature. For example, morphology has seen a number of localist, activation–based, race models of morphological processing (e.g., Baayen and Schreuder, 2000; Baayen et al., 2000) In these models, it is hypothesized that composition and retrieval processing routes compete to analyze forms. The route that completes analysis in the shortest time wins—see Sections 2.3.3 and 6.3.4 for further discussion. The speed with which each route is computed is determined by factors such as the frequency with which that route has won in the past. These models provide an algorithmic implementation of morphological processing—and it is plausible that Fragment Grammars could be implemented by

---

[26]From the perspective of computational linguistics see, for example, discussions in Jurafsky and Martin (2000); from a linguistic perspective see Chomsky (1956b); Huybregts (1985); Pullum and Gazdar (1982); Shieber (1985); and for a psycholinguistic perspective see Levy (2008).
Chapter 1: Introduction

A race–model implementation of Fragment Grammar would require finding a way to represent probabilities in time. From an evolutionary perspective, a brain which represents high–probability computations in such a way that allows for faster computation of more frequently encountered structure has obvious appeal.

As another example, Michael Ullman has argued extensively that compositional structure building and retrieval from memory correspond to the traditional psychological distinction between *procedural* and *declarative* memory (Ullman, 2001). It is entirely possible for such an account to be consistent with the present work. The formal mechanisms described in the next chapter—*stochastic memoization* and *stochastic lazy evaluation*—may be implementable in a way that reflects the procedural/declarative distinction.

### 1.5.4 Resource Tradeoffs versus Prediction Tradeoffs

The idea that choices about productivity and storage should be viewed as an inference is closely related to the view that they can be seen as an optimization problem. Many authors have adopted a perspective which treats the problem as one of optimal resource allocation. The resources typically discussed are the *time* it takes to complete a computation and the *space* in memory that is consumed by stored items (e.g., Baayen and Schreuder, 2000; Baayen et al., 2000; Bertram et al., 2000; Frauenfelder and Schreuder, 1992; Kuperman et al., 2010; Prasada and Pinker, 1993; Sandra, 1994; Stanners et al., 1979a; Yang, 2009, 2010). The intuition behind such approaches is simple: High frequency items are stored because they can therefore be processed faster, minimizing average processing time.

The intuition underlying the model advocated in this thesis is quite different. What is optimized by the present model is not the time it takes to compute an expression or the amount of memory used, but rather the ability of the system to *predict* which expressions and subexpressions will be reused in the future and what kinds of structure–building processes will give rise to novel forms.

I believe that the prediction perspective is preferable to the resource tradeoff perspective for several reasons. First, for practical reasons, resource tradeoff theories are difficult to formulate precisely. Such theories have to be grounded in measures of computation cost (in time) and memory cost for which we currently only have very rough estimates (e.g., Landauer, 1986; Valiant, 2005). Furthermore, such theories require a formulation of the fungibility between these two currencies—how can space in memory be traded for faster processing time or vice versa. Even in the area of formal computer science, *complexity theory*, which studies such questions, space and time tradeoffs are only partially understood (see e.g., Papadimitriou, 2003, for a general introduction). Fungibility of time and space in the brain is even less well–understood. By contrast, for the models studied in this thesis, there is only one currency, probability which has a clear interpretation: the likelihood that a particular form or structure–building process will be used in the future.

The problems just outlined are primarily methodological, however, there is a more serious scientific concern. Clearly human biology provides constraints on resources which must be respected by the linguistic system. A computation cannot happen faster than is possible in neural hardware; and it cannot happen so slowly that by the time it is finished it is of no use to the organism. Likewise, there is only a finite amount of space available for storage in the brain. However, while

---

27Implementation of Fragment Grammars in an activation–based framework would not be trivial. Fragment Grammars can easily represent hierarchical and recursive constituent structures because it is build on top of CFGs—the representation of such structures in activation–based frameworks is challenging, although some progress has been made (see e.g., Hadley, 2000; Tabor, 2009; Westermann et al., 2009). Also, most localist, activation–based, race models use a constant additive update rule. As I discuss in Section 2.3.3, probabilistic systems cannot be represented by such a rule.
biological hardware certainly provides boundary conditions on computation versus reuse, there are likely many other factors influencing the optimal rates of each. Language is subject to many other constraints. It seems likely that these factors play as much role in shaping what is optimal in terms of productivity and reuse as constraints on memory and time. For example, the structure of the environment that individuals wish to communicate about is at least as important: What meanings are repeated? How often does a new meaning need to be produced? Learnability is also likely to place strong constraints on productivity and reuse. A system which generates too much novelty will not witness enough repeatable structure to be learned. In short, while resource bounds are surely an important part of a theory of productivity and reuse, they are only one component of many.

Note that the prediction perspective neatly sidesteps many of these issues. The quantity optimized by the models studied in this thesis is the rate of reuse versus the rate of productive generalization. The reason that a particular form or computation is likely or unlikely to be reused in part or full can depend on many causes—communicative efficiency, learnability, processing costs, etc.—but this quantity collapses across all of these. From a scientific perspective, of course, we ultimately would like to provide an explanation for productivity and reuse in terms of all of the contributing factors. However, by collapsing them into a single, interpretable quantity, we can move part of the way towards a solution without understanding all of the complexities of the system.

Nevertheless, there is an interesting correlation between prediction and resource consumption. As I will discuss in the next chapter, the techniques used to formalize the framework in this thesis—stochastic memoization and stochastically lazy evaluation—have effects both on prediction and processing time. Intuitively, there is a close relationship between techniques which allow a system to predict future computations and techniques which allow a system to perform those computations more efficiently. Nevertheless, this relationship is not well-understood, and in the remainder of the thesis I will appeal to the prediction perspective to provide intuitions about modeling results.

---

28Being a social species further complicates these issues since these constraints are likely determined, at least in part, by frequency dependent selection. Suppose that the population is prewired for some computation/reuse tradeoff. When does a mutation increasing the productivity of the system or increasing the rate of storage spread through the population? When is it selected away? Under what regimes it is subject to neutral evolution? The system must remain learnable, mutually intelligible, and useful.
Chapter 2

The Framework

This chapter consists of four parts. In Section 2.1, I discuss the modeling idiom adopted in this thesis—structured Bayesian generative modeling. In Section 2.2, I discuss the psychological interpretation of this approach to cognitive modeling, focusing, in particular, on the Bayesian notion of *optimality*. In Section 2.3, I develop the modeling framework from a probabilistic programming perspective using the Church programming language (Goodman et al., 2008). This formalization makes explicit the relationship between the model and two important ideas from computer science—*memoization* and *lazy versus eager evaluation*. By adopting stochastic generalizations of these techniques, I show how Adaptor Grammars (AG) and Fragment Grammars (FG) can be formalized as generalizations of probabilistic context–free grammars (PCFGs). I then show how this approach can again be generalized to the space of all generative models. One theme that runs through this part of the thesis is that techniques that have consequences primarily in terms of the *efficiency* of programs in a traditional deterministic setting, can have effects on the *meaning* of programs (i.e., the function defined by the program) in a stochastic setting. Section 2.3 is likely to be of more interest to readers with a background in Bayesian modeling, the theory of programming languages, or probabilistic programming. Readers who wish to skip these sections may do so—nothing crucial to understanding what comes later in the thesis depends on them. Finally, in Section 2.4, I provide more details on the five formal models evaluated in this thesis.

2.1 Structured Bayesian Generative Modeling

The modeling framework used in this dissertation is known as *structured Bayesian generative modeling*. This paradigm makes use of Bayesian statistics to do inference over rich, structured representations. I discuss each part of the framework in more depth below.

2.1.1 Bayesian Statistics

Bayesian statistics are distinguished from the more well–known *frequentist* approach to statistics by a *subjective* interpretation of probabilities. While the frequentist interprets probabilities as the (limiting) relative frequencies of large numbers of observations of some process, the Bayesian interprets probability as a degree of belief in the possibilities of different outcomes (Bernardo and Smith, 1994; Jaynes, 2003; Sivia, 1996).

1Note that *subjective* in this context is a technical term from the philosophical and mathematical literatures on the interpretation of probability. It refers to the fact that in a Bayesian setting probabilities are assumed to measure
A basic concept in probability theory is the random variable. Intuitively, a random variable can be thought of as a placeholder that can take on particular values from some specified set, $X$, according to a given probability distribution, $p$. A random variable can be sampled to produce a single value (i.e., a draw) from $X$, with probability, $p(x)$, specified by the corresponding distribution. When we have sampled a value $x$ from set $X$ with probability $p(x)$ we say that $X$ has taken on the value $x$ and write $X = x$. Some random variables which might be used in models of language structure are described below.

- $D$ (linguistic data sets): The distribution over finite datasets which can be drawn from a specific language or a specific subdomain of some language (e.g., English inflectional morphology). This random variable could take on values such as the set of specific verbs which appear in the SWITCHBOARD corpus: $D = d_{SWBD}$ (Godfrey et al., 1992; Marcus et al., 1999), or all of the specific sentences that are spoken by a parent to a child by the age of 10: $D = d_{Age \leq 10}$. Sampling from this random variable might correspond to an act of writing or speaking.

- $G$ (possible natural language grammars): The distribution over possible grammars for natural language (i.e., any possible human language). This random variable could take on values such as the grammar of French, $(G = g_{French})$ or the grammar of English $(G = g_{English})$. One process which samples from this random variable is cultural evolution—whereby languages are learned iteratively by each generation of speakers; each child and each generation of children samples a possible human language grammar (conditioned on their input data, see below).

- $U$ (logically possible linguistic computation systems): The distribution over possible architectures for the computational system which underlies language ($U$ for universal grammar). This random variable can take on values, for example, which correspond to the approaches to productivity and reuse studied in this thesis: $U = u_{full-parsing}$, $U = u_{full-listing}$, $U = u_{exemplar-based}$, $U = u_{inference-based}$. Evolution can be thought of as sampling from this random variable, as can the scientist trying to determine the properties of universal grammar.

A probability distribution can be defined for multiple random variables: $P(D = d_{SWBD}, G = English)$ is the joint probability that the random variable $D$ takes on the value that is the set of verbs in the SWITCHBOARD corpus and that the random variable $G$ takes on a value which is the grammar for English. Defining joint distributions over multiple random variables allows inference problems to be represented as problems of probabilistic conditioning. Conditioning is a form of hypothetical reasoning which asks: what would the probability of an outcome be if something were true? In Bayesian approaches, conditioning formalizes all forms of belief update: learning, reasoning, abduction, induction, etc. The conditional probability $P(G = g|D = d)$ is the probability that the random variable $G$ takes on some value, $G = g$, given that another random variable $D$ has taken on value $D = d$. For example, it might be the probability that we assign to a grammar of English, given the evidence provided by the set of verbs that appear in the SWITCHBOARD corpus. Conditional probability is given by Equation 2.1.

$$P(G = g|D = d) = \frac{P(D = d, G = g)}{\sum_{g' \in G} P(D = d, G = g')} \quad (2.1)$$

Conditioning can be thought of in terms of a simple two-step algorithm. The algorithm starts with a joint distribution and a proposition about the joint distribution, called the conditioner. In general, the conditioner can encode any assumption, hypothesis, observation of data, or other beliefs—mental constructs which exist only in the mind of the believer—rather than measuring (putatively) physical properties of the external world, such as the limiting frequencies of large numbers of observations.
Bayes’ Rule

The chain rule, given in Equation 2.2, states that the joint probability of two random variables can be rewritten as the product of the conditional probability of one random variable given the other, multiplied by the prior probability of the other random variable.

\[
P(D = d, G = G) = P(D = d | G = g)P(G = g)
\] (2.2)

The chain rule allows us to rewrite the definition of conditional probability as follows.

\[
P(G = g | D = d) = \frac{P(D = d | G = g)P(G = g)}{\sum_{g' \in G} P(D = d | G = g')P(G = g')}
\]

This version of the formula for conditional probability is known as Bayes’ rule. The left-hand side of the equation is known as the posterior probability of the grammar \( G = g \) given the data \( D = d \). The first term in the numerator on the right-hand side is known as the probability of the data \( D = d \), given the grammar \( G = g \), or alternatively as the likelihood of the grammar, \( g \), given the data \( d \). The second term is known as the prior probability of the grammar \( G = g \). The denominator of the right-hand side of Bayes’ rule is known as the evidence or marginal likelihood of the data. Note that the marginal likelihood is just the probability of the particular dataset \( D = d \) summed, or marginalized, over all possible hypotheses. The marginal likelihood can be thought of as the prior probability of the data. Put another way, if a grammar is chosen at random from the prior over possible grammars, and, then, a random dataset is sampled from this grammar, the marginal likelihood gives the probability of the dataset. In Chapter 8, estimates of the marginal likelihood of the data will be used to study the prior distribution over possible datasets (samples of natural languages) implied by the Fragment Grammars model.

Bayes’ rule is important because it makes explicit a structure which is common in probabilistic models. Often models can be stated in terms of a set of unobservable hypotheses (in this case, grammars) and some prior distribution over those hypotheses. After observing data, drawn
from an unknown hypothesis in this space, Bayes’ rule can be used to update beliefs about which hypotheses were likely to have generated that data.

While Bayes’ rule is conceptually important, complex models frequently involve large numbers (often infinite) of random variables in complicated dependency relationships. In such settings, there are many potential conditional distributions which are of interest, and it is less useful to identify particular subsets of random variables as corresponding to likelihood or prior components of the model. As a result, it is often more natural to think of a complex model in terms of the joint distribution that it defines over all random variables, rather than in terms of Bayes’ rule.

### 2.1.2 Hierarchical Generative Modeling

There are different approaches to building probabilistic models. In this dissertation, I adopt a **hierarchical generative** approach to model-building—also known as **analysis by synthesis**. An alternative approach to building probabilistic models is **discriminative** modeling.

The difference between these two approaches can be best understood in terms of Bayes’ rule. As scientists or learners, we are often interested in the hidden causes behind the data we observe. In probabilistic terms we are interested, for example, in the distribution over (hidden) grammars given the (observable) data: $P(G = g | D = d)$. Discriminative models capture this distribution directly. For example, most classical regression models are discriminative—they provide a recipe for generating hypotheses (lines or curves) directly from the data (points).

Generative approaches, on the other hand, start from the observation that, because of Bayes’ rule, the probability of interest, $P(G = g | D = d)$, is proportional to $P(D = d | G = g)P(G = g)$; the factor which relates them is just the marginal likelihood of the data (the denominator in Bayes’ rule). Instead of giving a recipe to directly compute the probability of hypotheses given the data, $P(G = g | D = d)$, generative models specify $P(D = d | G = g)P(G = g)$ and then use conditional probability to reason about the former.

Hierarchical modeling refers to the construction of generative models in which the random variables are arranged in a hierarchical order which is meant to reflect the causal processes by which data is generated. For example, the three random variables discussed in Section 2.1.1 (i.e., $U$, $G$, and $D$) have a natural order which can be given a causal interpretation: Evolution selects a universal grammar $U = u$; Cultural evolution—through iterated learning of language by children—selects the grammar for a particular language $G = g | U = u$. Finally, specific samples of the language are generated by adults and provide the input to the next generation of learners: $D = d | G = g$. The joint distribution over all three random variables can be written as in Equation 2.3 where the ordering of the variables represents the dependencies in the causal chain.

$$P(D = d, G = g, U = u) = P(D = d | G = g)P(G = g | U = u)P(U = u) \quad (2.3)$$

### 2.1.3 Structured Models

The final element of the modeling framework in this thesis is the use of **structured** probabilistic models. The Bayesian approach treats probabilities as degrees of belief; however, it does not presuppose anything about the kinds of things that can be believed. There has been a tendency in the history of cognitive science to treat statistical approaches and more traditional symbolic approaches as fundamentally distinct. However, probability theory presupposes no such distinction—**structured**

---

3This has led, for instance, to unfortunate use of terminology—such as the use of the general term **statistical learning** to refer to what are essentially a class of Markov models (e.g., Safran, 2002, 2003).
modeling refers to the application of probability theory to traditional tools from classical cognitive science, logic, and artificial intelligence.

2.2 Probability Theory, Optimality and Cognitive Science

Bayesian approaches in cognitive science are often associated with the idea that models should be optimal. The search for optimal models in cognitive science can be interpreted in one of two ways: either as a hypothesis or alternatively as a methodological claim. The hypothesis has been stated most clearly by John Anderson in his Principle of Rationality (Anderson, 1990, 1991). In his words:

Principle of Rationality: The cognitive system optimizes the adaptation of the behavior of the organism.

In other words, the hypothesis states that cognitive systems provide (approximately) optimal solutions to the problems an organism faces in its environment. Anderson (1990) argues that this principle can be motivated from an evolutionary perspective: Since the brain has evolved to guide behavior, it might be expected that there has been pressure over evolutionary time for it to select solutions to behavioral problems which are optimal—within the constraints provided by biology.

Crucially, the principle of rationality is a hypothesis, not a conclusion that can be drawn a priori from evolutionary considerations. Evolution does not always optimize. While natural selection optimizes, there are other mechanisms of evolutionary change which do not—such as genetic drift (Gillespie, 2004). Even when evolution does optimize, evolutionary change is subject to strong constraints. Natural selection can only work with the material at hand. Moreover, prima facie, human cognition often seems to be highly suboptimal (Marcus, 2008).

It is sometimes argued that even if the principle of rationality turns out to be wrong for the human mind, it is still useful to develop models from an optimality viewpoint. Optimality is a concept which has two fundamental components. First, an optimal system has to be optimal with respect to some function or purpose that it serves. Second, for a system to be optimal, there must be some currency for measuring which solutions are better and which are worse. For a cognitive system, specifying a function means characterizing the information processing problem that the system solves. If the task of cognitive science is to reverse engineer the computational systems underlying cognition, the methodological perspective proposes that the first step is to understand the purpose of each system. This has been stated most clearly by Marr (1982).

An algorithm is likely to be understood more readily by understanding the nature of the problem being solved than by examining the mechanism (and the hardware) in which it is solved.

The intuition underlying this proposal is that understanding the function of the system provides strong constraints on the space of possible models of the system. Out of the many possible computations that a cognitive system could be performing, only a subset are related to its function. Understanding which solutions are better or worse further constrains the space of possibilities. John Anderson has used the term Rational Analyses to refer to this approach (e.g., Anderson, 1991). A rational analysis starts by precisely specifying the goals of the system and deriving the optimal behavior of the system with respect to these goals. If the actual performance of the system matches the predictions of the rational analysis, then this can be taken as evidence for its purported function, and the hypothesis that the system is optimal. On the other hand, when predictions do not bear
out, proponents of the approach argue that precisely specifying the problem being solved can be valuable in isolating flaws in the theory.

When rational analysis is adopted as a methodology in a Bayesian setting, the currency of optimality is probability: The optimal system is the one which makes the best predictions. An important reason that probability theory is often used in building rational models is the idea that probability theory is a normatively correct or optimal theory of uncertain reasoning. There have been two main classes of argument to this effect. In the first, the axiomatic approach, the laws of probability are derived from a set of axioms that are intended to characterize consistent, rational reasoning.

The first such axiomatic result was proven by Cox (1946) who proposed the following three axioms. Let $B(x)$ represent the degree of belief in some statement $x$.

1. Degrees of belief can be ordered. If $B(x) > B(y)$ and $B(y) > B(z)$ then $B(x) > B(z)$.

2. The degree of belief in a statement is related to the degree of belief in the negation of that statement. That is, there exists a function $f$ such that $B(\neg x) = f[B(x)]$.

3. The degree of belief in the conjunction of statements $B(x \land y)$ is related to the degree of belief in the conditional statement $x|y$ and the degree of belief in $y$. That is, there exists a function $g$ such that $B(x \land y) = g[B(x|y), B(y)]$.

Cox showed that probability theory is the unique theory that satisfies these three axioms—up to isomorphism. In subsequent decades, the axiomatic approach has been both extended and challenged in various ways (see e.g., Bernardo and Smith, 1994). Nevertheless the approach remains a cornerstone of theoretical justifications for Bayesian statistics.

A second class of arguments for the optimality of probability theory are so-called Dutch book arguments (de Finetti, 1974). Dutch book arguments start from the assumption that probabilities can be interpreted as degrees of belief in gambling outcomes. An individual’s degree of belief, $B(x)$, in a statement, $x$, is interpreted as the proportion of a wager that he or she would be willing to bet on $x$ being true. The term Dutch book refers to a set of bets which guarantees a net loss for one of the participants. de Finetti (1974) showed that a necessary and sufficient condition for guaranteeing that no Dutch book could be made against an individual was that all of his or her beliefs follow the laws of probability theory.

Axiomatic and Dutch book arguments both support the conclusion that using the laws of probability to reason about uncertain beliefs is normatively correct and mathematically optimal. In this thesis, I adopt both the methodology of rational analysis and the Bayesian notion that the correct measure of optimality is prediction accuracy. In later chapters, I show that the model proposed here explains a large number of phenomena in English morphology. This lends support to the idea that productivity and reuse can be viewed as an optimal inference.

2.3 The Modeling Framework from a Probabilistic Programming Perspective

In this section, I give a semi–formal description of the modeling framework from the perspective of probabilistic programming, using the Church language (Goodman et al., 2008). As I

---

4This discussion follows MacKay (2003).

5There are several extensive Church tutorials available at http://projects.csail.mit.edu/church/wiki/Church.
Chapter 2: The Framework

make clear in Chapter 3, the actual implementations of the models used for the simulations are slightly more complex than the Church presentations below and, for reasons of efficiency, do not make use of the Church inference engine. I have nevertheless included the Church formalization of the models in the thesis because it allows the models to be presented in a way which is clear, concise, and elegant. Because it is a stochastic variant of the $\lambda$–calculus, Church excels at the compact expression of recursive models. This, in turn, allows the ideas underlying the modeling framework to be built up in a compositional and stepwise fashion. I first define the structure–building recursion underlying Probabilistic Context–Free Grammars (PCFGs). I then extend this definition in steps, using stochastic generalizations of ideas from the theory of programming languages: first, *stochastic memoization* is used to extend PCFGs to Adaptor Grammars (AG); then, *stochastically lazy evaluation* is used to extend AG to Fragment Grammars (FG); and finally, these techniques are generalized to the space of all generative models. This stepwise progression is natural when the models are expressed in Church.

2.3.1 Distributions as Programs and Church

Section 2.1 describes the modeling framework of this thesis—structured Bayesian generative modeling—which emphasizes the definition of probability distributions over complex, structured objects. There are many different kinds of objects which have been used in structured probabilistic models: trees, graphs, logics, algebraic structures, real numbers, manifolds, etc. One way of unifying these disparate kinds of models is by viewing them all as programs for a natively probabilistic model of computation (Goodman et al., 2008; Mansinghka, 2009; Pfeffer, 2001; Roy, 2011).

A probabilistic model of computation can be defined by taking a deterministic model of computation and adding probabilistic primitive operations. For example, a well–known model of computation, the Turing machine, can be made probabilistic by giving it access to a primitive operation which flips an unbiased coin. As it performs a computation, the Turing machine can ask for the result of a random flip, and then change its behavior based on the outcome. Although they are very important as formal models, Turing machines are notoriously difficult to use as practical computing devices. Because of this, many approaches in probabilistic programming have chosen the $\lambda$–calculus as an underlying model of computation (Erwig and Kollmansberger, 2006; Goodman et al., 2008; Kiselyov and Shan, 2009; Pfeffer, 2001).

The $\lambda$–calculus was introduced in Church (1932) to formalize the notion of a *computable function*. John McCarthy showed that the $\lambda$–calculus could be made into a practical programming language (McCarthy, 1960), with the development of LISP (for List Processor). Programming languages based on the $\lambda$–calculus have come to be known as *functional programming languages*. In this thesis, I use a probabilistic variant of LISP called Church, after Alonzo Church (Goodman et al., 2008).

In functional programming, the basic units used to implement programs are *procedures*: recipes for computing various mathematical functions. Church uses a LISP–like syntax where function application is in Polish notation, and, therefore, the function comes first, followed by its arguments. Thus, in Church (and LISP more generally), the addition procedure $+$ applied to two arguments, 2 and 3, is written $\langle + \ 2 \ 3 \rangle$, rather than $2 + 3$.

The most important operator in the $\lambda$–calculus is $\lambda$ (which is written *lambda* in Church). This is the basic *abstraction* operator, which is used to define procedures. In Church, this has the syntactic form $\langle \lambda$ arguments body $\rangle$ where arguments is a list of input arguments to the procedure and body defines the computation that the procedure will carry out on those arguments. Figure 2.1 shows a simple example of a procedure definition. This procedure takes three arguments

---

6See also [http://pubs.doc.ic.ac.uk/ProbLambdaQPA/](http://pubs.doc.ic.ac.uk/ProbLambdaQPA/)
x, y and z, and adds them together. Procedures and other constructs can be named using the define
operator. This procedure is named add3.

(define add3
  (lambda (x y z)
    (+ x y z)))

Figure 2.1: A Simple Procedure in Church: A definition for a simple procedure. This procedure
adds the values of its three inputs.

In Church (and the \(\lambda\)-calculus) every expression has a value. The value is the outcome that
results from performing the computation defined by the expression. The the process of computing a
value for an expression is called evaluation. For example, the expression \((+ 1 2 3)\) evaluates to 6.
The expression \((\text{lambda} (x y z) (+ x y z))\) has a value which is itself a procedure. In functional
programming languages, procedures are first-class objects which means that they can be passed to
other procedures, returned by procedures, and stored in memory.

Probabilistic programming languages like Church add probabilistic primitives, called elementary random procedures (ERPs), to the language. A deterministic procedure always returns the
same value when applied to the same arguments. A random procedure, on the other hand, defines
a distribution over outputs given inputs. A familiar example of a random procedure is the \text{rand} or
\text{random} function of most programming languages, which typically returns a value in the interval \([0, 1]\)
with uniform probability. Another example of a random procedure is \text{flip} which flips an unbiased
coin, returning \text{true} or \text{false} with equal probability. A different version of flip takes an argument
which specifies the weight of the coin: \((\text{flip} 0.2)\).

An inventory of such basic ERPs can be used to construct more complex random procedures. Figure 2.2 shows a simple Church procedure which defines a function called noisy-or. With probability \(\text{epsilon}\), where \(0 \leq \text{epsilon} \leq 1\), noisy-or returns the result of applying \text{or} to its
arguments. However, with probability \((1 - \text{epsilon})\), it returns the opposite of that result (i.e., if
\text{(or arguments)} is true it returns false, and vice versa). Note that the syntactic form of an if
statement in Church is \((\text{if condition do-if-true do-if-false})\) where the first expression is a
logical condition, the second expression is evaluated if this condition is true, and, otherwise, the
third expression is evaluated.

(define noisy-or
  (lambda (epsilon boolean1 boolean2)
    (if (flip epsilon)
      (not (or boolean1 boolean2))
      (or boolean1 boolean2)))))

Figure 2.2: Church Code for Noisy Or Function: Church code defining a noisy version of
the logical operator \text{or}. This operator is identical to logical \text{or} except that with some probability,
\text{epsilon}, it reverses its output.

The probability of a particular evaluation of a complex random procedure is the product of all the
ERPs which are evaluated in the course of invoking the procedure. The distribution over outputs is
defined by these probabilities over all possible evaluation paths. For example, if noisy-or is applied to the arguments false and true (i.e., (noisy-or 0.01 false true)) then the probability of the outcome true is 0.99 and the probability of false is 0.01. In more complex examples, such as (noisy-or 0.01 (flip 0.7) (flip 0.7)), the probability of true depends both on the evaluation of the calls to flip that determine the two arguments, as well as the flip inside the procedure body.

A procedure in Church can be understood as a stochastic function—a mapping from the procedure’s inputs to a distribution over its outputs.7 If a procedure takes no arguments, then this mapping can be thought of as a probability distribution. If the procedure takes arguments, then it defines a family of probability distributions—one for each combination of arguments.8 Alternatively, a Church procedure can be seen as a sampler for a random variable. When a Church procedure is applied to some arguments, it will be evaluated to produce a sample from the distribution over outputs. For example, noisy-or can be understood as a sampler for a distribution over \{true, false\} which is determined by the arguments passed to the procedure.

Generative models for complex, structured representations can be constructed by using stochastic procedures defined in a probabilistic language like Church. An important technique for doing this is the use of recursive procedure definitions. A recursive procedure is a procedure which invokes itself somewhere in its own body. A well-known recursive function from mathematics is the Fibonacci function. This function maps from an integer \(n\) to the \(n\)th number in the Fibonacci sequence. The \(n\)th number in the Fibonacci sequence, \(F_n\), is defined inductively: \(F_0 = 0\), \(F_1 = 1\), and \(F_n = F_{n-1} + F_{n-2}\), as shown Equation 2.4.

\[
F_n = \begin{cases} 
0 & n = 0 \\
1 & n = 1 \\
F_{n-1} + F_{n-2} & \forall n > 1 
\end{cases} 
\quad (2.4)
\]

Figure 2.3 shows a recursive procedure that computes the \(n\)th Fibonacci number in Church.

(define fib
  (lambda (n)
    (case n
      (0 0)
      (1 1)
      (else
        (+ (fib (- n 1)) (fib (- n 2)))))))

Figure 2.3: Church Code for Fibonacci Function: Procedure to compute the \(n\)th Fibonacci number.

The computation of a Fibonacci number by the procedure shown in Figure 2.3 can be visualized as a tree; Figure 2.4 shows the tree resulting from applying this procedure to the input value 6. Each node in this tree represents an application of the fib function and the hierarchical organization reflects the recursive nature of the process.

Recursive procedures can be used to define distributions over infinite numbers of objects. For example, the geometric distribution is distribution over all non-negative integers. The probability of each integer \(i\) is given by \(p(1-p)^i\) where \(p\) is a probability: \(p \in [0, 1]\). The geometric distribution

---

7Stochastic functions are sometimes called a probability kernels.

8Families of probability distributions are often informally called “distributions.” For instance, the Gaussian distribution is actually an infinite family of related distributions each parameterized by a mean, \(\mu\), and a variance, \(\sigma^2\).
can be described with the following generative process. Start at 0; flip a coin with weight \( p \); if it comes up Heads, return 0 and stop; if it comes up tails, move on to 1 and repeat. The term \( (1 - p)^i \) in the expression \( p(1 - p)^i \) corresponds to the number of integers that were passed over by this process (i.e., where \( \text{flip} \) returned \text{false}). The term \( p \) corresponds to the choice which led to the procedure returning a value. In Church, this distribution can be represented by the simple recursive procedure shown in Figure 2.5.

This procedure samples from the integers, returning integer \( i \) with probability \( p(1 - p)^i \). The use of recursion allows this infinite distribution to be expressed with only a few lines of code. As discussed in the preceding chapter, models of linguistic structure make heavy use of recursion. I will exploit the ability of Church to succinctly represent these processes in the next section.

```
(define geometric
  (lambda (p)
    (if (flip p)
      0
      (+ 1 (geometric p)))))
```

Figure 2.5: Church Code for the Geometric Distribution: Probabilistic procedure that samples from a geometric distribution with parameter \( p \).

### 2.3.2 Structure Building with unfold

In Section 1.4.1, I introduced a simple model of computation, the PCFG (Probabilistic Context–Free [Phrase–structure] Grammars). PCFGs consist of a set of rules, like \( N \rightarrow \text{Adj} - \text{ness} \),
and a set of probabilities for each rule. Each rule in a PCFG can be thought of as a declarative statement about the possible composition of some constituent type. Implementing a PCFG as a computational model requires specifying a procedure which recursively uses the declarative knowledge in the rules to build expressions. This procedure is called unfold and takes either a terminal or nonterminal symbol as an argument. Recall that there may be many rules associated with a nonterminal in a PCFG. When unfold is passed a nonterminal as an argument, it samples one of the corresponding rules (i.e., a rules which has this nonterminal on its left-hand side). It then recursively calls itself on each symbol on the right-hand side of the chosen rule. Alternatively, if the symbol passed to unfold is a terminal, it simply returns the terminal, ending the recursion. This function can be defined succinctly by the Church procedure shown Figure 2.6.

```
(define unfold
  (lambda (symbol)
    (if (terminal? symbol)
        symbol
        (map unfold (sample-rhs symbol)))))
```

Figure 2.6: unfold: The unfold procedure implements the basic recursion underlying computation in a PCFG.

In Figure 2.6, the procedure application (terminal? symbol) returns true if symbol is a terminal and false otherwise. The procedure sample-rhs encodes the database of PCFG rules. It takes a nonterminal as an argument, and, when applied to this nonterminal, samples one of the rules that share this nonterminal on their left-hand side. One simple implementation of sample-rhs, which encodes the PCFG shown in Figure 1.7 of Chapter 1, is shown in Figure 2.7. Note that there are many equivalent ways to implement sample-rhs; the details shown in Figure 2.7 are just one possibility.

The procedure map takes two arguments. The first is a procedure and the second is a list of values: (map procedure list). It applies procedure to each element of list and returns the new list that results. For example, for a simple procedure which adds one to each input value, (define add1 (lambda (n) (+ n 1))), the procedure application (map add1 (list 1 2 3)) returns the list (2 3 4). In the body of unfold, map recursively applies unfold to each symbol on the RHS of the rule that is sampled by sample-rhs. For example, if (sample-rhs 'N) were to return (Adj -ness), (map unfold '(Adj -ness)) will result in the recursive calls: (unfold 'Adj) and (unfold -ness).

A particular evaluation of unfold can be visualized with a tree like the one in Figure 2.8. This figure shows one possible evaluation of unfold when applied to the nonterminal N, which has resulted in the sampling of the word: agreeability. Note that because unfold is a stochastic procedure, this is only one of many words that might have been sampled by this application.

Many linguistic generative models can be formalized as variants of unfold. The impor-

---

9 Note that in Church the quote symbol, ', tells the interpreter not to evaluate an expression. 'N is just the nonterminal symbol N. Without a quote, N is treated as the name of a variable referring to some other value. Likewise '(A B C) is a list with three symbols. The unquoted list (A B C) is treated as the procedure A applied to input arguments B and C.

10 The name unfold derives from the fact that this procedure is the dual of the fold procedure (also sometimes called reduce or inject) which is common in functional programming. Fold-type procedures take a structure and reduce them to a single value by recursively applying a function which combines substructures. Unfold-type procedures take a value and build a structure by recursively applying a function, like sample-rhs, that expands the input at each step. These operations can be generalized and formalized as category-theoretic constructs known as catamorphisms (fold) and
(define sample-rhs
  (lambda (nonterminal)
    (case nonterminal
      (('W) (multinomial (list (list 'N) (list 'V) (list 'Adj) (list 'Adv) ... )
            (list p W1 p W2 p W3 p W4 ...)))
      (('N) (multinomial (list (list 'Adj 'ness) (list 'Adj 'ity) (list 'electro 'N) (list 'magnet) (list 'dog) ...)
            (list p N1 p N2 p N3 p N4 ...)))
      (('V) (multinomial (list (list 'N 'ify) (list 'Adj 'ize) (list 're 'V) (list 'agree) (list 'count) ...)
            (list p V1 p V2 p V3 p V4 ...)))
      (('Adj) (multinomial (list (list 'dis 'Adj) (list 'V 'able) (list 'N 'ic) (list 'N 'al) (list 'tall) ...)
            (list p Adj1 p Adj2 p Adj3 p Adj4 p Adj5 ...)))
      (('Adv) (multinomial (list (list 'Adj 'ly) (list 'today) ...)
            (list p W1 p W2 ...) ))))

Figure 2.7: Procedure for Sampling the RHS of Rules: sample-rhs takes a nonterminal symbol as input and samples one of the possible RHSs associated with that nonterminal by the PCFG.

2.3.3 Remembering Earlier Work with Memoization

Memoization refers to a widely-used technique in computer science where the results of computations are stored for later reuse. To memoize a procedure, each call to the procedure is intercepted. If the procedure has been called before with the current arguments, the result is looked up in a table—called the memotable—and returned. Otherwise, the procedure is first evaluated on the arguments, the result is stored in the memotable, and, finally, it is returned to the caller.

Because computations—especially those involving recursion—often consist of frequently reoccurring sub-computations, memoization can sometimes lead to a significant improvement in efficiency. This idea can be best illustrated with an example. As I discussed in Section 2.3.1, the Fibonacci function can be defined as a recursion (see Figure 2.3). A naive, recursive implementation of the Fibonacci function is very inefficient because, during computation, fib is applied to the same arguments many times. Consider Figure 2.4 of Section 2.3.1. In the pictured tree (fib 2) is evaluated 5 times, and (fib 4) is evaluated twice. If, instead of recomputing these values each time they are needed, the values are stored after their first evaluation, and then subsequently reused, the time needed to compute a value will be greatly reduced. Figure 2.9 shows the considerable reduction in work that is possible by use of memoization.12 The naive implementation takes a

anamorphisms (unfold) (Meijer et al., 1991). In Appendix A, I discuss the possibility of associating a function with each rule of a PCFG in order to compute phonological or semantic structures. The composition of fold and unfold can be used to reduce a sampled tree into a phonological string, or an expression encoding the truth conditions of an expression: (phonological-fold (syntactic-unfold symbol))) or (semantic-fold (syntactic-unfold symbol))).

11The actual version of unfold for the MDPCFG model places a Dirichlet prior on the rule weights in the underlying grammar. See Section 3.1.4 for details.

12Note that I am assuming a depth-first evaluation order in Figure 2.9.
number of recursive calls which is exponential in the input, \( n \), while the memoized version is linear in \( n \).

In many programming languages, including Church, a procedure implementing memoization—called \texttt{mem}—is provided as a primitive in the language. The function \texttt{mem} is a higher-order procedure—a procedure which takes other procedures as an argument. In the case of \texttt{mem}, a memoized version of the input procedure is returned. Thus \texttt{(mem fibonacci)} will return a memoized version of \texttt{fibonacci} which intercepts calls and checks if they are in the memotable before repeating work.

Because of its potential to improve efficiency, memoization has been applied widely in the design of algorithms, especially dynamic programming algorithms, which figure prominently in linguistic applications such as parsing (see e.g., Jurafsky and Martin, 2000; Manning and Schütze, 1999).\(^{13}\) It has also played a prominent role in the implementation of lazy functional programming languages (see below).\(^ {14}\)

### Stochastic Memoization

In a deterministic setting, every function that is applied to some arguments has one unique output value (or none at all if the procedure never returns). Therefore, memoizing a deterministic procedure cannot change its meaning: The mathematical function which it computes remains the same. By contrast, in a probabilistic setting, a procedure samples from a distribution over values.

\(^{13}\)Also see Johnson (1995); Norvig (1991) for a discussion of the relationship between memoization in functional programming languages and CFG parsing algorithms, especially Earley parsing.

\(^{14}\)For example, Haskell: \url{http://www.haskell.org/haskellwiki/Haskell}
Therefore, memoization fundamentally changes the distribution over return values and, hence, the meaning of the memoized procedure. The first value sampled from a memoized stochastic procedure (applied to some particular arguments) will be drawn from the distribution defined by the procedure without memoization. However, subsequent calls to the procedure (with the same arguments) will deterministically return the same value. The models in this thesis, however, require the ability to reuse earlier computed values probabilistically. In order to do this, the notion of memoization must be generalized to stochastic memoization.

A deterministic memoizer builds a memotable which associates each input with a particular output. A stochastic memoizer associates each input with a distribution over output values (Goodman et al., 2008). This distribution, called the memoization distribution, specifies the degree of belief that each previously computed value will be reused on the next call. This distribution must also place some probability mass on the possibility of calling the underlying procedure to sample new values which can be added to the memotable.

Memoization is usually understood as a technique for improving program efficiency in modern computer science. However, the technique was originally introduced by Michie (1968) as a method of learning in artificial intelligence. In this paper, Michie explicitly highlights how memoization allows systems to mix rule–based and memory–based computations in a single domain.

A function is a function, and the means chosen to find its value in a given case is independent of the function’s intrinsic meaning. This is no more than a restatement of a mathematical truism, but it is one which has been lost sight of by the designers of our programming languages. By resurrecting this truism we become free to assert: (1) that the apparatus associated with any given function shall consist of a “rule part” (computational procedure) and a “rote part” (look-up table); (2) that evaluation in the computer shall on each occasion proceed either by rule or rote, or by a blend of the two, solely as dictated by the expediency of the moment; (3) that the rule versus rote
decisions shall be handled by the machine behind the scenes; and (4) that various kinds of interactions be permitted to occur between the rule and the rote part. Thus each evaluation by rule adds a fresh entry to the rote.

Michie applied memoization to the problem of learning in a control system. Michie’s notion of memoization, however, was not probabilistic, and, thus, he introduced various other techniques to model uncertainty.\(^\text{15}\) The technique of stochastic memoization proposed in Goodman et al. (2008) can be seen as extending Michie’s proposal to the setting where there is uncertainty about which computations to reuse and which to compute anew.

Stochastic memoization of probabilistic generative models—when combined with Bayes’ rule—provides a recipe for doing inference over a set of stored computations. When we condition a memoized procedure on some data, we are asking which set of stored fragments of computation were likely to have been (re)used to generate the data. The computations which were likely to have generated the data are assigned high probability, making it more likely that they are reused again in the future.

Balancing Novelty and Reuse with Pitman–Yor Processes

There are many possible distributions one could use to implement a stochastic memoizer. In this section, I describe the one used in this dissertation: the Pitman–Yor process. The problem investigated in this dissertation revolves around balancing productivity and reuse. This requires solving the inference problem: For a given linguistic process, how likely is it that the next instance generated will be novel, or, alternatively, how likely is it that the next instance will be identical to a former computation? There are at least two kinds of information that are relevant for making such predictions. First, a natural (purely distributional) predictor for reuse of a computation is how often it was (re)used in the past. Second, a natural (purely distributional) predictor of the productivity of a computation is how often it has generated new observations in the past. The former kind of information requires tracking the *token frequency* of computations. The latter kind of information requires tracking their *type frequency*. The Pitman–Yor process integrates exactly these two kinds of information (Pitman and Yor, 1995). Following Johnson et al. (2007a) and Goodman et al. (2008) I will use it as a memoization distribution.

The Pitman–Yor process (PYP) is a distribution from non–parametric Bayesian statistics which is most easily described in terms a of *sequential sampling scheme* using the metaphor of a restaurant.\(^\text{16}\) Imagine a restaurant with an infinite number of tables. The first customer enters the restaurant and sits at the first unoccupied table. The \((N + 1)\)th customer enters the restaurant and sits at either (i) an already occupied table with probability \(\frac{y_i}{N+b}\), or (ii) at a new table with probability \(\frac{K+b}{N+b}\). The variable \(N\) is the total number of customers in the restaurant; \(K\) is the total number of occupied tables, indexed by \(1 \leq i \leq K\); \(y_i\) is the number of customers seated at table \(i\); \(0 \leq a \leq 1\) is the *discount parameter* of the model; and, \(b \geq -a\) is the *concentration parameter* of the model.

\(^{15}\)For example, he proposed that the identity operator—which determines which inputs should correspond to the same outputs in the memotable—should be similarity based (rather than using strict equality), and that this similarity function should be learned.

\(^{16}\)The term *non-parametric* refers to statistical models whose complexity can grow with the data, rather than being specified in advance.

\(^{17}\)This construction of the Pitman–Yor process is called the *Chinese Restaurant Process* (CRP) in, for example, Pitman (2002). However, in the literature the term *Chinese Restaurant Process* is now generally reserved for a PYP where the parameter \(a\) is set to 0.
Customers sit at an already-occupied table with probability proportional to the number of individuals at that table (minus a constant discount factor $0 \leq a \leq 1$), or at a new table with probability proportional to $b$, plus the total number of previously occupied tables multiplied by the discount factor. This is illustrated in Figure 2.10.

\[ N=0 \begin{array}{c} ? \vdots \end{array} \]
\[ K=0 \]

$1$

\[ N=1 \begin{array}{c} v_4 \vdots \end{array} \]
\[ K=1 \]

\[ \frac{1-a}{1+b} \quad \frac{a\cdot1+b}{1+b} \]

\[ N=6 \begin{array}{c} v_4 \quad v_1 \quad v_{16} \vdots \end{array} \]
\[ K=3 \]

\[ \frac{3-a}{6+b} \quad \frac{1-a}{6+b} \quad \frac{2-a}{6+b} \quad \frac{a\cdot3+b}{6+b} \]

Figure 2.10: The Pitman–Yor Process: This figure shows one possible series of distributions that might result from the sequential sampling scheme for the Pitman–Yor process. Shown is the distribution over the next customer after $N$ customers have been seated at $K$ tables. Associated with each table is a value $v_i$ which is drawn from the underlying function that has been stochastically memoized with the Pitman–Yor process.

To use a PYP as a memoization distribution for some procedure $\text{proc}$, we associate a restaurant with each possible combination of arguments to $\text{proc}$. Each table in each restaurant is labeled with a value called a dish that is sampled from $\text{proc}$, and then shared by all customers seated at the table. When a Pitman–Yor process is used as a memoization distribution, each table represents a reusable computation type which was sampled from the procedure which has been memoized. The probability of sampling a new computation type depends on the number of new types that were sampled in the past. Customers represent particular token instances of reuse of the values at each table. The probability of reuse depends on the number of times each value was reused in the past. In what follows, I will assume a higher-order procedure $\text{PYmem}$ which implements a Pitman–Yor based stochastic memoizer. Thus, the procedure $\text{proc}$ can be memoized with the syntax $(\text{PYmem } a \ b \ \text{proc})$.

---

18 The underlying distribution from which dishes are sampled is called the base distribution in the literature.

19 Note that multiple tables can share the same label. This is a necessary consequence of the exchangeability of the PYP. Nevertheless, it is the case that the highest probability assignments of customers and dishes to tables will be those which assign a single unique dish to each table.
Properties of the Pitman–Yor Process

The usefulness of the Pitman–Yor process as a model of reuse and novelty lies in the way that it assigns probability mass to different kinds of outcomes. The first thing to observe is that the PYP implements a simplicity bias. It assigns a higher probability to restaurants: (i) that have fewer customers; (ii) that have fewer tables; (iii) that for a fixed number of customers, $N$, assign the customers to the smallest number of tables. Furthermore, the PYP implements a rich–get–richer scheme. The probability of reusing an existing table is proportional to the number of customers seated at that table. As the number of customers at a table increases, so does the probability that the next customer will be seated at that table. Consequently, the process prefers to seat observations with the same type at the same table whenever possible.

The PYP also encodes another important—and opposing—bias. The probability of a new table, $aK + b$, depends on $K$—the number of already occupied tables. The number $K$ is just the number of novel tables that the PYP–memoized process has generated in the past—the type frequency of the process. Thus, in a Pitman–Yor process, the probability of generating a novel table is a function of the number of novel tables that have been generated in the past: The more often that novelty has occurred in the past, the more likely it will be in the future. This bias is controlled by the discount parameter, $a$. Note that when $a = 1$, every customer will be seated at a their own table.

Relationship between Pitman–Yor Processes and Activation–Based Models in Psycholinguistics  In psycholinguistics, there has been a long tradition of models which makes use of activation–based, localist representations (e.g., Baayen and Schreuder, 1999, 2000; Bybee, 1988, 1995b; Chialant and Caramazza, 1995; Hay, 2003; Laudana and Burani, 1985; Marslen-Wilson et al., 1997; Morton, 1969, 1970; Schreuder and Baayen, 1995; Taft, 1994). These models are localist in the sense that they consist of units which correspond to linguistic constituents such as words or morphemes. They are activation–based in the sense that each of these units has an associated real–valued activation which determines the speed with which the unit is accessed, or the likelihood that it takes part in computation. As these units are used over time, their activation level can change, leading the systems to favor some units over others. Typically, such systems use an additive update rule which increments each unit by a constant factor, $\epsilon$, each time it is used.

For readers familiar with these psycholinguistic models, it may be useful to describe the dynamics of a Pitman–Yor process in terms of additive change in the probability of individual units (tables) as these are used during processing. As I will show below, this perspective also highlights several properties of the system which are less obvious in the sequential sampling presentation.

First, consider the case where the process reuses some existing table. Let $p_t$ represent the probability of reusing that table at time $t$. The probability after sampling another value from that table (at time $t + 1$), $p_{t+1}$, is given by Equation 2.5.

$$p_{t+1} = p_t + C \ast (1 - p_t) p_t$$  (2.5)

Because the Pitman–Yor process is described by a sequential sampling scheme in which the entire system remains a properly normalized probability distribution after each sample, the additive update cannot be a constant, $\epsilon$. Instead, the magnitude of the probability change to $p_t$ depends on $p_t$ itself, as show in Equation 2.5.

$C$ is a term—common to all the equations in this section—that guarantees that each update leads to a properly normalized probability. It is equal to $\frac{1}{N + b}$, which is just the normalizing constant of the entire distribution after the value is sampled (i.e., at time $t + 1$). This can be understood as a unit conversion: Because the total number of observations has changed, the units of belief at time $t$ must be converted into units at time $t + 1$. Furthermore, because $C$ has $N$ in its denominator, as
more data is observed, each datapoint will cause a smaller change in \( p_t \)—the units in which belief is measured get smaller as more data is observed.

\( C \) is weighted by the product of the probability of the table at time \( t \) (i.e., \( p_t \)) multiplied by the probability of all other tables that *might* have been sampled, but were not: \((1 - p_t)\). Observing a customer from a table always leads to an increase in the probability of that table. However, as the probability increases, the magnitude of the change reduces. The change is maximized when the \( p_t = (1 - p_t) \), that is, when there is equal probability that a customer will or will not be sampled from the table. In other words, the amount that the probability changes in either direction, the magnitude of the change will decrease.

Equation 2.6 shows the change in probability of each of the existing tables that were not reused after observing the sample at time \( t \).

\[
p_{t+1} = p_t - C \cdot p_t
\]

(2.6)

All of the other probabilities (including the probability of making a new table) are reduced by an amount proportional to themselves, times the normalizer \( C \). This means that while the probability of each *unobserved* outcome is reduced relative to the *observed* outcome, their probabilities stay identical relative to one another.

There is one more case to consider: the sampling of novel value and the creation of a new table. Equation 2.7 shows the change in probability of novel table creation after new table is created.

\[
p_{t+1} = p_t + C \cdot (a - p_t) p_t
\]

(2.7)

The change in probability of future novelty is related to the probability before sampling by the term \((a - p_t)\). Because \(0 \leq a \leq 1\), when the probability of novelty is greater than \(a\), sampling another novel value will *decrease* this probability. However, when the probability is less than \(a\), sampling a novel value will *increase* this probability. Thus the parameter \( a \) of the model can be interpreted as the *target* probability of novelty: In the limit, the probability of creating a new table will equilibrate towards the value of the \( a \) parameter.

Another interesting property of the \( a \) parameter is that, in the limit, it corresponds to the expected proportion of tables in the restaurant that will have only a single customer (Teh, 2006). In Section 6.3.3, I will discuss several measures of productivity which are based on *hapax legomena*. A hapax legomenon is a form which occurs only once in some sample. In the limit, \( a \) represents the proportion of (table) hapaxes in the Pitman–Yor process.\(^{20}\)

Finally, it is well-known that the frequencies of many natural language structures follow a power–law distribution (Baayen, 1992; Chitashvili and Baayen, 1993; Yang, 2009; Zipf, 1935). This means that the probability that a structure will occur with frequency \( f \) in a large corpus is proportional to \( \frac{1}{f^g} \) for some exponent \( g \).\(^{21}\) The Pitman–Yor process generates such a power law distribution over frequencies in the limit of many samples. Specifically, the probability of an outcome having a particular frequency \( f \) is proportional to \( \frac{1}{1 + a} \). As the \( a \) parameter increases, so does the frequency of low probability outcomes (Goldwater et al., 2009; Pitman, 2006).\(^{22}\)

\(^{20}\)Adjustment of the \( a \) parameter can be interpreted as causing the Pitman–Yor process to interpolate between using type and token frequencies in estimating the probability of various outcomes. Goldwater et al. (2006) and Teh (2006) show how this fact can be used to derive a popular heuristic for smoothing in \(n\)–gram models, knowns as Kneser–Ney, which makes use of both type and token information.

\(^{21}\)When \( g = 1 \) this is the *Zipf* distribution.

\(^{22}\)For natural language word frequency distributions \( g \) is estimated at \( \approx 1.8 \) by Goldwater et al. (2009).
2.3.4 Stochastic Memoization of unfold: Adaptor Grammars

In the preceding sections, I introduced the notion of stochastic memoization and defined and discussed the Pitman–Yor process as a memoization distribution, stressing the way in which it balances reuse and productivity. In this section, stochastic memoization is applied to the probabilistic unfold recursion that was discussed in Section 2.3.2. The resulting model, known as Adaptor Grammars, was introduced in Johnson et al. (2007a).

(define adapted-unfold
  (PYmem a b
    (lambda (symbol)
      (if (terminal? symbol)
        symbol
        (map adapted-unfold (sample-rhs symbol))))))

Figure 2.11: Adaptor Grammars: By stochastically memoizing unfold, the results of earlier computations can be reused. The resulting model is known as Adaptor Grammars.

Adaptor Grammars can be defined in Church as shown in Figure 2.11. Each time adapted-unfold is invoked, the call will be intercepted by the memoizer and either (i) a previously computed expression will be returned, or (ii) a new expression will be computed, stored in the memotable, and then returned. Although this model is very simple to express, it can result in surprisingly complex behavior. Because adapted-unfold is recursive, when the memoizer chooses to build a new value, the subconstituents of that value will be also sampled from memoized distributions. Therefore, even when novel expressions are produced, they will frequently contain parts which are reused.

A key to understanding the Adaptor Grammars model is recognizing the kinds of objects that can be stored by the memoizer. When a novel expression is sampled, the memoizer calls the underlying procedure and waits for it to return a complete expression. For example, if adapted-unfold is applied to the symbol \( \text{W} \), it can either choose to retrieve a complete word from the memotable, or it can build a new word from scratch (possibly reusing subparts). It can not, however, sample a partial word.

Because of this, Adaptor Grammars can be interpreted as a modern, probabilistic version of the full-listing models from classical lexicalist theories, discussed in the last chapter. In the simulations reported later in the thesis, I will use an implemented version of Adaptor Grammars to instantiate this theory.\(^{23}\)

2.3.5 Lazy and Eager Evaluation Strategies

In order to turn the \( \lambda \)–calculus into a practical model of computation, one must fix an evaluation strategy for programs. This can be illustrated with an example. Consider the procedure add3 discussed in Section 2.3.1.\(^{24}\)

Imagine that this procedure is applied to three complex expressions as arguments, for example, \((\text{add3} (\text{+} 1 2 3) (\text{*} 2 4) (\text{-} 3 1))\). There are two ways in which the final value of this procedure application can be computed. These are shown in Figure 2.12.

---

\(^{23}\)The version of Adaptor Grammars used in this thesis and developed originally in Johnson et al. (2007a), differs slightly from the presentation in this section. The actual implementation of Adaptor Grammars uses an underlying PCFG in which rule weights have a Dirichlet prior—see Sections 2.4.2 and 3.1.7 for details.

\(^{24}\)See Abelson and Sussman (1996) for a detailed introduction to evaluation strategies.
Chapter 2: The Framework

56

Eager Evaluation

Eager Evaluation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(add3 (+ 1 2 3) (* 2 4) (- 3 1))</td>
<td>→ (add3 6 8 2)</td>
</tr>
</tbody>
</table>

Lazy Evaluation

Lazy Evaluation

<table>
<thead>
<tr>
<th>Expression</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>(add3 (+ 1 2 3) (* 2 4) (- 3 1))</td>
<td>→ (+ (+ 1 2 3) (* 2 4) (- 3 1))</td>
</tr>
</tbody>
</table>

Figure 2.12: Evaluation Strategies: The eager evaluation strategy first evaluates all arguments to a procedure, and then applies the procedure to these arguments. The lazy evaluation strategy waits until an argument value is needed before computing it.

In the top row of the figure, each argument expression is first evaluated, and then these values are substituted into the body of the add3 procedure. Only after this is done, is the body of the procedure evaluated to produce a final output for the whole computation. An evaluation strategy like that shown in the first row of Figure 2.12 is known as applicative order or eager evaluation in functional programming.

By contrast, in the bottom row, the expression passed as arguments to the procedure are not evaluated immediately; rather, they are substituted into the body of add3 first. Only when their values are needed by +, are they evaluated. This evaluation strategy is known as normal order or lazy evaluation in functional programming. Under lazy evaluation, arguments to a procedure are delayed until another part of the program requires the values to continue the computation. These delayed objects are known as promises since they “promise” to return a value when needed. When the value of a delayed promise is needed, it is forced. Programming languages which make use of lazy evaluation force the value of delayed objects at two places: the top–level of the entire program, and whenever delayed values are passed to primitive procedures, such as +.

The Church–Rosser theorem for the deterministic λ–calculus states that the order in which expressions are evaluated does not affect the final result of the computation (Hindley and Seldin, 2008). Nevertheless, the choice of evaluation strategy can still have impact on the efficiency of computation. To see this consider the procedure in Figure 2.13.

(define silly-procedure
 (lambda (first-argument second-argument)
   (if true
     (do-something first-argument)
     (do-something-else second-argument))))

Figure 2.13: Advantages of Being Lazy: This figure shows a procedure where there is an advantage to a lazy evaluation strategy. In this example, the value of the second argument is never needed and therefore an eager evaluator wastes work computing it.

This procedure takes two arguments: It only evaluates the second argument when the condition in the if statement is false. However, in this example, the if statement is always true and, therefore, the second argument will never be used. Suppose that this procedure was applied to two complex expressions:

(silly-procedure expression1 expression2)
Under eager evaluation, both expressions will be evaluated before they are passed to `silly-procedure`. The work involved in evaluating `expression2` will be wasted. By contrast, a lazy evaluator will not do this extra work.

While this example is admittedly silly, there are many real-life cases in complex programs where the value of a particular argument is not be needed. This has led to the development of lazy functional programming languages such as Haskell. Lazy languages can lead to significant savings in computation.

```
(define silly-procedure2
  (lambda (argument)
    (do-something argument argument argument)))
```

Figure 2.14: Advantages of Being Eager: This procedure shows a situation where it is advantageous to be eager. A lazy evaluator evaluates the input three times, while an eager evaluator only does so once.

However, lazy evaluation also has disadvantages which are illustrated in Figure 2.14. The procedure `silly-procedure2` takes a single argument which it then uses three times. Under an eager evaluation strategy, this argument would be evaluated before it was passed to `silly-procedure2` and thus the computation involved would be performed only once and shared between the three uses in the body of the procedure. However, the lazy evaluator would repeat this work three times, unnecessarily.

Laziness and Eagerness in a Probabilistic Language

Like memoization, the choice of lazy versus eager evaluation strategies does not fundamentally change the meaning of a deterministic program. However, as is also true of memoization, it does have an impact on the meaning of probabilistic programs. This can be illustrated with the simple example shown in Figure 2.15.

```
(define check-equality
  (lambda (argument)
    (if (= argument argument)
      (do-something)
      (do-something-else))))
```

Figure 2.15: A Procedure with Different Meanings under Stochastic Lazy and Eager Strategies: This procedure takes a single argument and checks if it is equal to itself.

This figure shows a procedure called `check-equality` which takes a single argument and calls either `do-something` or `do-something-else` depending on whether or not the argument is

---

25 http://www.haskell.org/haskellwiki/Haskell

26 In real-world lazy languages, this problem is solved by memoizing the result of computations so that they are only computed once. This strategy is known as *call-by-need* evaluation. In a stochastic setting, a call-by-need strategy can restore applicative order semantics to a lazy evaluator (see the next section).
equal to itself. Consider what happens when this procedure is applied to the expression (flip):
(check-equality (flip)). Under an eager evaluation strategy, the expression (flip) is evaluated
once, before the value is passed into check-equality. In this case, the condition of the if statement
is always true, and, therefore, do-something will always be called. However, with a lazy evaluation
strategy the expression (flip) is reduplicated and evaluated twice. The condition is only true with
probability 0.5.

Thus, the choice of a lazy versus an eager evaluation strategy has semantic consequences
in a probabilistic language like Church. Under an eager evaluation strategy, when an argument is
needed in more than one place in a procedure body, its value will be shared. Under a lazy evaluation
strategy, parts of a procedure which use the same argument will share only the distribution over
values given by that expression. In a stochastic setting, eager evaluation corresponds to reusing a
previously computed value while lazy evaluation corresponds to preserving the potential productivity
of the expression.

Stochastic Lazy unfold

In this section, I propose a version of unfold which mixes lazy and eager evaluation strate-
gies probabilistically. By stochastically mixing lazy and eager evaluation, this procedure will be able
to return arbitrary, partial computations, and, in combination with stochastic memoization, will
result in the Fragment Grammars model.

To develop a stochastically lazy version of unfold, I assume a version of Church that
is primarily eager, but comes equipped with a delay statement which delays the evaluation of
an expression until its value is needed.27 In this version of Church, it is possible to write the
stochastically lazy version of unfold shown in Figure 2.16.

(define delay-or-unfold
  (lambda (symbol)
    (if (flip)
        (delay (stochastic-lazy-unfold symbol))
        (stochastic-lazy-unfold symbol))))

(define stochastic-lazy-unfold
  (lambda (symbol)
    (if (terminal? symbol)
        symbol
        (map delay-or-unfold (sample-rhs symbol)))))

Figure 2.16: Stochastic Lazy Unfold: This version of unfold is stochastically lazy. When
passed a nonterminal as an argument it first samples a RHS. For each symbol on that RHS, it
calls delay-or-unfold. This procedure randomly chooses to continue or delay the recursion.

Figure 2.17 shows a possible sequence of computations for stochastic-lazy-unfold ap-
plicated to the nonterminal N. The procedure first samples a rule for N, in this case: N \rightarrow Adj -ity.
Once this rule has been sampled, it decides for each symbol on the RHS whether or not to delay

27 I assume that the interpreter recursively forces delayed expressions at the top–level and when they are needed as
the arguments of primitive procedures. stochastic-lazy-unfold could also be implemented in a version of Church
which was fundamentally lazy, but contained a primitive procedure, non-recursive-force, which only forced a single
delayed value (force is typically conceived of as a recursive operator). The code would be identical to Figure 2.16
with delay replaced by non-recursive-force.
further computation. Because -ity is a terminal, the recursion stops on that branch. The procedure continues to recurse on Adj by sampling the rule Adj → V -able. Once again, it decides whether or not to delay each of the symbols on the RHS of this rule. In the case of -able, it does not delay, and, because -able is a terminal, computation ends. In the case of V, the procedure delays the recursive call and returns the delayed object. At this point there is no more work to be done, and the procedure returns.

\[
\begin{align*}
&\text{(unfold 'N)} \\
&\quad \text{(sample-rhs 'N)} \\
&\quad \text{(dealy-or-unfold 'Adj)} & \text{(dealy-or-unfold 'ity)} \\
&\quad \text{(unfold 'Adj)} & \text{(unfold 'ity)} \\
&\quad \text{(sample-rhs 'Adj)} & \\
&\text{(dealy-or-unfold 'V)} & \text{(dealy-or-unfold 'able)} \\
&\text{(delay (unfold 'V))} & \text{(unfold 'able)} \\
& & \text{'able}
\end{align*}
\]

Figure 2.17: Possible Computation of stochastic-lazy-unfold: stochastic-lazy-unfold chooses whether or not to delay each recursion on the RHS of sampled rules. Here it has expanded a noun with the affixes -able and -ity but delayed the sampling of V. The delayed branch of execution is highlighted in red.

The result is an object in which some of the potential recursive branches of computation have been completed, but others have been delayed. Such partially delayed objects correspond to derivational tree fragments in a PCFG. For example, the delayed object in Figure 2.17 corresponds to the partially evaluated PCFG computations discussed in Section 1.4.2.28

The procedure stochastic-lazy-unfold demonstrates how mixing lazy and eager evaluation strategies can lead to the sampling of partial computations during the evaluation of a probabilistic program. The return value of stochastic-lazy-unfold is a complex structure which includes delayed subparts. However, there is a still a problems with the system just defined. The delayed objects it generates are ephemeral: They cease to exist at the end of the particular evaluation which created them. What is required is a way of storing such partial computations for later reuse. In the next section, I show how this can be done with stochastic memoization.
(define delay-or-unfold
  (lambda (symbol)
    (if (flip)
        (delay (fragment-unfold symbol))
        (fragment-unfold symbol)))
)

(define fragment-unfold
  (PYmem a b
    (lambda (symbol)
      (if (terminal? symbol)
          symbol
          (map delay-or-unfold (sample-rhs symbol))))))

Figure 2.18: Fragment Grammars: The Fragment Grammars model results from stochastically memoizing stochastic-lazy-unfold. The resulting procedure can reuse partial computations across multiple calls.

2.3.6 Fragment Grammars

Having outlined the techniques of stochastic memoization and stochastic lazy evaluation, I can now define the model proposed in this thesis: Fragment Grammars. By memoizing stochastic-lazy-unfold, partial subcomputations can be shared across multiple calls to the procedure shown in Figure 2.18.

Each time fragment-unfold is called, one of two things will happen: The procedure will return a previously sampled partial computation from the memotable, or it will sample a new partial subcomputation from the underlying procedure. In either case, delayed subparts of this computation will ultimately be forced, and a complete expression will result. Like Adaptor Grammars, the sampled computations are defined recursively, and, therefore, when a novel partial computation is sampled, it may be built of previously stored parts.

2.3.7 A Generalization: fragment-lambda

In the preceding sections, I developed the Fragment Grammars model by stochastically memoizing a stochastically lazy version of the unfold recursion. A natural question is whether the same techniques can apply to other kinds of generative models. In fact, it is possible to give a generalization which can apply to any procedure in Church and—because Church is universal in the space of probabilistic generative models—can, therefore, apply in principle to any generative model.

I propose a novel language construct called fragment-lambda which—like \( \lambda \) in the \( \lambda \)-calculus—can be used to define procedures. It differs from \( \lambda \) in that the procedures it defines automatically stochastically reuse partial subcomputations as they are invoked. The operator fragment-lambda can be defined via a program transformation as shown in Figure 2.19.

---

28 The delayed call to \( \text{unfold} \ 'V \) will eventually be forced by the interpreter at the top-level, and a full expression will be sampled.

29 The version of Fragment Grammars used in this thesis differs slightly from the presentation in this section. In particular it (i) places a Dirichlet prior on the weights of underlying PCFG rules, and (ii) places a beta–distributed
Figure 2.19: Fragment Lambda: Shown here is a program transformation implementing a new form of abstraction operator—called fragment-lambda—which allows arbitrary procedure definitions to automatically reuse subcomputations when they are invoked.

Figure 2.20 shows this program transformation applied to the unfold recursion. Under this transformation, when the unfold recursion is applied to each symbol on the RHS of a sampled PCFG rule, it will sometimes return a delayed object representing the application itself, and sometimes recurse. The resulting model is very similar to the definition of Fragment Grammars given above.\(^{30}\)

The technique encapsulated by the fragment-lambda operator opens up the possibility of developing lexicalized models for other areas of cognition. For instance, an influential theory in the realm of human image understanding is the recognition–by–components theory—also known as the geon theory (Biederman, 1987). In this theory, it is hypothesized that objects are recognized as assemblages of simple geometric components such as blocks and cylinders (known as geons). An implementation of such a model with fragment-lambda would allow increasingly complex, reusable components to be built out of simpler ones—leaving certain subcomponents underspecified as variables. There are, of course, many open questions about modeling and inference with this approach which must be left to future research. However, fragment-lambda illustrates how the idea of reuse of partial subcomputations provides a domain–general way of thinking about the interaction of productive, structure–building processes and memory (see Taatgen, 2003, for a similar proposal).

2.4 The Models Examined in the Thesis and Related Work

In this section, I discuss the five formal models which I will evaluate in later portions of the thesis. The discussions of these models in Sections 2.3 and 1.4.4 elided some details which are prior on the probability of delaying each branch of subcomputation. See Section 3.1.8 for details.

\(^{30}\)In the fragment-lambda–defined model, it is possible for a category like N to be stored as a partial computation at a table in the memoizer. By contrast, Fragment Grammars only store partially expanded subtrees of at least depth 1 on tables in their memoizer.
crucial for understanding their behavior in later simulation results. Here, I will provide the intuitions necessary to understand their performance.

### 2.4.1 Full–Parsing: Multinomial–Dirichlet PCFG (MDPCFG)

Recall from Section 1.4.4 that the full–parsing approach to productivity and reuse stores only minimal–sized fragments which are equivalent to rules in the underlying CFG. This approach is formalized using Multinomial–Dirichlet Probabilistic Context–Free Grammars, which I will abbreviate as MDPCFG throughout the thesis. A MDPCFG is a probabilistic context–free grammar where the probabilities of individual rules have a Dirichlet prior distribution (Johnson et al., 2007b; Kurihara and Sato, 2006). In a Bayesian setting, putting a prior on these rule weights allows them to be learned from the data. Intuitively, a MDPCFG learns rule weights which are proportional to the token frequency of each rule in the training corpus.

In nearly all of the simulations presented in later chapters, the MDPCFG underperforms the other models. This is a function two properties of this model. First, because it only uses minimal–sized fragments of structure, this model cannot capture any combinations of structure which are frequently reused in the data. Second, even when minimal–sized fragments of structure are appropriate, it is often inappropriate to estimate their productivity based on token frequency. As I will explain in later chapters, a more appropriate estimator in many (but not all) such cases is the type frequency of a rule. However, MDPCFG cannot take advantage of this fact.

As I explained in Section 1.4.4, because it has long been understood that morphology is characterized by sharp constraints on the productivity of word–formation processes, the full–parsing theory does not correspond directly to any proposal from the literature; it is included in the thesis as a baseline. However, the full–parsing model does correspond more closely to many proposals in the domain of syntax. In particular, several studies employing computational models closely related to Fragment Grammars (see Section 2.4.4) have demonstrated excellent parsing results using grammars which store a number of fragments that is only a few times greater than the number in the minimal CFG (Cohn et al., 2010, 2009; Post and Gildea, 2009; Zuidema, 2007).

### 2.4.2 Full–Listing: Adaptor Grammars (AG)

In section 1.4.4, I discussed the full–listing approach to productivity and reuse, which stores all computations in their entirety after the first time they are computed. This model is formalized by the Adaptor Grammars (AG) model of Johnson et al. (2007a).

AG has a number of subtle properties which are important for understanding its behavior in the simulations reported in later chapters. Although AG deterministically stores each computation in its entirety when it is first computed, it reuses these stored computations probabilistically. There are two important points to make about how AG reuses stored computations. First, the probability of reusing a stored computation is proportional to its token frequency in the corpus.

Second, it is always possible for AG to choose to invoke the underlying PCFG to account for a form compositionally—even when that exact form is already stored in the memoizer. In general, AG will prefer to reuse previously computed forms, but this preference can be controlled by adjusting the $a$ and $b$ parameters of the Pitman–Yor memoization distributions. If, for example, the $a$ parameter for some memoizer is set to 1 that memoizer will never reuse earlier work, but will, instead, always use the underlying PCFG rules to construct forms. Johnson and Goldwater (2009) show that the

---

31 Note that this is a rule–of–thumb for high–probability posterior states; the exact probability will vary across different posterior states.

32 As was the case with MDPCFG, this is a rule–of–thumb for high–probability posterior states.
performance of AG can be improved by learning these hyperparameters for each nonterminal in the grammar. This technique allows the system to learn that some categories of constituent store their computations, while others always productively compute fresh forms. This approach differs from Fragment Grammars because it allows levels of productivity to be learned only on a per–category basis, rather than at the granularity of individual words and rules. In this thesis, only the simplest version of (maximum a posteriori) AG, which does not learn $a$ and $b$ parameter weights, is employed.

Another important aspect of the AG model is the way in which it estimates the probability of the rules in the underlying PCFG. These rules receive a probability which is proportional to the number of stored computations in which they take part—in essence, they are estimated based on type frequency. As I will explain in subsequent chapters, because type frequency is often a good approximate estimator of productivity, AG often exhibits good generalization performance. For example, an Adaptor Grammar will learn that novel forms in -ness (e.g., wugness) are more probable than novel forms in -ity (e.g., wugity) due to the higher type frequency of -ness. However, this phenomenon is limited to single rules in the underlying PCFG, such as $N \rightarrow Adj$ -ness. In particular, AG can never learn that a combination of morphemes such as -ability can also generalize productively.

Nevertheless, AG’s performance across most of the evaluations in this thesis is second only to FG, and, in a few cases, better. This is a result of the fact that in the domain of morphology, storage is pervasive; full–listing is a good approximation to the true state of affairs. This is further supported by the success of AG in a number of studies which have looked at other aspects of word structure—such as morpheme segmentation (e.g., Johnson, 2008a,b; Johnson and Goldwater, 2009).

Recall, from the previous section, that the full–parsing model provides reasonable performance on syntactic datasets. Taken together with the remarks above, this is a further argument for an inference–based approach: While full–listing does a respectable job for morphology and full–parsing does a respectable job for syntax, only an inference–based framework can work for both. In fact, the results reported in Cohn et al. (2010, 2009); Post and Gildea (2009); Zuidema (2007), together with the results in this thesis, imply that inference–based models do better in both syntax and morphology. This is a natural consequence of the mixture of productivity and reuse across all levels of linguistic structure which I discussed in Section 1.3.

### 2.4.3 Exemplar–Based Productivity and Reuse: Data Oriented Parsing (DOP)

In section 1.4.4, I discussed the exemplar–based approach to productivity and reuse, which stores all generalizations consistent with the data. In this thesis, I formalize the exemplar–based approach with two versions of the Data Oriented Parsing framework (Bod et al., 2003)

There are many different techniques which fall under the DOP umbrella. They differ primarily in how they estimate the probability of stored fragments. I will use two variants of DOP. The first, known as Data Oriented Parsing I (DOP1), sets the probability of a subtree proportional to its token frequency in the training corpus (Bod, 1998; Scha, 1990).

There a number of well–known problem with DOP1. For example, Johnson (2002) shows that the DOP1 estimator is both biased and inconsistent. Another problem with DOP1 is that it tends to overweight training data nodes which appear higher and in larger trees, and, thus, produces a pronounced bias towards larger stored items (see e.g., Goodman, 2003). For this reason, I also explore a second DOP estimator, called the Goodman Estimator for DOP33 (GDMN) (Bod, 2003; Goodman, 2003), which assigns equal weight to each training data node and each training data item. The GDMN estimator for DOP has been shown to significantly improve performance in syntactic parsing

---

33Named for Joshua Goodman (Goodman, 2003).
Chapter 2: The Framework

compared to DOP1, and is currently the version of DOP which performs best on standard parsing evaluations (Bod, 2003).

Although there are many other variants, in its classical form, DOP is committed to the storage of all subtrees, and it is this feature that corresponds to the exemplar–based approach. Some versions of DOP have relaxed the all–subtrees assumption by introducing techniques which penalize either large subtrees or long derivations (see various chapters in Bod et al., 2003). These approaches can be seen as moving toward the use of optimal sets of subtrees, rather than all subtrees. In other words, these approaches move in the direction of the inference–based approach advocated here. DOP1 and GDMN on the other hand, straightforwardly encode the idea of storing all generalizations consistent with the data.

Both DOP models differ from the other models examined in this thesis in one very important way: While all of the other models in this thesis commit to a single derivation for each piece of observed data, the DOP models spread their probability mass across all possible derivations which are consistent with the input data. This has profound consequences: DOP cannot commit to a belief that some form was derived productively or that it was stored, instead it hedges across both possibilities. In the coming chapters, there will be many examples of cases where this causes the two DOP models to underperform the other models.

Intuitively, the two DOP models differ from one another in a simple way: DOP1 assigns more probability mass to larger subtrees, and, therefore, tends to behave more like AG; GDMN assigns more probability mass to smaller subtrees, and, therefore, tends to behave more like MDPCFG. In general, this means that DOP1 tend to outperform GDMN on most of the morphological evaluations reported in this thesis. As I mentioned above, GDMN outperforms DOP1 when applied to syntax. These facts reemphasize the pattern I mentioned above: a bias towards reuse is more appropriate for morphology, while a bias towards productivity is more appropriate for syntax, but the best performance is exhibited by a system which can learn productivity and reuse on a case–by–case basis.

2.4.4 Inference–Based Productivity and Reuse: Fragment Grammars (FG)

The inference–based model advocated in this thesis, Fragment Grammars (FG), differs from the earlier models in that it tries to find the set of stored items and points of productivity which best explain the data. It inherits from AG the ability to store any subcomputation. However, it generalizes AG by allowing stored subcomputations to include variables which require productive computation. Whereas AG can only productively compute forms using the underlying PCFG, FG can productively compute forms by reusing stored structures with variables. Like AG, FG estimates the probability of stored items based on their token frequency and the probability of underlying PCFG rules based on their type frequency.

Another class of all–fragment or all–subtree approaches is based on the use of kernel methods. In such methods, a function—called a kernel—is defined which measures similarity between pairs of inputs (e.g., pairs of trees). Using the kernel, an input training data set can be transformed into an inner product space in which discriminative machine learning techniques can be applied (e.g., support vector machines or perceptrons). Collins and Duffy (2002) show how to define tree kernels which compute the similarity between two trees by considering all subtrees, but in a way that sidesteps many of the efficiency problems inherent in DOP.

Some other versions of DOP are aimed at different problems entirely. For example, Unsupervised Data–Oriented Parsing (Bod, 2006) is aimed at the problem of unsupervised grammar induction, and is, therefore, not strictly comparable to the present models.

Again, these are rules–of–thumb for high–probability posterior states.
Chapter 2: The Framework

FG is similar to AG and MDPFCG—and different from the two DOP models—in that it commits to a single derivation for each form in the corpus. However, it differs from the other models in an important way. Although the other models are probabilistic, and, therefore, inferential in some respect, they all follow a semi–deterministic update rule. Consider how the probability distributions of each model change with respect to a single, newly observed tree: MDPFCG always increases the probability of every minimal, abstract fragment that occurs in the tree; AG always increases the probability of the entire tree (whether it was reused whole, or composed and stored). The two DOP models increase the probability of every subtree of the target tree. In all of these cases, the set of subtrees (in the top–level) that will be updated is determined in advance by the definition of the model.

By contrast, for FG, the set of subtrees which are updated depends on earlier update choices. FG chooses an analysis of an input tree in which structure is shared in an optimal way with earlier analyses. One consequence of this is that FG exhibits a kind of attractor behavior. In particular, it prefers that all instances of individual word types be analyzed in the same way. For example, if a corpus contains 50 copies of a form such as goodness, FG prefers to treat these 50 copies as all retrieved whole or all composed. The system also prefers to treat all instances of a particular subform, such as -ness, in a consistent manner. If it decides that -ness is productive, then there is pressure to treat all forms which use it as composed; if it decides that -th is unproductive, then there is pressure to treat all forms which use this suffix as retrieved whole. Although these are only prior preferences—there can be exceptions—they are an important part of the explanation for why FG is able to account for the linguistic data.

Finally, because of the attractor behavior of FG, it can undergo sudden reorganization. When the evidence that a particular affix is productive or reused reaches a tipping point, it can cause many forms to be reanalyzed, leading to a reorganization of the whole system.

Other Inference–Based Models of Reuse

There are several other models in the literature which can be seen as explicitly adopting the idea that productivity and reuse should be treated as an (optimal) inference. Zuidema (2007) introduces Parsimonious Data–Oriented Parsing (PDOP), a version of DOP which explicitly eschews the all–subtrees approach in favor of finding a set of subtrees which best explains the data. PDOP uses a heuristic algorithm, similar to the inside–outside algorithm (see Manning and Schütze, 1999), to search for good sets of subtrees. Two other models, developed simultaneously with the current framework, make use of similar Bayesian non-parametrics to define generative models over sets of subtrees (Cohn et al., 2010, 2009; Post and Gildea, 2009). These two approaches differ from Fragment Grammars in that they do not define the distribution over stored structures recursively—a new stored item cannot be built out of other stored items. In addition, all of these related approaches have been evaluated primarily on syntactic datasets, and primarily in a natural language processing—rather than psychology—setting. I leave it to future work to explore their performance on the empirical domains studied in this thesis.

37Note that AG update is only deterministic for tree fragments stored in the top–level of the system. The counts associated with CFG base rules may or may not update, depending on whether a top–level tree was composed or retrieved from memory during computation.

38Another related model is presented by Blunson and Cohn (2010) who use Pitman–Yor processes (PYPs) to learn probabilistic dependency grammars. Johnson (2007) showed how bilexical dependency grammars can be transformed into an equivalent Context–Free Grammar representation. Blunson and Cohn (2010) use this transformation to learn a tree–substitution grammar which encodes non–local dependency relations across large subtrees via PYP–memoized backoff (see, e.g., Goldwater et al., 2006; Teh, 2006).
2.4.5 Categorial Refinement Approaches

All of the models used in this thesis are defined with respect to a space of possible trees which can be characterized using (Probabilistic) Context–Free Grammars (PCFGs). It is well–known that the strong conditional independence assumptions of PCFGs make them ill–suited to capturing linguistic structure (see, e.g., Jurafsky and Martin, 2000, for discussion), because PCFGs can only enforce dependency relationships between constituents that are local to a single rule. Fragment Grammars, Adaptor Grammars, and Data–Oriented parsing can be understood as models which relax the conditional independence assumptions of PCFGs by allowing the storage of larger fragments of structure which can capture dependency relationships that cannot be encoded by single PCFG rules.

In computational linguistics, there is another widely–used technique for relaxing the conditional independence assumptions of PCFGs: categorial refinement (e.g., Bansal and Klein, 2010; Collins, 1997; Johnson, 1998; Klein and Manning, 2003). Under this approach, a PCFG is enriched by adding additional information to categories (i.e., nonterminals) in the grammar—effectively splitting or refining these categories into subcategories, which encode more stringent selectional restrictions. There are many kinds of information which can be added to PCFG categories: information about morphological features such the tense of a verb phrase or the number of a noun phrase (e.g., Jurafsky and Martin, 2000; Sag et al., 2003), information about the heads of phrases or lexical items (e.g., Collins, 1997), and structural information such as the categories of the parents or grandparents of individual tree nodes in a training corpus (e.g., Johnson, 1998; Klein and Manning, 2003).

Category refinement gives rise to systems of categories which are similar in spirit to the traditional feature–based representations used in (nearly all) linguistic theories of syntax (e.g., Bresnan, 2001; Radford, 2004; Sag et al., 2003), morphology (see, Spencer, 1991), and phonology (see, Goldsmith, 1996). Recent work in computational linguistics has focused on machine learning techniques which learn the category refinements which best predict training corpora (e.g., Bansal and Klein, 2010; Petrov and Klein, 2007a,b).

One important set of techniques for learning category refinements are based on Bayesian non–parametrics similar to those used in this thesis (Finkel et al., 2007; Liang et al., 2007). For Fragment Grammars and Adaptor Grammars, non–parametric distributions are used to represent an unbounded set of tree fragments which can be reused in future computations. In non–parametric categorial–refinement models, these non–parametric distributions are used to represent an unbounded set of subcategories for a given training corpus category. For example, the category of noun phrases (NP), can be associated with a Pitman–Yor restaurant where each table represents one of a potentially infinite number of subcategories of NP, such as PLURAL NPs or DAYS–OF–THE–WEEK. Using such techniques, these systems can learn a set of refinements which best predict the distribution of constituents in a training corpus (Finkel et al., 2007; Liang et al., 2007).

One important aspect of categorial refinement is that, representationally, it includes storage of large substructures as a special case. For example, if Fragment Grammars store a tree fragment of the form (N (Adj V -able) -ity), this can be encoded as two PCFG rules with refined categories: N → Adj1 -ity and Adj1 → V -able. The refined category Adj1 enforces the rewriting of these two rules as a single unit, essentially simulating the single stored tree.

Therefore, categorial–refinement models, in general, and non–parametric categorial–refinement models, in particular, may be able to provide an alternate account of the phenomenon studied in this thesis. Many linguistic accounts of both the English past tense and English derivational morphology make heavy use of features on rules which enforce selectional restrictions (see Chapters 4 and 6), and these features could possible be learned via categorial–refinement techniques. For example, in

---

39 These techniques are often found under the heading of state–split PCFGs.
some approaches to morphology, the productivity of irregular past tense rules (e.g., /ɪ/ → /æ/ in sing/sang) is controlled by annotating such rules with features marking the specific stems to which the rules can apply (see Section 4.4.2). In the present framework, by contrast, the productivity of such rules is controlled by storing irregular stems together with their past tense rules. Similarly, theories of derivational morphology make extensive use of feature annotations to mark selectional restrictions on rules (see Section 6.3.1). Therefore, categorial–refinement approaches may be able to provide an account of the empirical domains studied in this thesis, and it is an open question for future work how such accounts compare to storage–of–large–structures approach espoused by Fragment Grammars.
Chapter 3

Formalization of the Models and Inference

In this chapter, I present the mathematical details of the models studied in this thesis. I also discuss the inference algorithms used for each of the models and various issues of practical concern for the simulations that I report later. This chapter can be skipped by readers not interested in the mathematical details of the models. Relevant sections are referenced later in the thesis, as needed.

3.1 Formal Details of the Models

The mathematical formalization of the models which I present in this section relies heavily on their development from the perspective of probabilistic programming which was described in Section 2.3. The order here also roughly follows the order in that section. It is recommended that the reader read this section after or together with Section 2.3.

3.1.1 Context–free Grammars

In Section 1.4.1, I explained that all of the models studied in this thesis use context–free grammars (CFGs) as their underlying model of computation. In this section, I formalize this model. Context-free grammars are a simple, widely-known, and well-studied formalism for modeling hierarchically structured computation (see Autebert et al., 1997, for a review of formal results on CFGs). Formally, a context-free grammar is a 4-tuple, $G = (V_G, T_G, R_G, W)$.

- $V_G$ is a finite set of nonterminal symbols.
- $T_G$ is a finite set of terminal symbols (distinct from $V_G$).
- $R_G \subseteq V_G \times (V_G \cup T_G)^*$ is the set of production rules.
- $W \in V_G$ is a unique, distinguished start symbol.

By convention, nonterminals are written with capital letters, and, when used to model linguistic structure, they represent categories of constituents such as “noun phrase” (NP), “verb” (V) or “noun” (N). The unique, distinguished nonterminal known as the start symbol is often written $S$ (for “sentence”). However, because this thesis focuses on the domain of morphology, I will use
\( \mathcal{W} \) instead—to refer to the category of “word.” This symbol represents the category of complete derivations: sentences for syntax, words for morphology. Terminals, written with lowercase letters, typically represent atomic words or morphemes (e.g., agree, or -ness).

Production rules, which are written \( A \rightarrow \gamma \), where \( \gamma \) is some sequence of terminals and nonterminals, and \( A \) is a nonterminal, define the set of possible computations for the system. For example, the rule, \( N \rightarrow \text{Adj} -\text{ness} \), says that a constituent of type \( N \) can be constructed by first constructing an adjective \( \text{Adj} \), and then concatenating this adjective with the morpheme -ness. The list of symbols to the right of the arrow is referred to as the right–hand side (RHS) of that production. The nonterminal to the left of the arrow is the rule’s left–hand side (LHS). The functions \( \text{lhs}(r) \) and \( \text{rhs}(r) \) return the left–hand side and right–hand side of rule \( r \), respectively. The language, \( \mathcal{L}_A \), associated with nonterminal \( A \) is the set of expressions which can be derived from that nonterminal.

A CFG encodes the possible choices that can be made in computing an expression, but does not specify how to make the choices: It chooses expansions for nonterminals non–deterministically. Like many other non-deterministic generative systems, CFGs also have a natural probabilistic formulation known as Probabilistic Context–Free Grammars (PCFGs).

### 3.1.2 Multinomial Probabilistic Context–Free Grammars

In this section, I consider multinomial probabilistic context–free grammars and the distributions over trees and expressions which they define. A multinomial distribution is the simplest way of specifying a distribution over a finite number of choices—it corresponds to a vector of probabilities. A multinomial PCFG is the classic version of probabilistic CFG in which the distributions over rule choices are governed by multinomial distributions.

As mentioned in Section 1.4.1, a derivation tree (or just derivation) is a tree representing the trace of the computation of some expression \( \vec{w} \), beginning from some nonterminal category \( A \). I will call a derivational tree complete if all of its leaves are terminal symbols. A derivation tree fragment is a derivation tree whose leaves may be some combination of terminals and nonterminals. Given a derivation \( d \), define the function \( \text{yield}(d) \) to be the function which returns the leaves of a derivation (complete or fragmentary) as a list. Define the function \( \text{root}(d) \) to be the function which returns the nonterminal at the root of the tree. Define the function \( \text{top}(d) \) to be the function which returns the production rule (i.e., depth–one tree) at the top of \( d \). We say that a nonterminal \( A \) derives some expression, represented as a list of terminals \( \vec{w} \in T^* \), if there is a complete derivation \( d \) such that \( \text{root}(d) = A \) and \( \text{yield}(t) = \vec{w} \).

Formally, a multinomial PCFG, \( \langle G, \{\theta^A\}_{A \in \mathcal{V}_G} \rangle \), is a CFG \( G \) together with a set of probability vectors \( \{\theta^A\}_{A \in \mathcal{V}_G} \). Each vector \( \theta^A \) represents the parameters of a multinomial distribution over the set of rules that share \( A \) on their left–hand sides. We write \( \theta^A_r \) or \( \theta_r \) to mean the component of vector \( \theta^A \) associated with rule \( r \). Because each vector, \( \theta^A \), represents a probability distribution, they satisfy Equation 3.1.

\[
\sum_{r \in \mathcal{R}} \theta^A_r = 1 \tag{3.1}
\]

By convention, I will label the \( k \) immediate children of a root node of some derivation \( d \) as \( \hat{d}_1, \ldots, \hat{d}_k \), that is \( \text{yield}(\text{top}(d)) = (\text{root}(\hat{d}_1), \ldots, \text{root}(\hat{d}_k)) \). The basic structure–building recursion for PCFGs, described in Section 2.3.2, can be expressed mathematically by the following recursive stochastic equation (i.e., probability mass function).
This equation states that the probability of a tree, \( d \), given by the multinomial PCFG, \( G \), is the product of the probability of the rules used to build that tree from depth-one subtrees, summed over all rules whose left-hand side matches the root of \( d \), and whose right-hand side matches the immediate children of the root of \( d \).\(^1\) Note that the distribution over full derived structures is given by \( G^a \).

Multinomial PCFGs make two strong conditional independence assumptions. First, all decisions about expanding a nonterminal are local to that nonterminal; they cannot make reference to any information other than the identity of the nonterminal and the multinomial distribution associated with it. Second, (as a consequence of the first independence assumption) expressions themselves are generated independently of one another. These assumptions together mean that, given some multinomial PCFG, \( G \), the counts of individual rules used in some set of computations are sufficient statistics for \( \{\theta^r\}_{k \in R_G} \). In other words, if we know the counts of individual rules, we can compute the probability of a set of derivations without knowing where or how the individual rules were used. The probability of a derivation \( d \) is just the product of the probabilities of the individual rules it contains, as shown in Equation 3.2.

\[
P(d|G) = \prod_{r \in d} \theta_r \tag{3.2}
\]

The probability that a PCFG assigns to a particular expression (i.e., sequence of terminals) \( \vec{w} \) is computed by marginalizing (i.e., summing) over all derivation trees which share that expression as their yield.

\[
P(\vec{w}|G) = \sum_{d} P(d|G) \tag{3.3}
\]

Given an expression \( \vec{w} \) and a rule \( r \) we define the inside probability of \( \vec{w} \) given \( r \) as:

\[
P(\vec{w}|r,G) = \sum_{d} P(d|G) \tag{3.4}
\]

The inside probability of a sequence of terminals \( \vec{w} \), given rule \( r \) is the probability that the sequence is the yield of some complete tree for \( \vec{w} \) whose topmost depth-one subtree corresponds to \( r \).\(^2\)

Define a corpus of derivation trees, \( D \), of size \( N_D \), with respect to a grammar to be the trees which result from deriving \( N_D \) expressions from the start symbol \( W \). Let \( X = \{\vec{x}^a\} \) be the set of count vectors for each rule used in \( D \). The function \texttt{counts} takes a set of derivation trees and returns the corresponding vectors of rule counts: \( \texttt{counts}(D) = X \). I will occasionally abuse notation

\(^1\)Typically, for a PCFG this set of rules will be singleton. However, there is no formal requirement that this be true, and the expression including this sum is more consistent with the other models presented below.

\(^2\)Note that I have defined inside probabilities with respect to rules; in most work, they are instead defined with respect to nonterminal categories. The inside probability of a sequence of terminals \( \vec{w} \) given non-terminal \( A \) is computed by additionally marginalizing over rules that share \( A \) on their LHS: \( P(\vec{w}|A,G) = \sum_{r} \sum_{d} P(d|G) \).
and use \texttt{counts} to mean the function that returns the rule counts associated with some particular derivation tree: \texttt{counts}(d(i)) = \texttt{counts}\{\{d(i)\}\} = x_d(i). The probability of a set of derivation trees, \( D \), under a PCFG can be computed from the count vectors, \texttt{counts}(D) = X, according to the function \texttt{pcfg} below.

\[
P(D|G) = \text{pcfg}(X; G)
= \prod_{k \in V} \prod_{r \in R_k} \left[ \delta_k^r \right]^{x_k^r} \quad (3.5)
\]

Where \( x_k^r \) is the number of times that rule \( r \) with LHS \( k \) was used in the corpus. That is, the probability of a corpus of derivations under a multinomial PCFG is given by the product of the probabilities of all the rules used in the corpus.

\subsection{3.1.3 Multinomial-Dirichlet Distributions}

In this section, I define what is known as the Polya–urn representation of the multinomial-Dirichlet distribution. A multinomial distribution over \( K \) elements is specified by a parameter which is given as a vector of probabilities: \( \theta \). A multinomial–Dirichlet distribution places a Dirichlet prior on the vector \( \theta \).

The resulting hierarchical multinomial–Dirichlet distribution can be represented by a finite analog of the rich–get–richer sequential sampling scheme described for the Pitman–Yor process in the last chapter.\footnote{The Dirichlet is commonly used as the prior because it is conjugate to the multinomial. The notion of conjugacy is an important one in Bayesian statistics (see, e.g., Gelman et al., 2003); it refers to pairs of distributions—such as the multinomial and Dirichlet distributions—where the posterior distribution over the parameters is in the same family of distributions as the prior. For example, the posterior distribution over \( \theta \), when the prior is Dirichlet, and the likelihood is multinomial, is also a Dirichlet distribution.} Define the following sequential process: the first observation, \( v^{(1)} \), is sampled according to the following equation.

\[
v^{(1)}|\pi_1, \ldots, \pi_K \sim \frac{\pi_1}{\sum_{i=1}^{K} \pi_i} \delta_{v^{(1)} = v_1} + \cdots + \frac{\pi_K}{\sum_{i=1}^{K} \pi_i} \delta_{v^{(1)} = v_K} \quad (3.6)
\]

Where \( \delta_X \) is a \( \delta \)-distribution which returns 1 if the proposition \( X \) is true and 0 otherwise, and the \( \pi \)'s are pseudo–counts, which can be thought of as counts of imaginary prior observations of each of the \( K \) possible outcomes.

After \( N \) observations have been sampled, the \((N + 1)\)th observation is sampled according to the following distribution:

\[
v^{(N+1)}|v^{(1)}, \ldots, v^{(N)}, \pi_1, \ldots, \pi_K \sim \frac{\pi_1 + x_1}{\sum_{i=1}^{K} [\pi_i + x_i]} \delta_{v^{(N+1)} = v_1} + \cdots + \frac{\pi_K + x_K}{\sum_{i=1}^{K} [\pi_i + x_i]} \delta_{v^{(N+1)} = v_K} \quad (3.7)
\]

Where \( x_i \) is the number of previous draws of value \( v_i \). This process implements a simple rich–get–richer dynamic which increases the probability of each outcome as more values of that outcome are sampled. Note that, unlike a multinomial, the draws are \textit{not} independent, rather the probability of the next draw is dependent on the history of previous samples. Nevertheless, the draws are \textit{exchangeable}. A sequence of random variables is exchangeable if it has the same joint distribution.

\footnote{The fact that this hierarchical process can be represented as a sequential sampling scheme is the result of de Finetti’s Theorem (de Finetti, 1974; Freer and Roy, 2009)—a fundamental result in Bayesian statistics which states that all \textit{exchangeable} sequences of random variables can also be defined in terms of a hierarchical process where observations are drawn i.i.d. (i.e., independently and with identical distribution) from a random measure.}
under all permutations. In other words, the joint probability over a fixed set of \( N \) observations does not depend on their order.

An alternate way of thinking about a multinomial-Dirichlet distribution is that it assigns probabilities to different partitions of \( N \) objects amongst \( K \) bins. Each bin represents an observation type and each object represents an observation token. Let \( \vec{x} \) be a vector of length \( K \) which counts the number of observation tokens for each observation type. The probability of a particular partition can be given by the following equation.

\[
P(\vec{x}|\vec{\pi}) = \prod_{i=1}^{K} \frac{\Gamma(\pi_i + x_i)}{\Gamma(\sum_{i=1}^{K} \pi_i + x_i)} \prod_{i=1}^{K} \Gamma(\pi_i)
\]

(3.8)

Where \( \Gamma(\cdot) \) is the gamma function—a continuous generalization of the factorial function; in particular, \( m! = \Gamma(m+1) \) for discrete \( m \).

### 3.1.4 Multinomial–Dirichlet Probabilistic Context–Free Grammars

In this section, I define a variant of probabilistic context–free grammars where the individual rule probabilities are drawn hierarchically from a Dirichlet distribution. This variant, known as Multinomial–Dirichlet Probabilistic Context–Free Grammar will serve as the formalization of the full–parsing theory throughout this thesis.

For multinomial PCFGs, discussed in Section 3.1.2, the set of weight vectors \( \{\vec{\theta}_A\}_{A \in V_G} \) is pre–specified as a model parameter. It is possible instead to draw these weight vectors from a Dirichlet prior (Johnson et al., 2007b; Kurihara and Sato, 2006).\(^5\) The resulting Multinomial–Dirichlet Probabilistic Context–Free Grammar (MDPCFG) associates a Dirichlet distribution with each nonterminal in its underlying CFG \( G \); these Dirichlet distributions are each parameterized by a vector of pseudo–counts. Thus, a MDPCFG, \( M = (G, \{\vec{\pi}_A\}_{A \in V_G}) \), is a context-free grammar together with a set \( \{\vec{\pi}_A\}_{A \in V_G} \) of vectors of pseudo–counts.

The distribution over derivations \( d \) described by an MDPCFG can be represented by the following recursive stochastic equations.

\[
G_{\text{mdpcfg}}^a(d) = \begin{cases} 
\sum_{r \in R_G: a \rightarrow \text{root}(d), \ldots, \text{root}(d)} \theta_{r} \prod_{i=1}^{k} G_{\text{mdpcfg}}(\hat{d}_i) & \text{root}(d) = a \in V_G \\
1 & \text{root}(d) = a \in T_G 
\end{cases}
\]

(3.9)

\[
\vec{\theta}_A \sim \text{dirichlet}(\vec{\pi}_A)
\]

(3.10)

Note that these equations are identical to those for the multinomial PCFG except that the probability vectors, \( \vec{\theta}_A \), are now drawn hierarchically from \( \text{dirichlet}(\vec{\pi}_A) \). The version of MDPCFG employed in this thesis uses the Polya–urn representation of the multinomial–Dirichlet distribution.

Given a corpus of derivation trees, \( D \), the counts of individual rule uses, \( X = \{\vec{x}_A\} \), provide sufficient information to compute the probability of the corpus. The probability of a corpus of derivations, \( D \), under an MDPCFG, \( M \), can be given in terms of these counts by the function \text{mdpcfg} below.

\[
P(D|M) = \text{mdpcfg}(X; M)
\]

(3.11)

\[
P(D|M) = \prod_{A \in V} \left[ \frac{\prod_{i=1}^{K} \Gamma(\pi_i^A + x_i^A) \Gamma(\sum_{i=1}^{K} \pi_i^A)}{\Gamma(\sum_{i=1}^{K} \pi_i^A + x_i^A) \prod_{i=1}^{K} \Gamma(\pi_i^A)} \right]
\]

(3.12)

\(^5\)These distribution can be equivalently expressed using a hierarchical (i.e., de Finetti) representation or as a sequential sampling scheme.
3.1.5 Data Oriented Parsing

As I described in Sections 1.4.4 and 2.4.3, in this thesis I formalize exemplar–based productivity and reuse with two variants of the Data Oriented Parsing (DOP) formalism for tree–substitution grammar estimation (Bod et al., 2003). A tree–substitution grammar (TSG) is a generalization of context–free grammars where the basic units of combination can be arbitrary tree fragments, rather than rules. A CFG can be seen as a special case of a TSG where all fragments are restricted to be depth–one trees. A Probabilistic Tree–substitution Grammar (PTSG), is a TSG where each subtree is equipped with a probability.

Define the function prefix(d) to enumerate all prefixes of a given derivation tree, that is, all fully–connected subtrees of the given derivation tree which include the root node. Given a tree prefix, s ∈ prefix(d), write the n subtrees of d at the leaves of s as s′₁, ⋯, s′ₙ, that is, yield(s) = (root(s′₁), ⋯, root(s′ₙ)) . The probability of a derivation d under a PTSG is defined by the following recursive stochastic equations.

\[
G^a_{\text{DOP}}(d) = \begin{cases} 
\sum_{s \in \text{prefix}(d)} \text{PROB}(s) \prod_{i=1}^{n} G^\text{root}(s'_i)(s'_i) & \text{root}(d) = a \in V_T \\
1 & \text{root}(d) = a \in T_T 
\end{cases}
\]

(3.13)

This equation is essentially the same as the recursive stochastic equation for PCFGs except that tree prefixes have been substituted for rules. The probability of a particular derivation is the sum over all possible derivation tree fragments which can be combined to give rise to the target derivation d.

DOP is a family of estimation techniques for PTSGs. DOP provides a way of estimating the probability of stored tree fragments, PROB(s), from a corpus of examples. There are many techniques which fall under the DOP umbrella; however, most DOP estimators for PROB(s)—including the two used in this thesis—are frequentist and based on corpus counts. See Section 2.4 for further comments on why these particular versions of DOP were chosen and their relation to the other models used in the thesis.

The first variant of DOP, known as Data Oriented Parsing I (DOP₁) (Bod, 1998; Schä, 1990), sets the probability of each derivation tree fragment proportional to its token frequency in the training corpus.\(^6\) Let \(F_D(s)\) be the frequency of derivation subtree s in a set of derivations \(D\). DOP₁ uses the following estimator.

\[
\text{PROB}_{\text{DOP}1}(s) \propto F_D(s)
\]

(3.14)

A well-known problem with DOP₁ is that it tends to overweight training data nodes which appear higher and in larger trees (see, e.g., Goodman, 2003). For this reason, I explore a second DOP estimator in this thesis. This estimator—known as the Goodman Estimator (GDNN)—assigns equal weight to each training data node and each training data item; effectively down-weighting larger derivation tree fragments (Bod, 2003; Goodman, 2003, also see the discussion in Section 2.4.3). If \(d \in D\) is some derivation tree in a corpus of derivations, then let \(F_d(s)\) be the number of times that a subtree (complete or fragmentary) s appears in d. Let subtrees(d) be the function which returns the set of all subtrees of derivation tree d, and |subtrees(d)| be the cardinality of that set. The

---

\(^6\)I am assuming that the training corpus consists of labeled derivation trees, rather than strings, for example.
Goodman estimator is given by the following expression.\(^7\)

\[
\text{PROB}_{\text{GDMS}}(s) \propto \sum_{d \in D} \frac{F_d(s)}{\text{subtrees}(d)}
\]  

(3.15)

In other words, the Goodman estimator re-weights each derivation tree fragment in the corpus by the total number of trees which appear in its derivation. As a result, nodes which appear higher in derivation trees in the training set will receive the same weight as nodes which appear lower in trees in the training set.

**PCFG Reductions of DOP Grammars**

The number of subtrees for a given derivation tree is, in general, exponential in the number of nodes in the tree. TSGs which result from DOP are enormous—far too large to allow for efficient algorithmic processing such as parsing. Early versions of DOP dealt with this problem by working with approximations (e.g., Bod, 1995, randomly excludes 95% of potential fragments from the TSG). However, Goodman (1998, 2003) showed that it was possible to give a PCFG which is linear in the number of nodes in the training corpus and which gives exactly the same distribution over derivations as the corresponding DOP tree–substitution grammar.

Figure 3.1: **Node Addresses**: The DOP PCFG reductions defined in Goodman (2003) require annotating training data nodes with unique node addresses like those shown on the right of the figure.

For simplicity, assume that all derivation trees in the set \(D\) are binary branching.\(^8\) Assume that each node in the set of training derivations is labeled by an integer address to distinguish it from other nodes with the same label. This is shown in Figure 3.1, where each node has been labelled with an integer address so that, for example, \(N_1\) refers specifically to a single node in a particular training data token.

Let \(a_j\) represent the number of subtrees which occur under node \(A_j\), let \(a\) be the total number of subtrees which occur under nodes labeled \(A\) in the whole corpus, that is, \(a = \sum_j a_j\), and let \(\bar{a}\) be the total number of nodes of category \(A\) in the training corpus. Consider a depth–one subtree.

For each such subtree in the training corpus, add the rules shown in Figure 3.2 to the PCFG reduction. Goodman (2003) shows that the resulting PCFG defines exactly the same distribution as

---

\(^7\)My thanks to Jelle Zuidema for providing me with this expression.

\(^8\)More accurately, that they conform to Chomsky normal form. That is, all internal nodes are binary branching and all terminals are introduced by a single unary production.
3.1.6 The Pitman–Yor Process

In Section 2.3.3, I introduced the notion of stochastic memoization and defined the Pitman–Yor process (PYP) for use as a memoization distribution. The PYP is usually described as a sequential sampling scheme using the metaphor of a restaurant. Imagine a restaurant with an infinite number of tables. The first customer enters the restaurant and sits at the first table. The \((N+1)\)th customer enters the restaurant and sits at either an already occupied table or a new, unoccupied table, according to the following distribution.

\[
\tau^{(N+1)} | \tau^{(1)}, \ldots, \tau^{(N)}, a, b \sim \sum_{i=1}^{K} \frac{y_i - a}{N + b} \delta[\tau^{(N+1)} = \tau_i] + \frac{Ka + b}{N + b} \delta[\tau^{(N+1)} = \tau_K + 1] \tag{3.16}
\]

Where \(N\) is the total number of customers in the restaurant; \(K\) is the total number of occupied tables, indexed by \(1 \leq i \leq K\); \(\tau^{(j)}\) refers to the table chosen by the \(j\)th customer; \(\tau_i\) refers to \(i\)th occupied table in the restaurant; \(y_i\) is the number of customers seated at table \(\tau_i\); \(\delta_X\) is a \(\delta\)-distribution which returns 1 if the proposition \(X\) is true and 0 otherwise; \(0 \leq a \leq 1\) is the discount parameter of the model; and \(b \geq -a\) is the concentration parameter of the model.

Note that when \(a\) is equal to 1, every customer will sit at their own table. When \(a\) is equal to 0 the distribution becomes the distribution known as the single-parameter Chinese Restaurant Process (Pitman, 2002). The \(a\) parameter can be thought of as controlling the productivity of a restaurant: how much sitting at a new table depends on how many tables already exist. On average, \(a\) will be the limiting proportion of tables in the restaurant which have only a single customer, and the limiting probability of sitting at a new table. The \(b\) parameter controls the rate of growth of new tables in relation to the total number of customers \(N\) (Teh, 2006). Customers sit at an already–occupied table with probability proportional to the number of individuals at that table, or

\(\footnote{Scheme code implementing these reductions can be made available upon request.}

Figure 3.2: PCFG Reductions for DOP1 (left) and GDMN (right): This figure shows the PCFG reductions given in Goodman (2003) for the DOP1 estimator (left) and GDMN estimator (right).
at a new table with probability proportional to \((aK + b)\). See Section 2.3.3 for more discussion of the properties of the Pitman–Yor Process.

Each table has a dish associated with it. Each dish \(v\) can be understood as a label on the table which is shared by all the customers at that table. When a customer sits at a new table, \(\tau_i\), a dish is sampled from another distribution, \(\mu\)—known as the base distribution of the Pitman–Yor process. After the first time a dish is sampled for a table, all future customers who are seated at that table share the dish.

Like the multinomial–Dirichlet distribution, another way of understanding the PYP is to think of it as defining a distribution over ways of partitioning \(N\) items (customers) into \(K\) partitions (tables), for all possible \(N\) and \(K\). It differs from the multinomial–Dirichlet distribution in that it allows \(K\) to be unbounded. The probability of a particular partition of \(N\) customers over \(K\) tables is the product of the probabilities of the \(N\) choices made in seating those customers. It can easily be confirmed that the order in which elements are added to the partition components does not affect the probability of the final partition (i.e., the terms of the product can be rearranged in any order). Thus the distribution defined by a PYP is again exchangeable like the other distributions we have seen in this chapter.

Given some vector of table counts \(\vec{y}\) (i.e., partition component assignments), a closed-form expression for the probability of a partition can be given as follows. First, define the following generalization of the factorial function, which multiplies \(m\) integers in increments of size \(s\) starting at \(x\).

\[
[x]_{m,s} = \begin{cases} 1 & m = 0 \\ x(x+s) \cdots (x+(m-1)s) & m > 0 \end{cases}
\]

In particular, \([1]_{m,1} = m!\). The probability of the partition given by the count vector, \(\vec{y}\), in a PYP is given by the following expression.

\[
P(\vec{y}|a,b) = \frac{[b + a]_{K-1,a}}{[b + 1]_{N-1,1}} \prod_{i=1}^{K} [1 - a]_{y_i - 1,1}
\]

This equation compactly represents the product that results from the sequence of choices that led to a particular seating arrangement.

### 3.1.7 Adaptor Grammars

In Section 2.3.4, I showed how the Adaptor Grammars (AG) model of Johnson et al. (2007a) could be understood as a PYP–memoized version of the basic PCFG unfold procedure. The full AG model used in this thesis adds Dirichlet priors to the rule weights in the CFG base system.

Formally, an Adaptor Grammar is a triple: 

\[
A = (G, \{\vec{\pi}_A\}_{A \in V_G}, \{(a^A, b^A)\}_{A \in V_G})
\]

Where \(G\) is a context free grammar; \(\{\vec{\pi}_A\}_{A \in V_G}\) is a set of pseudo–count vectors for multinomial-Dirichlet distributions for each nonterminal \(A\); and \(\{(a^A, b^A)\}_{A \in V_G}\) is a set of hyperparameters for the Pitman–Yor process associated with each nonterminal \(A\). This model can be expressed by the following recursive stochastic equations.

\[
G^*_A(d) = \left\{ \begin{array}{ll}
1 & \text{root}(d) = a \in V_G \\
\sum_{r \in R_G: a \rightarrow \text{root}(d_r), \ldots, \text{root}(d_k)} \theta_r \prod_{i=1}^{k} \text{mem}(G^*_A(\hat{d_i})) & \text{root}(d) = a \in T_G \\
\end{array} \right.
\]

\[
\vec{\theta} \sim \text{DIRICHLET}(\vec{\pi})
\]
\[ \text{mem}\{G^A_{\text{AG}}\} \sim \text{PYP}(a^A, b^A, G^A_{\text{AG}}) \] (3.21)

Here, \( \text{mem}\{G^A_{\text{AG}}\} \) represents memoization of the CFG recursion defined by \( G^A_{\text{AG}} \). As I discussed in Section 2.3.4, because the recursion \( G^A_{\text{AG}} \) always returns a fully expanded tree, Adaptor Grammars can only store and reuse \textit{complete} derivation tree fragments.

Because all of the distributions used to construct the \( \text{AG} \) model are exchangeable, so too is the overall model. Therefore, the probability of a particular set of \( \text{AG} \) analyses can also be given in terms of the counts associated with the individual multinomial–Dirichlet and Pitman–Yor distributions, that is, these counts are again sufficient. For a corpus of derivation trees \( D \), let \( X = \{x^A\} \) be the set of count vectors of CFG rules used to construct the trees stored in the PYP restaurants. Let \( Y = \{y^A\} \) be the set of count vectors of reused derivations stored on each table in each of the \( \text{AG} \)'s Pitman–Yor processes. I will use the shorthand \( A = \langle X,Y \rangle \) for these two kinds of counts since together they are sufficient statistics of the \( \text{AG} \) model. The probability of the corpus in terms of counts of rule and subtree uses under the Adaptor Grammars model is given by the function \( \text{ag} \) below.

\[
P(A|A) = \text{ag}(A,A) = \prod_{A \in V} \left[ \frac{\prod_{i=1}^{K^A} \Gamma(\pi^A_i + x^A_i) \Gamma(\sum_{i=1}^{K^A} \pi^A_i)}{\prod_{i=1}^{K^A} \Gamma(\sum_{i=1}^{K^A} \pi^A_i + x^A_i) \prod_{i=1}^{K^A} \Gamma(\pi^A_i) \prod_{i=1}^{K^A} [1 - \pi^A_i]_{y^A_i - 1,1}} [b^A + 1]_{N^A - 1,1} \right]^{a^A_{\text{AG}} - 1}
\] (3.22)

### 3.1.8 Fragment Grammars

As described in Section 2.3.6, Fragment Grammars generalize Adaptor Grammars by allowing the storage of \textit{partial} derivations at tables in the PYP–memoizers. This is achieved by implementing a \textit{lazy} version of the \textit{unfold} procedure.

The actual implementation of the \( \text{FG} \) lazy recurrence used in this thesis differs slightly from the Church definition given in Section 2.3.6. In that section, the decision to recurse or halt at a nonterminal on the right–hand side of a rule was made by flipping a fair coin—a \textsc{binomial} distribution with parameter 0.5. In the actual implementation of Fragment Grammars used in this thesis, this decision is made by flipping a biased coin—a binomial distribution with parameter \( \nu \) not necessarily equal to 0.5: \textsc{binomial}(\( \nu \)). Furthermore, a \textsc{beta}–distributed prior is placed on the weight \( \nu \) so that it too can be inferred during training. There is one such \textsc{beta}–\textsc{binomial} distribution for each nonterminal on the right–hand side of each rule in the underlying CFG. In other words, the system is able to learn a separate probability of recursing or delaying for each nonterminal on the right–hand side of each rule in the underlying CFG. Note that the beta distribution is the analog of the Dirichlet distribution for binomially–distributed random variables. Like the Dirichlet distribution it is parameterized by a vector of \textit{pseudo–counts}, \( \vec{v} \), which can be understood as imaginary prior observations of each of the two possible outcomes.

Formally, a fragment grammar is a 4-tuple: \( \mathcal{F} = \langle \mathcal{G}, \{\vec{\pi}^A\}_{A \in V}, \{(a^A, b^A)\}_{A \in V}, \{\vec{\psi}\}_{B \in \text{rhs}(r \in R_G)} \rangle \) where \( \mathcal{G} \) is a context free grammar; \( \{\vec{\pi}^A\}_{A \in V} \) are the vectors of multinomial–Dirichlet pseudo–counts for each nonterminal; \( \{(a^A, b^A)\}_{A \in V} \) are the vectors of multinomial–Dirichlet pseudo–counts for each nonterminal; \( \{\vec{\psi}\}_{B \in \text{rhs}(r \in R_G)} \) is the set of Pitman-Yor hyperparameters for each nonterminal; and \( \{\vec{\psi}\}_{B \in \text{rhs}(r \in R_G)} \) is the set of pseudo–counts for the beta–binomial distributions associated with the nonterminals on the RHS of each production rule in \( \mathcal{G} \). Fragment Grammars can be described
Chapter 3: Formalization of the Models and Inference

by the following recursive stochastic equations.

\[
G_{FG}^a(d) = \begin{cases} 
\sum_{s \in \text{prefix}(d)} \text{mem}\{L^a\}(s) \prod_{i=1}^{n} G_{FG}^{\text{root}(s'_i)}(s'_i) & \text{root}(d) = a \in V_G \\
1 & \text{root}(d) = a \in T_G 
\end{cases} \tag{3.23}
\]

\[
L^a(d) = \sum_{r \in R_G} \theta_r \prod_{i=1}^{k} \left[ \nu_{\text{root}(d_j)} G_{FG}^{\text{root}(d_i)}(d_i) + (1 - \nu_{\text{root}(d_j)})1 \right] \tag{3.24}
\]

\[
\hat{\theta}^k \sim \text{DIRICHLET}(\pi^k) \tag{3.25}
\]

\[
\hat{\nu}_b^a \sim \text{BETA}(\tilde{\nu}_b^a) \tag{3.26}
\]

\[
\text{mem}\{L^a\} \sim \text{PYP}(a^k, b^k, L^a) \tag{3.27}
\]

In these equations, \(L^a(d)\) implements the stochastically lazy recurrence described in Section 2.3.5. With probability \(\nu_b\), the recursion is delayed, and with probability \(1 - \nu_b\), the recursion is continued, and with probability 1 − \(1 - \nu_b\), the recursion is delayed. \(\nu_b\) is drawn hierarchically from a beta distribution; in practice, the \(\nu_b\)s are integrated out, and a Polya– urn representation of these draws is used. \(\text{mem}\{L^a\}\) is the stochastically memoized version of this lazy recurrence which stores partial derivation tree fragments using its Pitman–Yor Processes. The top–level recursion is represented by \(G_{FG}^a\), which implements a sum over all of the stored tree fragments in the memoizer, and forces computation at nonterminal leaves.

Once again, because all of the distributions involved in its definition are exchangeable, so too is the overall FG model; the probability of a particular set of FG derivations can be computed directly from the counts. Let \(X\) and \(Y\) be sets of count vectors for underlying rules, and lexical items as before. Let \(Z\) be the set of count vectors counting the number of times that each nonterminal on the RHS of a CFG rule was expanded in producing a lexical item, that is, the count vectors associated with the beta–binomial distributions described above. We will use the shorthand \(F = (X, Y, Z)\) for the set of all sufficient counts representing a corpus of Fragment Grammar derivations. The probability of a set of Fragment Grammar analyses is given by the function \(F_G\) below.

\[
P(F|\mathcal{F}) = F_G(F; \mathcal{F}) = \prod_{a \in V} \left[ \Gamma(\sum_{i=1}^{K^a} \pi^a_i + x^a_i + 1) \Gamma(\sum_{i=1}^{K^a} \pi^a_i) \prod_{i=1}^{K^a} \Gamma(\pi^a_i + 1) \prod_{i=1}^{K^a} \Gamma(\pi^a_i + 1 - 1, 1) \prod_{i=1}^{K^a} \Gamma(1, 1, 1) \right] \times \prod_{r \in R_G} \left[ \prod_{b \in \text{rhs}(r)} \frac{\Gamma(\psi_b + z_b) \Gamma(\psi_b' + (x_r - z_b)) \Gamma(\psi_b + \psi_b') \Gamma(\psi_b' + x_r)}{\Gamma(\psi_b + x_r + z_b) \Gamma(\psi_b' + x_r)} \right] \tag{3.28}
\]

3.2 Inference for Fragment Grammar

Inference for the FG model seeks to answer the question: What is set of stored derivation tree fragments that best accounts for some observed data? Given a corpus of derivation trees, \(D\), and the hyperparameters for a Fragment Grammar \(\mathcal{F}\), the posterior distribution over Fragment Grammar analyses for \(D, F\), can be expressed using Bayes’ rule.

\[
P(F|D, \mathcal{F}) \propto P(D|\mathcal{F}, F)P(F|\mathcal{F}) \tag{3.28}
\]
In a following section, I describe a Metropolis-Hastings inference algorithm for Fragment Grammars that samples from this distribution.

### 3.2.1 A Metropolis–Hastings Sampler

The Metropolis–Hastings algorithm outlined in this section is based on those described in Johnson et al. (2007a,b). This algorithm relies on several standard techniques in computational linguistics. Most importantly, it relies on techniques for enumerating the marginal distribution over derivations, given an input expression, and using this distribution to sample a derivation conditioned on the expression expression. In this section, I will only mention briefly how these algorithms work. For more details, I refer the reader to Johnson et al. (2007a,b) and Goodman (1998).

Given a corpus of derivations $D$, we wish to infer the posterior distribution over the set of (unobserved) stored derivation tree fragments which gave rise to that corpus. Metropolis–Hastings is a form of Markov Chain Monte Carlo (MCMC). Suppose that our goal is to sample from some distribution $P$ defined over some state space $S$. MCMC algorithms work by defining a Markov chain over $S$ which has $P$ as its stationary distribution. In order to do this, we introduce a proposal distribution $q$ which given some state $s \in S$ defines the probability over “next step” states $s' \in S$: $q(s'|s)$. This proposal distribution induces a Markov chain on the set $S$. When the stationary distribution over this Markov chain is equal to $P$, then $q$ can be used to define a correct MCMC algorithm. Metropolis–Hastings gives a formula (see below) for defining a correct MCMC algorithm for arbitrary proposal distributions.

Because the Fragment Grammar defines an exchangeable distribution, any expression can be treated as if it were the most recently observed datapoint in the corpus. The MH algorithm exploits this fact by (re)sampling a new Fragment Grammar analysis $f^{(i)}$ for each $d^{(i)} \in D$ conditioned on the rest of the FG analyses, $F_{-f^{(i)}}$. Thus, each local step in the Markov chain is defined by resampling an analysis for a particular observed form, conditioned on the current analyses for the rest of the forms in the corpus. The inner loop of the algorithm works as follows: (i) Each FG analysis, $f^{(i)}$, is removed in turn from the current state of $F$. (ii) A new analysis, $f^{(i')}$, is sampled for the $d^{(i)}$. (iii) The FG state is updated and the algorithm repeats the procedure for $d^{(i+1)}$.

Ideally, it would be possible to exactly sample a new analysis for each $d^{(i)}$, conditioned on the current state of the system—that is, ideally, we could implement a Gibbs sampler. However, for reasons discussed below, it is impossible to efficiently sample exactly from the conditional posterior. Instead we sample new derivations from an approximation to the exact conditional posterior which takes the form an approximating PCFG: $G'(F_{-f^{(i)}}, F)$.

\[
\begin{align*}
    f^{(i')} | d^{(i)}, F_{-f^{(i)}} &\sim P(\cdot|d^{(i)}, F_{-f^{(i)}}, F) \\
    &\approx P(\cdot|d^{(i)}, G'(F_{-f^{(i)}}, F)) \\
    &= pcfg(\cdot; G'(F_{-f^{(i)}}, F))
\end{align*}
\]  

(3.29) (3.30) (3.31)

Because these proposals are not exactly from the desired conditional posterior, they must be corrected in order to guarantee that the Markov chain’s stationary distribution is equal to the full posterior of the FG model. This is achieved by accepting or rejecting these proposals using the Metropolis–Hastings criterion. Proposals are accepted with probability $P(f^{(i)}, f^{(i')})$, and rejected otherwise.

\[
P(f^{(i)}, f^{(i')}) = \min \left\{ 1, \left[ \frac{fg(F_{-f^{(i)},+f^{(i')}}; F)}{fg(F; F)} \right] \times \frac{pcfg(f^{(i)}; G'(F_{-f^{(i)}}, F))}{pcfg(f^{(i')}; G'(F_{-f^{(i)}}, F))} \right\}
\]

(3.32)
3.2.2 The Approximating PCFG

As discussed in Section 3.1.2, PCFGs make strong conditional independence assumptions. One important consequence of these conditional independence assumptions is that there are efficient dynamic programming algorithms available for solving the PCFG parsing problem. These algorithms rely on the fact that the distribution over parses for an expression, $\vec{w}$, decomposes into a product over sums for subexpressions of $\vec{w}$.

Unfortunately, this decomposition does not hold in the case of MDPCFGs, AGs and FGs. For each of these formalisms, the probability of a rule or derivation tree fragment depends on the number of times that rule or fragment was used in the past. If a rule or fragment is used several times in the same derivation, these probabilities must be updated. This destroys the dynamic-programming contract, which depends on the independence of these choices from one another.

For these reasons, it is impossible to exactly sample a new derivation for each expression in the training corpus. Instead, the MH algorithm samples a new derivation for each expression from a PCFG which closely approximates the conditional posterior, but can be dynamically-programmed: $G'_{F-f(o)}, F)$. The idea behind this approximation is that after removing the counts associated with the expression to be resampled, the rest of the counts are “frozen,” and a PCFG which is a snapshot of the frozen state of the FG is read off.

Intuitively, a stored derivation fragment with root category, $A$, and leaves, $V$-able -ity, can be thought of as a context–free rule of the form $A \rightarrow V$-able -ity, which ignores the internal structure of the fragment. For a Fragment Grammar $F = (G, [\vec{a}^k]_{A \in V_G}, ([a^k, b^k])_{A \in V_G}, [\vec{v}^k]_{B \in \text{rules}(R \in R_G)})$, we introduce a PCFG rule, $A \rightarrow \gamma$, into $G'(F_{-f(o)}, F)$ for each possible root $A$ and each sequence of terminals and/or nonterminals, $\gamma$, found at the leaves of a stored derivation fragment or on the RHS of an underlying rule from $G$. The probability of this rule is given as follows.

$$\theta_{\nu, A} = \sum_{v \in \text{mem}(L^\prime)} \frac{y^k_v - a^k}{N^k + b^k} + \sum_{r \in R_G} \left[ \frac{K^k a^k + b^k}{N^k + b^k} \times \frac{x^k_r + \pi^k_r}{K^k + \sum \pi^k_r} \right]$$

The first term represents the probability of all stored fragments with root category $A$ which have $\gamma$ as their sequence of leaves. The second term represents the probability of all underlying CFG rules with $A$ on their LHS which share $\gamma$ as their RHS. The second term has two components: The first component represents the probability of sitting at a new table in the PYP restaurant associated with $A$. The second component represents the probability that this particular rule $r \in R_G$ was chosen as the topmost portion of the fragment which will label the corresponding table.

Given this approximating grammar, we can efficiently compute the distribution over derivations given an expression. The proposal distribution, $P(f^{(i)} | d^{(i)}, G'(F_{-f(o)}, F))$, is computed in the following steps.

1. Compute the inside table over analyses of $d^{(i)}$ given $G'(F_{-f(o)}, F))$. This table includes the inside probability of every possible constituent in every possible derivation of $d^{(i)}$. The inside table is computed with a version of the CYK algorithm.

2. Sample a derivation using the inside table. Starting at the goal item, sample a derivation for $d^{(i)}$ by recursively sampling from the distribution over back-pointers that results from normalizing inside scores. This algorithm is described in more detail in Johnson et al. (2007b).
3. Sample a FG analysis, \( f^{(i)} \), using the approximating grammar derivation. The rules used in the approximating grammar collapse across Fragment Grammar derivation fragments and underlying CFG rules. A FG derivation is recovered by sampling conditionally on the collapsed derivation; undoing the marginalization that was performed when the probability of \( \rho^A \) was calculated. This sampling is performed bottom–up along the structure of the approximating grammar derivation. A new table is created each time that an approximating derivation node corresponds to a underlying CFG rule in the Fragment Grammar.\(^{10}\)

Once a proposal derivation has been sampled in this manner, the MH score is computed and the proposal is accepted or rejected accordingly.

3.2.3 Implementation

The MH sampler just described has been implemented in the OCAML programming language and can be made available upon request to the author.

3.3 Inference for Other Models

For the simulations in this thesis, maximum a posteriori (MAP) grammars were used for the MDPCFG and AG models. The MAP grammar refers to the most probable grammar in the posterior distribution over grammars, given the data. All of the training corpora used in this thesis provide labeled derivation trees for all inputs, and inference for all models was conditioned on these trees. How this was accomplished for FG is described in Section 3.5.4. However, one consequence of conditioning on these trees is that it was possible to directly compute the (approximate) maximum a posteriori (MAP) grammars for the MDPCFG and AG models.\(^{11}\) For both AG and MDPCFG, the posterior probability of a grammar is maximized when the amount of possible sharing between rules and/or stored fragments is also maximized. Because the derivations provided in the training corpus were considered to be a gold–standard, the MAP AG and MDPCFG grammars could be computed directly from the counts associated with fragments of the input derivation trees. The resulting MAP grammars were used to compute all results reported later in this thesis.

Although MAP grammars were computed directly for the AG and MDPCFG models, they were only approximated for the FG model by using the MH sampler (described in the preceding section) to perform stochastic search (see Section 3.5). Therefore, to the degree that the quality of inference affects the results reported in subsequent chapters, this should provide an advantage to the MDPCFG and AG models.

The two DOP formalisms are based on estimators which are defined directly in terms corpus counts. As described in Section 3.1.5, the simulations in this thesis made use of PCFG reductions for these models which could, therefore, also be computed directly from the input corpora.

\(^{10}\)Note that because this portion of the algorithm operates bottom–up along the structure of the analysis, it is impossible to create a loopy table.

\(^{11}\)Although the simulations in this thesis use MAP approximations for AG and MDPCFG, there are algorithms in the literature which implement fully Bayesian (i.e., full posterior) inference for these two models (see, e.g., Johnson et al., 2007a,b).
3.4 Computation of Conditional Probabilities

In the succeeding five chapters, I will make use of a large number of different conditional probabilities and other values—such as entropies, gains, and expected numbers of constituents—which are computed against the posterior grammars for each model (or the estimated grammar in the case of the DOP models). In this section, I briefly describe how these values were computed.

First, all computations were performed using PCFG representations of the (approximate) MAP grammars for each model. In the case of MDPCFG, the grammar was already represented as a PCFG. The two DOP models were represented by PCFG reductions as described in Section 3.1.5. For AG and FG approximating PCFGs like those described in Section 3.2.2 were used (see also Section 3.5.2).

Second, I extended my OCAML implementation of CYK to allow parsing against wildcard values. This allowed me to compute marginal scores for values such as “the set of all forms ending in -ity” which could be represented by a sequence such as * -ity. The conditional probability, for example, of a particular stem, given -ity, could then be computed as the ratio of the marginal score of the entire word over the marginal score of all words ending in -ity. The parser was further enhanced to allow for the computation of arbitrary expectations against the conditional distribution over derivations. This allowed the easy implementation of the computation of quantities such as the entropy of the set of derivations or the expected number of constituents in a particular form.

Third, using the PCFG MAP approximations and the enhanced CYK parser described above, the marginal probabilities for large numbers of test forms (including wildcard forms) were scored. These were then used to compute the exact values reported in various parts of the thesis, with a large suite of Ruby scripts. These scripts can be made available upon request.

3.5 Practicalities of Inference for FG

Inference in the Fragment Grammar model is difficult.\(^\text{12}\) The space of possible derivations for a corpus of forms is large, and CYK parsing against large grammars is slow.\(^\text{13}\) The sampler can also easily become trapped in modes of the posterior distribution. For these reasons, this section presents a number of practical techniques for improving FG inference.

3.5.1 Initialization

A set of analyses for each expression in an input corpus must be initialized before the MH algorithm described in Section 3.2.1 can be run. As I described in the last section, the Fragment Grammar inference algorithm can easily become trapped in modes of the posterior distribution. In order to maximize the possibility of finding a good posterior state, I used the batch initialization procedure described in Johnson and Goldwater (2009). Under batch initialization, every node in the input corpus is assigned in parallel to its own table in a PYP restaurant, and all decisions about the distribution of variables are randomly sampled from the prior. This has the effect of starting the sampler at a completely random point in posterior space. When combined with the selective model averaging technique described in the next section, search performance can be greatly improved.

\(^{12}\)Note that the difficulty of doing efficient inference over models which store arbitrary subtrees—such as Fragment Grammars—was one of the motivations for the decision to store only complete subtrees in Adaptor Grammars (Johnson et al., 2007a). By only storing complete subtrees, the number of rules in the approximating PCFG for the Adaptor Grammars model does not grow in the number of stored items.

\(^{13}\)CYK is hard against a $O(n^3|G|^2)$ bound, where is $n$ is the length of the input and $|G|$ is the size of the grammar. FG approximating grammars frequently have tens of thousands of rules.
3.5.2 Selective Model Averaging

The problem of the sampler becoming trapped in modes of the posterior distribution can be further mitigated via the technique of selective model averaging (SMA). In SMA, a large number of different posterior states are averaged according to their posterior probability—to produce an approximation to the MAP grammar. The intuition behind this approach appeals to an empirical observation about Fragment Grammar performance. When a probability–ranked list of fragments is compared across multiple FG sampler runs on the same training corpus, the top–ranked fragments are nearly always identical. Fragments which are very useful—highly shared, or highly frequent—are rediscovered by all MCMC chains. However, lower ranked fragments—those which appear further down in the ranking—show more variability.

To mitigate noise in the estimation of MAP probabilities for low–ranked fragments, a large number of individual posterior modes can be averaged over a large number of sampler runs. In the simulations reported below, each sampler instance tracked the top 20 distinct posterior modes that it encountered during sampling, and the approximating PCFG which corresponded to each of these states was written to disk. A large number of sampler instances were run for each training set, and the resulting approximating PCFGs for all runs were averaged.

The distribution defined by a mixture of two (multinomial) PCFGs over the same support (i.e., the same set of terminals and nonterminals) is just the PCFG which mixes over individual rules according to the weights of the two grammars. This can be seen by considering the recursive stochastic equations below.

\[
\begin{align*}
    w_1 G_1^A(d) + w_2 G_2^A(d) &= w_1 \left[ \sum_{A \rightarrow \text{root}(\hat{d}_i), \ldots, \text{root}(\hat{d}_k)} \theta_{r_1} \prod_{i=1}^{k} G_1^{\text{root}(\hat{d}_i)}(\hat{d}_i) \right] \\
    & \quad + w_2 \left[ \sum_{A \rightarrow \text{root}(\hat{d}_i), \ldots, \text{root}(\hat{d}_k)} \theta_{r_2} \prod_{i=1}^{k} G_2^{\text{root}(\hat{d}_i)}(\hat{d}_i) \right] \\
    &= \sum_{A \rightarrow \text{root}(\hat{d}_i), \ldots, \text{root}(\hat{d}_k)} \left( w_1 \theta_{r_1} + w_2 \theta_{r_2} \right) \prod_{i=1}^{k} \left[ w_1 G_1^{\text{root}(\hat{d}_i)}(\hat{d}_i) + w_2 G_2^{\text{root}(\hat{d}_i)}(\hat{d}_i) \right]
\end{align*}
\]

This rearrangement of terms depends only on the two sums in the top row being over the same set of rules. Note that for any two PCFGs which are defined with respect to the same set of terminals and nonterminals, the sum can be made identical simply by introducing rules with weight 0 in each case that one grammar has a rule and the other does not. For each training set, a single average approximating PCFG was produced by averaging a large number of posterior samples—each of which was produced by a chain initialized to a random point in posterior space.

Note that selective model averaging can potentially introduce error into estimates of the MAP approximating grammar. For example, if the posterior is strongly bimodal, averaging could lead to a mixture grammar whose parameters lay between the two modes in a low probability point in posterior space. However, in practice, the sampler seems to produce steeply peaked posteriors, and high–probability posterior states typically share a large proportion of their approximating rules. Moreover, the average grammars used in the analyses in subsequent chapters were typically dominated by a very small number of especially good samples. These considerations suggest that the selective model averaging technique is approximating the MAP solution for high posterior–probability fragments.

In the simulations reported in subsequent chapters, both the number of sweeps through each training corpus per sampler instance, and the number of individual sampler runs, varied greatly from...
simulation to simulation—according to availability of time on the MIT CSAIL cluster. In general, however, dozens of runs and thousands of posterior–mode grammars were averaged in this way to produce an approximate MAP grammar for each of the analyses described in subsequent chapters.

### 3.5.3 Type/Token Binning

A third technique for mitigating the degree to which the sampler runs become trapped in posterior modes is type/token binning. This is best illustrated with an example. The derivational morphology simulations reported in Chapter 7 were run over approximately 1.7 million word tokens. Resampling each of these tokens amounts to a very small step in posterior space. However, these 1.7 million word tokens are distributed over only approximately 20,000 word types. One way of taking larger steps in posterior space is to resample many tokens—which correspond to the same type—simultaneously. In order to make this possible, the MH sampler was enhanced with the ability to take inputs in bins each of which corresponded to $N$ tokens of some particular word type. The tokens in these bins were then resampled simultaneously by the sampler.

Binning input tokens leads to a tradeoff. The more aggressively one bins tokens into types, the larger the steps in posterior space, and the more work done per unit time during sampling. However, all the tokens in a bin are treated by the sampler in an identical manner. Therefore, the more aggressive the binning, the greater the number of posterior states that are rendered inaccessible to the sampler. In particular, states where individual tokens within a bin are split between different analyses cannot be reached. In the limit, where all tokens of each type are in a single bin, many analyses become unreachable (i.e., the Markov chain is not ergodic). Therefore, to take advantage of binning while not eliminating too many possible states from the posterior, different Markov chains were run at a variety of bin sizes. Some simulation runs binned aggressively, and, thus, took large steps in posterior space, while other simulation runs binned less aggressively, and, therefore, took smaller steps, but left more states accessible. These multiple runs were averaged using selective model averaging as described in the last section.

### 3.5.4 Conditioning on Constituent Information

Another technique used to improve inference performance was conditioning on constituent information. The training corpora described in later chapters included derivation trees for all input forms. To speed parsing, the CYK parser implementation was adapted to condition on the constituent information in these gold–standard derivations. In particular, I implemented span–wise conditioning (SWC). The parse chart was seeded with the set of spans present in the gold–standard derivation. The parser was then only permitted to hypothesize spans that matched the gold–standard spans.

Note that this technique is not identical to exactly conditioning on the full gold–standard derivation trees. In particular, a rule in the approximating PCFG for a FG state collapses across the internal structure of the fragments to which it corresponds. For example, the approximating PCFG will represent a fragment such as $(N (~adj~ V ~able~ -ity))$ with a rule like $N \rightarrow V ~able ~ity$. In this example, the internal ~adj~ node in the fragment is suppressed in the approximating PCFG.

---

14 Many individual chains ran for weeks, and, typically, there were several dozen or more such chains running at one time.

15 Note that while binning renders the chains non–ergodic, the posterior scores for selective model averaging remain accurate because they can be computed from the counts alone.

16 I thank Mark Johnson for suggesting this technique.
rule. Even if the gold–standard parse requires an Adj node over the span which includes V and -able in the input, the parser will not disallow this rule because this rule matches a larger span and suppresses the internal Adj constituent. In other words, SWC enforces category matching in any position where the FG model makes a random choice; by contrast, places where the FG model does not make a random choice are invisible to these parser constraints. SWC significantly improves parsing speeds.

3.5.5 Auxiliary Variable Slice Sampling

A final inference optimization was auxiliary variable slice sampling. The main time bottleneck for inference is parsing: The main cause of high parsing costs is the large size of approximating PCFGs for each FG state. To mitigate this problem, I employed the following technique (Van Gael et al., 2008). First, before removing the counts and resampling analysis \( f \) during the outer loop of the MH algorithm, I found the approximating PCFG rule with the lowest probability \( p \) which was used in that analysis. I then sampled \( p' \sim \text{UNIFORM}(0, p) \). Before reparsing \( f \), I removed all rules from the approximating PCFG which had probability less than \( p' \). This ensured that the MH reverse proposal probability for \( f \) was still computable, while, in some cases, significantly reducing the size of the approximating PCFG.

3.5.6 Hyperparameters

Hyperparameter values for all simulations in this thesis were identical.

1. \( \tilde{\pi} \): The multinomial–Dirichlet pseudo–counts, \( \tilde{\pi} \), for underlying CFG rules were set to 1 for all rules in the MDPCFG, AG, and FG models.

2. \( a \): The Pitman–Yor \( a \) parameter was set to 0.5 for all restaurants in both the AG and FG models.

3. \( b \): This Pitman–Yor \( b \) parameter was set to 100 for all restaurants in both the AG and FG models.

4. \( \tilde{\psi} \): The Beta–binomial pseudo–counts, \( \tilde{\psi} \), which govern the probability that a nonterminal on the right–hand side of a base rule is expanded or delayed, was set to 50 for all nonterminals on the RHS of all rules for the FG model.

The values for the Pitman–Yor process \( a \) and \( b \) parameters are relatively high, allowing for easy storage of novel stored structures. They were chosen as the result of informal experimentation. However, they were chosen prior to running the actual simulations reported in this thesis, and their values were not optimized for these datasets.

---

17I would like to thank Trevor Cohn for suggesting this technique.
Chapter 4

The English Past Tense: Abstraction and Competition

4.1 Introduction

This chapter reviews the empirical and theoretical literatures on one of the most intensively studied phenomena in psycholinguistics: the English past tense. Because of the broad range of empirical facts which are known about the phenomenon, the English past tense provides a rich test case for the models studied in this thesis. The English past tense is especially suited to evaluate issues of productivity and reuse because it consists of a highly productive, regular rule as well as a number of markedly less productive irregular processes. This sharp dichotomy in productivity makes it an especially important target for the learning models presented in the next chapter.

The first part of the chapter reviews the empirical literature on the past tense, focusing, in particular, on differences between regular and irregular forms. After this review, I examine different theoretical accounts of the phenomenon. This discussion is organized around three high-level points. First, I emphasize that any theory of the past tense must provide mechanisms capable of representing abstract generalizations. Second, I argue that merely providing the possibility of abstract generalizations is insufficient, and that a theory of the past tense must provide a quantitative account of how competition between generalizations is resolved. Finally, I discuss the residual issue of how to best represent low-productivity generalization (such as those found in irregular past-tense forms), arguing that, at present, this is still an open question.

4.2 The English Verbal Paradigm

English verbs are obligatorily marked with inflections that carry four kinds of syntactic and semantic information:

1. **Tense**: The tense of the verb; for example, present (*walk*) or past (*walked*).
2. **Person**: The person of the subject of the verb; for example, first-person (*am*) or third-person (*walks*)
3. **Number**: The number of the subject of the verb; for example singular (*is*) or plural (*are*)
4. **Aspect**: The completedness of the verb; for example, progressive (*walking*) or imperfect (*walks*).
Like many inflectional systems, the English verbal system is highly *syncretic*: many combinations of the features above can share a single marking. For example, aside from a few exceptional verbs—such as forms of *be*—person is only marked on third-person present singular verbs (e.g., *walks*), and, therefore, other person features share the same inflections across other feature values. Similarly, the markings for perfective and past forms are the same for most verbs (+ed in *walked/have walked*), although a small number of irregulars distinguish these forms (e.g., *ate/eaten*). Linguists typically organize the set of verb forms which share a stem into a table, called a *paradigm*. 1

Paradigms for several English verbs are shown in Figure 4.1.

<table>
<thead>
<tr>
<th>VBD (past tense)</th>
<th>GO</th>
<th>EAT</th>
<th>GIVE</th>
<th>SING</th>
<th>RING</th>
<th>WALK</th>
<th>USE</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>went</em></td>
<td>ate</td>
<td>gave</td>
<td>sang</td>
<td>rang</td>
<td>walked</td>
<td>used</td>
<td></td>
</tr>
<tr>
<td>VBG (progressive)</td>
<td>gone</td>
<td>eaten</td>
<td>given</td>
<td>sung</td>
<td>rung</td>
<td>walked</td>
<td>used</td>
</tr>
<tr>
<td>VBZ (3rd singular present)</td>
<td>goes</td>
<td>eats</td>
<td>gives</td>
<td>sings</td>
<td>rings</td>
<td>walks</td>
<td>uses</td>
</tr>
<tr>
<td>VB (base form)</td>
<td>going</td>
<td>eating</td>
<td>giving</td>
<td>singing</td>
<td>ringing</td>
<td>walking</td>
<td>using</td>
</tr>
<tr>
<td>VBP (simple present)</td>
<td>go</td>
<td>eat</td>
<td>give</td>
<td>sing</td>
<td>ring</td>
<td>walk</td>
<td>use</td>
</tr>
</tbody>
</table>

Table 4.1: *English Verbal Paradigm*: This table shows the six forms of the English verbal paradigm, with examples for a number of verbs.

This table shows six possible forms for seven English verbs. The first column gives the abbreviation for the verb form used in the tagset of the Penn TreeBank corpus (Marcus et al., 1999). I will use these abbreviations throughout this and later chapters. Each column is labeled with its *stem* (e.g., *GO*). Note that these stems refer to the abstract entities underlying the different forms of a verb. For example, there is no particular phonological relationship between the present (*go*) and past (*went*) forms of *GO*. 2 English verbs fall into a number of distinct inflectional classes.

---

1I am using the term *paradigm* purely descriptively. The necessity for paradigms and their psychological reality are controversial topics in linguistics and psycholinguistics (see e.g., Bobaljik, 2008; Carstairs-McCarthy, 2001; Moscoso del Prado Martín et al., 2004; Stump, 2001b). In word–and–paradigm theories, they are considered first–class theoretical objects; while in other theories, they are considered epiphenomenal (e.g., Kiparsky, 2005; Noyer and Harley, 1999).

2I divide verbal paradigms into six cells. A more usual organization of English verbs collapses simple presents (VBP) and base forms (VB) (see e.g., Carstairs-McCarthy, 2001). However, these six verb types were available in the training data (i.e., the Penn TreeBank Marcus et al., 1999), and nothing crucial hinges on the number of paradigm cells.

3 The use of the term *stem* is typical of so–called *stem–and–arrangement* or *morpheme–based* approaches to morphology (see, Hockett, 1954). These are approaches to morphology which focus on morphemes and take concatenation as the most basic structure building operation. In so–called word–and–paradigm or *lexeme–based* approaches to morphology, the *stem* would be called a *lexeme*. Word–and–paradigm approaches (i) make use of the concept of abstract *lexemes*—the abstract nodes that represent whatever is shared between forms in a paradigm (but do not necessarily have any phonological content) and (ii) stress the importance of operations which transform phonological structure (in various ways that can be more sophisticated than concatenation). Note that the term *lexeme*, as used in the word–and–paradigm literature, is not to be confused with the same term as used in lexical processing theories. In the latter (see, e.g., Caramazza, 1997), the term typically (and confusingly) means something nearly the opposite of what it does above: a node which represents *only* the phonological structure of words (i.e., a node shared between homophones).

My use of *stem* does not indicate a commitment to the item–and–arrangement view of morphology. On the contrary, the modeling framework here is consistent with a variety of formal systems. The particular formalization of the English past tense that I adopt in this chapter has aspects of both approaches, but the computational model could be adapted to be fully consistent with either approach.
An inflectional class is a set of verbs which mark the different cells of a paradigm in the same way. Two classes visible in Figure 4.1 are shown in Table 4.2.

<table>
<thead>
<tr>
<th></th>
<th>I</th>
<th>II</th>
</tr>
</thead>
<tbody>
<tr>
<td>VBD</td>
<td>+/d/ (walk → walked)</td>
<td>/s/ → /s/ (sing → sang)</td>
</tr>
<tr>
<td>VBZ</td>
<td>+/z/ (walk → walks)</td>
<td>+/z/ (sing → sings)</td>
</tr>
<tr>
<td>VBN</td>
<td>+/d/ (walk → walked)</td>
<td>/s/ → /s/ (sing → sang)</td>
</tr>
<tr>
<td>VBG</td>
<td>+/w/ (walk → walking)</td>
<td>+/w/ (sing → singing)</td>
</tr>
<tr>
<td>VBP</td>
<td>+∅ (walk → walk)</td>
<td>+∅ (sing → sing)</td>
</tr>
<tr>
<td>VB</td>
<td>+∅ (walk → walk)</td>
<td>+∅ (sing → sing)</td>
</tr>
</tbody>
</table>

Table 4.2: Inflectional Classes: Two inflectional classes of English verbs. Class II contains verbs such as *sing* and *ring* while class I contains the regulars.

Each cell of table in Figure 4.2 shows a particular rule that is applied to the stem in that class to compute the surface form that appears in the cell. For example, class I is the class of regular verbs, which simply concatenate /d/ to form their past participles (VBN). In contrast, the past participles (VBN) of class II verbs (e.g., *sing*, *ring*) are formed by taking the stem /s/ vowel and transforming it into /s/.

The English verbal paradigm is of interest for the present study because of the great variability in the levels of productivity of the rules in various inflectional classes. The progressive marker *-ing* that forms verbs in the VBG cell is completely regular; there are no verbs which mark this cell in a different way. A few cells in the paradigm, such as the third-person singular present (VBZ), have a tiny number of exceptional forms (e.g., *have*/has), but are otherwise regular. The most interesting cases, however, are the past (VBD) and past participle (VBN) cells which show highly variable regularity and productivity. I discuss these in the next section.

### 4.3 Empirical Review of the English Past Tense

The vast majority of English verbs, the *regulars*, mark their past tense and past participle via concatenation of the suffix *-ed*. However, about 180 verbs, the *irregulars*, form their past tense and past participles in other ways. Importantly, the existence of an irregular form typically *blocks* the possibility of using the regular form for a given stem: The regular possibility is considered infelicitous or ungrammatical by speakers when an irregular exists (e.g., *went*/goed). There are, however, a small number of stems which seem to have both irregular and regular past forms. In some cases, the two forms have subtly different meanings (e.g., *fit* in *The suit fit well* and *The tailor fitted the suit*), but in a small number of other cases, speakers seem to accept two variants of a stem with identical meaning (e.g., *dreamed*/dreamt, *leaped*/leapt, *dived*/dove). These later verbs are known as *doublets*.

This following sections review the empirical literature on the English past tense, highlighting differences between regular and irregular forms in five areas: regularity, productivity, frequency, processing, and development.
Chapter 4: The English Past Tense: Abstraction and Competition

4.3.1 Transparency and Systematicity

The regular, +/d/, rule is completely phonologically transparent.4 The irregulars, by contrast, vary widely in their transparency and systematicity. While a few past tense forms are unrelated to their stem, and to other irregulars—for example, the suppletive from, go/went—many irregular verbs exhibit one of the following types of semi–regularity.

First, many irregulars can be organized into classes according to the transformation their stem undergoes to form the past. For example, there are a number of irregulars which form their past participles according to the following rule: /i/ → /s/ (e.g., sing/sung). Another class of irregulars, the weak verbs, are identical in stem and past forms (e.g., cut, spread). A third class of irregulars lax their stem vowels and append a /t/ to form the past (e.g., lose/lost, sleep/slept).

Second, the stems in each irregular inflectional class often share phonological structure with one another. For example, most of the verbs which alternate between /I/ and /æ/ (e.g., fling/flung) or /æ/ (e.g., sing/sang) end in the velar nasal: /ŋ/. Similarly, all members of the class of weak verbs end in /t/ or /d/. The structure shared between stems, in many cases, constitutes a family resemblance category (FRC; Pinker and Prince, 1996). In a FRC, there are a number of conditions which predict membership in a class, and stems which meet more of the conditions are more likely to be in the class. However, no one condition, or set of conditions is necessary and sufficient for predicting membership. For example, although verbs which alternate between present /I/ and past /s/ (e.g., fling/flung) tend to end in the velar nasal consonant, /ŋ/, the class also contains forms that end in non–nasal velars (dig/dug) as well as forms which end in non-velar nasals (win/won).

Third, irregular past forms tend to share structure with other past forms, including regulars. For example, the class of irregulars which laxes the stem vowel and appends a /t/ (e.g., sleep/slept), the weak verbs, and all regulars share a final coronal stop (i.e., /t/ or /d/).

4.3.2 Productivity

Perhaps the most important difference between the regular and irregular classes of verbs is their great disparity in productivity. A large number of studies have shown generalization of the regular rule to novel stems in both production and rating tasks (Albright and Hayes, 2003; Ambridge, 2010; Berko, 1958; Bybee and Moder, 1983; Gordon and Miozzo, 2008; Prasada and Pinker, 1993; Ramscar, 2002).5 The regular rule can even be applied to forms which are phonotactically odd for English (e.g., to a stem ending in a velar fricative, as in Bached). In fact, because the regular past ending can be attached to any verb stem, regular past tense forms are less phonotactically regular, on average, than irregular past tense forms (Buzio, 2000; Pinker and Prince, 1988). The regular rule also applies to stems that are formed by other morphological processes. For example, it applies when a verb has been created from a nominal form, even if nominal form is, itself, derived from an irregular verb: fly → fly ball → fly out → flied out (not flew out) (Huang and Pinker, 2010; Kim et al., 1994, 1991). The regular rule is sometimes also over–applied, resulting in overregularization errors such as breaked (Stemberger, 1983).6 The regular rule applies to newly coined verbs, and,

4The regular rule appears in a number of phonologically conditioned allomorphs. For example, it is devoiced after voiceless consonants (e.g., miss/missed). However, the distribution of allomorphs is completely complementary and phonologically predictable.

5Note that these studies— with the exception of Berko (1958)— are concerned with how differing levels of phonological or semantic similarity between novel forms and existing forms influences generalization. Although the studies focus on varying aspects of this question, they all find evidence for generalization of the regular rule.

6Also see discussion of adult overregularization in Marcus et al. (1992, p. 45).
therefore, new past forms entering the language typically enter as regulars. For all of these reasons, the regular rule is often described as a default, applying when no other form is available. Irregulars, on the other hand, are far less productive. Nevertheless, rating and wug studies have shown that some classes of irregular verbs can be generalized under limited circumstances (Albright and Hayes, 2003; Ambridge, 2010; Bybee and Moder, 1983; Bybee and Slobin, 1982; Prasada and Pinker, 1993). These studies find that irregular classes which contain greater numbers of stems and exhibit more systematicity tend to generalize more easily. Critically, the applicability of an irregular inflectional process to novel stems is highly sensitive to the phonological structure of the stem, and, in all cases, the regular past form is acceptable. The applicability of the regular rule, by contrast, is much less sensitive to the phonological makeup of the stem (Prasada and Pinker, 1993).

Historically, a small number of verbs which were formerly regular have become irregular (e.g., wear, spit, and dig, Bybee, 1985). In contrast, regularization has affected thousands of verbs (see, e.g., Lieberman et al., 2007). Accidental overirregularization in speech occurs as well (e.g., bite/bote), but it appears to be a marginal phenomenon (Bybee, 1985; Xu and Pinker, 1995). In summary, while irregular inflectional classes show some generalizability, their generalizability is far less than the regular rule, and is highly constrained by the structure of the stem.

4.3.3 Frequency

Like most structures in natural language, verb forms follow a power–law frequency distribution. Power–law distributions are characterized by two properties. First, the highest frequency words are many times more frequent than lower frequency words, that is, the distribution is steeply peaked at the high frequency end of the spectrum. Second the decline in frequency moving away from the peak is slow, that is, there is a long tail containing many infrequent words.

Irregular past forms are found primarily towards the peak of the word frequency distribution, while the long tail consists of regulars. Irregulars are, for the most part, very high frequency verbs, and the most frequent English verbs are irregular. In fact, this general pattern is characteristic of the frequency distributions associated with productive and unproductive processes across languages (see e.g., Baayen and Lieber, 1996; Chitashvili and Baayen, 1993).

As a consequence of the relationship between frequency and productivity just outlined, frequency is also predictive of (over)regularization during language development and use. Lower

---

7In judgment tasks, participants have also been shown to prefer novel past tense forms which attach the regular rule to stems which are more canonically or prototypically regular in terms of their phonology (Albright and Hayes, 2003; Ambridge, 2010). However, the magnitude of this effect is much smaller than in the case of irregulars; it shows different developmental trajectory (Ambridge, 2010); and, even when a stem is not prototypically regular, the regular rule can always be applied to it—even if the output is phonotactically ill-formed. These facts can be handled by a probabilistic approach which allows for a mixture of rules of differing levels of specificity, such as the present approach or that of Albright and Hayes (2003). As I will discuss in Section 4.4.1, in order to account for the past tense data, a model must be able to represent a fully abstract, general regular rule (i.e., “attach +ed to any verb”) that will apply when all else fails. It is not crucial, however, that this rule be the only way—or even most frequent way—of producing regular forms. Some proportion of regulars—possibly even a large proportion—in everyday usage may be retrieved directly from memory (as I will discuss in greater detail below), or built by special subcases of the general rule which are sensitive to phonological structure (as argued by Albright and Hayes, 2003). This raises a general point to which I will return below: The necessity of an abstract regular rule is demonstrated by generalizations which occur in rare boundary cases (e.g., phonotactically illicit stems, words formed by onomatopoeia, etc.). The fact that tighter generalizations can be found for highly attested patterns is irrelevant to this point.

8I am not aware of any study which has looked at rates of overirregularization errors in spontaneous adult speech. However, the strangeness of the examples given in the cited sources suggests that it is significantly less common than overregularization.
frequency irregulars are more frequently overregularized during development (Marcus et al., 1992). Higher frequency regulars and irregulars are more resistant to speech errors in adults (Bybee and Slohín, 1982; Stemberger and MacWhinney, 1986). Furthermore, lower frequency forms are more susceptible to regularization over language change (Bybee, 1985; Lieberman et al., 2007).

### 4.3.4 Processing

Regulars and irregulars also pattern differently in a number of experimental paradigms that tap into the cognitive processes involved in producing and comprehending verb forms. First, priming studies have shown stem priming effects are, in general, stronger for regulars than irregulars (Fowler et al., 1985; Kempley and Morton, 1982; Marslen-Wilson and Tyler, 1997; Münte et al., 1999; Napps, 1989; Stanners et al., 1979a). This fact is generally interpreted to mean that, while irregular stems and past forms have independent lexical representations, regulars share some aspect of their representation (e.g., their stems are represented separately from the past tense rule, and shared with forms of the verb in other tenses). Such priming effects are also stronger for more phonologically transparent irregulars (Kieler et al., 2008).

Production tasks have repeatedly demonstrated that the effects of frequency are different for regular and irregular past forms. In these experiments, subjects are presented with a stem and asked to say its past tense form as quickly as possible. High frequency irregular past tense forms are named faster than low frequency irregular past tense forms. For regulars, there is either no effect of frequency, or, sometimes, there is an anti-frequency effect where low frequency regulars are produced faster than high frequency regulars (Beck, 1997; Prasada et al., 1990; Seidenberg and Bruck, 1990; Shenkman, 1994). The difference in frequency effects is taken as evidence that irregular forms are stored independently and can, therefore, accumulate strength in memory, while regular forms are composed. I discuss anti–frequency effects in Section 5.3.3.10

This pattern of frequency effects also arise when participants are asked to judge the naturalness of past tense forms. Ullman (1993, 1999) regressed subjects' judgments of naturalness for past tense forms against their frequency—controlling for judgments of naturalness for the bare stem forms. He found that while irregulars showed a positive correlation with frequency, regulars did not. Instead, the naturalness ratings for the regular past tense forms were strongly predicted by the naturalness ratings for the stem, suggesting regulars are constructed via a compositional route.

Finally, Alegre and Gordon (1999a) and Gordon and Alegre (1999) have demonstrated frequency effects for high frequency regulars using a lexical decision paradigm. This has been widely interpreted as showing that even fully regular forms can be stored if they are sufficiently frequent.11 Frequency effect–based evidence for storage of fully regular inflected forms has also been demonstrated for a wide variety of other inflectional systems including: English nominal number marking (Sereno and Jongman, 1997), Italian nominal number marking (Baayen et al., 1997a), and Dutch and Finnish nominal (case and number) marking (Baayen et al., 1997b; Bertram et al., 1999).

---

9Prasada et al. (1990) is described in Pinker (1991) and Pinker (1999), and Seidenberg and Bruck (1990) is described in Seidenberg and Hoefnner (1998) and Pinker (1999).

10Another naming study, MacKay (1976), finds a relationship between naming latency and the phonological complexity of the transformation that relates stems to past forms. Past forms which are more phonologically dissimilar to their stems take longer to produce.

11Alegre and Gordon argue that the threshold for storage is about 6 per million, however, this has been disputed by Baayen et al. (2007), who argue for much more pervasive storage (although they do not provide evidence for the English past tense).
4.3.5 Development

A final way in which regular and irregular forms differ is in their developmental trajectory. One longstanding observation is that children sometimes overregularize irregular forms, producing forms such as goed in place of went (e.g., Chamberlain, 1906; Ervin, 1964; Ervin and Miller, 1963). An issue which has been the focus of much attention is whether overregularization follows a U-shaped developmental trajectory. Under a U-shaped developmental trajectory the first overregularization is preceded by a period of early correct performance on irregulars, and followed by eventual recovery to correct usage. The interest in U-shaped development lies in the way that it seems to support a view whereby children’s grammatical representations reorganize when the regular rule is discovered. I will discuss the historical debate surrounding this issue in more detail below; here, I focus on the available evidence.\(^{12}\)

Although there were earlier studies of child language examining questions of overregularization and U-shaped development (e.g., Cazden, 1968; Kuczaj, 1977), the most comprehensive study of the phenomenon, and the only one to provide strong quantitative evidence for U-shaped development in the past tense, is the corpus study of child speech presented by Marcus et al. (1992). Marcus et al. (1992) show that rates of overregularization vary greatly between individual children and across verb forms. At least one child in the corpus (Abe) overregularizes at high rates throughout the sample, whereas some other children overregularize at much lower rates (e.g., Adam). There is also a large amount of variation between verbs, with some verbs nearly always overregularizing, some verbs nearly never overregularizing, some verbs showing very clear U-shaped development patterns, and some verbs oscillating chaotically between correct performance and overregularization (Marcus et al., 1992, pp. 54–55).

Marcus et al. (1992) also discuss several global properties of overregularization. First, they point out that overall overregularization rates are fairly low. In their sample, the overall median rate across children was only 2.5%. This result has been challenged, for instance by Maratsos (2000) and Yang (2002), who claim that rates of overregularization are higher, nevertheless even these authors concede that rates are relatively low (around 10%). A reanalysis of the Marcus et al. (1992) data by Hoeffner (1996) supports this general conclusion.

Second, Marcus et al. (1992) find that—in the age range covered by their samples—there is no general evidence for robust recovery from overregularization errors, although one child (Abe), who has the highest rates of overregularization, does show evidence of recovery. Although recovery must occur eventually, the process appears to be gradual.

A more controversial claim made by Marcus et al. (1992) is that there is a global (i.e., verb-general) period of early correct performance before the onset of overregularization. In the Marcus et al. (1992) sample, three of the children showed a statistically significant early correct period. An aggregate test over all children was also significant. The controversy surrounding global early correct performance stems from the inability of some theoretical models (e.g., Hoeffner, 1996; MacWhinney and Leinbach, 1991; Plunkett and Juola, 1999; Plunkett and Marchman, 1993, 1996) to exhibit this effect without recourse to questionable assumptions about the input (see Section 4.4.1). Some connectionist authors have argued that the empirical data is better characterized by so-called micro-U-shaped learning: the U-shaped learning of some verbs, but not others (e.g., Hoeffner, 1996; Plunkett and Marchman, 1993). Nevertheless, all authors agree that there is clear

\(^{12}\)There have been several recent results in computational learning theory in the identification in the limit framework (Jain et al., 1999) that show that, for some classes of learnable languages, U-shaped learning is a necessary property of the learning trajectory. That is, under some criteria for correct language identification (e.g., a bound on the number of correct grammars between which the learner is allowed to vacillate once the language has been identified) the learner must pass through a period where they hypothesize an incorrect grammar (Baliga et al., 2008; Carlucci et al., 2005, 2006; Fulk et al., 1994).
period of early correct performance for some verbs, and that there is a clear period of global early correct performance for some children.

Another controversial claim made by Marcus et al. (1992) is that the onset of overregularization coincides with the emergence of reliable, obligatory past–tense marking of regular forms. English speaking children go through a period during which they sometimes leave finite verb forms unmarked for tense (e.g., Mary walk home in past context). Marcus et al. (1992) argued that the onset of the first overregularization is correlated with the emergence of reliable obligatory tense marking on regular forms. They performed two analyses to support this claim. First, they showed that rates of reliable regular marking are significantly lower than 50% before the first overregularization, but significantly higher after the first overregularization. Second, they showed that rates of reliable obligatory marking are higher after the first overregularization than before. Hoeffner (1996) criticized these analyses on the grounds that, because reliable marking is increasing throughout the period, the choice of the first overregularization as a dividing point is somewhat arbitrary: There are a large number of dividing points which could lead to the same results. Furthermore, he showed that obligatory regular marking is better predicted by age than the assumption of the emergence of an all−or−nothing regular rule. However, Hoeffner’s analysis supports the more general observation that overregularization increases during the period when reliable regular marking is increasing. In Section 4.4.2, I discuss the theoretical significance of this debate in more detail.

Another important (uncontested) finding of Marcus et al. (1992) is that overregularization is not predicted by sudden discontinuous increases in regular forms in the input to the learner; nor is it predicted by sudden increases in the use of regular forms by children. There is, of course, a logically necessary increase in the proportion of vocabulary which is regular. Only about 180 verb forms are irregular; therefore, as the child learns more verbs, the proportion which are irregular must at some point begin to decrease. However, there is not a sudden jump in the number of regulars which corresponds to the onset of regularization.

Finally, Marcus et al. (1992) discussed several factors that protect individual verbs from overregularization. First, they found that higher frequency irregulars show less overregularization. Second, they found that verbs from robustly attested inflectional classes (i.e., verbs which are phonologically similar to other high−frequency irregulars) are protected from overregularization. However, similarity between an irregular stem and regular forms does not cause greater rates of overregularization. These results are consistent with the facts discussed in Section 4.3.2: The applicability of irregular inflections depends on the phonology of the stem to a much greater degree than the applicability of the regular inflection. Marcus et al. (1992) also find that the phonological transparency of the stem–past mapping (i.e., the amount of change or complexity inherent in the irregular inflectional class) does not predict the rate of overregularization. Finally, Xu and Pinker (1995) showed that rates of developmental overirregularization—the misapplication of irregular inflection to regular stems (e.g., bite/bote)—are very small.

4.4 Theories of the Past Tense

In this section I review previous theories of the past tense, with a special focus on computational models. My review is organized around three high−level points. First, much of the historical discussion of the past tense occurred in the context of the debate between connectionist and rule−based approaches to the phenomenon. Although many issues were raised in this discussion, I will focus on one: the necessity for mechanisms that support abstract generalization. In particular, theories of the past tense must provide the means to represent an abstract generalization like: “attach −ed to any verb.” Traditional linguistic models, dual−mechanism accounts, more recent probabilistic rule−based models, rich analogical models, and some connectionist...
ist models allow for such generalizations. However, classical feedforward connectionist networks do not. The failure of these models to adequately account for the empirical data, especially with regard to the regular rule, is a direct result of not providing the means for this abstraction.

Second, merely providing the means to represent abstract generalizations is not enough. An empirically adequate theory must also provide a mechanism for resolving competition between generalizations at differing levels of abstraction. Furthermore, phenomena such as overregularization and the differential generalizability of past tense inflectional classes indicate that the mechanism must be quantitative. This discussion will also highlight an important principle of competition: the elsewhere condition. This principle states that when multiple generalizations are available, the most specific one should be preferred. My discussion in this section will highlight how many different approaches have advocated versions of this principle; in Chapter 8, I will show how a generalized, probabilistic version of the elsewhere condition follows as a natural consequence of the modeling assumptions in this thesis.

Third, I discuss a remaining question: how to best represent less productive generalizations, such as the irregular inflectional classes. In particular, it is unclear whether these phenomena are best handled by analogical or associative mechanisms or by quantitative rule–based mechanisms. I will argue that a more fine–grained understanding of the real and superficial differences between these different computational proposals is needed to answer this question.

4.4.1 The Necessity for Abstract Representations

Much of the historical discussion of the past tense occurred in the context of a debate between connectionist and rule–based accounts of the phenomenon. In an influential paper, Rumelhart and McClelland (1986) presented a connectionist model of the acquisition of the English past tense, which, they claimed, was able to account for past tense acquisition in a way that obviated the need for traditional rule–based accounts.

Although the Rumelhart and McClelland (1986) model and similar connectionist approaches have been critiqued on many grounds (see, e.g., Lachter and Bever, 1988; Marcus, 2001; Marcus et al., 1992; Pinker, 1999; Pinker and Prince, 1988; Sproat, 1992), I will focus on one aspect of these critiques: An adequate theory of productive linguistic processes, like the regular past tense rule, must provide mechanisms capable of representing abstract generalizations.

Rumelhart and McClelland (1986) and Related Connectionist Theories

To understand the difficulties faced by Rumelhart and McClelland (1986) and related theories, we must clarify the aspects of linguistic rules that are absent in these models. Traditional linguistic rules can be understood as functions which accept arguments from a certain domain and map them to outputs according to a specified algorithm. A fundamental feature of the types of rules used in linguistic theories is that they are often specified in terms of abstract features or categories. For example, traditional treatments of English verbal inflection have frequently made use of a very general rule which accepts any verb (stem) and concatenates the regular ending, +/d/, to the stem. Using abstract categories in rules (such as the category: VERB) is critical because it allows a rule to treat an entire class of inputs uniformly, ignoring any structure (e.g., the phonology of the stem) which is irrelevant for the operation it performs. Importantly, the kinds of abstract categories or features used in traditional rules are not directly observable from any particular input. A stem is verbal by merit of its relation to the rest of the system of English grammar; this fact cannot be deduced with certainty merely by examining the phonology of a single stem.

Skousen (1992) provides a particularly clear and general definition of (probabilistic) rules.
The Rumelhart and McClelland model also provides a mapping from inputs to outputs. However, it defines this mapping over a very specific class of representations. The Rumelhart and McClelland model is a simple feedforward network—also called sometimes called a multilayer perceptron. Such networks operate by strengthening connections between a distributed representation of input forms and a distributed representation of output forms. In the case of the Rumelhart and McClelland model, the input and output representations were sets of trigrams of phonological features. Any generalization made available by this representation scheme is statable terms of associations between directly observable, phonological features of the input and output. Because this representational scheme did not utilize abstract categories to distinguish classes of inputs or outputs, or to distinguish subparts of words (e.g., stems or suffixes), Rumelhart and McClelland’s claimed that their modeling results showed that there was no need for such categories to account for linguistic generalizations.

However, there are a number of problems with this argument. For their model, every correspondence between stem and past form—regular or irregular—correlates some phonological subsequences in the input with some phonological subsequences in the output. Because irregular inflectional classes often reside in relatively coherently defined and specific parts of phonological space, the model could often successfully acquire these patterns. However, the regular rule applies to any verb form which is not irregular (and in the case of overregularization to some forms which are irregular). Because every regular mapping presents evidence of some phonological correlations, the model is only able to acquire a general regular rule if there is sufficient variability in the phonological structure of regulars to “wash out” spurious correlations. This fact leads to a number of systematic difficulties.

First, because there are frequent phonological correspondences between stem and past forms, which are irrelevant for regulars, the model sometimes hypothesizes bizarre mappings—called blends—that do not resemble any structure that a native speaker would recognize (Pinker and Prince, 1988, give the example: mail/membled). Second, the model often has trouble generalizing the regular rule to stems which are phonologically very dissimilar to existing regulars (Prasada and Pinker, 1993, give the example: ploamph). As I discussed earlier, in such cases, native speakers nearly always employ the regular rule. Third, because the model employs only phonological structure, it cannot distinguish between phonologically identical words of differing morphological origin. It will uniformly produce flew as the past tense of fly regardless of whether it is used with its standard meaning or in the phrase fly out.

Subsequent connectionist approaches, which make representational assumptions similar to Rumelhart and McClelland’s model (e.g., Daugherty and Hare, 1993; Daugherty and Seidenberg, 1992; Hoeffner, 1996; MacWhinney and Leinbach, 1991; Plunkett and Juola, 1999; Plunkett and Marchman, 1993, 1991), also suffer from the same problems with the regular rule (see, Marcus, 2001; Pinker, 1999, for reviews). Although these models have made various enhancements to the original Rumelhart and McClelland work—such as adding additional layers to the network, or providing semantic or orthographic features on inputs and outputs, they all fail to provide mechanisms by which abstract categories such as VERB or STEM can be represented, and therefore they suffer similar problems to those just described.15

---

14See Marcus (2001) for a discussion of this class of network architectures from a psychological perspective. For a mathematical overview, see Bishop (1995).

15These weaknesses all follow from the scheme which the model uses to represent mappings, thus these limitations do not necessarily invalidate other aspects of the model. As I will discuss in the next section, the model improves on traditional linguistic approaches by giving a precise account of how various generalizations compete with one another and by providing a theory of learning.

It is also important to emphasize that the claim of traditional linguistic models is not that abstract categories
Feedforward Networks and Overregularization  In Section 4.3.5, I mentioned several controversies regarding the interpretation of the developmental data presented in Marcus et al. (1992). In particular, advocates of connectionist networks have called into question the evidence for a global period of early correct learning, and the correlation between the emergence of reliable regular marking and overregularization. In this section, in order to clarify the issues underlying these debates, I focus in detail on the way in which feedforward networks account for overregularization (see also Section 4.4.2).

Because feedforward networks require high variability in the structure of stems in order to wash out spurious generalizations, the original Rumelhart and McClelland model was able to account for overregularization only by introducing a discontinuity in the training regime. After initial training on a set of forms which were mostly irregulars, the model was flooded with a large number of regular forms. The sudden increase in the variability associated with the regular generalization led to overregularization errors. This move was heavily criticized as unrealistic (see, e.g., Marcus, 1995; Marcus et al., 1992). Children do not receive input which is discontinuous in this way, but, rather, are exposed to a dataset of gradually increasing size with an overall cumulative increase in the number of regular types.

There have been many subsequent attempts to achieve U–shaped learning in feedforward networks without an input discontinuity. MacWhinney and Leinbach (1991) showed that a feedforward network can exhibit overregularization and recovery without an input discontinuity, but were unable to demonstrate early correct performance. Plunkett and Marchman (1991) achieved similar results, showing, in addition, that the rates of overregularization for various verbs are highly dependent on the type frequencies of inflectional classes in the inputs. They were also able to achieve early correct learning for some verbs, but no global effect (i.e., micro–U–shaped learning). Hoeffner (1992) showed that U–shaped learning is possible without abrupt input discontinuities in an attractor–network based connectionist model. However, the overall rates of overregularization and the rate of recovery in this model are inconsistent with the findings in Marcus et al. (1992).

Plunkett and Marchman (1993) were able to achieve overall early correct behavior, but this appears to be attributable to a more subtle training discontinuity: They first trained the network to perfect performance on a subset of twenty (mostly irregular) verbs before expanding the input set (Marcus, 1995; Plunkett and Marchman, 1996). Furthermore, their model exhibits significant rates of overirregularization—the misapplication of irregular inflection to regular forms—a phenomenon which is very infrequent in child language acquisition (Xu and Pinker, 1995). Based on a systematic analysis of the Plunkett and Marchman models, Hoeffner (1996) argues that the Plunkett and Marchman (1996) results do not generalize to more realistic input settings.

Hoeffner (1996) further explored a large number of factors which may affect U–shaped development in feedforward networks. The overall result was that none of the manipulations provided robust U–shaped learning except for manipulations that involve changes in semantic and phonological representations during learning. In particular, he found that phonological manipulations, which obscured the structure of regulars, and semantic manipulations, which obscured the identity of the form being processed, could lead to correct U–shaped trajectories. A final feedforward model, Plunkett and Juola (1999), demonstrated correct U–shaped learning, but only by employing a training schedule where the size of the training set expanded exponentially rather than linearly—an obvious absurdity for real language learning.

cannot be partially predicted from phonological or semantic correlates but, rather, that these abstract categories exist. Of course, any system must have a way of learning which words are verbs, and which parts of words correspond to stems and which parts correspond to suffixes. Without abstract categories with which to associate these properties, however, generalization cannot be handled correctly.
Related Approaches which Allow for Abstraction

Although feedforward connectionist networks in the tradition of Rumelhart and McClelland (1986) cannot represent abstractions such as *VERB*, this is not a necessary property of the connectionist framework. In the first part of this section, I briefly review some connectionist networks that extend the feedforward architecture with mechanisms able to make categorical distinctions between regulars and irregulars.

Connectionist approaches are often grouped together with analogical models (see, e.g., the introduction of Skousen et al., 2002), and, in the second part of this section, I review the application of these models to the English past tense. For both sets of models, the ability to represent abstract classes of inputs means that they can avoid some of the problems discussed above.

Other Connectionist Approaches A number of connectionist models have introduced mechanisms which go beyond the simple feedforward assumptions of the approaches described above. As a result, these models have been able to avoid some of the problems (e.g., blends, failure to generalize to phonologically odd irregulars, etc.) characteristic of the feedforward approaches (see, Marcus, 2001; Pinker, 1999, for more detailed reviews of many of models mentioned below).

Some of these connectionist models have made explicit use of a bipartite architecture which has a component designed to handle the regular generalization (Hare et al., 1995; Westermann and Goebel, 1995). By providing two components, such models can sort verbs into two classes (corresponding to regulars and irregulars), and handle generalization differently for each class. Another kind of connectionist network, which has been more successfully applied to the English past tense, is the classifier network (Forrester and Plunkett, 1994; Hare et al., 1995; Nakisa and Hahn, 1996; Plunkett and Nakisa, 1997). Most feedforward networks implement a function which maps from a phonological representation of input stems to a phonological representation of past tense forms. Classifier networks, instead, start out with a set of past tense inflectional classes (e.g., /s/ → /s/, +/d/) and learn to select the correct class for each input stem. These networks do not have to learn segmentation for stems and affixes or the phonological transformations associated with different inflectional classes. Instead, they select which features of a stem are diagnostic of each class. By separating the problem of constraining the input domain of each class from the problem of generating output forms, such models can explicitly learn which input features to ignore, implementing a form of abstraction. Another very interesting extension of feedforward models is the constructivist neural network (Westermann, 1998; Westermann et al., 2009). These networks learn their structure (i.e., the number and connectivity of nodes) in addition to association strengths during training. When applied to the English past tense, constructivist neural networks are able grow a network topology in which regulars and irregulars are handled by different components of the system (Westermann, 1998). In all of these examples, the addition of a mechanism which allows for a categorical distinction between regulars and irregulars allows the models to produce more accurate generalization of the regular rule.

Connectionist approaches that provide such mechanisms are also, in some cases, better able to account for the developmental trajectory of overregularization. For example, constructivist neural networks can achieve U-shaped performance because they gradually learn to divide the input space between specific irregular processes and a general rule, exhibiting reorganization during training (Westermann, 1998). Likewise, O’Reilly (1996) presented a connectionist framework which can produce U-shaped development by employing two different learning mechanisms in the same network: error-driven (back-propagation) learning and association-based (Hebbian) learning. In this model, the network is provided with an additional +PAST feature, marked on past tense forms. The associationist learning mechanism learns to associate this feature directly with the common ending of regulars (i.e., +/d/) —effectively ignoring the phonological structure of regular stems.
Error–driven learning, similar to that in Rumelhart and McClelland (1986), accounts for irregulars. Both O’Reilly’s and Westermann’s models provide a mechanism by which regulars and irregular forms can be segregated and explain U–shaped learning by allowing different trajectories for each set.

**Analogue Models** In addition to rule–based and connectionist approaches, the English past tense has also been studied using a number of analogical models. These models specify algorithms by which input forms can be classified based on their similarity with large sets of stored exemplars. For example, in the applications to the past tense which I describe below, an input stem is compared with a large number of stored stems, each of which is labeled with an associated inflectional class (e.g., +/d/, or /l/ → /x/). When a stem is phonologically similar to some stored stem (or set of stems), it is assigned to the same inflectional class as that stem. These models differ from one another in how they compute similarity, and how they determine the categorization of inputs based on the similarity.

Skousen (1992) developed an exemplar–based approach, known as *Analogue Modeling of Language* (AML), which stores all input forms and generalizes to novel instances by finding the subset of stored exemplars (called a context) that most reliably predicts output forms (see, also, Skousen et al., 2002). Derwing and Skousen (1994) applied AML to the English past tense but found that their model was unable to reliably assign the correct past tense allomorphs to stems. Furthermore, it produces overregularization rates far in excess of those seen empirically. However, another study applying AML to the past tense, Eddington (2000), was more successful. It produced accurate classification of input stems and captured the fact that the generalizability of irregular inflections depends upon the phonological structure of the stem to a greater extent than the regular rule (Prasada and Pinker, 1993).

Eddington (2000) also explored another analogical model: *memory–based learning* (MBL). MBL is a form of nearest–neighbor classification which is augmented with mechanisms that discount unimportant dimensions of similarity (see, Daelemans and van den Bosch, 2005, for details). This model performed similarly (although slightly worse) to AML on the data in Eddington’s study.

Albright and Hayes (2003), applied Nosofsky’s exemplar–based *Generalized Context Model* (GCM) to the problem of the English past tense (Nosofsky, 1990). GCM computes feature–based similarity between an input and stored exemplars, weighting each exemplar by a measure of its representativeness. Albright and Hayes (2003) found that, for adult novel verb generalization data, this model did not account as well as an alternative probabilistic, rule–based model (see Section 4.4.2, for a discussion of the rule–based model). Albright and Hayes (2003) argued that the GCM and other analogical approaches, such as AML and feedforward connectionist networks, fare poorly on the English past tense because they cannot ignore spurious generalizations that result from incidental similarities between inputs and stored exemplars.

However, Albright and Hayes’ criticisms may only apply to the specific versions of the analogical models which they discussed. In general, analogical models provide recipes for calculating the similarity between inputs and stored exemplars, and for using these calculations to classify inputs; they do not specify how inputs and stored exemplars should be represented. Although the models just described (and considered by Albright and Hayes) used relatively simple phonological representations, there is nothing in principle about analogical models which prevents the use of

---

16Roughly speaking, AML operates by finding a set of exemplars (called a context) whose features match the input and all predict the same outcome. For example, if the input stem was “fring”, a context might be a set of stored stems which all end in “-ng” and are members of the /l/ → /x/ inflectional class. The set of all such contexts is computed, and the largest context, which does not contain a smaller context with a different predicted outcome, is chosen to classify the input.
more abstract structures, such as morphological categories. In fact, the version of AML used in Eddington (2000) likely out–performed that used in Derwing and Skousen (1994) because it employed richer, more realistic phonological representations. Therefore, while the shortcomings of these models illustrate, once again, the importance of abstract representations, they do not necessarily invalidate other aspects of the analogical approach.

4.4.2 The Necessity for a Quantitative Account of Competition

The preceding sections discussed how many connectionist and analogical models fail to account for the English past because they lack abstract generalizations in their hypothesis spaces. Of course, this is not a problem for traditional, rule–based accounts, and, in this section, I will identify a different problem that they face: the failure to provide an adequate, quantitative theory of competition between the many generalizations which are consistent with the data. I will then give an overview of several existing theories which do provide an account of competition, emphasizing some common features in how they resolve competition.

Classical Deterministic Theories

Classical linguistic theories of the past tense posit that there are abstract rules for each kind of inflectional marker, including irregulars (e.g., Chomsky and Halle, 1968; Halle and Marantz, 1993; Kiparsky, 1982b; Mohanan, 1986). The application of these rules is constrained by conditions on their inputs, such as sensitivity to the phonological structure of stems. In many cases, several rules serving the same function will be available for a given input. When this scenario arises, the theories must provide a means to resolve the competition between rules.

A general mechanism often invoked for handling competition is the *elsewhere condition*, also called *Païnini's principle* (Anderson, 1969; Kiparsky, 1973; Stump, 2001b). The elsewhere condition states that a rule with more specific input conditions takes precedence over a rule with more general conditions (e.g., Kiparsky, 1982b; Stump, 2001b, for use in ranking morphological rules.). For example, some theories (e.g., Chomsky and Halle, 1968; Noyer and Harley, 1999) allow rules to be annotated with lists of stems to which they can apply. These annotations can be used to associate irregular inflectional classes (e.g., $/i/ \rightarrow /s/$) with specific sets of stems (e.g., *fling*). Under such an account, the regular rule is not annotated for specific stems, but can apply to any verb. Because the input conditions on the regular rule (i.e., any verb), are less specific than the input conditions on the irregular rule (e.g., the stem *fling*), the elsewhere condition prefers the irregular rule when it can apply, and regular rule *elsewhere*.

Another strategy for handling competition between rules is to posit that the lexicon is organized into ordered *strata* or *levels*. Morphological processes that operate in earlier levels take precedence over processes assigned to a later level (Halle and Mohanan, 1985; Kiparsky, 1982b,c; Mohanan, 1986). In the case of the past tense, the regular rule might be assigned to a later stratum than the irregular rules (see also Section 6.3.5), and, thus, only apply if no irregular rules have applied earlier. In order to handle rule competition that arises within a level, stratal approaches often need to make additional assumptions—such as a variant of the elsewhere condition (e.g., *Kiparsky*, 1982b) or an *avoid synonymy* principle (e.g., Kiparsky, 1982c).

Both of these approaches to handling rule competition offers certain insights into linguistic structure, however, all of these principles or mechanisms are *deterministic* and therefore they do not provide natural accounts of *quantitative* aspects competition between rules, especially those involved with learning. For example, these principles do not provide an adequate account of the phenomenon of overregularization. Individual children produce both overregularized and correct past forms of irregular stems (Marcus et al., 1992). Because the the principles just discussed are all—or—
nothing, they predict that children should produce one or the other for a given stem. Moreover,
these principles do not explain why the rate of overregularization varies depending on the frequency
of forms and their similarity to other irregulars.

These deterministic principles also have difficulties accounting for the occasional generalization
of irregular rules. Although past–tense irregular rules are much less productive than the
regular rule, they can occasionally be applied to novel stems. Traditional deterministic theories
predict that in such cases one, and only one, of the possible rules will always apply (whether regular
or irregular). Therefore, they do not naturally account for cases where speakers produce several
different forms at different rates, or cases where speakers’ judgments of acceptability reside on a
cline (e.g., fring → frang/frung/fringed). Another challenge for these traditional theories is that
they do not have a direct way of incorporating frequency, and, thus, it is difficult to link them to
processing results such as those discussed in Section 4.3.4. Finally, these traditional theories shed
little light on the dynamics of language change: They do not provide a mechanism which allows
low–frequency irregulars to become regularized over time, and they do not explain why irregularity
persists in the first place.

The Blocking and Retrieval Failure Hypothesis of Marcus et al. (1992)

Recognizing the limitations of classical deterministic linguistic theories, many researches have adopted dual–mechanism approaches. One standard instantiation of the dual–mechanism ap-
proach is the blocking–and–retrieval failure (BRF) hypothesis of Marcus et al. (1992). The BRF
posits an architecture consisting of two components. Regulars are handled via a rule component,
similar to classical linguistic theories, and irregulars are stored in an associative memory system.
Competition between the regular rule and irregular past tense forms is handled by the blocking
principle which states that when both the regular rule and a stored irregular form are available for
a particular stem, the stored form should be preferred. The existence and rate of overregularization
is accounted for by assuming that the associative memory system is probabilistic, and stored forms
have a probability of being correctly retrieved which is related to their frequency (and similarity to
other stored forms). When a stored irregular is insufficiently strong (e.g., when it has been witnessed
too infrequently or is too dissimilar to other stored irregulars), retrieval can fail, and, as a result,
blocking will not apply, which will cause an overregularization error.

The BRF theory is designed to simultaneously account for the productivity of the abstract
regular rule and explain varying rates of overregularization for different irregular forms. Under this
account, because adults have been exposed to vast amounts of data, they posses robust memory
traces for irregular verbs, and, therefore, nearly always produce the appropriate past forms. Occa-
sional overirregularization (e.g., bite/bote) in this theory proceeds via generalization processes made
available by associative memory, explaining its low frequency in learners (and even lower frequency
in adult speech). U–shaped development is explained by a combination of two factors. Because irreg-
ular verbs constitute the highest frequency verbs in English, learners will often initially store them
whole and they will have relatively strong memory traces. At the point in time when the regular
rule is acquired, it will begin to compete with these stored irregulars. However, rates of overregu-
larization will in general be low because blocking will only fail probabilistically, and failure will be
less common for more frequent verbs. Furthermore, because irregulars are stored in a structured,
associative memory, they will be protected from overregularization by similarity to other irregulars.
As the child hears more examples of irregular forms, memory traces are strengthened—and recovery
will gradually occur.

The Rule–Learning Component in BRF and Overregularization

Because the morphological rules of a particular language cannot be innate, to provide a complete account of past tense
acquisition, a dual–mechanism theory like BRF must be paired with a theory of how productive rules, such as the regular past tense rule, are learned. In this section, I consider a detailed example of how this might be accomplished, with two goals in mind.

First, I will argue that the controversy surrounding the relationship between reliable regular marking and the onset of overregularization (see 4.3.5) is largely the result of a misconception. This debate has been driven by the assumption that theories with rule–based components, like the BRF, must predict that the acquisition of the regular rule results in a sudden increase in forms generated by that rule (also see Section 8.3). I will show, however, that when integrated with a plausible account of rule learning, the BRF is consistent with a wide variety of developmental trajectories. Second, even when integrated with an account of rule–learning, the BRF still leaves open the question of how to resolve several kinds of competition. I will argue that providing precise, quantitative theories of these kinds of competition is necessary to empirically distinguish between models.

There have been a number of theories of inflectional rule learning that could plausibly serve as the rule learning component of BRF (e.g., MacWhinney, 1978; Pinker, 1984). Here I consider the algorithm presented in Pinker (1984). This algorithm is built around a paradigmatic representation of inflectional structure. Paradigms are tables with dimensions labeled by semantic, phonological, and formal attributes, each with a number of specific levels. For example, a dimension might be labeled by the semantic attribute TENSE with levels PAST and PRESENT, by the phonological attribute NUMBER-OF-STEM-SYLLABLES with levels ONE and MORE-THAN-ONE, or by the formal attribute DECLENSION with levels I and II. The values of unobservable, purely formal attributes are also marked on the lexical entries for individual words to allow the algorithm to learn the structure of languages with arbitrary subclasses of inflections (e.g., nominal declensions, verbal conjugations, etc.). The theory makes use of both word–specific paradigms, where cells are filled with individual words, and word–general paradigms, which represent inflectional rules that apply to classes of words. Both individual forms and paradigms have associated strengths which are related to factors such as their frequency in the input data.

Words are initially stored whole in the lexicon, and associated (via sampling) with an appropriate cell in a word–specific paradigm. For example, if a child hears walked in a past tense context, then this word will be stored, and entered into a word–specific paradigm. The particular cell in which the word is entered is sampled randomly from one of the attribute–level combinations which was observable in the input (e.g., for walked: PAST on the TENSE dimension, or COMPLETED on the ASPECT dimension, etc.). The particular dimensions and levels which are available to the learner are drawn from a universal set, and are sampled by the learner according to prior probabilities that encode the prominence of each dimension across inflections in the world’s languages.

As children hear forms used in more contexts, they gradually expand the number of dimensions associated with each word–specific paradigm. Other mechanisms then segment the forms in word–specific paradigms, and extract the common phonological material across several word–specific paradigms in order to construct word–general paradigms (i.e., rules) from the shared material.

Competition is handled via two mechanisms. First, multiple words are prevented from being stored in the same cell of a word–specific paradigm. This is known as the unique entry principle (UEP). Second, a distinction—marked by a pre–emptability diacritic—is maintained between forms generated from word–general paradigms (i.e., computed by rule) and words retrieved from memory. When a retrieved form is available, it pre–empts the rule–generated form, implementing blocking.

Although the algorithm is not a probabilistic generative model, its details suggest that it could readily be reformulated as one. The sampling procedure used in the algorithm is essentially a form of rejection sampling from the posterior over paradigms, conditioned on the input data. The prior weights assigned to various attribute dimensions, combined with principles like the Unique Entry Principle and the ordering in which various general paradigms are iteratively constructed, conspire to approximate a prior distribution. A fully Bayesian model would encode such preferences not as specific search procedures, but, rather, as a simplicity prior over the space of possible inflectional
Note that both principles are needed: The UEP handles competition between multiple stored forms, while the pre–emptability diacritic handles competition between stored forms and rules. Both mechanisms are also probabilistic and defeasible. If the memory traces of two forms are strong enough (i.e., there is sufficient evidence), they may both be stored in a single word–specific paradigm cell. Likewise, if the strength of a word–specific paradigm is insufficiently strong, it may not preempt the use of a word–general paradigm.

Unsurprisingly, many of the basic assumptions of this theory are similar to those of the BRF model. Under both models words are initially stored whole in memory, retrieval from memory is probabilistic and related to frequency, and the existence of a stored form can block the use of a regular rule to express a particular meaning. The Pinker (1984) algorithm, however, provides additional mechanisms that allow rules to be abstracted from input forms in a way that remains consistent with early correct performance: The algorithm starts by storing all forms (and form specific paradigms) and, only later, abstracts general rules.

There are two points to take from these discussions. First, the Pinker (1984) algorithm is consistent with a wide variety of rule–learning trajectories. In the model, both individual forms and paradigms are weighted, the principles which handle competition are probabilistic and defeasible, and the process of abstracting rules is iterative. These facts mean, for example, that it is possible for this algorithm to reach a state where a version of the regular rule has been learned, but where this version is not yet fully general (i.e., it may include an unnecessary phonological, semantic, or formal condition), and therefore does not apply across the board to all past tense forms. Likewise, because the UEP and pre–emptability are probabilistically defeasible, the model allows for a variety of interactions between the regular rule and irregular forms. The model can produce overregularization if a memory trace for a form in a word–specific paradigm is insufficiently strong to pre-empt a word–general rule. The rates with which this will happen will depend on precisely how memory strength and pre-emption are implemented in the model. Most importantly, the model does not predict a sudden increase in the use of the regular rule at the moment when this rule is first abstracted. While it can produce this behavior in principle, a sudden increase is not a logical necessity. I discuss this issue more broadly in Section 8.3.

Second, even when combined with the BRF, this model leaves unspecified how several kinds of competition should be resolved. First, the mechanisms of abstraction that learn word–general paradigms allow the model to acquire word–general rules with overlapping domains of applicability. For example, because the algorithm can hypothesize rules which are sensitive to phonological structure, as well as rules which are sensitive to arbitrary, purely formal structure (e.g., declensional classes), there is nothing to stop it—in principle—from learning several versions of the regular rule with disjoint or partially overlapping domains of applicability. Second, the model can learn rules which correspond to irregular inflectional classes (e.g., /I/ → /2/). Under the BRF hypothesis, these kinds of generalizations are handled by associative memory. This raises the question of whether irregular generalizations should be handled by the rule component, the associative memory component, or both, and, how the system decides in each case. I return to this issue in Section 4.4.3.

More generally, both of the issues just discussed are important because distinguishing between theories of overregularization requires making precise quantitative predictions about the trajectory of the phenomenon. Although they differ in how they represent generalizations, all theories predict overregularization for the same basic reason: competition between irregular forms and a generalization corresponding to the regular rule. The only factor which can empirically distinguish theories is the degree of competition at different stages of learning. Thus, theories need to specify precise, quantitative mechanisms for competition between competing generalizations.

systems for a language.
The Dual–Mechanism Model of Taatgen and Anderson (2002)

An implemented, computational model of the past tense is explored by Taatgen and Anderson (2002). This model, formulated in the ACT–R framework (Anderson and Lebiere, 1998), makes use of ACT–R’s distinction between declarative and procedural memory, and, thus, is a variant of the dual–mechanism approach. In ACT–R, declarative memory consists of a database of stored facts, while procedural memory consists of a set of production rules which operate over those facts.

To formalize the past tense in this framework, Taatgen and Anderson provide their model with three initial production rules which correspond to three strategies for generating past tense forms. The first production rule directly retrieves facts which encode the past tense of stems from memory. For example, this rule might retrieve a fact like: past[sing] = sang. The second production rule formalizes a kind of analogical reasoning over past tense forms. Past tense forms are represented as consisting of two slots, corresponding to stem and inflection. The analogical rule can inflect an input stem by randomly sampling another past form stored in memory, and using the inflection on that form. The third production rule implements a do-nothing strategy which, rather than inflecting an input stem, simply returns the stem without inflection.

Like Fragment Grammars, ACT–R provides mechanisms which compile frequently co–occurring sequences of computations into single production rules (Taatgen, 2003). Using these mechanisms, a version of the regular past tense rule can be discovered by combining steps which retrieve a regularly inflected form from memory and then apply it to a new stem using analogy. Thus, in Taatgen and Anderson’s ACT–R model of the past tense, the system has three options in deciding how to inflect a stem: It can do nothing; it can retrieve the corresponding past form stored for a stem direction from memory, or it can choose an inflection by analogy with stored forms. This last possibility is enhanced by the compilation mechanism, which allows particularly valuable analogies (such the use of the the regular ending) to become single production rules.

Given an input stem, the model chooses which production rule to use based on that rule’s expected outcome. In ACT–R, the expected outcome of a production rule is based on four factors: (i) the prediction accuracy of the rule on previously generated past tense forms (ii) the utility of the output which the rule computes (presumably constant for all forms in these past tense simulations), (iii) a fixed cost (measured in time) associated with each production rule, and (iv) random noise. The rule with the highest expected outcome is chosen to inflect an input form. The high–frequency of irregulars means that the retrieval route is favored for these forms because of its high prediction accuracy in the past (Factor i). For regulars, the model eventually learns that the compiled, abstract regular rule has the highest prediction accuracy.

The model achieves impressive results in approximating the final state of the learner. It associates stems with the correct past tense inflectional allomorph and generalizes the regular rule to novel stems. It is also able to correctly capture U–shaped development. In the model, the do–nothing rule is assigned lower cost than analogy or retrieval. As a result, like children, early in learning, the model produces uninflected verb forms at high rates. Although it has low cost, the do–nothing rule also has poor accuracy in predicting the past tense forms of stems. By contrast the other rules are able to predict the past tense of stems with higher accuracy, despite their higher cost. Therefore, as more data is observed, they begin to have higher expected outcome, and, thus, be preferred. This happens first for high–frequency irregular forms, leading to a period of early correct performance. Eventually the model discovers the regular rule, and begins correctly inflecting low–frequency and novel regular stems. At the same time, overregularization begins to occur. Recovery is gradual, and relative rates of overregularization are realistic.

Taatgen and Anderson’s model is similar to Fragments Grammars in several respects. First, the representations used for the past tense are similar to those which I describe in the next chapter: They do not take into account phonological structure, and they treat past forms as being composed of
Chapter 4: The English Past Tense: Abstraction and Competition

stems and inflections. Second, the model makes use of mechanisms which compile frequent sequences of rules into single reusable units. Third, and most importantly, like Fragment Grammars, this model explicitly treats the problem of past tense inflection as an optimization. It finds the best way to account for the data by optimizing the expected outcome of production rules. As a result, like Fragment Grammars, blocking is a consequence of the ability of the retrieval rule to more reliably predict irregular past forms, and, unlike classical linguistic theories or the BRF, is not an independently stated principle of the system.

Nevertheless, Taatgen and Anderson’s model also differs from Fragment Grammars in a number of ways. First, Fragment Grammars are defined by a single, uniform, probabilistic generative model: Unlike the ACT–R framework, there is no distinction between facts stored in memory and production rules, instead, there is a single generative process that can choose to reuse larger or smaller units. Second, in the framework described in this thesis, inference is based on conditioning this generative model on observed data. Therefore, the only quantity optimized by Fragment Grammars is prediction performance (see Section 1.5.4). The Taatgen and Anderson (2002) model, in contrast, optimizes a complex function which includes not only prediction performance but also quantities such as the cost of production rules and the utility of goals. Furthermore, during learning, these quantities are updated using heuristics, rather than by probabilistic conditioning. Third, in ACT–R, compilation of frequently co–occurring sequences of productions rules is handled by a special, distinct mechanism. In contrast, for Fragment Grammars, compilation is part of the process of generating observations. Finally, the Taatgen and Anderson (2002) model makes use of a larger number of parameters than Fragment Grammars, and, thus, allows extra degrees of freedom in accounting for the past tense data. Of particular importance are the costs which Taatgen and Anderson (2002) assign to different production rules. As I discussed above, the ability of their model to capture U–shaped development hinges crucially on the fact that they assign the do–nothing rule a lower cost than other rules. This difference in cost seems somewhat arbitrary from a psychological point of view: It is not obvious that the using the bare stem form of a verb should be significantly less costly than using other forms.

Nevertheless, despite the differences between Taatgen and Anderson (2002) and the present framework, they both adopt a view of productivity and reuse which involves optimizing prediction performance. Taken together with the results in the next chapter, these two models provide strong support for the idea that productivity and reuse can be inferred from distributional information in the data available to language learners.

Probabilistic Rule–based Theories

Several recent models of the past tense have elaborated on the traditional linguistic rule-based approach by employing probabilistic measures of rule applicability. For example, Yang (2002) describes an approach which employs rules for all inflectional classes (as in Chomsky and Halle, 1968), but which enhances rules with two probabilistic mechanisms not present in traditional accounts. First, the matching of stems to rules is probabilistic; that is, a stem matches a rule (with appropriate input conditions) with some probability $p$. When matching fails a default rule is applied. Second, even after matching a stem to a rule, the application of the rule is noisy, and with some probability it fails, again leading to the use of a default. Yang (2002) argues that the model can account for the standard effects of frequency and similarity on the generalization of regular and irregular classes (see Section 4.3), but he does not provide a computational implementation of his model or an evaluation of these claims (see Section 4.4.3 for further discussion of Yang’s arguments).

The Yang (2002) model is similar to Fragment Grammars in that it focuses on the problem of learning which inflectional classes apply to a stem, rather than learning the phonological content of the classes. However, it differs from Fragment Grammars in that it makes use of an explicit notion
of a default rule but does not specify how the learner decides which generalization is the default. Furthermore, it does not provide a precise specification of the way in which the relevant distributions should be estimated, although it is clear from Yang’s discussion that frequency plays a role.

Albright and Hayes (2003) described another probabilistic rule–based model. Unlike the preceding approaches, their model is concerned with learning the correct form of phonological generalizations for past tense inflectional classes. Their model uses a minimal generalization learning algorithm (Albright and Hayes, 2002) which constructs rules by aligning training examples and abstracting common phonological structure. Rules are assigned scores using a statistic which compares the number of items in the dataset to which a rule can apply (i.e., its abstractness or level of generalization) to the number of items to which it does apply (i.e., its reliability). This quantity is modulated by a frequentist confidence score, which reduces the strength of rules abstracted from fewer datapoints. Like the current model, the Albright and Hayes (2003) framework makes no explicit use of a blocking principle, but, rather, relies on confidence scores to resolve competition. Albright and Hayes found that their model closely predicts people’s judgments about the generalizability of past tense inflectional classes to novel stems. An important additional finding, is that their model learns several versions of the regular rule, some of which are conditioned on particular phonological features of the stem (e.g., it inflects the set of regulars with stems ending in voiceless fricatives with a special version of the regular rule).

One fundamental difference between Albright and Hayes (2003) and the other models discussed in this section is that it does not provide an explicit mechanism whereby irregular rules can be directly associated with stems (although they do suggest that such a mechanism must exist). Rather, their rule scores can best be understood as frequentist estimates of the probability that a particular rule applies given that its input conditions are met and that the stem is novel. That is, their theory is a theory of novel form generalization, rather than reuse of existing forms. In this vein, Albright and Hayes (2003) evaluate the model against behavioral data that consists entirely of novel stems. Because the model lacks a way to directly associate stems with irregular classes, it seems likely that the model will overgeneralize irregular classes to existing stems when these stems closely match the input conditions on an irregular rule (e.g., bring/brang). Conversely, the Albright and Hayes model does provide important insights into how the phonological structure of past tense rules can be learned, and how these generalizations can be applied to novel stems.

Other Models of Rule Induction

Finally, there have been several other computational models which have focused on the problem of inducing the phonological structure of past tense rules. Yip and Sussman (1996, 1997) provide a model of phonological rule learning in the constraint propagation framework. Inputs and outputs are encoded as vectors of (binary) features, and (binary) constraints are learned between these vectors of features. For example, a constraint might be learned that requires that an output feature take on the value TRUE whenever some particular input feature takes on the value FALSE. Constraints may also ignore features in the input and output vectors—allowing a constraint to represent generalizations which abstract over classes of input and outputs. Yip and Sussman (1996) and Yip and Sussman (1997) provide an algorithm which iteratively constructs increasingly general constraints from a corpus of examples.

18Compared to Fragment Grammars, for existing stems, the Albright and Hayes (2003) model seems likely to underestimate the probability of the regular rule and overestimate the probability of irregular rules. Note that a correlation with human data on only novel stems—like that reported in Albright and Hayes (2003)—will not reveal this issue since the baseline productivity of all inflections (when evaluated on novel stems) is absorbed by the intercept. Such a correlation will only reveal whether the relative fit of different processes (on novel stems) is accurate.
Competition between constraints is handled by computing an activation level for each constraint on a given input vector. Because constraints are activated in proportion to the number of input feature values which they match, this mechanism indirectly encodes a version of the elsewhere condition, selecting more specific constraints when they are available. Yip and Sussman (1996) and Yip and Sussman (1997) show that their algorithm is able to efficiently build a system of rules capable of correctly inflecting stems for past tense using only a small number of training examples. They also note that learning becomes faster as more constraints are generalized. Their system is robust to noise, but is not sensitive to the token frequency of forms in the input, and they do not provide evaluation on specific experimental or developmental datasets. While their algorithm may provide an elegant account of the induction of the structure of phonological rules, its insensitivity to token frequency makes it unlikely to be able to account for many of the specific phenomena discussed in Section 4.3.

Molnar (2001) presents a related algorithm for learning phonological rules which consists of two components, called generalize and sift. Generalize iteratively constructs increasingly abstract rules from the input corpus by finding and factoring shared phonological substructure between forms.\footnote{This notion of construction of generalizations via alignment of structures and identification of common subparts is the bedrock of distributional learning and is nearly universal in the literature on learning algorithms for linguistic structures. Pinker (1984) makes use of a similar mechanism for abstracting general paradigms. Yip and Sussman (1996) and Albright and Hayes (2003) also use similar techniques. Formal implementations of the idea (for inducing phrase-structure grammars) go back at least to Solomonoff (1959). There is an enormous literature on these techniques in the context of syntax induction. It should be noted that, despite historical discussions of the limitations of such approaches (see, e.g., the discussion of discovery procedures in Chomsky, 1965), recent work has shown that they can be (provably) successful even in quite complex domains (e.g., Clark, 2006, 2010; Clark and Lappin, 2011).} The sift component associates a probability with each rule, proportional to the frequency with which it is used to generate forms in the input. Molnar (2001) does not provide any evaluation on experimental or developmental data.

Ling and Marinov (1993) presented a model of past tense (phonological) rule induction which learns a mapping between (fixed length) input and output vectors of (binary) phonological features. This mapping is implemented as a decision tree: Vectors of output features are computed by considering each input feature in a specific order, with later decisions depending on earlier ones. The algorithm which learns feature orderings is based on an information gain criterion: Input features which provide greater amounts of information about outputs are used before those which provide less information. Because input features associated with irregulars (e.g., features encoding the irregular stem sequence -IN) are more predictive of specific output features (e.g., features encoding the irregular past sequence -æN), they are ordered first in decision trees. This leads the algorithm to enforce a version of the elsewhere condition whereby more specific, predictive features are checked first. Ling and Marinov (1993) showed that their system was able to learn to produce past tense forms for existing stems with high accuracy. However, it was not able to produce accurate U-shaped learning without additional manipulation of a noise parameter.

A final phonological rule learning algorithm for the past tense, developed in an inductive logic programming (ILP) framework, is presented by Mooney and Califf (1995). This algorithm iteratively generalizes Horn clause predicates\footnote{A Horn clause is a disjunction of literals with at most one positive literal. These can be expressed as predicates of the form, \((p \land q \land \cdots \land t) \Rightarrow u\), and thus naturally correspond to classical linguistic rules of the form: \(u \rightarrow p q \cdots t\). In the logic programming literature, grammars expressed in terms of Horn clauses are referred to a \textit{definite clause grammars}.} which correspond to regular and irregular inflectional classes. The most important aspect of this study is that in addition to standard ILP generalization mechanisms, they find that they need to enforce a constraint whereby more specific predicates are ordered before more general ones—again, an elsewhere condition to handle rule competition.
Chapter 4: The English Past Tense: Abstraction and Competition

The preceding models all differ from the models studied in the thesis in that they are focused on the problem of phonological rule induction, rather than learning the productivity of morphological processes. Moreover, for the most part, these studies have not provided evaluations on psychological datasets. For these reasons, they are not comparable to the current framework. However, the techniques exploited in these models may be useful in formalizing an account which integrates the tools explored in this thesis with rule induction (see also Appendix A). Furthermore, in almost every one of the preceding models, some version of the elsewhere condition was employed to handle competition between different forms.

4.4.3 How to Handle Less Productive Generalizations

The preceding sections have argued that a successful model of the English past tense must provide at least two components: mechanisms which allow for abstract generalizations and a quantitate account of competition between the multiple generalizations which are consistent with the data. There remains a residual issue which has been the focus of some debate in the literature: how less productive generalizations, such as the irregular inflectional classes, should be handled.

Dual–mechanism approaches posit that the systems which handle productive, regular generalizations and less productive irregular generalizations are different in kind. Generalization of regulars is handled by a rule–based component whereas generalization of irregulars proceeds via associative memory. By contrast, probabilistic, rule–based accounts propose that all generalizations are handled by probabilistic rules (Albright and Hayes, 2003; Yang, 2002).

Probabilistic rule–based theorists such as Albright and Hayes (2003) and Yang (2002) have offered several criticisms of dual–mechanism models. Albright and Hayes (2003) focused on the need for rules which handle highly prototypical subclasses of regulars. However, as discussed in Section 4.4.2, dual–mechanism theories (such as the BRF of Marcus et al., 1992) must be paired with an account of rule induction, and, furthermore, such an account must provide mechanisms which can learn rules with specific phonological and semantic input conditions. Therefore, the observations of Albright and Hayes (2003) can be taken as identifying an empirical gap in dual–mechanism theories, rather than as identifying a fundamental flaw with two–component architectures.

Yang (2002) critiques a different aspect of the dual–mechanism approach. He argues that a probabilistic rule–based account provides better fit to the rates of overregularization observed for different classes of irregulars than is possible for a dual–mechanism model. Yang considers two issues. First, he argues that the effects of frequency within particular irregular inflectional classes show a nearly perfect (inverse) correlation with rates of overregularization, but that, when all irregulars are considered, irrespective of class, this correlation is much lower. Second, he argues that low-frequency irregulars from high–frequency classes display a free–rider effect: They are overregularized at low rates despite their low frequency. He suggests that dual—mechanism models are unable to account for these effects because they do not provide rules which unify the irregular classes, but store irregulars in separate memory traces.

This claim is difficult to evaluate because the exact mechanisms underlying associative memory in dual–mechanism models are not fully specified. Sometimes, it is suggested that associative memory could be similar in architecture to a connectionist network (e.g., Pinker and Prince, 1988). However, the existence of languages with more complex mixtures of inflection marking raises a serious question. There are many cases across languages where several inflections are available to mark some function. These inflections are typically sensitive to different phonological, morphological, and semantic properties of the stem in ways which allow overlap in their domains of applicability. Moreover, unlike the past tense, there often several allomorphs which are (semi–)productive (when the relevant selectional restrictions are met). Cases which have been studied in detail in psycholinguistics include Italian verbal inflection, the German and Arabic plural system, and Polish, Russian,
and Serbo-Croatian case marking (Clahsen et al., 2004, 1992; Dąbrowska, 2001, 2004, 2008; Hahn and Nakisa, 2000; Kapatinski, 2007; Köpcke, 1988; Marcus et al., 1995; Nakisa and Hahn, 1996; Orsolini and Marslen-Wilson, 1997). These systems raise a problem for dual-mechanism theories since they do not seem to provide a clean divide between regular and irregular forms, but, rather, suggest a productivity cline amongst a number of partially overlapping rules.

A dual-mechanism account has two options in such cases. One option is to posit that these different ways of inflecting words are all represented in the rule system. As I discussed above, algorithms like the one presented in Pinker (1984) are able to learn systems with several different (potentially overlapping) classes of inflectional processes. However, if a dual-mechanism theory allows this for other languages, it is difficult to see why at least some irregular English inflectional classes cannot also be represented in the rule system as well. If this is allowed then the theory can account for Yang’s criticisms much in the same way as the theory that Yang advocates.

Another option for the dual-mechanism model is to posit that only the most productive generalizations are handled by the rule system, and that the remaining semi-productive regularities are handled by whatever mechanisms are available for generalization in associative memory. However, this approach calls into question the idea that associative memory can be organized in a manner that is similar to simple feedforward connectionist networks. Many of the semi–regularities present in languages other than English are productive within their restricted domains. Frequently, they are sensitive to abstract morphological, semantic, or phonological structure—such as the gender or animacy of stems. This suggests that if associative memory is like a neural network, it must have a more complex architecture than simple feedforward approaches. It would, at the very least, need to contain “nodes” representing various morphological categories, and it would likely need to be able to perform general operations on stems to handle semi-productive cases of generalization.

Alternatively, associative memory could be modeled using a more expressive analogical framework—such as memory-based learning, or analogical modeling of language—discussed in Section 4.4.1. However, as I have argued, it is unclear exactly what predictions these models make (in principle) that are different from (probabilistic) rule-based approaches since they do not specify a representation format—only a way of using similarity between representations. In any case, once a sufficiently rich analogical system is provided for associative memory, it could, again, answer Yang’s criticisms.

The preceding discussion highlights the problem with assigning processes to separate modules based on their productivity (i.e., the representational approach). As I argued in Chapter 1, productivity and reuse are properties which vary across all levels of linguistic structure. This is illustrated by the English past tense. Although the regular rule is far more productive than irregular inflections, irregulars can sometimes be generalized. This fact forces theories to provide mechanisms which can account for this generalization. However, if these mechanisms are sufficiently powerful, it becomes unclear how they differ from productive rules. Conversely, some regulars may also be handled by rules which are constrained by additional phonological or morphological selectional restrictions. Once mechanisms are provided which can learn such constrained rules, it becomes unclear why they cannot also account for irregulars. These problems are exacerbated in morphological systems which, unlike the past tense, show complex mixtures of rules with overlapping domains of applicability and multiple levels of productivity.

If differences in productivity can not provide decisive evidence for differences in mechanism, what other kinds of data can be brought to bear on the issue? One kind of data which has been offered is the existence of family resemblance structure (also called variegated similarity by Albright and Hayes, 2003). In a rule-based approach, all inputs to a rule need to be similar to one another in

---

21 A similar point is made by Marcus et al. (1992, p. 133) who point out that regular rules need not be condition-free, and that there can be mixtures of productive rules of varying levels of selectivity.
the same way. In a family resemblance category, items can be similar to one another along multiple dimensions without having any single defining characteristic. Many theorists have argued that this kind of structure is the hallmark of associative or analogical learning mechanisms (e.g., Albright and Hayes, 2003; Pinker and Prince, 1996; Prasada and Pinker, 1993; Ullman, 1999). Because of this, such family resemblance structure has been offered as evidence that irregulars are handled by an associative memory and not by rules (see especially, Pinker and Prince, 1996, for detailed arguments).  

However, the claim that only analogical and associative models can handle family resemblance categories appears to be incorrect. As emphasized by Albright and Hayes (2003), probabilistic rule–based models that allow for mixtures of rules of differing degrees of abstraction and productivity can also explain variegated similarity. For example, one class of irregulars are those which change /o/ (or another vowel to) /u/ in the past tense (e.g., blow, grow, know, throw, draw, slay). This class of verbs has many of the characteristics of a family resemblance category (Pinker and Prince, 1996). First, the class contains prototypical members (e.g., blow) and less prototypical members (e.g., draw). Second, there are a number of non–defining characteristics which are shared between members, but which may be absent—for example, these forms tend to end in /o/ and tend to start with a consonant cluster, but some possess one feature and not the other. Finally, there are unclear cases (e.g., slay). Nevertheless, it is possible to construct a system of overlapping probabilistic rules which can characterize this class. For example, such a system might consist of three rules, all of which transform a stem vowel to /u/ in the past tense, but differ in their input conditions: one rule sensitive to stem final /o/, another rule sensitive to initial consonant clusters, and a third sensitive to both. The degree to which a form is a member of the family resemblance category can be computed by summing the probability of all the rules which apply. This will lead to graded membership and prototypicality effects.

In fact, the main thrust of the Albright and Hayes (2003) analysis is that mixtures of probabilistic rules account for human generalization data better than analogical models. Again, however, it is unclear whether this criticism applies to all analogical approaches or to the particular analogical model which Albright and Hayes (2003) studied.

The challenge in resolving this debate is that, once abstract structures are allowed, and a quantitative account of competition is provided, it is not clear what precise properties distinguish analogical, associative and rule–based theories. The property of feedforward networks, in the tradi-

---

22 Dual–mechanism theorists have also argued against rule–based representations of irregulars on other grounds. For instance, MacKay (1976) presents a theory and empirical data which suggest that the complexity of a rule is proportional to the amount of change a stem undergoes in transformation to its past tense form, and that more complex transformations should block the regular rule with more difficulty. Marcus et al. (1992) show that children’s rates of overregularization are not (inversely) correlated with amount change from stem to past forms. However, the assumptions underlying MacKay (1976)—that rule complexity can be gauged by the amount of changed phonological material—are not natural in a probabilistic framework, and seem to be related to the historically–unsuccessful attempt to relate processing costs to derivational complexity, and so I do not consider them here.

23 One issue which this account raises is why it should be the case that a system such as the English past tense is so dichotomous, with a highly productive regular rule, on one hand, and a number of irregulars which seem to require mixtures of numbers of weak rules, on the other. The degrees of freedom allowed by the mixtures–of–probabilistic—rules account seem to suggest that we should instead expect to see many more classes with much greater variability in productivity. There are two possible answers to this. First, many languages do seem to provide more complex mixtures of semi–productive rules, and therefore the English past tense may be somewhat exceptional in this regard. However, another potential answer is discussed in Chapter 8. Under an inferential model of productivity and reuse the highest prior probability languages—that is, the most optimal and learnable systems—will be characterized by a variety of different (often partially overlapping) unproductive, irregularities at the high end of the word–frequency distribution, while the tail of the distributions will be characterized by a small number regular and productive generalizations. Thus, the inferential approach to productivity predicts that system which are more dichotomous will be favored over the cultural evolution of language.
tion Rumelhart and McClelland (1986), that underlay their inability to correctly account for the past tense was the lack of generalizations in their hypothesis spaces which could ignore irrelevant phonological or semantic structure in inputs. Similarly, classical linguistic theories could not account for scalar productivity, graded naturalness judgments, and variable overregularization because of their determinism. However, we have not yet identified similar properties which allow us to distinguish between theories once these faults are remedied. It could turn out, in fact, that any sufficiently rich, probabilistic analogical system can simulate a probabilistic rule–based system—and vice versa. Alternatively, it may turn out that there is some property which does distinguish these two kinds of systems, and that both are necessary for accounting for human generalization, but that the relevant property is orthogonal to productivity and reuse.

4.5 Conclusion

In this chapter, after reviewing the empirical literature on the past tense, I have discussed the three most important issues—from the perspective of this thesis—that have arisen in theoretical discussions of the topic. First, any theory of the past tense must make available abstract generalizations to account for the regular rule. Second, any theory of the past tense must provide a quantitative account of competition between the many generalizations which are consistent with the data—one recurrent theme is the use of various forms of the elsewhere condition. Third, I have argued that the residual issue of how to handle less productive generalizations, and the related question of how many kinds of mechanism are needed, is still open.
Chapter 5

The English Past Tense: Simulations

5.1 Introduction

The last chapter argued that an adequate theoretical account of the English past tense must provide both the capability of representing abstract generalizations and a quantitative account of how such generalizations compete. The framework presented in this thesis automatically provides mechanisms for abstract generalization by building on tools from the theory of computation and classical cognitive science. This chapter examines the further question of whether this framework can provide an account of competition between generalizations for the English past tense.

The model proposed in this thesis, Fragment Grammars (FG), treats productivity and reuse as an inference. In Sections 1.4.4 and 2.4, I described four other approaches to productivity and reuse (i) full-parsing, implemented with Multinomial-Dirichlet PCFGs (MDPCFG), (ii) full-listing, implemented with Adaptor Grammars (AG), (iii) a version of the exemplar-based approach, implemented with Data-oriented Parsing 1 (DOP1), and (iv) a second version of exemplar-based approach, implemented with Data-oriented Parsing: Goodman Estimator (GDMM). This chapter focuses on evaluating these five models against prior empirical studies of the English past tense system.

To provide an adequate account of competition between inflectional classes for the English past tense, a model must explain several phenomena. First, a model must treat the regular rule as a default—applying in cases where no other inflectional process is available. Second, it must exhibit blocking of the regular rule by the irregular forms. Third, it must be able to capture these effects quantitatively, explaining why blocking sometimes fails during language acquisition (i.e., overregularization). Fourth, it must explain how the frequency distribution of forms in the input gives rise to this pattern of probabilistic defaultness and blocking.

In the following sections, I will first give a general overview of the modeling assumptions used for the simulations of the English past tense system, and will then evaluate the five models on a corpus of adult speech. These evaluations will show that only Fragment Grammars capture the correct pattern of defaultness and blocking and, furthermore, that the way in which they use frequency is consistent with empirical evidence from processing experiments. Finally, I will present a developmental evaluation of the five models, and show that only Fragment Grammars predict overregularization phenomena which are consistent with findings in the literature.
Chapter 5: The English Past Tense: Simulations

5.2 The Simulations

The five models studied in this chapter used identical training sets and made identical assumptions about the starting state of the system. In this section, I describe and justify these assumptions.

5.2.1 The Starting CFG and Input Representation

Each of the five models explored in this thesis start from an underlying CFG which provides the space of all potential structures which can be generated. In this section, I describe the CFG used for the past tense simulation results reported below. The modeling decisions behind the CFG, and how they correspond to assumptions about learning, can be summarized by the following points.

1. The systems are provided \textit{a priori} with stems for all verbs.

2. The systems are provided \textit{a priori} with an inventory of all possible inflections in the English verbal paradigm, such as $+/d/$ and $/i/ \rightarrow /æ/$.

3. The systems are provided \textit{a priori} with knowledge of the correspondence between paradigm cells and each marking process; that is, the underlying grammar specifies, for example, that $+/d/$ can either mark past tense (VBD) or past participle (VBN).

4. The systems are provided with inputs which specify the stem, the marking process, and which paradigm cell is being marked (i.e., the tense of the verb).

5. The systems are \textit{not} provided with information about how productive each process is or which stems go with which inflectional markers.

6. The systems are \textit{not} sensitive to any phonological or semantic conditions on the applicability of each marking process to the stem. For example, each model is free to hypothesize that $/i/ \rightarrow /æ/$ can apply to the stem \textit{WALK}, despite the fact that the application of such a rule to that stem is phonologically impossible.

\begin{verbatim}
V →→ Stem Inflection
Stem →→ EAT
Stem →→ GO
Stem →→ WALK
...
Inflection →→ +∅ VB
Inflection →→ +∅ VBP
Inflection →→ +/\i/ VBG
Inflection →→ +/z/ VBZ
Inflection →→ /\i/ → /æ/ ∧ +/z/ VBZ
Inflection →→ WENT-SUPPLEMENTATION VBD
Inflection →→ /i/ → /æ/ VBD
...
\end{verbatim}

Figure 5.1: Context–Free Grammar for Past Tense Simulations: A context–free grammar (CFG) for the English past tense simulations.
These assumptions lead to a CFG like the one in Figure 5.1. A full list of the rules used in the simulations is given in Appendix B. These rules were constructed by hand, starting from the lists of irregular inflectional classes given in Pinker and Prince (1988). Note that the rules in Figure 5.1 for each inflectional class are of the form \textit{Inflection} $\rightarrow$ \textit{phonological–form paradigm–cell}.\footnote{In fact, the rules used in the simulations divide the \textit{phonological–form} into two parts; the first part specifies the transformation applied to the stem (if any). The second part specifies the suffix attached to the stem (if any). However, all stem–transformations and suffixes are bound into single rules—the system cannot learn them independently.} As a result, each input training form used in these simulations contained terminal symbols identifying both the phonological transformation associated with the form and the tense of the form. Figure 5.2 shows a few possible derivation trees generable by this grammar.

![Figure 5.2: Example Trees for Past Tense CFG](image)

Figure 5.2: Example Trees for Past Tense CFG: This figure shows some examples of trees that can be generated by the grammar in Figure 5.1.

The simplifying assumptions made in modeling the past tense provide both more information and less information to the models than is available to the child learner. On one hand, all stems and inflections are available \textit{a priori}, and forms are explicitly labeled with their tense, unlike in real speech. On the other hand, the CFG allows any \textit{Inflection} to appear with any \textit{Stem}, even including phonologically impossible cases, as noted above. Another simplification is that under this grammar all forms are bi–morphemic, consisting of a stem and a past tense inflectional marker, and, therefore, the models can not represent verbs as unanalyzed wholes, or choose to not inflect a verb for tense.

Taken together, these assumptions mean that the starting state is massively \textit{overproductive}. Furthermore, the only kind of information available to constrain this productivity over the course of learning is the distribution of stem–inflection co–occurrences in the input. As discussed in Section 1.5.1, the decision to provide the full morphological structure of all input forms—including stem segmentations and the abstract identity of morphemes—was made in order to isolate the question of how the distribution of these units affects inferences about productivity. Also as noted in Section 1.5.1, the lack of constraining phonological and semantic selectional restrictions is \textit{adversarial} to the problem of correctly inducing the pattern of productivity in the past tense system. Because these additional sources of evidence are correlated with productivity (e.g., because less phonologically transparent rules tend to be less productive, and vice versa), including this information should only improve simulation performance (see also Section 8.2).

### 5.2.2 Training

The following sections describe the results of training the models with inputs derived from two corpora: the SWITCHBOARD corpus (Godfrey et al., 1992; Marcus et al., 1999) and CHILDES (MacWhinney, 2000). I describe data preparation common to both sets of simulations in this section.
Both corpora provided part–of–speech tagged utterances. From these utterances, all verbs were extracted along with their part of speech (i.e., VB, VBP, VBG, VBD, VBN). Each verb was stemmed based on its form and part of speech (e.g., sang:VBD → SING). A first pass of stemming was performed using the WordNet stemmer (provided in the Natural Language Toolkit Bird et al., 2009; Fellbaum, 1998), and then mistakes and missing stems were corrected by hand. The stemmed verbs were then paired with the appropriate inflectional marker (e.g., SING:VBD was paired with the rule /s/ → /æ/).

Verbs from the training samples, together with their frequencies, were input as trees of the following form:

$$(\text{START} (V (\text{Stem stem}) (\text{Inflection phonology tense}))).$$

Simulations were run as described in Chapter 3, Sections 3.3–3.5. See chapter 3 for a detailed mathematical description of the parameters for the five models, and Section 3.5.6 for description of the values used for each parameter for all simulations.

5.3 Adult Data: SWITCHBOARD Simulations

To provide an adequate account of competition between the inflectional classes of the English past tense, a model must explain several phenomena. First, a model must explain why the regular rule operates as a default—applying in cases where no other inflectional process is available. Second, it must explain blocking—the phenomenon whereby the availability of an irregular makes a regular form infelicitous. Third, it must provide an account of these two phenomenon which is quantitative. Both defaultness and blocking are robust for adult English speakers; however, they are not deterministic. There are cases where both a regular and an irregular form exist side–by–side (i.e., doublets). Furthermore, adults produce speech errors, such as overregularizations and (more rarely) overirregularizations, which shows that the processes implementing blocking must admit low probability exceptions (Bybee, 1985; Stermerger, 1983; Xu and Pinker, 1995). Similarly, while novel verb rating and elicitation studies consistently demonstrate robust generalization of the regular rule to novel verbs, they also find that irregular classes can sometimes be generalized (Albright and Hayes, 2003; Ambridge, 2010; Bybee and Moder, 1983; Bybee and Slobin, 1982; Prasada and Pinker, 1993). Importantly, the model should allow for such variability while simultaneously capturing the fact that cases of overregularization and overirregularization are marginal, and, thus, that the system is nearly deterministic. A model which admits variability at the price of failing to correctly generalize the regular rule, or at the price of failing to block the regular rule’s application to irregulars, is inadequate. An account of competition should also explain the relationship between defaultness, blocking, and the frequency of linguistic structures in the input. In particular, it should be able to explain the differential frequency effects for regulars and irregulars found in the literature on past tense processing (see Section 4.3.4).

I will show in this section that Fragment Grammars (FG) are the only model, of the five considered in the thesis, which can explain both defaultness and blocking and can do so in a way that explains frequency effects discussed in the experimental literature.

5.3.1 Data Preparation and Simulations

The SWITCHBOARD corpus (Godfrey et al., 1992) consists of a large number of telephone conversations between strangers. The Penn Treebank provides a version of this corpus annotated for part–of–speech and other information (Marcus et al., 1999). To create this corpus, pairs of strangers were chosen randomly, connected, and given a topic of conversation to discuss; thus, SWITCHBOARD provides a large sample of (reasonably) naturalistic adult speech. All verbs,
Chapter 5: The English Past Tense: Simulations

except forms of be, have and do, were extracted from the corpus, stemmed, and paired with their past tense inflectional class as described in Section 5.2.2. Table 5.1 summarizes the counts of forms in the corpus.

Table 5.1: Counts of Verbs: This table shows the token (left) and type (right) counts of verbs from the SWITCHBOARD corpus used in the simulations. There were 2117 stem types in total. Note that the one irregular VBZ form is says.

110 FG simulations were run on the training set, each with 10000, 15000, 20000, and 25000 inputs as described in Sections 3.3–3.5. These ran for between 20 and 700 sweeps through the whole corpus, depending on time available on the MIT CSAIL cluster. Results were averaged to estimate the MAP approximating PCFG as described in Section 3.5.2. The approximating PCFGs for the other models were computed directly from the input corpus as described in Section 3.3.

5.3.2 Simulation Results

In this section, I will first discuss the overall performance of the models, and then turn to a more detailed analysis of their behavior on regular and irregular past tense forms. Following this, I will consider how the FG model can account for frequency effect data from the literature. I will discuss these results, and then conclude by examining two empirical phenomena, anti–frequency effects and storage of regular doublets, which are not directly reflected in the SWITCHBOARD simulations, but which can also be explained by FG.

Overall Performance

The appropriate quantity for measuring the ability of each model to correctly inflect verb stems is the conditional probability of inflectional markers given the stem. This quantity, which measures the probability that a randomly chosen stem will be correctly inflected for tense by each model, was computed for VBD (past tense) and VBN (past participle) forms. Because results were similar for both sets of forms, all numbers and plots in the rest of the chapter collapse across the two sets. Three kinds of VBD and VBN test items were considered:

---

2Be, have, and do are often used as both auxiliary and main verbs, and, thus, are extremely frequent. I excluded them for the practical reason that, because of their high frequency, doing so significantly reduced the size of the training corpus, improving simulation run time. However, there is no theoretical reason why excluding these verbs should have any effect on the simulation results, and a large pilot study, in which they were included, exhibited exactly the same pattern of results that I report below.

3I excluded forms from these analyses for which there was no clear correct past marker, such as doublets, and forms for which there were errors in the training data that were not discovered prior to running the simulations (e.g.,
1. **Attested items**: The set of regular and irregular forms in the training data.

2. **Novel items with Known Stems**: A set of novel forms which were computed by finding those stems in the input training corpus which appeared in other tenses (i.e., VBP, VB, VBZ, VBG) but did not appear as either a past tense (VBD) or a past participle (VBN). The correct VBD or VBN form was identified by hand. There were approximately 1100 such forms.

3. **WUG tests**: A completely novel wug-stem. The regular rule was considered correct for this stem.

Table 5.2 shows the probability that each model correctly inflects stems from each of the categories above, broken down into regular and irregular subsets (where appropriate). Note that these numbers weight forms which are more frequent more highly than forms which are less frequent, and, therefore, place greater importance on correctly inflecting more common stems.

The first row shows performance over all categories of tests. The FG model performs best overall. The next three rows show performance on forms which appeared in the training set. Again, the FG model performs best on these forms. In the next two rows, the pattern changes. These two rows show the performance of each model on novel VBD and VBN forms with known stems. Note that the majority of such forms are regular. In these categories, the best performance is exhibited by the DOP1 model. However, FG comes a close second in each case. This pattern is repeated for the next line: completely novel wug stems. These results show that DOP1 has learned a robustly generalizable regular rule. However, DOP1 only correctly inflects 0.057% of attested irregulars, showing that this model is incorrectly applying this regular rule to irregular forms as well. The final line shows performance on novel irregulars with known stems. Because these forms are never attested in the training corpus, all of the models necessarily fail to provide correct inflection on these test items.

<table>
<thead>
<tr>
<th>Test Set</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall</td>
<td>0.999</td>
<td>0.227</td>
<td>0.993</td>
<td>0.875</td>
<td>0.380</td>
</tr>
<tr>
<td>Attested Overall</td>
<td>0.999</td>
<td>0.209</td>
<td>0.995</td>
<td>0.874</td>
<td>0.375</td>
</tr>
<tr>
<td>Attested Irregular</td>
<td>0.999</td>
<td>0.042</td>
<td>0.995</td>
<td>0.057</td>
<td>0.175</td>
</tr>
<tr>
<td>Attested Regular</td>
<td>0.999</td>
<td>0.488</td>
<td>0.995</td>
<td>0.998</td>
<td>0.662</td>
</tr>
<tr>
<td>Novel (Known Stems) Overall</td>
<td>0.907</td>
<td>0.454</td>
<td>0.804</td>
<td>0.937</td>
<td>0.488</td>
</tr>
<tr>
<td>Novel (Known Stems) Regular</td>
<td>0.931</td>
<td>0.472</td>
<td>0.824</td>
<td>0.983</td>
<td>0.510</td>
</tr>
<tr>
<td>Regular WUG</td>
<td>0.889</td>
<td>0.486</td>
<td>0.828</td>
<td>0.989</td>
<td>0.536</td>
</tr>
<tr>
<td>Novel (Known Stems) Irregular</td>
<td>0.010</td>
<td>0.025</td>
<td>0.012</td>
<td>0.002</td>
<td>0.026</td>
</tr>
</tbody>
</table>

Table 5.2: Performance of Models on Past Tense Dataset: This table shows the probability that forms from different test sets will be correctly inflected by each model. The best scoring model is highlighted in bold in each row.

Examining the pattern of results more generally, FG demonstrates the best performance with regard to the *defaultness* and blocking phenomena discussed above. It is able to correctly inflect irregular forms (i.e., block the regular rule) while correctly generalizing the regular to the various novel test cases. Although DOP1 also generalizes the regular rule robustly, it does so at the expense of correct performance on irregulars. Of the remaining models, MDPCFG and GDMN both perform relatively poorly on all tests. The performance of AG is more interesting. While it underperforms FG alternate British and American spellings of words such as focussed/focused, incorrect past forms, etc.).

4These numbers are approximations to the maximum *a posteriori* probabilities for each model.
on all categories, it exhibited the same overall pattern of performance. It is able to correctly handle
irregulars, while generalizing the regular stem.

The five models studied here differ in the kinds of structures that they store, and this
difference is what leads to the patterns of performance just described. FG (inference–based), DOP1
(exemplar–based), and GDMN (exemplar–based) all allow the storage and reuse of fragments of structure
of any size. AG (full–listing) can only store full trees; however, it can also choose to account for any
form compositionally, using the rules in the underlying PCFG and stored subderivations. MDPCFG
(full–parsing) can only use minimal–sized units (i.e., units like those in the underlying PCFG).

Expected Number of Constituents Each of the five models makes different choices about which
computations to store, and what parts of a derivation to compute on the fly. To understand the
pattern of results above, we must investigate these choices in more detail. One useful way of doing
this is to examine the expected number of constituents for each form under each model. Given a
particular form, each model defines a distribution over all of the derivations which could give rise to
that form. By counting the number of fragments used in each derivation, weighting this number by
the probability of the derivation, and, finally, summing over all derivations, the expected number of
constituents can be computed for the form. If a form is retrieved whole with high probability, then
this number will be low. Alternatively, if a form has a high probability of being built out of small
pieces, then the number will be high. If, instead, a model assigns equal probability to a number of
derivations, some of which contain many fragments, and some of which contain few, the number will
be intermediate. Each form in the training data has a tree structure like that shown below.

```
      START
     /   \
    V    \
   /|   |
Ste|  Inf
   o   l
```

Since a derivation fragment boundary can fall (in principle) at any of the non–leaf nodes
of this tree, the maximum number of possible constituents for a form is 4 (i.e., fragment boundaries
at the START, V, Stem, and Inflection nodes). Because a fragment must be chosen at the START
node, the minimum number of constituents is 1.

Figure 5.3 plots the expected number of constituents predicted by FG for the forms in the
test sets described above. The x–axis is the frequency (log scale) with which the form appeared in
the training set. Note that novel forms with existing stems and wug forms appear at frequency 0 on
the left–hand side of the plot. Irregular forms are plotted as open blue squares while regulars are
plotted as solid red circles (wug items are considered regular). The size of the point for each form
is scaled according to the probability that the form was correctly inflected by the model; smaller
points represent forms less likely to be inflected correctly.

First, consider the irregulars (open blue squares) in Figure 5.3. Correctly inflected irreg-
ulars, represented by larger blue squares, have 2 expected constituents. The FG model has learned
to represent these forms with trees which store their stems together with their inflections, like the
following.

```
  V
 /\   |
St/  Inf
   o   l
```

Note that, for these forms, the model is treating the initial portion of each training tree
Figure 5.3: **Expected Number of Constituents for FG**: The expected number of constituents for each test form. Irregulars are shown as open blue squares, and regulars are shown as solid red circles. Points are scaled such that forms which were less likely to be correctly inflected are represented by smaller points.

(i.e., \( \text{START} \rightarrow V \)) as composed, rather than as stored together with the material below the \( V \) node. This is most likely because this portion of the tree is shared identically across both regular and irregular training forms. The consequence of treating the topmost tree fragment as composed is that the expected number of constituents for correctly inflected irregulars is 2 rather than 1.

There are also a number of incorrectly inflected irregulars towards the lower end of the frequency spectrum which appear with 3 expected constituents. These forms are overregularized by the model, due to their low token frequency in the training set. Because the regular rule is represented by a fragment like that shown below, these overregularized forms consist of three constituents (corresponding to derivation fragment choices at the \( \text{START} \), \( V \), and \( \text{STEM} \)).

\[
\begin{array}{c}
\text{v} \\
\text{ Stem \ Inflection} \\
+/-/Y \text{ED}
\end{array}
\]

Turning to the regulars (filled red circles), Figure 5.3 shows that the majority of regulars, including novel (frequency 0) forms, also have 3 expected constituents. As was the case with the overregularized irregulars, this reflects choices at the \( \text{START} \), \( V \), and \( \text{STEM} \) nodes in each derivation. However, there are also a considerable number of regulars, especially towards the high frequency end of the spectrum, which have only 2 expected constituents and are, therefore, stored like irregulars.
I will return to the storage of high–frequency regulars in more detail later in the chapter.

Figure 5.4: Expected Number of Constituents for MDPCFG and AG: The expected number of constituents for each past tense form. Irregulars are shown as open blue squares, and regulars are shown as filled red circles. Points are scaled such that forms which were less likely to be correctly inflected are represented by smaller points.

Figure 5.4 shows the expected number of constituents for the MDPCFG (left) and AG (right) models. In Table 5.2, the MDPCFG (i.e., full–parsing) model performed poorly on all tests. This follows from the fact that this model parses all forms in full and therefore cannot learn the contingencies between stems. This is shown on the left–hand side of Figure 5.4—all forms have exactly 4 expected constituents under this model.

Because AG (i.e., full–listing) memorizes each form in the dataset, it performed well on attested items in Table 5.2. The right–hand side of Figure 5.4 demonstrates that most forms in the AG model are treated as wholes, and have an expected number of constituents near 1, especially more frequent forms. Note that unlike FG, AG does not treat the START → V portion of each tree as compositional. Recall that, in addition to storing the entire derivation for each input form, AG also recursively stores all (complete) subderivations of each form. In the case of infrequent forms, the probability associated with the complete stored tree is relatively low, and, therefore, analyses which make use of stored subderivations can contribute significant probability mass to the total score of a form. The left–hand side of the AG plot demonstrates this effect: Forms which appeared infrequently in the training set have an expected number of constituents greater than 1, although this number never reaches noticeably beyond 2. By contrast, forms which did not appear in the training set at all (i.e., frequency 0 forms) are at the upper limit of 4 expected constituents, because AG cannot reuse any stored subtrees for these forms.

AG’s treatment of some low frequency forms as partially composed, also explains the somewhat surprising fact that FG outperforms AG on attested items. Because AG deterministically stores all derivations and subderivations, it might be expected that it would be at ceiling for all attested items. However, while AG stores all (sub)derivations, it reuses stored items probabilistically, always reserving some probability mass for the possibility of composing trees using rules from its underlying PCFG. Figure 5.4 shows that, compared to FG, AG systematically overweight the possibility of
composition for some low frequency irregular forms, leading to overall lower performance on attested items.

As mentioned above, AG also performed well on both regular novel items with known stems and *wag*-tests. This is a result of the way in which AG estimates the weights of rules in its underlying PCFG. As I discussed in Section 2.4.2, each underlying rule of the AG model has a probability which is (roughly) proportional to the type frequency of the items in which it appears. Because the regular rule is the most type-frequent inflectional marker in the English past tense system, AG can generalize this rule correctly with reasonably high probability.

![DOP1 (exemplar-based) vs. GDMN (exemplar-based)](image)

**Figure 5.5: Expected Number of Constituents for DOP1 and GDMN:** The expected number of constituents for each past tense form. Irregulars are shown as open blue squares, and regulars are shown as filled red circles. Points are scaled such that forms which were less likely to be correctly inflected are represented by smaller points.

The performance of the two exemplar-based models is more complex. Figure 5.5 shows the expected numbers of constituents for each. Both models show significantly more variation in this quantity than the previous three models. This is a result of the fact that exemplar models do not commit to a single analysis of each form, but, rather, spread their probability mass over all generalizations consistent with the input. In both models, there is a tendency for irregulars to consist of fewer constituents, and regulars to consist of a larger number of constituents. However, this dichotomy is not nearly as sharp as with the FG model, which accounts for the inability of these models to simultaneously correctly classify both regulars and irregulars (as seen in Table 5.2).

As discussed in Section 2.4.3 (see also Section 3.1.5), the DOP1 estimator favors larger subtrees and therefore, on average, it produces lower expected numbers of constituents than GDMN. This fact explains the robust performance of DOP1 on the regular rule. Because it favors larger subtrees, it is able to place more probability mass on the frequent subtree corresponding to the regular rule. By contrast, GDMN places more mass on depth-one trees corresponding to rules in the underlying CFG, and, therefore, overestimates the probability that stems will combine productively with irregular inflectional classes.
Whole–form Frequencies and Stem Frequencies

In Section 4.3.4, I discussed the differential frequency effects for regulars and irregulars that arise in production and naturalness judgment tasks. In production tasks, high frequency irregulars are named faster than low frequency irregulars, while there is no such effect for regulars (or sometimes there is an anti-frequency effect, see Section 5.3.3, below. Beck, 1997; Prasada et al., 1990; Seidenberg and Bruck, 1990; Shenkman, 1994). In a set of naturalness judgment tasks, Ullman (1993) and Ullman (1999) found that, when stem judgments are partialled out, participants' judgments of irregulars are predicted by their whole–form frequencies, while their judgments of regulars are not. These differences in frequency effects are taken as evidence that irregular forms are stored independently and can therefore accumulate strength in memory, while regular forms are composed.

In this section, I examine the relationship between stem and whole–form frequencies and the probabilities assigned to individual forms by each model. The specific quantity on which I focus is the marginal likelihood. The marginal likelihood of a form is the total probability assigned to that form by a model—the sum of the probabilities of all possible derivations. It can be understood intuitively as a summary of each model’s expectation that a form will appear again in the future.

Table 5.3 shows the results of a partial correlation analysis similar to that presented in Ullman (1999). The numbers in the table represent the correlations between marginal likelihood for each form and its frequency in the training set, with the stem frequency partialled out of both values. Note that the production and judgment experiments cited above were run with existing English words on adult, English–speaking participants, who (presumably) knew the correct past tense forms for each word. However, the sets of test items discussed in the previous section included items for which each of the models failed to correctly learn the past tense inflection. In order to compare the models with the adult data, the analyses in Table 5.3 are restricted to past forms which were correctly inflected 85% of the time. Out of 3961 potential existing (attested and novel known–stem) English past forms from in the preceding section, the following numbers of forms were excluded for failing to meet this criterion: FG: 143, MDPCFG: 3961, AG: 1348, DOP1: 229, GDMN: 3640. Note that in some cases this left no forms for the partial correlation analysis, and, therefore, that the corresponding table entries are marked “NA.”

<table>
<thead>
<tr>
<th>Forms</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Irregulars</td>
<td>0.999</td>
<td>NA</td>
<td>0.988</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>Regulars</td>
<td>0.499</td>
<td>NA</td>
<td>0.831</td>
<td>0.650</td>
<td>0.551</td>
</tr>
</tbody>
</table>

Table 5.3: Partialed Correlation of Marginal Likelihood and Whole–form Frequency Controlling for Stem Frequency

Of the five models, only FG and AG correctly inflected a sufficient number of both regulars and irregulars to perform the analysis for both sets of forms. For AG, both regulars and irregulars show a strong correlation with whole–form frequency when stem frequency is controlled. By contrast, for FG, there is a large difference between the correlations for these two subsets of forms. Irregulars show an almost perfect correlation with their stem frequencies, while irregulars show a much lower correlation. The FG result is more consistent with the results in Ullman (1993) and Ullman (1999).

To understand the pattern of results in Table 5.3, consider the way that the FG model accounts for regulars and irregulars. Most regulars are composed: The probability of each such compositional derivation is the product of the probabilities of its parts; in particular, it is the product of the probability of the stem and the regular past tense ending. These probabilities, in turn, are proportional to the frequencies of the stem and regular past ending in the training data. ⁵ Across

⁵This is true, on average, across high probability posterior states.
composed regulars, the past tense ending is shared, and therefore has constant probability. Thus, any variance in the probabilities of composed regulars will be driven primarily by stem frequency, which is partialed out in these analyses. By contrast, irregulars are stored as wholes by the FG model. Their probabilities are (roughly) proportional to their whole form frequencies in the input. The AG model stores all forms as wholes, and thus shows a strong correlation between marginal likelihood and whole–form frequency for both regulars and irregulars. The results in Table 5.3 are plotted for the FG and AG models in Figure 5.6.

![Figure 5.6: Effect of Full–Form Frequency on Marginal Likelihood for FG](image)

The y–axis in Figure 5.6 plots the residuals of the marginal likelihoods of each form after partialed out stem frequency. The x–axis plots the residual whole–form frequencies after partialed out stem frequency. Irregulars are plotted with open blue squares and regulars are plotted with filled red circles. The differences between FG and AG emerge clearly in these plots. For FG, irregulars fall along an almost perfect line with positive slope, while regulars seem to fall into two large groups: Most low and medium frequency regulars show little (partialed) correlation with whole–form frequency. A smaller number of high–frequency regulars towards the upper right of the FG plot seem to show a somewhat stronger positive relationship (see below). By contrast AG, shows a strong positive relationship between (partialed) marginal likelihood and (partialed) whole–form frequency for all forms.

Turning to the regulars, in Section 5.3.2, I showed that the FG model stores some high–frequency regular forms as wholes (see Figure 5.3). Experimental evidence for the storage of high–frequency, regular forms has been reported by Alegre and Gordon (1999a) and Gordon and Alegre (1999), who showed, using a lexical decision paradigm, that these forms exhibit a frequency effect which is not exhibited by lower frequency items. Under the FG model, forms which are stored whole should exhibit a whole–form frequency effect. This is born out in Figure 5.6, where the cluster of regular forms in top right of the FG plot appears to show a stronger positive relationship with (partialed) whole–form frequency than the larger group of regulars in the center of the figure. This
effect is confirmed by dividing the regulars into stored and composed subsets (based on expected number of constituents) and correlating each with partialed frequency. Stored regulars correlate more highly with partialed frequency ($r = .466$) than composed regulars ($r = 0.336$).

Of the five models, only FG produces a pattern of results which exhibits the same relationships as the experimental results between regulars and irregulars, and between stored and composed regulars. What are the psychological implications of these results? The measure used as a proxy for experimental data in these analyses was marginal likelihood. This quantity can be understood as a summary of each model’s expectations that a form will be used again in the future (either as a whole or as a combination of reusable parts). Moreover, because of the assumptions made in modeling the domain, each model’s estimation of marginal likelihoods depended on just the distributions of stem and inflection morphemes in the input data. Of course, experimental participants’ actual expectations about reuse will take into account many other factors: semantic structure, phonological structure, pragmatics, style, memory limitations, attention, priming, etc. However, by isolating just the distributional factors, these analyses provide strong evidence that people’s frequency–based inferences about productivity use the same distributional cues as the FG model.

Discussion

In the preceding two sections, I showed that the FG model was able to provide the most accurate performance on the adult SWITCHBOARD data of the five models considered. It was able to achieve this performance by storing derivations which combined stem and inflection for irregulars (and high–frequency regulars), while composing the majority of regular forms. This section ties this behavior back to the intuitive discussions in Chapter 1.

Section 1.2.1 discussed how the FG model can be understood as balancing a tradeoff between productivity and reuse. The storage of a partial derivation fragment which includes a variable is a prediction of future novelty, while the storage of a specific combination of (sub)computations is a prediction that this combination will be reused as a unit. One intuition which is helpful in understanding how FG manages this tradeoff is the idea that it compares expected and observed frequencies.

The probability, $p$, that FG assigns to a particular past–tense form, $f$, can be understood as a prediction that in a sample of size $N$, there will be approximately $pN$ expected observations of that form. When the observed frequency of $f$, $n_f$, exceeds the expected frequency, $pN$, FG will adjust the set of stored fragments to increase $p$. Conversely, when the observed frequency, $n_f$, is less than the expected frequency, $pN$, FG will adjust the set of stored forms to decrease $p$. By storing large derivations, $p$ can be increased for forms which use these large derivations. By contrast, by breaking stored derivations into smaller parts, $p$ can be reduced for the corresponding forms.

The probability of composed forms will be given by the probability of the stem multiplied by the probability of the inflection. These, in turn, will be proportional to the frequency of the stem and the frequency of the inflection across different composed verb forms. For stored forms, the probability of the entire structure will be proportional to its frequency in the training set.

Irregulars tend to be stored whole by the model because their frequency exceeds what would be expected by the product of their parts, if they were composed. By contrast, many regular stems are relatively infrequent, occurring only a few times in the corpus, therefore their observed frequency is well–approximated by the product of the probabilities of these stems and the regular affix.

\footnote{For example, it has been observed that regular past forms tend to be phonotactically less expected than irregular past forms (Burzio, 2000; Pinker and Prince, 1988). This may lead to lower rates of storage for regular forms (see, Hay, 2003), and, therefore, lower correlations between the probability of these forms and their whole–form frequencies. As another example, in naturalness judgment experiments, speakers are likely to base their judgments partially on the plausibility of each form in its frame sentence. This will depend on semantic, pragmatic, and stylistic factors.}
The case of high-frequency regulars is more interesting. Each of these regulars can be derived compositionally with reasonably high probability. However, for some of these forms, the observed frequency still exceeds the expected frequency under the compositional route. To account for this discrepancy, the model stores additional copies of these forms as wholes in memory. Importantly, the model only needs to account for the difference between the observed frequency and the expected frequency under the compositional route. As a result, more evidence will be required, in general, for the storage of regular forms than for the storage of irregular forms. Furthermore, because the model must only account for the difference in observed and expected frequencies, some token instances of each high-frequency regular will be assigned whole-form derivations, and some instances of each will be composed, leading to competition between the two possibilities (see Section 5.3.3).

The conditions under which FG chooses to store a regular form explain an additional empirical effect reported by Alegre and Gordon (1999a). Alegre and Gordon showed that storage of high-frequency regulars is modulated by the overall stem-frequency of the form. Regular forms transparently share their stems with other cells in the English verbal paradigm, and there is a large amount of variability between verbs in the total frequency of the stem across all paradigm cells. Alegre and Gordon (1999a) performed an analysis of their reaction time data where they compared subsets of high-frequency regulars with high and low cross-paradigm stem counts. They found that high-frequency regulars with high cross-paradigm stem counts showed less of a frequency effect than high-frequency regulars with low cross-paradigm stem counts. In other words, when a stem is shared robustly across many forms of a verb, individual forms of the verb are harder to store. This can be explained by again considering expected and observed frequencies. As the frequency of the stem increases so does the expected frequency of the composed form. Therefore, for the same whole form frequency, an increase in the stem frequency should reduce the rate of storage, which is exactly the reported effect.

More broadly, the tradeoff between productivity and reuse in the model cannot be described in terms of any absolute whole-form or stem frequencies. As I will discuss in Section 6.3.4, a more accurate characterization of the conditions under which the model chooses to store or compose a form can be given in terms of the relative frequencies of the form and its parts. However, this too is only an approximation. In general, the decision to store or compose in the FG model depends on the complex set of tradeoffs which are captured by the generative model.8

7This is Jennifer Hay’s relative frequency hypothesis. See Section 6.3.4 for details.

8This is related to an important debate in the past tense literature. As I reviewed in the last chapter, connectionist models rely on variability across regular forms to acquire a general regular rule (see Section 4.4.1). In English, the productive default (i.e., the regular rule) is also the most type-frequent past inflection, allowing these models to abstract (to some degree) over the particular details of the phonology of regular stems. Type frequency has also been proposed as a predictor of productivity by other theorists (e.g., Bybee, 1995b).

However, Marcus et al. (1995) discuss the case of the German plural, where the default rule appears to be the least frequent allomorph in both type and token frequencies, and, therefore, challenges theories in which type frequency is the main determinant of productivity. A number of other similar cases such as the Arabic sound plural and Polish inflectional marking have also been considered in the literature (Dąbrowska, 2004; Forrester and Plunkett, 1994; Plunkett and Nakisa, 1997).

An important aspect of Fragment Grammars is that, in general, they do not predict that the most productive processes should be the most type frequent. It is not type frequency per se which matters, but the tradeoff between the number of novel forms using a particular word-formation process (a quantity measured by type-frequency) and the distribution of repetitions of individual words using the process. Because of this tradeoff, when comparing word-formation processes for sets of words that have similar token-frequency spectra, type frequency can often serve as a reasonable approximate measure of productivity. However, there are many cases where it leads to inaccuracies (see Section 6.3.3).
5.3.3 Other Analyses

In this section, I consider two additional phenomena from the past–tense literature: the storage of regular doublets and anti–frequency effects. I present these analyses in a separate section, because, while they relate the behavior of the FG model to proposals from the literature, they do not directly involve evaluation of simulations, and are, therefore, more speculative.

Doublets

A closely–related phenomenon to the storage of high-frequency regulars is the phenomenon of storage of regular doublets. A doublet is a verb which appears in the past tense in both irregular and regular forms. For example, for many native speakers of English, both dreamed and dreamt are grammatical as past tenses of the stem dream.

Ullman (1993) provides evidence from a naturalness rating task that both the regular and irregular past–tense forms of doublets correlate in naturalness with their whole–form frequencies (when stem judgments are partialed out), suggesting that both forms are stored in memory. While it is unsurprising that irregular doublet forms (e.g., dreamt) are stored in memory, the storage of regular doublets (e.g., dreamed) is somewhat unexpected. In general, these forms are not extremely frequent, which rules out an explanation that they are stored for the same reasons as other high–frequency regular forms (Pinker, 1999; Ullman, 1993).

An alternative account is offered by Pinker (1999) who argues that storage of regular doublets is the result of the need to overcome the blocking caused by irregular forms. Intuitively, since dreamt must be stored, dreamed should be blocked (under a theory like the blocking and retrieval failure hypothesis, see Section 4.4.2). However, dreamed appears in the input, despite being blocked by dreamt. The only mechanism available to the learner to account for its appearance is to store dreamed in memory.

Doublets are relatively rare, and the sample from the SWITCHBOARD corpus only contained a single example (i.e., proved/proven), which was insufficiently frequent for the model to correctly learn both the regular and irregular forms. Therefore, to examine how FG treats doublets, I conducted simulations on a series of artificial datasets generated from a vocabulary of 1000 stems. All (but one) training stems appeared with a single suffix (corresponding to the regular rule). One training stem appeared with both the regular suffix, and a (distinct) irregular suffix, representing an (idealized) doublet. Stems were assigned ranks, and training data sets (of size 2001) were constructed by sampling from a power–law distribution over ranks. The percentage of the doublet forms that appeared as regulars or irregulars was varied between 0 and 1 across training sets, and FG simulations were run at each percentage.

Figure 5.7 shows the breakdown of storage and composition for input forms in these simulations. The x–axis represents the proportion of doublet instances that appeared as regulars. At the far left–hand side of the plot, the doublet did not appear as a regular, and, therefore, is best understood as simply a (high–frequency) irregular. At the far right–hand side of the plot, the doublet did not appear as an irregular, and, therefore, is best understood as simply a high–frequency regular form. In between, the doublet appeared in intermediate mixtures of regular and irregular instances.

The FG model is predicted to store all irregular instances of the doublet. Regular instances of the doublet, however, can be split between whole–form storage and composition. The y–axis shows the percentage of doublet instances for each of these three possibilities (i.e., stored irregulars, composed regulars, stored regulars). Irregular instances of the doublet appear as open blue squares.

9For each doublet proportion, 20 random training datasets were generated, the sampler was run for 200 sweeps, and the results were averaged.
Figure 5.7: **Doublets:** The $y$–axis of this figure shows the percentage of doublet forms that were treated as (i) composed regulars (filled red circles), (ii) stored regulars (open red squares), and (iii) stored irregulars (open blue squares). The $x$–axis is the percent of doublet forms which were regular in the training input.
As expected, for each training set, all irregulars were stored, and, thus, the open blue squares form a line with slope of $-1$.

Regular instances of the doublet are shown in red. Filled red circles represent the percentage of regular instances which were composed. Open red squares show the percentage of doublet forms which were stored as wholes in memory.

When the percentage of regular doublets in the input is at 10%, or less, all regular doublet forms are composed. As the percentage of regular doublet forms increases, there is a small increase in whole-form storage for the doublet, but composition also increases. Interestingly, however, the percentage of the regular doublet forms which are composed plateaus at 20%. Beyond this proportion, all additional regular forms are stored whole (with the exception of the case where the doublet is 100% regular).\(^{10}\)

This behavior can be understood in terms of the comparison of observed and expected frequencies that was discussed in Section 5.3.2. The expected frequency of a composed, regular doublet form is proportional to the number of stem instances appearing in composed forms multiplied by the frequency of the regular rule. In these simulations, this expected frequency is approximately 20% of the doublet forms which appear in the input. As was the case with high-frequency regulars, the model must account for additional instances of the regular forms by storing them whole. This can be seen clearly in Figure 5.7.

However, there is an important difference between the case of doublets and the case of high-frequency regulars. Because irregular doublet forms are always stored, their stems cannot be shared, reducing the probability that the stems combine with the regular rule. This, in turn, reduces the expected number of regular forms associated with the stem, making storage of any regulars which are observed and do use the stem more likely. Thus, competition with irregular doublet forms causes FG to store regular doublets at a higher rate than other regular forms.

In summary, FG predicts that regular doublets should be stored as wholes at a lower frequency threshold than regulars in general, due to competition with the corresponding irregular forms. This competition with the irregular form of each verb can be understood as a probabilistic version of blocking. The existence of irregular forms leads the model to predict that the corresponding regular should not appear in the training set. Therefore, when regular forms do appear in the training set, despite this prediction, they must be stored whole. The explanation of storage of regular doublets provided by Pinker (1999) works in a similar way: Because of blocking, the regular doublet forms are unexpected, and must be stored. Therefore, FG can be seen as formalizing a probabilistic version of this account.

**Anti–Frequency Effects**

In production experiments where participants are asked to produce a past tense form from its stem, regular forms are often produced faster than irregulars (e.g., Beck, 1997; Prasada et al., 1990; Seidenberg and Bruck, 1990; Shenkman, 1994).\(^{11}\) Furthermore, regular forms sometimes exhibit an anti-frequency effect in which high-frequency regulars are produced more slowly than lower frequency regulars (e.g., Beck, 1997; Prasada et al., 1990).

\(^{10}\)When the doublet is 100% regular, the entire training set is regular. Under this condition, the regular rule is exceptionless and can account for a higher percentage of the input forms than when there is an irregular competitor.

\(^{11}\)While these elicited naming experiments show an advantage for regulars, this is not generally the case for all latency-based methods. For instance, in the primed lexical decision experiments reported in Stanners et al. (1979a), regular past–tense primes are associated with longer latencies than irregulars. This suggests that the speed advantage for regulars may be task–specific.
Pinker (1999, p. 303) discusses an account of anti-frequency effects which is based on competition between stored and composed regular forms (see, also, Balota et al, 2000). Because high–frequency regular forms are sometimes stored whole in memory, and stored forms block the output of the regular rule, competition is expected between stored and composed high–frequency regular forms. This competition, in turn, slows the processing these forms.

As discussed in Section 5.3.2, the storage of high–frequency regulars by the FG model can lead to ambiguity for these forms. Since the model assigns a reasonably high probability to both composition and retrieval for these forms, such forms exhibit competition which is not present for irregulars or low–frequency regulars. This competition in the FG model can be quantified by examining the entropy of the distribution over derivations for each form. Entropy can be understood as a measure of the uncertainty in a distribution—if a form is sampled from a distribution, how surprising is any single outcome on average? Figure 5.8 plots the entropy for items from the SWITCHBOARD training set as a function of their whole–form frequency.

![Figure 5.8: Entropy of Regular Forms by Frequency in FG Model](image)

This figure shows the entropy (uncertainty) over derivations for regular and irregulars forms as a function of their whole–form frequency. Irregulars are shown as blue squares, and regulars are shown as red circles.

In Figure 5.8, both irregulars and low–frequency regulars have low entropy. The model is very confident that they should be retrieved whole from memory, and composed, respectively. However, as the frequency of regulars increases, so does their entropy. Entropy is highest for those regulars in the middle–high frequency range. For these forms the whole–form retrieval route is available, but is not yet strong enough to overwhelm competition with a compositional route.

In the absence of a processing model which integrates priming and other factors, it is impossible to say precisely how the competition revealed in Figure 5.8 will affect processing latencies; however, the results are broadly consistent with the theory proposed by Pinker (1999). One aspect of the phenomena in Figure 5.8, which goes beyond the existing experimental literature, is that it predicts that the effect of competition will be greatest for regulars in the middle–high fre-
frequency range, but be reduced for the highest–frequency forms. Previous studies (e.g., Beck, 1997) have only reported mean production latencies for two groups of verb forms: low–frequency regulars (whose frequency falls in the low–frequency range in Figure 5.8), and high–frequency regulars (whose frequencies fall in both the middle–high and high frequency ranges in figure 5.8). Based on these previous studies, it is impossible to say whether the anti–frequency effect is attenuated for the highest–frequency regular forms. Testing this prediction of FG is left to future work.

5.4 Developmental Data: CHILDES Simulations

In this section of the chapter, I study the developmental predictions of the five models. To do this, I will examine the behavior of each model as it is (independently) trained on a sequence of naturalistic samples of English verbs, of increasing size. Each step in this sequence can be understood as an idealization of the increasing amount of data that children are exposed to as they grow older. The simulations, therefore, examine how predictions about productivity and reuse change for each model as it exposed to more information about the English verbal system.

The most important empirical phenomenon in past tense development is the overregularization of irregular verbs. In Section 4.3.5, I discussed five important empirical properties of overregularization. First, all English speaking children overregularize at some point during development. Second, the overall rate of overregularization is relatively low. Third, irregulars are protected from overregularization by their frequency and their similarity to other irregulars. Fourth, at least for some children and some verbs, there is a period early correct performance preceding the onset of overregularization. Fifth, the onset of overregularization is correlated with an increase in the rate of correct, obligatory marking of regular forms.

As I will discuss below, some of these phenomena are beyond the scope of the present study because they involve unmodeled kinds of structure (e.g., phonology) or developmental change in the space of possible representations (e.g., the discovery of the regular rule)—rather than change in the relative productivity of existing representations. However, I will show that for those phenomena which concern competition between productive and unproductive past tense forms, and are, therefore, within the scope of these simulations, only the FG model is able to produce an account which is consistent with the empirical facts.

5.4.1 Data Preparation and Simulations

The following simulations were performed on data extracted from the CHILDES corpus (MacWhinney, 2000). Data from the following corpora were used: Bloom (1970, 1973), Brown (1973), Clark (1978), Demetras (1989a,b), Gleason (1980), Higginson (1985), Kuczaj (1977), Nelson (1989), Demuth et al. (2006), Sachs (1983), Suppes (1974), MacWhinney (2000), and Weist et al. (2009). Files which were annotated for part–of–speech information (i.e., were marked with MOR tags, MacWhinney, 2008) and were marked for target–child age were chosen from these corpora and processed further.

To simulate the input available to a child, all relevant adult speech was used. All verbs excluding forms of be, have and do, were extracted from the corpus. MOR tags for verbal categories were mapped into their corresponding Penn TreeBank verbal categories (i.e., VBD,VBN,VB,VBP,VBZ).  

12In particular, any utterances tagged with the following roles were excluded: Target_Child, Child, Playmate, Non_Human, Environment, and Camera_Operator. All other utterances were included.

13This mapping was complex: It involved taking into account MOR categories (i.e., v, aux and part) and distinguishing fusional markings and suffixation.
Figure 5.9: **Total Cumulative Regular and Irregular Tokens and Types:** This figure shows the cumulative total number of past tense tokens (left) and types (right) for each training increment used in the CHILDES simulations.

The verbs were lemmatized and paired with the appropriate past tense inflection as described in Section 5.2.2.

Verb tokens were divided into sets according to the age of the target child in the file in which they were found, and a corresponding empirical relative frequency distribution was constructed for each month from 1;6 to 5;0. The result was an empirical estimate of the relative probabilities of each verb form in the speech of adults speaking to (or around) children of that age. These relative frequency distributions were then used to construct a series of 43 training sets, each with approximately 1000 tokens, by multiplying the relative frequency of each verb by the total number of tokens. The input to each model at each time step consisted of the cumulative verb tokens in all of the training sets up to, and including, the target age. Figure 5.9 shows the cumulative types and tokens in the input datasets for each age in months.

Figure 5.10 shows the breakdown in cumulative percentages of tokens and types for regulars (filled red circles) and irregulars (open blue squares). What can be seen is that, aside from a small amount of variability in the initial few samples—likely due to the small sample sizes associated with these time steps—there are no sudden discontinuities in the proportion of regulars in either types or tokens. Rather, the input data is characterized by a gradual overall increase in the proportion of regular types, and an overall greater proportion of irregular tokens at most ages.

6 FG simulations were run, each with 10000, 15000, 20000, and 25000 inputs, for each training set (i.e., for each age) as described in Sections 3.3–3.5. These ran for between 20 and 700 sweeps through the whole corpus, depending on time available on the MIT CSAIL cluster. Results were averaged to estimate the MAP approximating PCFG as described in Section 3.5.2. Approximating PCFGs for the other models were computed directly from the input corpora as described in Section 3.3.
Figure 5.10: Percent Cumulative Regular and Irregular Tokens and Types: This figure shows the cumulative proportion of irregular (open blue squares) and regular (filled red circles) tokens (left) and types (right) for each training increment used in the CHILDES simulations.

5.4.2 Results

In this section, I first consider the performance of each model in terms of the trajectory of overregularization. I will show that only FG produces a trajectory which is consistent with empirical findings. I then show that the FG model also predicts the importance of frequency in protecting irregular verbs from overregularization. Next, I consider two other important empirical phenomena: early correct performance and the correlation between reliable obligatory regular marking and overregularization. I will argue that, in both cases, these phenomena involve representational capacities or representational changes which cannot be captured by the present models, although they are not inconsistent with the results presented here. Finally, I discuss the findings.

Trajectory of Overregularization Rates

Figure 5.11 plots the rates of overregularization produced by each model at each developmental increment. The upper left-hand plot in this figure shows 1 minus the average total overregularization rate for Adam, Sarah, Eve, and Abe as reported in the appendices of Marcus et al. (1992). The equivalent quantities for each model are displayed (in blue) in the remaining plots in the figure. Each plot also includes the Marcus et al. (1992) data in light grey for comparison.

As I discussed in Section 4.3.5, two empirical facts about overregularization for which there is a broad consensus in the literature are: (i) All children exhibit some overregularization for some verbs, and (ii) the overall rate is relatively low (from 2.5% to around 10% as calculated by various

---

14 This plot was produced by taking the total numbers of overregularizations across all all four children at each age in months and dividing by the total number of past tense forms produced at that age.
authors. Hoeffner, 1996; Maratsos, 2000; Marcus et al., 1992; Yang, 2002). Examination of the plots in Figure 5.11, shows that only FG is in accord with these two empirical observations. The MDPCFG (full-parsing), DOP1 (exemplar-based), and GDMN (exemplar-based) models produce overregularization rates which are far in excess of empirical findings. By contrast, AG (full-listing) produces virtually no overregularization at any point during learning, simply memorizing all irregulars perfectly.

These results show that under an inference-based approach to productivity and reuse, there is sufficient evidence in the data for the correct inflection of irregular forms in the majority of instances. However, there is also sufficient ambiguity about the appropriate inflection for there to be occasional overregularization throughout learning.

The results reported in Section 5.3.2 show that with time and evidence, FG will eventually correctly learn the pattern of productivity and reuse in the English verbal system. A trend in this direction is apparent in Figure 5.11: Rates of overregularization for FG appear to be declining with time. However, this decline is gradual and chaotic, suggesting that, in these datasets, there is still significant ambiguity about the applicability of the regular rule to some irregular stems, even at relatively late time steps.

As noted in Section 4.3.5, Marcus et al. (1992) found considerable variability in overregularization rates between verbs in their samples, and this result has been supported by other studies (e.g., Hoeffner, 1996). The FG model produces similar behavior, showing a wide variety of different trajectories for different verbs. Figure 5.12 shows trajectories for a number of examples. These are informally sorted into four rough classes: verbs that show overregularization followed by recovery (Row I), verbs that show little overregularization (Row II), verbs that show high rates of overregularization throughout (Row III), and verbs that behave chaotically (Row IV).

### Protection from Overregularization

Marcus et al. (1992) show that individual verbs are protected from overregularization by two factors. First, higher frequency irregulars are overregularized at a lower rate than lower frequency irregulars. Because FG stores irregular forms based on the high frequency of co-occurrence of their stems and inflections, high-frequency irregulars should be less likely to overregularize. This is confirmed by a Pearson correlation of the log rate of overregularization with the log frequency of each past tense form in the input ($r = -0.90$, $p < .001$).

Second, Marcus et al. (1992) report that irregulars are protected from overregularization by similarity to other irregulars. Specifically, irregulars which come from inflectional classes which are attested more frequently in the input are overregularized at lower rates. Because the models studied in this chapter do not represent the phonological selectional restrictions associated with irregular inflection, or the internal phonological structure of stems and inflections, it is not possible to test the model for this empirical phenomenon, and I leave it to future extensions of the framework to explore this prediction.

---

15 And sufficient simulation run time. The results reported here involve 43 different training-data sets, and therefore each point involves fewer simulations, run for less time, than the SWITCHBOARD results.

16 Marcus et al. (1992) report that there is no significant trend towards recovery from overregularization in the samples used in their study. It is possible that the gradual and chaotic recovery of FG is therefore correctly capturing some aspect of child language. However, there are many factors which affect the performance of the model—the most important for present purposes are the size of the training sets, and simulation time. Therefore, the gradual recovery of the model should be interpreted with caution.

17 The verbs in the figure were classified into these four classes by hand inspection of plots for all irregulars in the CHILDES training set. The four classes are the same as used by Marcus et al. (1992) when presenting examples of Abe’s overregularization trajectory on individual verbs. In three of the cases—ate, said, and won—the example is the same as the one given in Marcus et al. (1992) for the class.
Marcus et al. (1992)

FG (inference–based)

MDPCFG (full–parsing)

AQ (full–listing)

DOP1 (exemplar–based)

GDMN (exemplar–based)

Figure 5.11: 1 minus Overregularization Rate: The upper left plot shows one minus the overregularization rate for irregulars (blue squares) from Abe, Sarah, Adam, and Eve from the appendices of Marcus et al. (1992). The remainder of the plots show the same quantity (open blue squares) for each of the five models. These plots also include the (Marcus et al., 1992) data in light grey for comparison.
Chapter 5: The English Past Tense: Simulations

Figure 5.12: Examples of Trajectories for Individual Verbs: This figure shows examples of the trajectory of overregularization errors for individual verbs. These are informally sorted into four rough classes: verbs that show overregularization followed by recovery (row I), verbs which show low rates of overregularization throughout (row II), verbs which show high rates of overregularization throughout (row III), and verbs which behave chaotically (row IV).
Early Correct Performance and the Discovery of the Regular Rule

In Section 4.3.5, I described two controversial empirical phenomena described by Marcus et al. (1992): the existence of a global (i.e., verb–general) period of early correct performance, and a correlation between the onset of regularization and an increase in the rate of reliable marking of regular forms.

Examination of the results for FG in Figure 5.11 reveals no evidence for a clear global period of early correct performance. However, this is not surprising. Under all rule–based and dual–mechanism accounts (e.g., Pinker and Prince, 1988; Yang, 2002), as well as some connectionist accounts (e.g., MacWhinney and Leinbach, 1991), early correct performance is driven, at least in part, by the memorization of irregular past–tense forms as unanalyzed wholes. All theories of language learning need to provide mechanisms for the acquisition of monomorphemic words; once such a mechanism is provided, there is nothing to prevent it from accidentally treating some morphologically complex forms as monomorphemic. Under most theories, this will be especially true for high–frequency forms like most irregular verbs. The simulations in this chapter made a number of simplifying assumptions, in order to focus on competition between different past tense allomorphs. Most importantly, all inputs were already analyzed as morphologically complex forms. Thus there is no way for any of the models to capture the most prevalent account of early correct learning.18

Similar remarks apply to most theoretical explanations of the correlation between regular marking and overregularization. Under both rule–based and connectionist accounts, the onset of overregularization is predicted to correlate with the emergence of a generalization covering regular forms—whether this generalization is represented as a rule or as a pattern of connection weight (and whether or not its emergence requires unrealistic training schedules). In the present simulations, however, all (and only) the inflections of the English verbal system were present from the beginning of learning. What is learned by the models is not the set of available generalizations, but the productivity of existing generalizations. Therefore, any increase in overregularization which depends on the sudden discovery of the regular generalization cannot be captured by the present models.

However, it is possible that FG may exhibit weaker versions of early correct learning and the emergence of the regular rule that do not require the kinds of representational change described above. One of the ideas which underlies the model is that the mere existence of an abstract generalization does not mean that the generalization must be used productively. The model must learn which generalizations are productive from patterns in the input data. In the present simulations, at early stages of learning, many abstract generalizations were available, and the model was uncertain about the productivity of the regular rule. At the first training step, its performance on wug tests was just below 60% correct. At the same time, there were several high–frequency irregular verbs which were inflected correctly with probability 0.95 or higher. As the amount of training data increased, FG’s confidence in the productivity of the regular rule also increased. During the same set of training steps, some of the initially high–performing irregulars showed an increase in their overregularization rates.

18Of the models reviewed in Chapter 4, the only fully–implemented, computational proposal which was able to capture early correct learning was Taatgen and Anderson (2002).

This model was able to capture these phenomena by providing an alternate past tense marking rule which returned a stem without marking it for tense. This rule was assigned a low ACT–R cost, and, therefore, when their model had been exposed to a relatively small amount of data this rule was preferred. As their model received more evidence, the cost in prediction accuracy of using the null–marking rule increased, eventually eliminating the possibility of unmarked tensed forms.

Although assigning null marking a low a priori cost is somewhat arbitrary, the Taatgen and Anderson (2002) model and the current framework are very similar in other respects (see Section 4.4.2). These results suggest that, provided with similar mechanisms, FG may also exhibit a period of early correct performance.
Figure 5.13: Overregularization Rate and *Wug*-test Performance for Irregulars with Early High Performance (FG): This figure shows one minus the overregularization rate (open blue squares) for irregulars which were overregularized at a rate less than 5% or lower on their first appearance in the training set. *Wug*-test performance is shown with filled red circles.

This is illustrated in Figure 5.13, which shows 1 minus the overregularization rate (open blue squares) for verbs which were correctly inflected in the sample in which they first appeared (at a rate of 95% or better). Also shown is the rate of correct performance on *wug*-test items (filled red circles).

The example trajectories for individual irregular verbs given in 5.12 informally suggest that that the increase in productivity of the regular rule shown in Figure 5.13 may be the cause of the increase in overregularization rates for some initially correctly–marked verbs. For example, in Figure 5.12 the verb *saw* appears to increase overregularization around the same training steps where the model is showing improved *wug*-test performance in Figure 5.13.

However, the statistical reliability of this interaction between the productivity of regular rule and overregularization is difficult to investigate. It is necessary to first establish that the rates of overregularization for each verb, at each training step, are statistically significant, and not the result of noise during inference or idiosyncrasies of the training data. While it is possible, in principle, to generate confidence intervals for Bayesian models like FG, in practice, the large number of simulations involved and the difficulty of inference make this extremely challenging. I therefore leave the possibility of this interaction as a suggestion, and its investigation as a problem for future work.
Discussion

In this section, I showed that, of the five models examined in the thesis, only FG provides an account of the productivity of past tense inflections which is consistent with the observed developmental trajectory. Specifically, in simulations designed to idealize the increasing input data available to the child learner, only FG produced a low, but significant rate of overregularization. Furthermore, I showed that for FG—like children—overregularization is less frequent for more frequent verbs.

In Chapter 1, I argued for a view of productivity in which this property is inferred from the patterns of novelty and repetition in the input data. Under such a view, the availability of some abstract generalization does not oblige a system to use the generalization productively. In the present simulations, even irregular inflectional classes were represented fully abstractly in the underlying computational system. Despite this fact, FG only produced moderate rates of overregularization—even on the earliest and smallest training sets. Because the space of representations in the model does not develop or change during learning, this low rate of overregularization can only be a feature of the pattern of productivity and reuse in the data itself. As I have discussed, extensions to the model may allow it to more accurately capture phenomena like early correct performance. However, the fact that FG produces the right pattern of overregularization, even within the limits of its representational assumptions, is strong evidence that there is rich information about productivity available to learners, and, therefore, that an inferential approach to productivity and reuse is plausible.

5.5 Conclusion

In this chapter, I have presented the results of two sets of simulations on the English past tense. These results show that, of the five models considered, only Fragment Grammars can correctly capture the relevant patterns of productivity and reuse, accounting for the sharp dichotomy between the highly generalizable regular rule and the irregular inflectional classes. Furthermore, FG can capture the correct pattern from just the distribution of morphemes in the data. In the next two chapters, I consider a very different morphological system: English derivational morphology. Unlike the past tense, this system exhibits a broad cline of productivity, wide variability in phonological and semantic transparency, and rules with highly overlapping domains of applicability.
Chapter 6

English Derivational Morphology: Productivity and Conspiracies

6.1 Introduction

This chapter reviews the literature on productivity and reuse in English derivational morphology. English derivational morphology presents an interesting and important domain in which to examine questions of productivity and reuse for three reasons. First, in the history of generative grammar, it was here that the problem of controlling the productivity of generative processes was first highlighted as an independent question. Second, English derivational morphology presents a much greater diversity of phenomena than the domain of English verbal inflection. In particular, it provides examples of morphological processes occupying a wide cline of levels of productivity, from extremely productive suffixes such as -ness, to affixes of more limited and contextually-dependent productivity such as -ity, to affixes which can only be used productively for humorous or poetic effect, such as -th. This latter aspect of the system—its wide range of productivity levels—provides an important counterpoint to the near-determinism of the past tense system. Third, in Section 1.3.3, I discussed the fact that morphological systems often exhibit learning conspiracies: Several different kinds of evidence available to the language learner—such as semantic and phonological transparency, parsability, and the frequency distribution of words—all conspire to make similar, redundant predictions about the productivity of word formation processes. English derivational morphology provides a number of particular clear examples of such learning conspiracies, and the discussion in the following sections will be organized around these correlated properties.

6.2 Derivational Morphology

Although there is some debate about whether derivational and inflectional morphology should be handled by different formal machinery in linguistic theories (see e.g., Spencer, 1991; Stump, 2001a), the two kinds of morphology are nearly always distinguished on a descriptive level. Stump (2001a) offers the following general criteria for distinguishing between the two.¹

¹As Stump (2001a) stresses, these are rules-of-thumb rather than hard criteria for distinguishing between the two kinds of morphology.
1. **Change in lexical meaning and/or part of speech:** Inflectional morphology generally preserves the meaning of a stem (or lexeme) and the part–of–speech of a word. For example, with the exception of adjectival uses of participles, all of the forms of the English verbal paradigm discussed in Chapter 4 typically function as verbs. Furthermore, they all preserve the core lexical meaning and standard argument structure restrictions of the corresponding stem. Derivational processes, however, tend to change the category and meaning of their inputs in more drastic ways (e.g., *work*/worker, *true/truth, bland/blandness*).

2. **Syntactic determination:** Inflectional structure is often syntactically determined. For example, English perfect constructions, which use forms of the verb *have* (e.g., *have gone, has eaten, etc.*), require that the main verb occur in the past participle form. The form of the meaning–bearing element is completely determined by the syntactic context. English derivational structure is not syntactically specified to the same degree. Instead, derivational processes can be employed when an alteration of meaning or part–of–speech is required for semantic or pragmatic reasons. As a consequence, inflectional morphology is also often *obligatory* whereas derivational morphology is typically *optional*.

3. **Productivity:** In general, inflectional structure is more productive than derivational structure. This generalization is subject to important caveats, however. There are many highly productive derivational processes (see below), and, as seen in Chapter 4, some inflectional processes have very low productivity.

4. **Semantic regularity:** Usually, inflectional structure is highly semantically regular or transparent, while derivational structure is less so. For instance, English past tense allomorphs have a precisely specified meaning with respect to tense and aspect, while the precise meaning of English derivational affixes, such as *-ity*, can vary in ways which are not strictly compositional.

5. **Closure:** In general, the addition of inflectional morphology to a word means that further derivational structure cannot be added. While the agentive suffix *-er* can be added to a verb stem in English (e.g., *work/worker*) it cannot be added to an inflected verb stem (e.g., *worked/*workeder*). The converse is not true; for example, derivationally derived words are easily be pluralized (e.g., *worker/workers*).

Another difference between inflectional and derivational morphology is that while inflectional morphology can often be organized into paradigms, as I discussed in Chapter 4, derivational morphology typically cannot. There are a few cases where English derivational morphology seems to show a quasi-paradigmatic structure. For instance, verbs ending in *-ate* often drop this morpheme when forming an adjective in *-able* (e.g., *calculate/calculable, graduate/graduable* rather than *calculatable/graduitable*). However, in general, paradigmatic structure is far less evident in derivational morphology.

From the perspective of the models studied in this thesis, another very important way in which English derivational and inflectional morphology differ is the degree of hierarchical embedding which they permit. There is no upper bound on the size of words which can be generated by derivational processes: *agree, disagree, disagreeable, disagreeability*. Derivational morphology even permits recursive structure: *missile defense, anti-missile defense, anti-anti-missile defense*. By contrast, English inflectional morphology exhibits limited hierarchical structure and no recursion.

The existence of complex embedded structures in derivational morphology makes it necessary to introduce some additional terminology. When discussing inflectional morphology in Chapters 4 and 5, I distinguished only between a *stem* and the inflectional process applied to that stem. In contrast, in discussing derivational morphology, it is necessary to distinguish between a *stem*, a
base, and the derivational processes applying to bases. A stem refers to a morpheme without internal structure, that is, an atom. A base refers to a word or morpheme to which a derivational process applies—whether or not that word has already undergone some other derivational process, and whether or not it has internal structure.

While many English stems can stand on their own as words (e.g., agree/agreeable), others cannot (e.g., *vive/revive/survive). The latter class of stems are referred to as bound stems, while the former are referred to as free stems. In morphological theories, it is often assumed that stems are annotated with features indicating whether or not they are bound. This mechanism, or a similar one, is necessary to prevent bound forms from standing on their own. Note, however, that this presents the child with a learning problem. To the degree that children eventually learn that a prefix like re- is shared between forms with bound stems like revive and forms with free stems like redo, they must have correctly segmented the bound stem. However, once this stem is segmented, they must also learn that it can only be reused with a limited number of other prefixes or suffixes (e.g., survive) and that it must not be reused on its own. Thus the problem of determining the distribution of bound and free stems is simply another instance of the problem of productivity and reuse.

As I emphasized in the last chapter, one reason that the English past tense provides a natural test case for the models presented in this thesis is the great disparity that it presents between the productive regular rule and less productive irregular processes. There, I showed that by treating productivity as an inference this sharp disparity in productivity could be acquired from the input data—despite the simplifications made in modeling the domain.

The evaluation against derivational morphology provides a more robust test of each model’s ability to handle two other important aspects of linguistic structure. First, derivational morphology allows us to examine the consequences of learning patterns of productivity and reuse in a domain with considerable hierarchy and embedding. As I showed in Chapter 2, the productivity–as–an–inference model can be elegantly integrated into the kinds of recursive generative processes necessary for representing natural language. However, the past tense does not make full use of this property of the model. Complex, embedded structures define a vastly larger space of possible tradeoffs between productivity and reuse, and, thus, provide a more challenging test for the different models studied in this thesis.

Second, in contrast to the past tense, derivational morphology is not characterized by a sharp dichotomy between productive and unproductive processes. Rather it presents a cline between the two extremes, with many processes taking intermediate values. Thus, by extending the productivity–as–inference model to derivational morphology, I hope to establish that the model can account not only for a dichotomous system but also for a system requiring a more graded representation of productivity.

### 6.3 Empirical and Theoretical Background

The preceding section described some of the critical differences between derivational morphology and inflectional morphology, with a focus on how English derivational morphology differs from the English past tense. In this section, I describe five specific properties of the English derivational system (and the various theoretical accounts which have been proposed to handle these properties). These properties are:

1. **Selectivity**: English derivational affixes are characterized by a large number of selectional restrictions—semantic, phonological, morphosyntactic, and even pragmatic—governing the bases with which they can combine.
2. **Transparency:** English derivational affixes show high variability in their amount of phonological and semantic transparency or regularity. Some affixes, such as -ness or adverbial -ly are almost completely regular and transparent both in phonology (they simply concatenate to their base) and semantics (the meaning of the combination of base and affix is compositional). Other affixes, such as -ity or -ive, trigger phonological changes in the structure of the base. Some of these changes are systematic (like stress shift, e.g., normal/normality), and some of these changes they are idiosyncratic; (e.g., the final stem consonant in persuade/persuasive). Less phonologically transparent affixes also are often less semantically transparent. For example, while curiousness can only mean the general property of being curious, curiosity has an additional meaning—something which is curious (e.g., an odd or interesting object).

3. **Productivity:** Derivational affixes vary widely in their ability to form new words. Some affixes, such as -ness or (adverbial) -ly, are very productive and can be applied to form new words easily (pine-scentedness); other affixes, such as -th (e.g., width, length), do not form novel words easily.

4. **Parsability:** Affixes vary in their composability or parsability, as shown in processing experiments which are designed to examine the issue of whether words are retrieved whole from memory or whether they are built on the fly.

5. **Ordering:** When a word has more than one suffix or prefix, there are tight restrictions on the order in which these affixes can occur. In particular, of all the logically possible orderings, only a tiny subset occur in practice.

Critically, all five properties are highly correlated. In general, more productive affixes are also more parsable, show fewer selectional restrictions on their distribution, are more phonologically and semantically transparent, and tend to appear later in ordered sequences of multiple affixes. Less productive affixes, by contrast, show more evidence of whole–form retrieval, are less phonologically and semantically transparent, and tend to appear inside of more productive affixes in complex forms.

To some degree these correlations are logically necessary. For instance, an affix with tighter selectional restrictions will apply to a smaller set of bases and therefore it will have less of an ability to form novel words than will a less restrictive suffix. Such selectional restrictions will also affect the possible orderings among affixes. Nevertheless, even given these considerations, the properties described above are more strongly correlated than is strictly required by the logic of the system. In the remainder of the chapter each of these properties will be discussed in its own high–level section.

6.3.1 **Selectivity**

The distribution of English derivational affixes is governed by a number of selectional restrictions on bases and affixes. The most obvious kinds of restrictions are those on the category of the base. Affixes are sensitive to the morphological (or syntactic) category of their base and produce outputs with specific categories. For example, the categorial restrictions of -ness can be notated: $\text{Adj}>\text{N}$, signifying that this suffix takes adjectives (Adj) as inputs and returns nouns (N) as outputs. Categorial restrictions can be thought of as part of the definition of the affix. For instance, English has several different affixes which have the phonological form -ly. These includes one with category $\text{Adj}>\text{Adv}$ (e.g., quick/quickly, stupid/stupidly) and another with category $\text{N}>\text{Adj}$ (e.g., man/manly).

Many affixes also have phonological restrictions on the bases to which they can be attached. For example, the affix -en:$\text{Adj}>\text{V}$ (e.g., red/redden, short/shorten) strongly prefers its base to be monosyllabic. Furthermore it only attaches to bases which end in an obstruent (see e.g., Fabb, 1988; Plag, 2003, for discussion of these restrictions.). The suffix -ak$\text{V}>\text{N}$ (e.g., surprise/surprisal,
refuse/refusal, arrive/arrival) only attaches to bases which have final stress (e.g., *enter/*enteral Plag, 2003). Word formation is also governed by numerous semantic restrictions on affixation. For example, the agential affix -erV>N (e.g., baker) only attaches to verbs which have an agent as subject. The suffix -eeV>N (e.g., employ/employee) is restricted to refer to sentient entities (Plag, 2003).

There are also a number of restrictions on word formation which appear to be purely morphological. For example, some English affixes appear to be sensitive to whether their base is of Latin or Germanic origin: -akN>Adj (e.g., development, development/distributional) may only attach to either bare stems or bases ending in Latinate suffixes such as -ment or -ion (see, Fabb, 1988, for discussion and arguments against this analysis).

A second example of a morphological constraint on affixation is the base–driven selectional restriction, whereby affixes already attached to a base control which other affixes can further attach to the base. For example, to nominalize a verb ending in -ize, native English speakers prefer the ending -ation (e.g., nominalize/nominalization), despite the availability of many other suffixes which can be used to form abstract nouns from verbs (e.g., -ment, -age, -al)—some of which are more productive than -ation in other contexts. If a speaker further wants to form an adjective from an abstract noun ending in -ization, they will usually use the affix -alN>Adj (nominalizational) in preference to, for example, -aryN>Adj (e.g., inflationary).

Base–driven selectional restrictions have been the focus of much linguistic research. Aronoff and Schuvenewelt (1978) give several other examples, noting that -ousness is preferred to -osity (e.g., humorousness v. humorosity) and -iveness and preferred to -ivity (e.g., restrictiveness v. restrictivity), while -ility is preferred to -leness (e.g., servility v. servideness), and -icity is preferred to -icness (e.g., concentricity v. concentricness). They provide experimental support for some of these claims, showing, in a judgment task, that speakers prefer -iveness to -ivity for both existing words and novel bases ending in -ive. Such preferences remain even when extraneous factors such as phonological transparency are eliminated (see, Anshen and Aronoff, 1981; Cutler, 1980). Other cases where a less productive affix is preferred in some morphological context include -ability over -ableness (e.g., reversibility v. reversibleness), and -ality over -alness (e.g., centrality v. centralness) (e.g., Anshen and Aronoff, 1981; Embick and Marantz, 2008).

Critically, most of the kinds of selectivity discussed above appear to be viable preferences. Many phonological selectional restrictions are violable—awaken seems to violate the restriction that -en attaches to monosyllables. Likewise, the relationship between various base–driven selectional restrictions is also not absolute. Some preferences are quite strong (e.g., furiousness v. furiosity), while others are weak (e.g., ability v. ableness, tolerance v. toleration). This competition between possible derived forms is another example of a way in which derivational morphology differs from the English past tense. As I noted in Section 4.3, in the case of the past tense, there are typically (nearly) deterministic blocking relationships between an irregular and the corresponding regular form. Except for the rare case of doublets, when an irregular exists, it completely blocks the regular form. In derivational morphology, by contrast, the large number of graded and violable selectional restrictions mean that blocking must be viewed as a tendency, in the words of Kiparsky (1982b), rather than as a hard–and–fast rule (see Chapter 8).

Violability of selectional restrictions extends even to categorial constraints on bases. For example, -ness is a highly regular and productive affix which typically converts adjectives to nouns, but, in the case of the word forgiveness, it appears to take forgive, a verb, as a base. Likewise, the agential -er affix combines with nouns (e.g., slaver), verbs (e.g., baker), and even bound stems
The problem of violable categorial restrictions is further complicated by the wide-spread availability of zero-derivation or conversion in which the category of a lexical item can change without any overt morphological marking (e.g., The committee will table the resolution).

Chapters 4 and 5 emphasized that semi-productivity and semi-generalizability demand a quantitative account of competition. Any theory must explain why mostly unproductive processes, such as the irregular inflectional classes, can sometimes be generalized. The need for such a variable, quantitative approach is even greater in the case of English derivational morphology. In contrast with the past tense, the derivational system is characterized by competition between formation rules which often allow several forms, all of which have a reasonable degree of acceptability.

### 6.3.2 Transparency

Affixes vary considerably in their phonological and semantic regularity. Some affixes, such as -ness are phonologically simple—they combine with a base via concatenation and cause no phonological changes to its structure. By contrast, other affixes are phonologically opaque: The phonological structure of the base is different in affixes and unaffixed words. For example -ity triggers a number of phonological changes in the base: It moves the stress of the base (e.g., normal/normality), it sometimes changes the quality of base vowels (e.g., inane/inanity), and it sometimes changes the final consonant of the base (e.g., electric/electricity). Cases of phonological opacity vary in their systematicity. Some changes, such as stress shift, always apply when applicable. Some changes only apply in the context of other specific affixes, but do so systematically within that restricted context. For example, all words ending in -ic change /k/ to /s/ when combined with -ity (e.g., atomic/atomicity, electric/electricity). Other changes are associated idiosyncratically with particular stems or sets of stems. For example, a set of Latinate stems ending in /d/ changes this final consonant to /s/ before the affix -ive (e.g., pervade/pervasive, persuade/persuasive, invade/invasive, etc.).

Affixes also vary in their semantic regularity or transparency. Words such as curiosity, ability, and ethnicity have two different meanings. On one hand, they can be synonymous with curiousness, ableness, and ethnicness—referring to the general quality of possessing the corresponding adjective. On the other hand, they can also mean a particular specialized, countable instance of this quality: an albino rhinoceros is a curiosity, juggling is an ability, Italian is an ethnicity. Importantly, formations using -ness cannot have these more specialized meanings, nor can some words which end in -ity (e.g., electric/electricity).

Again, cases of semantic opacity vary in their systematicity. The ability of -ity to refer to a countable instances of the general quality seems to apply to most words using this suffix. However, complex derived words can also often accrue completely idiosyncratic meanings. For example, orientation refers to the act of orienting (e.g., We used a compass for orientation), an introduction to some new surroundings (e.g., During the first week on the job, he attended an orientation), and the state of being oriented (e.g., I lost my orientation in the woods), amongst many other things.

Semantic and phonological transparency are correlated. Words which are more transparent in one dimension of structure tend to be more transparent in the other. In Section 1.1, I discussed the case of -ness, -ity, and -th. Productive affixes, such as -ness tend to be transparent both in terms of their phonology (simple concatenation) and their semantics. Less productive affixes, which often change the phonological structure of their base, also tend to accumulate more idiosyncratic meanings. -ity is less phonologically and semantically transparent than -ness, and -th is less phonologically and semantically transparent than -ity. This phenomenon extends down to individual words: As a word’s meaning becomes idiosyncratic, it often changes pronunciation. Aronoff (1976) discusses the case of prohibition, which, in American English, can refer either to the historical period (1919–1933) when various legal restrictions were in place on alcohol, or, alternatively, to the general state or condition
of something being prohibited. When it is used to refer to the idiosyncratic meaning, the /h/ is usually deleted and the first i is usually reduced to a /a/. On the other hand, when used to refer to the general meaning, the /h/ is pronounced and the first i is reduced to /i/.

There is also a systematic relationship between transparency and frequency. Aronoff (1976, 1983) points out that less transparent words—especially less semantically transparent words—tend to be highly frequent. I mentioned above that there was a logically necessary relationship between productive processes and transparency. There is a corresponding, logically necessary relationship between idiosyncrasy and frequency. Whereas a completely productive process must be transparent in order to apply in novel circumstances, a completely idiosyncratic process must be attested, at least once, in order to be learned at all. In other words, regularity is necessary for interpreting novel forms whereas frequency is necessary for learning idiosyncratic forms. However, again, the correlation between (non)transparency and frequency goes beyond this logical necessity: As frequency increases so does irregularity (see, e.g., Aronoff, 1983).

Baayen and Lieber (1996) explored the frequency distributions of transparent and non-transparent processes by systematically analyzing the Dutch prefix ont- (equivalent to English un-). They find that Dutch words using this prefix can be divided into several classes which exhibit distinct semantic and categorial selection structure, as well as differing levels of productivity. Based on this, Baayen and Lieber (1996) argued that there are several distinct allomorphs of ont-, which differ in their word frequency distributions. Unproductive allomorphs of ont- have an approximately log-normal word–frequency distribution—characterized by thin tails at both the upper and lower ends of their (log) frequency ranges. Productive allomorphs of ont-, on the other hand, have long–tailed frequency distributions. Notice that this pattern is also observed for productive and unproductive past tense allomorphs (Section 4.3.3). Productive processes give rise to distributions with a large number of low probability events, whereas unproductive processes are characterized by a distribution which is dominated by existing forms.

Most importantly, from the point of view of the present discussion, Baayen and Lieber (1996) presented a detailed analysis of the lexical semantics of the various classes, using a version of the formal framework of Jackendoff (1990). They demonstrated that the more productive allomorphs of ont- share more aspects of their semantic representations with other words in Dutch than the less productive allomorphs do. Specifically, the most productive classes of ont- words, which form verbs from nouns, have a primitive CAUSE element and incorporate their base nouns as themes, which is very common structure for Dutch verbs in general.

In summary, there is wide variation in the level of phonological and semantic transparency associated with English derivational affixes. However, the two are correlated. More phonologically transparent structures are more semantically transparent and vice versa. There is also a relationship between transparency and the shape of word–frequency distributions (cross-linguistically). More transparent affixes are associated with frequency distributions which have longer tails, appearing in greater numbers of low frequency forms. Less transparent affixes tend to be associated with more normally–shaped distributions (in the log domain).

### 6.3.3 Productivity

The English past tense, discussed in the previous two chapters, is characterized by a sharp dichotomy in productivity between regular and irregular inflectional process. The regular rule is broadly productive, applying in any case where an irregular form is not available, while the irregular

---

3A lognormal distribution is a distribution which is normal in the logarithm of some random variable

4Baayen and Lieber (1996) speculate that this may be Poisson distributed in log frequency.
inflectional classes have extremely limited productivity and are only rarely generalized to novel stems. English derivational morphology, by contrast, demonstrates far greater variability: There are very productive affixes (e.g., -ness, -ly), semi–productive affixes (e.g., -ity, -ive) and unproductive affixes (e.g., -th, -ster). In this section, I examine the productivity of English derivational morphemes in more detail. I will first give an overview of evidence for the generalizability of various derivational morphemes, and, then, discuss several theoretical approaches to the question of productivity. After discussing these issues, I will present a detailed review of probabilistic measures of productivity from the literature.

There is ample anecdotal evidence that English speakers are able to use some affixes (such as -ness, -ly, -y and -er) to form new words. In some cases, these intuitions are supported by experimental results. For example, Berko (1958) showed in her famous study that agentive -er could be generalized to novel verbs by both adults and children. In a study of a corpus of speech “errors,” MacKay (1979) found that speakers spontaneously generalize a number of derivational affixes to create novel forms. Furthermore, Anshen and Aronoff (1988) found that participants spontaneously generate more novel forms for -ness than they do for -ity (see, also, Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978; Cutler, 1980, discussed in Section 6.3.1 above).

The psychological reality of productivity distinctions is further reinforced by the way in which speakers sometimes fail to differentiate between potential and existing words for highly productive processes (Aronoff, 1983). For example, Aronoff and Schvaneveldt (1978) and Anshen and Aronoff (1988) found that, for words involving highly productive affixes, people have difficulty distinguishing words that they (probably) have never heard from words that they (probably) have.

From the point of view of this thesis, the fundamental question of productivity is to explain why certain processes more readily form new expressions than others. Two broad possibilities have been suggested (see Bauer, 2005). One view, which I will call the deterministic view, sees productivity as an all–or–nothing phenomenon. A structure building process is either productive—and can therefore be used to form new expressions—or it is not. Under the deterministic view, apparent differences in the productivity of various processes are due either to differences in their selectional restrictions or to extragrammatical factors, such as the pragmatic utility of various forms (Bauer, 1992). Under this theory, for example, an affix like -th forms fewer novel words than -ness because it is more selective about the bases with which it can combine. This view is exemplified by classical linguistic theories such as those discussed in the context of the history of derivational morphology in Section 1.3 and in the context of the past tense in Section 4.4.2.

A second view is the scalar or gradient view of productivity. Under this view, the readiness with which a word formation process can be used to form novel words is considered to be a measurable and gradient property of the process itself. This gradient property is argued to be a fundamental part of speakers’ linguistic knowledge of each process and not a mere epiphenomenon of interacting deterministic constraints (Hay and Baayen, 2005).

Numerous authors have given arguments in favor of a scalar view of productivity. For example, Bauer (1992) studied the affix combination -lily (e.g., holily, i.e., holy+ly), and showed that, even taking into account selectional constraints on the distribution of adverbial -ly, there is still variability in readiness with which it attaches to different classes of words ending in adjectival -ly. Likewise, Plag (2003) argued that frequency dependent differences in parsability across words using the same affix require a scalar account of productivity (see discussion of parsability below).
In recent years, the scalar view has been endorsed by many theorists in the field (see e.g., Aronoff and Fudeman, 2005; Bauer, 2005; Hay and Baayen, 2005).

Productivity is a gradient property in all of the models examined in this thesis. However, the gradience of productivity is not a first principle of these models, but, instead, follows as a consequence of two more basic assumptions. First, the models take as their goal the problem of predicting future novelty and reuse. Second, the models measure uncertainty about these predictions using probability. Since the productivity of a process is the readiness with which it can produce novel forms, productivity is quantified in the models as the probability that they assign to novel forms. In the next section, I discuss in more detail measures of productivity from the literature and the relationship between these measures and quantities computed by the models in this thesis.

Measuring Productivity

In this section, I address two questions. First, if productivity is a scalar property of word formation processes, how should it be quantified and how can it be measured empirically? Second, how does it relate to familiar statistical quantities which can be computed from corpora, such as the type and token frequencies of words and affixes? As a point of reference, I will first introduce and discuss several conditional probability distributions which capture different aspects of the notion of productivity. I will then given a historical overview of measures of probability, in terms of these distributions.

Conditional Probabilities that Measure Productivity

There are several probabilistic quantities which measure the readiness with which word formation processes can be used to generate novel forms. Consider some word formation process \( p \in P \), where \( P \) is a random variable which ranges over all possible word formation processes. For example, \( p \) might be the process which affixes \(-ness\) onto the end of an adjective, while the random variable \( P \) includes: this process, \(-ity\) affixation, \(-th\) affixation, etc. Let \( N \) be a boolean random variable which ranges over \{true, false\} and specifies whether a word form is novel or not with respect to some dataset. For instance, if the word in question was \textit{happiness}, and this word appeared in the dataset in question, then \( N = F \); if the word was \textit{pine-scentedness}, which did not appear in the dataset, then \( N = T \). Define the quantity \( \Pi_p \):

\[
\Pi_p = P(N = T, P = p)
\]  

Equation 6.1 is the joint or marginal probability that \( p \) is used and that the result is a novel form. Note that this value is summed over all possible novel words using \( p \) and over all possible computations (derivations) over each of these words. In other words, it is the total probability of the joint event of a word being novel and using \( p \), ignoring all other factors (such as the identity of the base).

There are two things to note about this quantity. First, any way of computing or estimating \( \Pi_p \) that is based on some corpus \( C \) must be inferential or model-based. The quantity measures the probability mass on novel linguistic forms, and, by definition, these cannot be observed in the corpus itself. Therefore any estimator for \( \Pi_p \) will necessarily be based on some specific assumptions about how probability is projected from a corpus of examples to unseen structures. Second, if the estimator is based on a textual corpus, it will need to make fairly radical simplifying assumptions. There are many factors which govern the use of novel forms—pragmatic utility, selectional restrictions on bases, various kinds of frequency, and, of course, the architecture of the computational system which underlies language comprehension and production. Many of the relevant kinds of information are simply unavailable in a text corpus, and, for those that are, our current state of theoretical
understanding is limited. In this discussion, I will focus on estimating \( \Pi_p \) and related quantities from distributional aspects of the corpus, that is, various kinds of type and token frequencies of (parts of) words.

There are two other quantities which are closely related to \( \Pi_p \) and which capture intuitive aspects of productivity (Baayen, 1992, 2001, 2003). The first, \( \Pi^*_p \), focuses on the probability that a particular word formation process \( p \) will be used to form novel words.

\[
\Pi^*_p = P(P = p|N = T) = \frac{P(N = T, P = p)}{P(N = T)} = \frac{\Pi_p}{\sum_{p'} \Pi_{p'}}
\]  

(6.2)

\( \Pi^*_p \) is the conditional probability of \( p \) given that a novel form is generated. For example, it might measure the proportion of all possible novel words which use the suffix -ness. This quantity can be interpreted as the proportion of overall vocabulary growth at some moment in time that is attributable to word formation process \( p \) (see Baayen, 2001). Because of this, Baayen (2006) calls this value the expanding productivity. \( \Pi^*_p \) is appropriate for comparing productivity between different word-formation processes, such as word formation with -ity versus -ness. Note that \( \Pi^*_p \) is simply the joint probability, \( \Pi_p \), normalized by the sum over all word formation processes: \( \sum_{p'} \Pi_{p'} \).

Like all conditional distributions, some information is lost relative to \( \Pi_p \). In particular, \( \Pi^*_p \) does not consider the base rate of the process \( p \). Also note that \( \sum_{p'} \Pi_{p'} \) is a constant with respect to a set of individual processes \( p' \). Thus, if we are only concerned with the relative values of \( \Pi^*_p \), then we are free to use the joint probability \( \Pi_p \), because the two quantities are related by a constant factor (i.e., \( \frac{1}{\sum_{p'} \Pi_{p'}} \)). I will make use of this fact in the next chapter.

A second quantity which is closely related to \( \Pi_p \) is the conditional distribution with the random variables reversed: \( \hat{\Pi}_p \).

\[
\hat{\Pi}_p = P(N = T|P = p) = \frac{P(N = T, P = p)}{P(P = p)} = \frac{\Pi_p}{\sum_{t \in \{T,F\}} P(N = t, P = p)}
\]  

(6.3)

\( \hat{\Pi}_p \) is the conditional probability of a novel form, given that a word formation process \( p \) is used. Intuitively, this quantity measures the (weighted) proportion of all possible words using \( p \) which are novel. For example, it measures the proportion of all possible words ending in -ness, weighted by their probability, which are not observed in the training corpus. \( \hat{\Pi}_p \) also has an interpretation in terms of growth rate: It measures the instantaneous rate of growth in the part of the vocabulary which uses process \( p \) (see Baayen, 2001). A third way to understand \( \hat{\Pi}_p \) is as the amount of probability mass associated with \( p \) which is held–out or reserved to generate novel forms in the future; that is, the potential productivity of \( p \) (in the terminology of Baayen, 2006). As was the case with \( \Pi^*_p \), some information is lost relative to the numerator \( \Pi_p \). In particular, this measure cannot take into account the base rate of the process \( p \); it only specifies the probability that a new form will be generated once \( p \) has been chosen. For example, a very common process—such as -ness affixation—may have the same \( \hat{\Pi}_p \) value as a less common process—such as affixation with -fold. Many more novel words using -ness will appear over time than words using -fold, because -ness affixation has a higher base rate; however, \( \hat{\Pi}_p \) will not be sensitive to this difference. Also note that, unlike \( \Pi^*_p \), \( \sum_{t \in \{T,F\}} P(N = t, P = p) \) is not constant across affixes, thus, if we wish to compare values of \( \hat{\Pi}_p \), we must compute it exactly.

**Measures of Productivity from the Literature** There is a long discussion in the literature about how to best estimate the productivity of morphological processes from data. In this section,
I will focus on estimators which make use of distributional quantities such as the type and token frequencies over bases and affixes.

As I have already noted (see e.g., Sections 1.3.3, 4.3.3, 6.3.2), there is a clear relationship between type and token frequencies and levels of productivity. Less productive word formation processes are characterized, on average, by a smaller number of word types and a higher token frequency for each type, while more productive processes are characterized, on average, by a larger number of word types each with lower token frequency (Baayen, 1992, 1993, 1994, 2001, 2006; Baayen and Lieber, 1991; Baayen and Renouf, 1996; Barnwell, 2010; Bauer, 1992, 2001, 2005; Du and Zhang, 2010; Fernández-Domínguez et al., 2007; Gaeta and Ricca, 2006; Hay and Baayen, 2002; Plag, 2004; van Marle, 1992; Yang, 2005).

This relationship reflects a tradeoff: For a given number of words, productivity is predicted by the shape of the distribution of tokens over types, and, therefore, any measure of productivity which is based only on type or token frequencies of words (or parts of words) will be inaccurate in some cases. Bybee (1995b), for example, proposes the type frequency (also known as the morphological family size) of an affix as a predictor of the productivity of morphological processes. However, this measure suffers from not taking into account token frequency. It is possible for a process to apply to a relatively large set of forms and still be unproductive, if each of those forms is frequent. Baayen (2006) points out, for example, that the Dutch verbal prefix *ver-* has greater type frequency than the Dutch agentive suffix *-ster*, but that the former is judged less productive than the latter by speakers. He argues that this is because most types using *ver-* have a high token frequency. This has led a number of researchers to propose more complex statistical measures which combine type and token frequencies in various ways.

An early account was given by Aronoff (1976) who proposed that the productivity of a process should be measured as the proportion of existing word types which use the process divided by the number of potential word types using that process. Aronoff’s proposal does not take into account token frequencies per se, however, under the assumption that all potential and existing words are equiprobable, Aronoff’s proposal is equal to $1 - \Pi_p$.\footnote{A related criticism of Bybee’s proposal is made with respect to the German plural in Marcus et al. (1995). The most generalizable allomorph of the German plural has the lowest frequency in terms of both type and token counts.}

Another index of productivity which attempts to combine variability and token frequency is given by Barnwell (2010), who defines an estimator of the productivity of variable positions in frame–based constructions such as the the. Barnwell proposes that the productivity of a variable in a construction should be measured by the product of the log frequency of the construction and the empirical entropy of the slot. Entropy is maximized when the empirical distribution is uniform over all forms, and, thus, this measure assigns high productivity to frequent frames with more variability. Although this measure is likely to be better than raw type frequency, it will still suffer from being unable to take into account the fact that specific high frequency forms detract from the productivity of a process (Aronoff, 1983).\footnote{Aronoff’s proposal is actually an index of non–productivity, rather than an index of productivity. Also, unlike the measures described below, Aronoff’s proposal is more conceptual than practical since he does not specify a way of enumerating potential words.}

Yang (2005) proposes an account of productivity based on a serial search model of lexical

\footnote{Although Barnwell’s measure uses probabilistic quantities, it does not seem to have a probabilistic interpretation. Multiplication by log frequency corresponds mathematically to exponentiation, and, therefore, a product of log frequency and entropy does not appear to have an interpretation in terms of the standard operations of probability theory (i.e., taking expectations/marginalization, or conditioning). As a result, the measure does not seem to correspond to an estimate of the entropy, relative entropy, mutual information, etc. of any probability distribution(s) associated with a corpus.}
access. In Yang’s model, each word formation process, such as -ity or -ness affixation, is represented by a rule. These rules always include a list of exceptional bases to which they can apply. The rules may also include an optional default case, which applies to any base (that meets the categorial and other selectional restrictions of the rule). To analyze a word, a rule first checks the list of exceptional bases in order of decreasing frequency. Only when no exceptional base matches does the default apply (when present). For Yang (2005), a rule is productive if it provides a default case.

Yang argues that a rule will have a default case if the expected time required to apply the rule with the default is less than the expected time when there is no default. Time is measured in the number of steps taken to analyze a form. In the case of a form which is handled by the default, this will correspond to the number of listed exceptions that must be checked and rejected before the default can apply. Because exceptions are ordered in terms of frequency, the expected number of steps which it takes on average to analyze any form will depend on the distribution of low and high frequency forms using the rule.

Consider how this system balances token frequency against type variability. When a process applies productively to a large number of low frequency forms, a default case will be favored. A rule will tolerate exceptions so long as their number is relatively low, and their token frequency is relatively high—because the expected time searching through the list of exceptions prior to applying the default case will remain low. However, when the number of exceptions is high, and/or their frequency is relatively balanced, the expected time searching the list of exceptions will overcome the advantage of including the default. Yang derives an estimate of the threshold at which a rule with a default case will be favored over one without: Based on the assumption that word–frequency distributions are Zipfian, Yang shows that when the number of exceptions, $M$, is less than the total number of words, $N$, divided by the logarithm of the total number of words (i.e., $M < \frac{N}{\ln(N)}$) a productive default case will be favored.

Yang’s approach is consistent with general observations about productivity and word–frequency distributions. When the distribution has a small number of very high–frequency forms and a very thin tail, or when it has a large number of forms with moderately high, and nearly identical frequency, the rule will be treated as an unproductive. When a rule is characterized by a long tail of low–frequency forms, it will represented as a productive rule with a default. However, there are also a number of problems with Yang’s model. First and most importantly, it cannot provide quantitative estimates of differences in productivity judgments between two processes which both have a productive rule. Under this model, a word–formation process is either productive (has a default) or it is not. Second, it is not clear how the model generalizes to the case of words with hierarchical or recursive structure.


$P$ and $P^*$ are estimators for the quantities $\hat{\Pi}_p$ and $\Pi_\infty^*$, described above. These estimators are based on the theory of Good–Turing estimation (Good, 1953). Recall that one problem with defining measures of productivity is that they must estimate the probability of events which are unobserved—the probability that various word–formation processes will generate novel words. Good–Turing is a well–known (frequentist) statistical technique for estimating the probability of unseen types from a population of observed types. Imagine that we have some corpus $C$ of $N$ words.

---

10 As explicitly noted by Yang, this is a version of the elsewhere condition (see Section 4.4.2).
together with their frequencies. Let \( V(1,N) \) be the number of words appearing in corpus \( C \) (of size \( N \)) with frequency 1. In the literature, these word types, which appear only once, are known as hapax legomena or hapaxes for short. Let \( E[V(1,N)] \) be the expectation of \( V(1,N) \).

Good–Turing estimation states that a good estimate of the probability of observing a novel word type on the next draw is given by the following expression:

\[
\frac{E[V(1,N)]}{N}
\]

In other words, under Good–Turing estimation, the probability of a novel word is estimated by the proportion of already observed words which appear with frequency 1—the proportion of hapaxes. Good–Turing estimation has proven very useful for smoothing natural language frequency distributions (i.e., accounting for probability mass which belongs to words unseen in the training sample) and is widely used for such purposes in computational linguistics (see e.g., Chen and Goodman, 1998; Goldwater et al., 2009; Teh, 2006).

Like any estimator for unobserved events, Good–Turing is based on mathematical assumptions which can be adjusted in various ways (see e.g., Gale and Sampson, 1995; Orlitsky et al., 2003), and given various interpretations. For example, Baayen (2001) discusses the fact that this estimator has a geometrical interpretation as the slope of the tangent line to the growth curve of vocabulary size at sample size \( N \).

Good–Turing is also closely related to the Pitman–Yor processes discussed in Section 2.3.3. For Pitman–Yor processes, the proportion of probability mass on novel events in the limit of infinite observations is given by the Pitman–Yor \( \alpha \) hyperparameter. This value is also the limiting expected proportion of hapaxes for the process (i.e., the proportion of tables with a single customer. Goldwater et al., 2009; Teh, 2006). This fact provides a Bayesian justification of Good–Turing estimation.

Good–Turing can be used to provide estimates of \( \Pi^*_p \) and \( \hat{\Pi}_p \). The estimator of the first quantity is called \( P^* \) by Baayen. This quantity is also sometimes referred to as the hapax–conditioned degree of productivity or the expanding productivity (e.g., Baayen, 2006). Let \( E[V_p(1,N)] \) be the expected number of forms which use morphological process \( p \) and which are represented by exactly one token in a corpus \( C \) of size \( N \). The \( P^* \) estimator of \( \Pi^*_p \) is given by:

\[
P^* = \frac{E[V_p(1,N)]}{E[V(1,N)]}
\]

In other words, \( P^* \) is just the proportion of all hapaxes in the corpus which are attributed to process \( p \). Like the quantity it estimates, \( \Pi^*_p \), \( P^* \) is useful for comparing the differential productivity of various affixes.

The estimator of \( \hat{\Pi}_p \), which Baayen provides, is called \( P \), the category–conditioned degree of productivity, or the potential productivity. Let \( E[V_p(N)] \) be the expected frequency of all words

\[11\] The close relationship between Pitman–Yor processes and Good–Turing estimation could suggest that it is inevitable that the Fragment Grammars model and Baayen’s measures return similar results, but this is not the case. There are two reasons for this. First, the relationship between Good–Turing and Pitman–Yor processes concerns the probability of a new table. The probability of generating a new form in Fragment Grammars is most closely related, instead, to the probability of a variable in some stored derivation fragment. Once a fragment has been stored with a variable, the relationship between use of that fragment and new–table probability is complex. Second, the relationship between Good–Turing and Pitman–Yor processes holds for a single Pitman–Yor process. Fragment Grammars consist of a system of recursively defined Pitman–Yor processes. When applied to data set, recursively defined or hierarchical PYPs can have very different properties from a single process. For example, Blunson et al. (2009) show that approximations to related Bayesian nonparametric models, which hold in the case of a single process, do not hold when models are defined recursively or hierarchically, and that, in such cases, the approximations can be highly inaccurate.
in the corpus that make use of word formation process \( p \). Then the category–conditioned degree of productivity is given by:

\[
P = \frac{E[V_p(1, N)]}{E[V_p(N)]}
\]  \hspace{1cm} (6.6)

In other words, \( P \) is the proportion of all words in the corpus which are attributable to \( p \), and which appear only once. Like the quantity which it estimates, \( \Pi_p \), \( P \) measures the rate at which the set of words which are derived using \( p \) is growing.

Baayen has argued extensively that both quantities are needed to capture the intuitive notion of productivity. While \( P \) measures the readiness with which a process forms new words, it cannot take into account the base rate of the process and thus overestimates the intuitive productivity of low–frequency affixes. \(^{12}\) Likewise, \( P^* \) cannot take into account the overall rate at which words are being added to the vocabulary and is only appropriate for comparing the productivity of different processes.

In the next chapter, I will compare values of \( P^* \) and \( P \) computed from corpora with estimates of \( \Pi_p \) and \( \hat{\Pi}_p \) computed directly from the models studied in the thesis. It is therefore appropriate to say a few words about the theoretical and psychological interpretation of both measures.

There are a number of reasons to believe that Baayen’s measures correctly capture at least some aspects of morphological productivity. First, Baayen’s measures have been the subject of a long discussion in the literature, and, in many cases, seem to fit with the intuition of linguists (Baayen, 1993, 2006; Baayen and Lieber, 1991; van Marle, 1992). Second, their derivation in terms of Good–Turing estimation gives them a firm theoretical foundation. Third, Baayen’s estimators are also reasonably replicable across corpora. For example, Baayen and Renouf (1996) show that estimates of the two quantities from two different corpora achieve similar (although not identical) results. Fourth, there is some empirical evidence that these measures correlate with experimental data. Baayen (1994) presents a production experiment in which speakers were asked to generate as many forms as possible using particular Dutch affixes. He found that the category conditioned degree of productivity (i.e., potential productivity), \( P \), was a good predictor of the number of novel words produced.\(^{13}\) Finally, as I will discuss below in greater detail, Baayen’s quantities correlate with a number of other factors, such as measures of parsability and ordering, which are theoretically predicted to be related to productivity.

Nevertheless, it must be emphasized that comparison between model predictions in this thesis and theoretical measures of productivity such as Baayen’s can only serve as evidence for theoretical convergence. To the degree that a number of different measures agree, we can conclude that they are likely to be tapping into a common underlying cause. However, when they disagree, there are many possible explanations. A true, gold–standard evaluation of morphological productivity requires wide–coverage experimental data of a kind which do not yet exist.

\(^{12}\)Gaeta and Ricca (2006) make a similar point and propose that values of \( P \) should be calculated relative to a corpus which is stratified in terms of affix frequency. (Baayen, 2006) points out that their measure gives similar results to \( P^* \).

\(^{13}\)Note that, because in such experiments the participants are provided with an affix and then asked to generate forms, the conditional probability of novelty given the affix (i.e., \( \hat{\Pi}_p \) estimated by \( P \) is the appropriate predictor.
6.3.4 Parsability: Decomposition and Retrieval

A fundamental question in psycholinguistics is whether morphologically–complex words are retrieved whole from memory or composed on the fly. The results of over forty years of intense research on this question have definitively shown that both composition and retrieval processes are employed. One case which has been studied extensively is English prefixation. Using a lexical decision task, Taft and Forster (1975) showed that bound stems from truly prefixed words (e.g., juvenate from rejuvenate) take longer for subjects to reject than pseudo–bound stems (e.g., repertoire from repertoire). They concluded that prefixed words are represented in terms of their parts. Further evidence for this conclusion comes from experiments using different manipulations (e.g., prefixed non–words versus unprefixed words) and other experimental paradigms (e.g., primed lexical decision. See, e.g., Forster and Azuma, 2000; Taft et al., 1986). However, Stanners et al. (1979b) presented evidence from an identity priming task that prefixed words with bound stems elicit weaker priming of their stems than prefixed words with free stems, suggesting that words with bound stems are represented as wholes.

Similar results hold for other languages. For example, in a lexical decision study, Taft and Zhu (1995) showed that Chinese also exhibits a mixture of decomposition and retrieval. In particular, Chinese characters which are shared across several words showed evidence of decomposition while characters appearing only in a single word showed evidence of retrieval. Bertram et al. (1999) and Bertram et al. (2000c) found evidence for a mixture of computation and retrieval in both Dutch and Finnish.

These results show that there is pervasive variability in parsability—whether a word is composed on the fly or retrieved whole from memory. This variability extends down even to the level of individual words. For example, Hay (2001) showed that composition and retrieval can vary across individual words which share the same affix (e.g., abasement versus enticement). Although there is a broad consensus in the literature that lexical processing mixes parsing and retrieval, the factors which predict each mode of computation are numerous and complex. In the rest of this section, I will give a brief review of the three factors which appear to be most relevant: frequency, productivity, and semantic and phonological transparency. I will first consider experimental and theoretical work which attempts to relate parsability and frequency, focussing on Hay’s relative frequency hypothesis (Hay, 2001, 2003). After discussing this literature, I will consider the relationship between parsability and productivity. Finally, I will conclude with a discussion of the relationship between parsability and transparency.

Parsability and Frequency: Measures and Models

It has long been known that both base (or stem) frequencies and whole–form frequencies play a role in morphological processing. High–frequency words are more likely to be retrieved, while low–frequency words are more likely to be composed. This fact, which was discussed in the context of the English past tense in Section 4.3.4, has also been demonstrated with a wide variety of experimental methods, including lexical decision, identity priming, speeded naming, and free reading. (e.g., Alegre and Gordon, 1999a; Baayen et al., 1997b, 2003, 2007; Beck, 1997; Bertram

14Mandarin Chinese words are largely bimorphemic, while characters typically represent a monosyllabic morpheme, so most words consist of two morphemes/characters.

15The model presented in this thesis allows variability to extend even down to the level of individual uses of individual words. In fact, the Fragment Grammars model makes crucial use of this fact. It is the average rates of parsing which give rise to scalar productivity and parsability effects in the model. Allowing this kind of variation may be important for modeling differences between experimental tasks. For example, Andrews (1986) shows that decomposition of morphologically complex words is modulated by the inclusion of compound words in tests.
et al., 2000b,c; Bradley, 1980; Burani and Caramazza, 1987; Burani et al., 1984; Clahsen et al., 2004; Colé et al., 1989, 1997; Gordon and Alegre, 1999; Manelis and Tharp, 1977; Meunier and Segui, 1999; Moscoso del Prado Martín et al., 2005; Prasada et al., 1990; Rubenstein et al., 1970; Sereno and Jongman, 1997; Taft, 1979; Vannest and Boland, 1999). For example, Niswander et al. (2010) found that whole–form and base frequencies predict fixation time in a free reading task: a pattern that has also been documented in a number of languages with very different morphological systems. For example, using a lexical decision task, Burani and Caramazza (1987) found that Italian, which is a more highly inflected language than English, evidences both whole–form and stem frequency–effects. While most studies have focused on the processing of simple stem–affix combinations, more recently it has been shown that the frequencies of even deeply–embedded word constituents can have effects on processing latencies (Kuperman, 2008).

As was the case for the English past tense, processing latencies for high–frequency forms are often better predicted by their whole–form frequencies, while latencies for low–frequency forms are often better predicted by the frequencies of their parts. Using a lexical decision task, Colé et al. (1997) showed that the processing of morphologically complex words in French exhibits an interaction between base and whole–form form frequencies. When the frequency of the base is less than the frequency of the whole form, latencies are better predicted by the base frequency. However, when whole–form frequency is greater than that of the base, latencies are better predicted by the whole–form frequency. Using similar French materials, Meunier and Segui (1999) showed that only low–frequency suffixed words prime their stems in an identity priming task.

Whole–form frequency is well–established as a good predictor for processing latencies of high–frequency words, and also explains part of the variance in processing times for even relatively low–frequency words (Baayen et al., 2007). There has been more debate, however, about which kinds of part–word frequencies should be used to explain the processing of low–frequency, potentially–composed words. Historically, most studies have made use of the token frequency of bases or stems to predict the processing of low–frequency forms (e.g., the studies discussed in the preceding paragraphs). However, a number of more recent studies have emphasized the use of base type frequency, sometimes called the morphological family size (MFS) of a word. For example, Bertram et al. (2000a) found that MFS predicts the processing time of complex Dutch words and that this effect is driven by semantically transparent family members, suggesting that MFS is specifically linked to regular and productive composition. Similarly, Moscoso del Prado Martín et al. (2005) demonstrated an effect of MFS for both Dutch and Hebrew and, further, showed that this effect is based on shared morphological structure, rather than just shared semantics. In fact, the MFS of a base even predicts processing time when the base is presented in isolation, suggesting that a word’s morphological family is important even when other affixes are not present (Baayen et al., 2006; Schreuder and Baayen, 1997).

Baayen et al. (2007) argue that only morphological family size—not base token frequency—is relevant for predicting the processing times of low–frequency words. In a study of low–frequency English words, they demonstrated that, while whole–form token frequency and morphological family size both have a facilitatory effect on lexical decision latencies, base token frequency has no effect when these (and other) factors are controlled by inclusion in a linear model. This result should be interpreted with caution. A linear model assumes that the relationship between variables such as the (log) morphological family size and (log) base or stem token frequency can be captured as a linear combination of their values. However, in the present case, this assumption is clearly wrong. The type frequency of a base is the number of distinct words in which a base was used, and, therefore, provides a lower bound on the token frequency of the base. This relationship cannot be expressed as a linear combination. In particular, for low–frequency forms, like those studied by Baayen et al.
Information–Theoretic Models of Parsability There have been a number of recent attempts to provide theoretical accounts which predict processing measure from various frequencies in a way that does not (implicitly or explicitly) assume a linear model. One such class of models incorporates various information theoretic measures such as entropy, Kullback–Leibler divergence, and mutual information (Kostić, 1995, 2003; Kuperman, 2008; Kuperman et al., 2010; Moscoso del Prado Martín et al., 2004). For example, Moscoso del Prado Martín et al. (2004) propose that the processing latency associated with a word should be related to the difference in two factors: the the negative log probability of a word (also known as surprisal; see Equation 6.7) and the entropy of the word’s morphological family (which they call its paradigm, \( P_w \); see Equation 6.8). That is, they propose that processing latency should be positively correlated with the expression shown in Equation 6.9.

\[
I(w) = -\log[p_w] \tag{6.7}
\]

\[
H(P_w) = -\sum_{x \in P} P(x|P_w) \cdot \log[P(x|P_w)] \tag{6.8}
\]

\[
I(w) - H(P_w) \tag{6.9}
\]

\( I(w) \) approaches 0 as the the whole–form frequency of the word increases, and, thus, captures the intuitive idea that as the whole–form frequency of a word increases, its processing latency decreases. The second term, \( H(P_w) \), increases as the set of words in a morphological family gets larger and more uniform in probability. Thus, this model predicts a facilitatory effect for words which are members of large morphological families with members that are similar in frequency. Note that the two terms represent a tradeoff: Increasing the frequency of a single form, \( w \), will decrease its surprisal, \( I(w) \), and decrease the entropy of its morphological family, \( H(P_w) \) by making it less uniform (note the signs in the equations above). Thus, this model predicts that processing of complex words should be fastest either when a complex word is very frequent, or when it is the member of morphological family that provides robust evidence of composition. The slowest complex words will be those which are infrequent and members of morphological families that give little evidence for composition.

Measures of Parsability based on Models of Lexical Processing Another category of frequency–based predictors for processing data are those derived from models of lexical access. The best known quantity is Baayen’s \( A \) (Baayen, 1993; Hay, 2001). This quantity is derived from a dual–route, activation–based, race model of morphological processing. In such a model, there is competition between two ways of building a word: composition and whole–form retrieval from memory. In Baayen’s original formulation (Baayen, 1993), it is assumed that for each word there

---

16Two of the models examined in this thesis—Fragment Grammars and Adaptor Grammars—use MFS of stems when estimating the probabilities of rules in their underlying PCFG base systems. Both models also use token frequencies to estimate the probability of reusable stored derivation fragments. Thus, in these systems, type and token frequencies are related hierarchically: Type frequencies are only used (and updated) when a new stored structure is generated. As a result, these models enforce the correct functional relationship between types and tokens. In contrast, nearly all parametric statistical methods which are used in psychology are based on linear models with random error terms, and, therefore, cannot represent this functional relationship. Even more powerful generalizations of these tools (e.g., hierarchical and mixed–effect models) are conservative in this respect.
is some frequency threshold $\theta$ above which the retrieval route will dominate, and below which the compositional route will be used. Each unit in the model (i.e., word or morpheme) has an associated resting activation. The quantity $A$ is interpreted as an approximation to the resting activation of a morpheme, and is computed by summing the frequencies of all forms which use that particular morpheme and have a frequency less than the parsing threshold $\theta$. In other words, $A$ is the total frequency of all parsed forms which use a particular morpheme. Note that this definition captures the intuition that each form should count exclusively as evidence either for parsing or for retrieval. As discussed in Section 2.4, Fragment Grammars share this assumption, but the exemplar–based approaches studied in this thesis do not.

Hay and Baayen (2002) call into question the empirical adequacy of the assumption that $A$ can be computed based on a fixed frequency threshold, $\theta$, for all affixes. Adopting Hay’s relative frequency hypothesis (RFH; discussed in more detail below), they argue that $\theta$ should be related to the ratio of the whole–form frequency and the base frequency of words which use an affix. Furthermore, they show that a recent version of Baayen’s dual–route, activation–based, race model, known as MATCHCHECK (Baayen and Schreuder, 2000; Baayen et al., 2000), is consistent with the RFH, and produces parsability decisions which depend on these ratios, rather than than a fixed threshold $\theta$.

The Relative Frequency Hypothesis

The preceding sections emphasized that parsability depends on a complex interaction between the frequencies of words and the frequencies of their parts. One quantitative theory of this interaction is Hay’s relative frequency hypothesis (RFH) (Hay, 2001). The RFH proposes that whether a word is composed or retrieved depends on the ratio between the frequency of the base and the frequency of the whole form. When the ratio of base to whole–form frequency is low, the form is likely to be stored and retrieved whole, and whole–form frequency is likely to be a good predictor of processing latencies. Conversely, when this ratio is high, the form is likely to be composed, and base (token) frequencies are likely to be predictive of processing time variance.

Critically, Hay argues that although many researchers have stressed the importance of absolute frequencies, when examined in detail, their models actually predict that parsability depends on the ratio between whole–form and part–form frequencies (Hay, 2003, p. 60).

While the proponents of many of these models emphasize the role of surface frequency, examination of the models themselves reveals that they predict an interaction between the surface frequency of the complex form, and the frequency of the parts. Maximally decomposable forms should be those which are much less frequent than the parts they contain. Non-decomposable forms should be those which are more frequent than the parts they contain.

Hay highlights two important corollaries of the RFH. First, whether a word is parsed or retrieved as a whole is a property that should vary from word to word depending on the precise distribution of the word and its parts. Second, the baseline rate of parsing for an affix should vary from affix to affix, depending on the proportion of forms which contain that affix and are parsed. Affixes which are used in many parsed forms tend to favor more parsing as a baseline, while affixes used in mostly retrieved forms, tend to prefer to be retrieved.

Hay (2001) reports the results of an experimental study supporting the RFH. Participants were asked to rate the complexity of various English derived word forms. The study matched words using identical affixes which differed in whether their whole–form frequency was greater than or less

---

17 Di Sciullo and Williams (1987) make a similar point.
than their base frequency. Hay found that words for which the base frequency was greater than the whole-form frequency were rated as more complex than the corresponding matched words for which the opposite frequency relation held.

**Parsability and Productivity**

In my analyses of the English past tense in Chapter 5, I showed how FG could provide an account of both the pattern of processing latencies which arise in experimental tasks such as speeded naming and lexical decision (i.e., parsability), as well as the ability of the regular rule to generalize to completely novel stems (i.e. productivity). Key to unifying parsability and productivity was the assumption that a single kind of representation underlies both composition of _existing_ forms and generalization of _novel_ forms—namely, partial computations involving a variable. While this assumption (which is implicit in much work in morphology) is natural in compositional accounts of linguistic structure and leads to parsimonious theories, it is important to recognize that it is a hypothesis. There is a logically possible theory where novel generalization and composition of existing forms are handled by fundamentally different mechanisms.

Unfortunately, for practical reasons, it is very difficult to directly test this hypothesis quantitatively (and with wide coverage). On one hand, such a test would require data on the productivity of a reasonably large number of affixes (e.g., _wug_ tests or naturalness ratings). Databases of affix generalization data do not yet exist. On the other hand, such a test would also require measures of parsability for a number of affixes. However, the parsability of an affix is not a directly measurable quantity. In experimental studies, composition or retrieval of existing forms is inferred from patterns of processing latencies using frequency effect analyses or priming/interference manipulations. There is no way, for example, to directly assess whether some particular word has been composed or retrieved whole by an experimental subject. As a result of these difficulties, there are very few studies which attempt to directly correlate productivity and parsability for derivational affixes. Instead, researchers have used indirect methods to examine the question.

For example, Hay and Baayen (2002) investigated the relationship between parsing and Baayen’s measures of productivity: \( P \) and \( P^\ast \). As described in Section 6.3.3, these are corpus-based measures of the morphological productivity of individual affixes. To generate parsability scores for affixes, Hay and Baayen used MATCHCHECK—a dual-route, activation-based model of morphological processing designed to predict processing latencies (see Baayen and Schreuder, 2000; Baayen et al., 2000). Hay and Baayen first show that MATCHCHECK produces parsability predictions for individual affixes which are consistent with the relative frequency hypothesis. In particular, in MATCHCHECK, whether a form is retrieved (or not) is an (increasing) function of the ratio of whole-form to base frequency for that form. Hay and Baayen then show that the number (and proportion) of forms for a given affix which were parsed by MATCHCHECK correlated highly with \( P \) and \( P^\ast \).

These results must be interpreted with caution. \( P \) and \( P^\ast \) are corpus-based measures of productivity, rather than direct measures of subjects’ willingness to generalize particular affixes. Likewise, although MATCHCHECK has been evaluated against some processing datasets (e.g., Baayen and Schreuder, 2000), it is still a theoretical model, and therefore the parsability scores it produces are only indirectly linked to processing data. Nevertheless, this study contributes converging evidence to the widely-held assumption that similar mechanisms underlie parsability and productivity.

---

18 Specifically, they showed that \( P \) correlates highly with the _proportion_ of words which are parsed by MATCHCHECK, while \( P^\ast \) correlates highly with the _absolute number_ of forms which are parsed by MATCHCHECK. This follows from the fact that \( P \) is an estimate of the probability that a given affix will generate a novel form, while \( P^\ast \) is an estimate of the relative productivity of different affixes.
Parsability and Transparency

Numerous experimental studies have shown that parsability is also correlated with both phonological and semantic transparency. Unsurprisingly, the less transparent an affix, the less likely it is to be parsed. For example, MacKay (1978) found that the latency with which speakers produced a derived word (e.g., *decision*) from a base (e.g., *decide*) is related to the complexity of the phonological change. Bradley (1980) demonstrated that less transparent affixes (e.g., *-ion*) show a whole–form frequency effect while more transparent affixes (e.g., *-ness*) show a base frequency effect. Using a lexical decision task, Vannest and Boland (1999) found that less phonologically transparent affixes (i.e., *level 1* affixes—see Section 6.3.5) showed a larger effect of whole–form frequency, while more transparent affixes showed a larger effect of stem frequencies, providing evidence that the latter are decomposed.

Tapio (2008) argues that the relationship between transparency and parsability is not limited to changes at the phonological level but extends even to low–level phonetic structure. A wealth of prior evidence demonstrates that segments which lie at syntactic constituent boundaries tend to take longer to pronounce than segments which do not. Tapio (2008) showed that boundary effects also appear within words: Segments at boundaries within composed words take longer to pronounce than segments within retrieved words. This fact may shed some light on the long term dynamics of phonological change: Over time, the initial segments of affixes which are typically stored will weaken and be susceptible to changes which may be later phonemicized (i.e., grammaticalized by the phonological system).\footnote{Jurafsky et al. (2002) showed that another kind of lexical structure—polysemy—also shows frequency driven variation between forms. They found that high–frequency senses of the polysemous word *to* show greater levels of segmental reduction than low–frequency senses.}

Lexical decision experiments have also found a correlation between parsability and semantic transparency. Using primed lexical decision, Feldman and Soltano (1999) showed that semantically opaque affixed forms behave more like unrelated words, while semantically transparent forms show more robust priming. Marslen-Wilson et al. (1994) found that semantically transparent forms prime relatives independently of phonological transparency, while semantically opaque forms do not.

Transparency and parsability are also correlated in metalinguistic segmentation tasks. Wheeler and Schumsky (1980) asked subjects to provide segmentations for morphologically complex words. They found that less transparent (and less productive) affixes were often not segmented, while more transparent affixes were segmented at higher rates. Derwing and Baker (1979) found that subjects’ judgments as to whether a derived word “comes from” a base were predicted by the semantic and phonological similarity of the base and derived word (as rated by a separate group of subjects), with semantic similarity having a stronger effect.

Hay discusses the relationship between transparency and parsability within the framework of her *relative frequency hypothesis* (RFH; see, Hay, 2003, and Section 6.3.4). Hay notes that forms which are less parsable (i.e., forms whose frequency is greater than the frequency of their base) should be stored and therefore should be able to accumulate phonological and semantic irregularities. This argument is similar to the arguments made by Pinker and Prince (1988), Burzio (2000), and others, that for the English past tense, irregulars have more regular phonotactics than regulars. In a dictionary study, Hay (2001) found that those derived forms which are more frequent than their bases showed greater evidence of semantic drift—the accumulation of semantic non–transparencies. Further support for this hypothesis comes from experimental and corpus–based evidence demonstrating that highly–decomposable affixes are associated with low–probability phonotactic junctures and that these words tend to have fewer polysemous senses (Hay, 2003).

Overall, consistent with my discussions throughout the chapter, parsability is found to be closely related to, and correlated with, transparency.
6.3.5 Ordering

As noted earlier, a fundamental property of derivational morphology is that it can give rise to multiply–embedded hierarchical structures, producing words with several affixes, (e.g., disagree-ability). The number of such affix sequences which are attested, however, is much smaller than the number of logically possible sequences (even when categorial restrictions are taken into account). Therefore, a long–standing theoretical problem in derivational morphology is accounting for which sequences of affixes are possible and which are not.

One very influential theory of affix ordering is known as the level–ordering generalization (LOG). LOG has its origins in Chomsky and Halle’s detailed treatment of English phonology in The Sound Pattern of English (Chomsky and Halle, 1968). As noted above, some English affixes cause changes to the phonology of the bases to which they join (e.g., the stress shift and consonant sibilantization triggered by -ity in atomic/atomicity). To account for this fact, Chomsky and Halle (1968) proposed that different English affixes create different kinds of boundaries within a word. Transparent affixes, such as -ness create word boundaries. Non–transparent affixes, such as -ity create formative boundaries. Phenomena such as stress–shift can be triggered across formative boundaries, but not word boundaries. The LOG hypothesis results from the observation, first made by Siegel (1974), that the distinction between word and formative boundary affixation also predicts the ordering of affixes. In particular, when both formative–boundary affixes and word–boundary affixes occur in a single word, the former generally appear before the latter. Siegel also noted that word–boundary affixes were more productive than formative–boundary affixes and that only formative–boundary affixes combined with bound stems.

To explain these facts, Siegel (1974) proposed a theory of the lexicon which divided the process of word generation into a number of ordered levels or strata. Formative–boundary affixes are joined with bases in the first level of the lexicon; only after derivation in this stratum is complete, are word–boundary affixes concatenated onto the resulting forms in the second level of the lexicon. Other properties, such as the availability of different affixes for productive generalization, can also be associated with strata. Note that level–ordering is an example of the representational approach in which differences in productivity are taken as evidence for different generative submodules (see Section 1.3).

Level–ordering (or stratal) theories of the lexicon reached their most complete state of development in lexical phonology (Halle and Mohanan, 1985; Kiparsky, 1982a,b,c, 1985; Mohanan, 1986). Lexical phonologists proposed several different architectures which varied in number of levels and the assignment of morphological processes to levels. For example, Kiparsky proposed three levels of morphological processes. The first level handled formative–boundary derivational affixes (e.g., -ity) and irregular inflections such as the irregular past tense forms of English verbs; the second level handled word–boundary derivational affixes (e.g., -ness) and compounding; the third level handled regular inflection such as the English past tense +/d/ rule. Halle and Mohanan (1985) and Mohanan (1986) proposed a theory with four levels which moved compounding to level 3 and regular inflection into level 4.

However, despite their wide–spread use, level–ordering theories suffer from several empirical problems. First, under these theories, some affixes must be assigned to multiple levels. For example, consider the affix -able, attached to the verb compare. There are two acceptable pronunciations of the word, one which includes a stress shift and vowel reduction (i.e., ‘comparable’) and one which does not (i.e., com’parable) (Plag, 1996). The former pronunciation indicates that the affix must reside in an early level of the lexicon, while the latter indicates that it should reside in a later level.

\[20\] Note that stratal theories of the lexicon, including Siegel’s, are lexicist: They assume that the lexicon is a generative component of the language system. Thus, the use of the term lexicon here refers to this component and not, for example, the inventory of stored items.
Giegerich (1999) provides extensive examples of such multiple-level assignment problems and argues that they represent a pervasive and decisive challenge to (affix-centered) stratal theories.

Another problem with level-ordering is that it gives rise to bracketing paradoxes (Aronoff and Sridhar, 1983; Strauss, 1982; Williams, 1981). For example, the suffix -ity, which attaches to adjectives, is usually assigned to the first lexical level, while the prefix un-, which also selects for adjectives, is usually assigned to a higher level. Consider the word ungrammaticality. Since -ity has triggered a stress shift, grammaticality must be output by a level before un- is prefixed, which gives rise to the bracketing: [un- [grammatical -ity]]. However, this violates the selectional restrictions for un-, since grammaticality is a noun, and, by hypothesis, un- only attaches to adjective (or verbs). However, the other possibility [[un- [grammatical] -ity], violates the assumption that un- is assigned to a higher level than -ity. Thus, either bracketing violates a prediction of the LOG.

There is another more serious problem with level-ordering: The theory both under- and overgenerates. Undergeneration occurs when the theory rules out affix combinations which are, in fact, attested. In many attested English words, a lower-level affix appears after a level higher-level affix. For example, many English words involve the affix combinations: -ist+ic, -able+ity, and -ment+al (e.g., Aronoff and Sridhar, 1983; Fabb, 1988; Giegerich, 1999; Hay, 2002). In all three cases, the outermost affix triggers stress shift, and, therefore, must be classified as level 1 affixes, while the innermost affix does not trigger a stress shift and therefore appears to be a level 2 affix. Even more problematic is that fact that level ordering massively overgenerates, predicting many combinations of affixes that are never attested in English. In an influential study of 43 English affixes, Fabb (1988) showed that there are 663 pairs that are possible when categorial restrictions are taken into account. Taking into account several additional phonological and semantic selectional restrictions reduces this number to 614, and the addition of level-ordering restrictions further reduces this number to 459. However, Fabb (1988) showed that only 50 of these 459 combinations are actually attested. The LOG does not rule out enough possibilities.

These problems with LOG have led researchers to gradually retreat from the theory. The following sections describe two alternative theories which have been proposed in the literature: selectional-restriction based theories and complexity-base ordering.

Selectional-Restriction Based Theories An alternate approach to LOG is to reject the idea of lexical levels altogether and, instead, propose that ordering effects are an epiphenomenon of a large number of affix-specific selectional restrictions. In his paper criticizing the overgeneration of level-ordering theories, Fabb (1988) proposes that affixes can be categorized into four classes, each of which has different affix-specific selectional restrictions. Class I consists of those suffixes which never attach to a word with an existing affix (e.g., Fabb claims that -ion and -ary are in this class).

That is, Class I affixes select for morphologically simple bases. Class II consists of a set of affixes which can attach outside of a set of specific affixes (e.g., Fabb claims that deadjectival -ary only attaches to -ion). Class III are those fully productive affixes (e.g., -ness) which can attach to any bases rather than affixes, as in the traditional approach. The intuition behind this approach is that certain affixes (e.g., -ity) seem to be selected by bases ending in certain other suffixes (e.g., -able). Organizing lexical levels around bases accounts for such facts.

21It is also arguable that the semantics of this bracketing are incorrect.

22In fact, most examples of bracketing paradoxes in the literature concern cases of interaction between prefixes and suffixes. One possible solution is to assume that level ordering applies differently to the two kinds of affixes (Strauss, 1982). This solution, however, still leaves problems for any theory of how morphological composition gives rise to semantic composition.

23A third approach is provided by Giegerich (1999), who argues that lexical strata should be organized around bases rather than affixes, as in the traditional approach. The intuition behind this approach is that certain affixes (e.g., -ity) seem to be selected by bases ending in certain other suffixes (e.g., -able). Organizing lexical levels around bases accounts for such facts.
word. Class IV contains the residue of problematic cases which do not seem to fit into any of the earlier classes (6 such cases).

While Fabb’s selectional theory diverges from LOG by dispensing with the notion of lexical strata, it still attempts to account for affix ordering with a small number of classes. In contrast, Plag (2002, 2003) develops a selectional–restriction based theory of ordering which involves a large number of different, intersecting selectional restrictions, some of which are borne by only a single affix. Plag (1996) criticizes Fabb’s classification scheme, pointing out many examples where it both over- and undergenerates like the LOG. In Plag’s theory, these issues are resolved by positing selectional restrictions with morphological, phonological, semantic, and even pragmatic content.

### Complexity–Based Ordering

Another theory of affix ordering is Hay’s complexity–based ordering (CBO) which builds on her relative frequency hypothesis (RFH) (Hay, 2002, 2003; Hay and Plag, 2004) (see Section 6.3.4). Hay argues that affix ordering can be explained by an appeal to affix parsability. In short, CBO proposes that more parsable affixes (e.g., -ness) should appear outside of less parsable affixes (e.g., -ive). CBO is similar to stratal theories, and different from selectional–restriction based theories, in that it is a global theory of ordering. It posits that a global property of affixes—their parsability (or productivity)—should predict which affixes appear earlier or later in sequences.

Hay stresses that, because CBO is built on the relative frequency hypothesis, it allows for variability between words which can lead to apparently exceptional instances of affix ordering. For example, Plag (2002) points out that some pairs of affixes appear in both possible orders, for example, -ion and -al in sensational and colonialization. He suggests that such cases may be problematic for CBO, because on average either -ion is more parsable than -al or vice versa. Recall, however, that RFH allows the parsability of an affix to vary on a word–by–word basis, depending on whether a word is more or less frequent than its base. Hay and Plag (2004) show that the cases where -ion attaches after -al are instances where the base to which -ion attaches (e.g., colonial) is more frequent than its own base (i.e., colony). In other words, these exceptions are explained by word–by–word differences in parsability—words such as colonial which are not very parsable are more likely to take additional, productive suffixes.

CBO correlates well with earlier theories, such as the level–ordering generalization and Fabb’s theory, in those cases where these theories make the correct predictions. Hay (2002) shows that parsing rates for individual affixes (given in Hay and Baayen, 2002, see Section 6.3.4.) are higher for level 2 affixes than level 1 affixes, and that CBO is also consistent with Fabb’s categories.

However, CBO goes beyond these earlier theories, making correct predictions in several difficult cases. For example, recall that -mental is an example of an affix combination that violates LOG. According to CBO, -al should be able to attach to words ending in -ment precisely when there is a strong boundary between -ment and its base. For example, CBO predicts that a frequent word, like government, whose whole–form frequency is greater than its base frequency, should be able to affix -al. In contrast, a word like wonderment whose base frequency is greater than whole–form frequency, will likely be parsed, and therefore should not be able to affix -al. In an acceptability rating task, Hay (2002) found that -ment forms with high whole–form to base frequency ratios (e.g., government) are more acceptable with -al than forms with low whole–form to base frequency ratios (e.g., wonderment).

Hay and Plag (2004) further tested this hypothesis by examining a set of 15 English suffixes. Using the set of attested suffix pairs in a corpus of English words, Hay and Plag construct an affix

---

24Fabb (1988) also discusses the verbal affix -en separately, pointing out the fact, mentioned earlier, that it has rather tight phonological selectional restrictions.
hierarchy such as the one represented by the graph shown in Figure 6.1 (taken from Hay and Plag, 2004).

![An Affix Ordering Graph](image.png)

**Figure 6.1: An Affix Ordering Graph:** This graph compactly represents the empirical orderings of affixes found in the sample studied in (Hay and Plag, 2004). Affixes lower in the hierarchy tend to come after affixes higher in the hierarchy.

In the graph shown in 6.1, affixes are placed as early in the hierarchy as possible, without violating an ordering which is attested in the sample corpus. Hay and Plag (2004) measured the parsability of each affix using a composite quantity which combines type and token parsing ratios (i.e., whole-form to base ratios computed over both types and tokens) and the productivity scores for affixes (i.e., values for Baayen’s $P$, reported in Hay and Baayen, 2002). Hay and Plag showed that these parsability scores predicted the ordering shown in Figure 6.1, with more parsable affixes appearing later in the hierarchy. Importantly, the hierarchy represented by the graph in Figure 6.1 is derived from actual combinations of suffixes attested in a corpus, while the parsability scores are computed from the affix and base frequencies. There is no *a priori* reason why one should predict the other, therefore, this result provides support for the CBO.

Plag and Baayen (2009) further develop the graph-theoretic tools used in Hay and Plag (2004), and apply them to a larger set of English suffixes. Their results provide further support for CBO (see below). Zirkel (2010) applied these techniques to English prefixation, showing that parsability is correlated with ordering in that domain, and Manova (2010) showed that this correlation also holds for Bulgarian suffixation.

It should be noted that the measures of parsability and productivity in these studies are corpus-based. A more ideal evaluation of the CBO would correlate position in the ordering hierarchy directly with native speaker parsability data (e.g., judgment data, reaction time latencies, etc.). Nevertheless, the parsability and productivity measures used in these studies were motivated, developed, and computed without any reference to ordering phenomena. Thus, the fact that they predict
ordering is surprising and can be taken as evidence in favor of the relationship between ordering and parsability predicted by CBO.

**The Mean Rank Statistic** Plag and Baayen (2009) extend and formalize the graph-theoretic approach to ordering presented in Hay and Plag (2004), and develop a summary measure of an affix ordering which they call its *mean rank*. In the next chapter, I will make use of the mean ranks of various suffixes, and, therefore, I describe this summary statistic in more detail here.

The starting point for computing the mean rank of a suffix is the *adjacency matrix* for a given set of affixes. In Plag and Baayen (2009), the adjacency matrix is used to represent the presence or absence of particular affix combinations in some corpus. Figure 6.2 shows the adjacency matrix from the dataset studied by Plag and Baayen (2009).

![Figure 6.2: Adjacency Matrix over Affixes from Plag and Baayen (2009):](image)

Each row/column combination that contains a 1 in this matrix signifies that the sequence $a_{\text{row}} \prec a_{\text{col}}$ was attested in the corpus examined by Plag and Baayen (2009).

Each 1 in the matrix signifies that the combination of affixes specified by that row and column was attested in the corpus (in row-column order). Note that affixes appears in the same order in both rows and columns (i.e., the adjacency matrix preserves the diagonal). As previously noted, CBO predicts that affixes should be ordered according to parsability, and therefore that if $a_1$ precedes (notated as: $a_1 \prec a_2$), the opposite case ($a_2 \prec a_1$) should not be attested. If this is true, then it should be possible to reorder the rows and columns of the matrix in such a way as to place all the 1's above the diagonal of the matrix.

In practice, this turns out to be impossible. There are cases of *cycles* in the set of adjacency relations (e.g., cases where $a_1 \prec a_2$ and $a_2 \prec a_1$). Because of this, it is not possible to reassign indices to the affixes in the matrix shown in Figure 6.2 in such a way that *every* 1 appears above the diagonal. As an alternative, Plag and Baayen (2009) seek an ordering of rows and columns which *minimizes* the number of such cycles. One such ordering of rows and columns from Plag and Baayen (2009) is shown in Figure 6.3.
Affixes which are ordered low on x- and y-axes, and therefore have low rank, tend to appear before many other suffixes, whereas suffixes with high rank tend to appear after many other suffixes. Plag and Baayen (2009) show that the minimal number of violations possible for their data set is 10. They also show that such a low number of violations of acyclicity is highly unlikely to be due to chance alone, thereby giving strong support to the idea that affixes can be ordered in a hierarchy—a point strongly consistent with CBO.

Note that it is possible for there to be many reassignments of affixes to indices that are minimal. For example, any exchange of two indices which does not move a 1 below the diagonal in Figure 6.3 will also be a consistent minimal ordering. Because of this problem, Plag and Baayen (2009) average the rank of each affix over a large number of minimal orderings and, thus, define the mean rank of each affix. Intuitively, the mean rank of an affix can be thought of as the average number of other affixes which appear before the affix in some corpus of words, taking into account transitivity. For example, if a₁ appears after a₂ but a₂ appears after many other affixes, the greater number of affixes after which a₂ appears will influence the ranking for a₁ as well.

Although I will make use of the mean rank statistic in the next chapter, it should be noted that it has several several limitations. First, it does not take into account differences between affixes

---

25 Note there are a large number of row/column orderings. In particular, there are N! such orderings where N is the number of rows. Searching for the minimal number of acyclicity violations—a problem known as the feedback arc set problem—cannot be solved efficiently (in polynomial time) by any known algorithm. Therefore the number 10 for the minimal number of violations from Plag and Baayen (2009) is an empirical estimate derived by running a heuristic search algorithm, known to work well in practice.
which do not violate acyclicity. Suppose that in some corpus $a_1$ appears after affixes \{a_3, a_4, a_5\}, but $a_2$ only appears after affix $a_3$, and does not appear before affixes \{a_4, a_5\}. Under these conditions, the rank of $a_1$ and $a_2$ with respect to one another is unspecified in any particular ordering. Intuitively, this seems incorrect: $a_1$ seems to appear later in words “more often” in some sense. Second, the mean rank is computed from an adjacency matrix which is not weighted by frequency. Thus, highly infrequent combinations (e.g., \textit{-nessless: happinessless}) are given the same weight as more frequent combinations (e.g., \textit{-lessness: cluelessness}).

Despite these limitations the mean rank statistic still captures a great deal of information about affix ordering and provides a useful summary statistic with which to correlate other measures. For this reason, I will make use of the values of this statistic reported by Plag and Baayen (2009) in the next chapter.

**Complexity–Based Ordering and Selectional Restrictions** CBO predicts that ordering should be correlated with parsability, but it does not preclude that additional phonological, semantic, morphological or other selectional restrictions might still apply to constrain the combination of affixes. Hay and Plag (2004) explore the relationship between the global predictions of the CBO and the local predictions of selectional–restriction based theories. For the set of 15 English affixes described above, they determined both the combinations that would be allowed by a detailed specification of the selectional restrictions for each affix and the combinations that would be allowed under the CBO. Both sets of predictions were compared to the set of attested forms. Their findings were surprising. For a small number of affix combinations, the two theories were complementary: several ordered pairs of affixes which are allowed under CBO were correctly ruled out by selectional restrictions. However, the vast majority of combinations are either ruled–out, or allowed, by both theories. Thus, the two theories are largely redundant. This redundancy is another illustration of the correlation between selectivity, transparency, parsability, and productivity that has been discussed throughout the chapter.

### 6.4 Conclusion

In this chapter, I have reviewed the relevant literature on English derivational morphology focusing on a group of correlated properties of the system: productivity, selectional structure, parsability, and ordering. The correlation between these properties is an example of what might be called a learning conspiracy—several different kinds of evidence available to the language learner all conspire to make similar, redundant predictions about the productivity of word formation processes.

A conspiracy like this suggests that these apparently unrelated phenomena may be linked by a hidden cause. In the next two chapters, I will argue that the underlying factor unifying these phenomena is the fact that the system must infer the unobservable levels of productivity of different generative processes from the observable data.
Chapter 7

English Derivational Morphology: Simulations

7.1 Introduction

This Chapter reports simulation results examining four phenomena from English derivational morphology: productivity, parsability, ordering, and base-driven selectional restrictions, for a set of 338 English suffixes. Although there is a substantial body of work focusing on these phenomena (see Chapter 6), quantitative modeling in this domain remains difficult for several reasons. First, as I will describe in the next section, construction of an appropriate training corpus presents a number of challenges. Second, and more critically, for productivity and parsability, there are no broad-coverage datasets available to directly test model predictions. Because, in English, there is a clear notion of the “correct” past tense form of most verbs, the performance of the five models studied in the thesis could be measured directly in Chapter 5. English derivational morphology, by contrast, offers few such clear-cut cases. Instead, it presents wide variability in the acceptability of different affixed forms: including sharp preferences (e.g., furiousness v. *furiosity), weaker preferences (e.g., expectation/?expectedness), and many doublets (e.g., tolerance v. toleration) (e.g., Kiparsky, 1982b). The appropriate data to evaluate such a system—for example, databases of naturalness judgments or wug-tests—do not yet exist.

The analyses presented in this chapter can be grouped into three categories. First, a number of the analyses are exploratory: They are designed compare and contrast the behavior of the five models studied in the thesis and generate predictions for future empirical studies. Second, several of the analyses test the convergence of theoretical predictions between the models studied here and theories from the literature. In particular, I will examine the degree to which the present models make similar predictions to Baayen’s theory of productivity, as represented by his measures $P$ and $P^*$, Hay’s relative frequency hypothesis (RFH), and Hay’s complexity-based ordering (CBO). Third, in several cases, empirical data is available for the direct evaluation of model predictions. This is especially true in the case of affix-ordering phenomena, which are directly observable in corpora of English words, and base-driven selectional restrictions, which have been the subject of a number of experimental studies (Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978; Cutler, 1980).

In the first section of this chapter, I explain the construction of the training data used for these simulations, highlighting the difficulties involved in providing morphological analyses for large numbers of English words. In the next section of the chapter, I turn the the related issues of productivity and parsability. I present an exploration of the distribution of productivity scores

165
for different affixes, and, then, show that the scores predicted by Fragment Grammars are highly correlated with Baayen’s $P$ and $P^*$, demonstrating convergence between these two differently motivated theories. After giving some examples of productive affixes learned by each theory (with an eye towards future experimental work), I analyze predictions about parsability for existing forms. I show that three of the models studied in the thesis (Fragment Grammars and the two exemplar–based models) are all consistent with Hay’s RFH. I also explore an important difference between Fragment Grammars and the two exemplar–based models: Fragment Grammars demonstrate attractor–like behavior, preferring to treat forms as nearly always composed or nearly always retrieved. I conclude the sections on productivity and parsability with a general discussion which ties these results back to the literature and explains the relationship between the RFH and Fragment Grammars.

In the next section, I turn to the phenomena of affix ordering and base–driven selectional restrictions. I first show that Fragment Grammars are the model most consistent with the main prediction of CBO: More productive affixes tend to appear later in affix sequences. I then evaluate the performance of the models against empirical ordering measures reported by Plag and Baayen (2009). I show that Fragment Grammars provide the best fit to the summary mean rank statistic, but that Adaptor Grammars (full–listing) provide the best fit to the raw corpus frequencies of various affix combinations. However, I also highlight a way in which Fragment Grammars capture structure which Adaptor Grammars cannot: In some cases, Fragment Grammars learn that specific combinations of suffixes (e.g., -ate+ion, i.e., -ation) are generalizable as units. I examine this phenomenon in detail in the next section, showing that Fragment Grammars capture the exact pattern of -ity and -ness generalization which has been reported in the literature (Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978; Cutler, 1980). The other models fail to account for these data in ways which highlights several subtleties of Fragment Grammar inference. Finally, I conclude the sections on ordering and selectional restrictions with a general discussion of the phenomena, and an explanation of the relationship between Fragment Grammars and CBO.

7.2 The Training Corpus and Simulations

This section describes the underlying CFG, input representations, and training corpus used by all of the models evaluated in this chapter. A particular challenge in conducting these simulations was constructing an appropriate training sample. I will describe how this was accomplished below. First, I describe the general representational assumptions and starting CFG used by all models.

7.2.1 The Starting CFG and Input Representations

All five models used a very simple input CFG, a fragment of which is shown in Figure 7.1. This CFG was derived from the set of morphological analyses made available as part of the CELEX database (Baayen et al., 1993). The sets of lexical categories (nonterminals) and stems/affixes (terminals) are drawn from modified CELEX morphological analyses, which I describe in Section 7.2.2. This CFG gives rise to simple derivation trees like the one shown in Figure 7.2.

This system has several noteworthy properties. First, affixes are treated as heads, which take their bases as arguments. This is consistent with a number of morphological theories which treat morphological structure as constituent–based and headed (e.g., Halle and Marantz, 1993; Lieber, 1980; Selkirk, 1982; Williams, 1981). Second, as with the past tense, no attempt was made to represent any of the phonological or semantic structure associated with each affix or stem (e.g., stress shift triggered by -ity). Therefore, the units in this system are best understood as representing the abstract identity of morphemes, without their associated phonological and semantic properties.

Under this system, any combination of morphemes satisfying the categorial restrictions
\[ W \rightarrow N \]
\[ W \rightarrow V \]
\[ W \rightarrow Adj \]
\[ W \rightarrow Adv \]
\[ N \rightarrow Adj -ness \]
\[ N \rightarrow Adj -ity \]
\[ N \rightarrow dog \]
...\[ V \rightarrow N -ify \]
\[ V \rightarrow Adj -ize \]
\[ V \rightarrow re-V \]
\[ V \rightarrow agree \]
...\[ Adj \rightarrow dis- Adj \]
\[ Adj \rightarrow V -able \]
\[ Adj \rightarrow N -ic \]
\[ Adj \rightarrow tall \]
...\[ Adv \rightarrow Adj -ly \]
\[ Adv \rightarrow today \]
...
encoded by the CFG can be generated (under all five models). Therefore, as was the case with the past tense simulations, the system is massively overproductive, and the only kind of information available to constrain productivity is the distribution of morpheme co-occurrences in the input. Furthermore, the lack of constraining phonological and semantic selectional restrictions is adversarial to the problem of correctly inducing the pattern of productivity in English derivational morphology. As I discussed in Chapter 6, these additional sources of information are highly correlated with productivity, and, therefore, their inclusion would likely improve the ability of the models to induce the correct patterns of productivity and reuse.

### 7.2.2 Input Corpus

The input data for the derivational morphology simulations was derived from the CELEX database which contains a large sample of English words gathered from dictionaries and newswire (Baayen et al., 1993). CELEX provides morphological analysis information for a subset of the words in the database, and these analyses served as the kernel of the training dataset. Because of a number of problems with these CELEX analyses (described below), this kernel set of parsed words was supplemented and revised via a combination of heuristic parsing and hand-correction (described below). The resulting data set contained 338 suffixes, over 25,000 word types, and over 7.2 million word tokens.

Extending the CELEX Sample

As I discussed in earlier sections of the thesis (see e.g., Sections 1.3.3, 4.3.3, and 6.3.2), productive word-formation processes are characterized by long-tailed frequency distributions over forms. As a result, dictionaries and other word databases tend to undersample high-productivity affixes (see Evert and Lüdeling, 2001, for a discussion of this issue in German morphology). CELEX, in particular, only provides morphological analyses for words which appeared with a frequency threshold of at least 15 occurrences in 18 million. Consequently, relying solely on the analyzed portion of the CELEX database would make critical cues to productivity unavailable to the models (see, also, Hay and Baayen, 2002).

Along with the set of morphologically analyzed forms, CELEX provides a separate list of all word strings which appeared in the original samples used to construct the database. This list contains many more words and, therefore, provides a more representative sample of low-frequency, productively-derived forms. To find potentially interesting, suffixed forms, a heuristic parser (implemented in Ruby) was applied to this larger set of words.

This parser starts at the end of each word and attempts to match each of the suffixes found in the kernel set of CELEX analyses, one at a time, in frequency order. Once the parser matches a suffix, it recursively applies itself to the resulting base. This process continues until the parser cannot match any further suffixes. If the stem that results from this process is contained in the set of kernel morphological analyses, then the word is considered correctly parsed, and a tree representation of the parse is returned. The parser is conservative in that it attempts to recursively segment only

---

1. The exact size of the training dataset varied very slightly between simulations due to rounding error in the binning of types and tokens.

2. The code for this heuristic parser can be made available upon request. Note that the problem of segmenting morphological structure and identifying the underlying form of morphemes is a major research problem in its own right, from both an engineering and a scientific standpoint. The parser represents a (somewhat crude) attempt to solve this problem. A more principled solution would integrate a phonological segmentation and rule-learning model (e.g., Chan, 2008) with a model of productivity and reuse (such as Fragment Grammars). It seems likely that such integration could lead to significant improvement in the performance of both kinds of system.
those suffix combinations which give rise to categorically well-formed parses with known stems as their innermost base.\(^3\)

Many of the words in CELEX are annotated for transformations which apply to a base before attaching an affix. For example, CELEX might specify that a word, such as *curate*, loses its final *e* before appending *-or*, when deriving the word *curator*. When constructing the set of possible suffixes used by the heuristic parser, I also associated the set of such transformations with each suffix. During the suffix stripping process, these transformations were inverted, and the resulting base (e.g., with restored final *e*) was passed recursively back to the parser as input to the next stage of stripping. Thus, the heuristic parser was able to exploit systematic spelling changes (e.g., loss of final *e*, or the change of final *y* to *i* before *-ness*).

The set of heuristically-parsed forms was added to the set of morphological analyses provided by CELEX.

**Problems with the CELEX Morphological Analyses** The set of analyses in any morphological database, such as CELEX, represents an implicit theory of morphological structure. The set of analyses provided by CELEX reflect a number of theoretical decisions which are particularly problematic for its use in studying productivity and reuse. Several of these are outlined below.

1. Only bases which appear as independent words in CELEX are segmented from their affixes. In particular, bound stems are never segmented (e.g., *possible* is not segmented).

2. Certain suffix combinations are represented as single units. For example, a common suffix combination in English is *-ate+ion* (i.e., *-ation*). In many cases, there is no corresponding word ending in *-ate* for an existing word ending in *-ation* (e.g., *form*/ *formate*/ *formation*). In such cases, CELEX often represents the suffix combination *-ate+ion* as a single suffix *-ation*.

3. Certain derivational affixes exhibit quasi-paradigmatic structure (see Section 6.2) with systematic correspondences between a set of affixes across morphological categories or meanings (e.g., *-ism*/ *-ist*, *-ent*/ *-ence*). In such cases, CELEX often codes a form with one ending as the base of forms that have the other ending. For example, nouns ending in *-ence* (e.g., *competence*) are coded with the corresponding *-ent* form (i.e., *competent*) as their base, rather than deriving both *-ent* and *-ence* forms from the bound stem *compet-*.

4. Many semantically non-transparent deverbal forms are coded with verb particle constructions as their base. For example, one analysis of the word *reliable* in CELEX gives the verb-particle construction *rely on* as the base, presumably reflecting the coders intuitions that *reliable* is semantically most related to this combination.

Many of these choices are driven by, or interact with, considerations of productivity and reuse. For example, in the CELEX corpus, *-ation* was likely introduced as a single affix (in some forms) to reflect that fact that this combination is frequent, (semi)productive, and that, in some cases, an intermediate *-ate* form does not exist. However, because the goal of the present study is to examine how patterns of productivity can be inferred from the distribution of morphemes in the input, the starting state of the system should expose as much combinatorial structure as possible. Therefore, when possible, the heuristic parser was extended to segment bound stems (e.g., *possible*) and suffix combinations (e.g., *-ation*), collapse across related bases (e.g., *rely* and *rely on*), and represent families of related words as sharing the same base (e.g., *competence*/ *competent* share

\(^3\)Because I did not analyze prefixes in this study, they were not handled by the parser. However, segmented prefixes which were provided in the kernel CELEX sample were included during training.
In many cases, however, it was not possible to make such modifications automatically, and, therefore, hand correction was employed.

**Hand Correction** Because many of the problems described in the last section could not be addressed automatically, and, because the unparsed portion of CELEX is very noisy, containing many misspellings and non–words, approximately 10,000 training forms were corrected by hand from the combined set of original CELEX analyses and heuristic parses. These 10,000 forms were chosen based on whether they contained one of a number of suffixes of interest. Note, however, that the corrected forms represent less than half of the training set, so a fair amount of noise remained in the corpus.

**Forms Included in Training**

The 338 suffixes examined in these simulations were compiled from the many citations discussed in Chapter 6. A list of all suffixes which appeared in at least 10 words types (and are, therefore, less likely to reflect errors in the training corpus) can be found in Appendix C. The training set used for all simulations consisted of all forms which (i) included one of these suffixes, or (ii) included a stem which appeared with one of these suffixes (in another form).

### 7.2.3 Training

Word forms along with their frequencies were input to all of the models as trees of the following form.

\[
(W (N (Adj (V agree) -able) -ity))
\]

80 FG simulations were run on the training set, each with 105,000; 95,000; 85,000; 75,000; 65,000; 55,000; 45,000; 35,000; 25,000; and 15,000 as described in Sections 3.3–3.5. Each simulation ran for between 1 and 115 sweeps through the training corpus, depending on time available on the MIT CSAIL cluster. Results were averaged to estimate the MAP approximating PCFG as described in Section 3.5.2. The approximating PCFGs for the other models were computed directly from the input corpus as described in Section 3.3. See chapter 3 for a detailed mathematical description of the parameters for the five models and Section 3.5.6 for a description of the values used for each parameter in the simulations.

### 7.3 Simulation Results

In this section, I present analyses of simulations performed on the training corpus described in the last section. This section is divided into two parts. In the first, I describe analyses examining the related issues of productivity and parsability. For the reasons noted in the introduction, these analyses will focus on explorations of model predictions and tests of convergence with theories from the literature. In the second part of this section, I turn to affix ordering and base–driven selectional restrictions. Because these represent phenomena which are either directly observable in corpora (ordering) or have been investigated experimentally (selectional restrictions), this discussion will present several empirical evaluations, although I will continue to discuss relationships between the models and theories from the literature and compare and contrast model predictions.
7.3.1 Productivity and Parsability

As I discussed in Section 6.3.4, many morphological theories—including all of the models studied in this thesis—unify the phenomena of generalization of novel forms (i.e., productivity) and composition of existing forms (i.e., parsability) by assuming that both kinds of forms are computed using representations that include abstract variables. In this section, I examine the predictions made by each model for these related issues.

First, I compare and contrast the five models by examining the distribution over productivity scores that each predicts for a large number of English suffixes. Second, I explore the degree to which each of the models makes similar predictions to Baayen’s theory of productivity, showing that Fragment Grammar is highly correlated with his measures $P$ and $P^*$. Third, to ground intuitions and generate predictions for future empirical work, I give examples of highly productive suffixes for each model. Fourth, I turn to the question of parsability, and show that 3 of the models studied in this thesis (FG and the two exemplar-based approaches) are consistent with Hay’s relative frequency hypothesis (RFH). I also discuss a crucial difference between Fragment Grammars and the two exemplar-based models: Fragment Grammars predict a strong preference for parsing or retrieval rather than an intermediate mixture of the two. Finally, I discuss these results, focusing especially on the relationship between Fragment Grammars and the RFH.

The Distribution of Productivity Scores

In Section 6.3.3, I discussed the scalar view of productivity (e.g., Aronoff and Fudeman, 2005; Bauer, 1992, 2005; Hay and Baayen, 2005): the idea that productivity is an inherent, gradient property of morphological representations. All of the models studied in this thesis adopt this view, and, in this section, I compare the distribution in productivity scores over suffixes predicted by each model, when trained on a corpus of English words.

Section 6.3.3 discussed two conditional probabilities which can be used to measure productivity. The first, $\hat{\Pi}_p$, represents the probability that a novel stem will be sampled, conditioned on a particular suffix, $p$: $P(N = T | P = p)$. This can be understood as a measure of the rate at which a suffix is adding new words (relative to the total number of words using that suffix)—what Baayen calls the potential productivity and measures with $P$. The second, $\Pi_p^*$, is the conditional probability of a suffix given that a form is novel: $P(P = p | N = T)$. This can be understood as the proportion of the overall growth in vocabulary (for all suffixes) which is due to a particular suffix, $p$—what Baayen calls the expanding productivity and measures with $P^*$. Recall that $\Pi_p^*$ is proportional to the joint probability of a novel stem and a suffix: $\Pi_p = P(N = T, P = p)$. Therefore, when comparing between suffixes, we can use $\Pi_p$ in place of $\Pi_p^*$.

These productivity scores were approximated by computing the probability that each model assigned to the combination of a suffix and a novel wug–stem, which was not present in the training set. The total probability of such a combination is an approximation to the joint probability of the suffix and a novel stem: $\Pi_p$. Values for $\Pi_p$ were computed by normalizing the $\Pi_p$ values by the total (marginal) probability of each suffix across all stems. Since, $\Pi_p^*$ is proportional to $\Pi_p$ (when comparing across suffixes), I use the latter in place of the former.

---

4This is an approximation because the probability of the novel wug–stem was the same across all affixes, meaning that the probability of “novelty” was assumed to be a small, fixed constant. As a consequence, differences in productivity scores in this section only take into account the probability that each affix can compose with a novel stem, not the probability of novel stems themselves. Computing an exact value for quantities such as $\Pi_p = P(N = T | P = p)$ would require taking into account the precise probability of all of the stems which are available to be composed with each suffix. This, in turn, would require taking into account diverse (and unmodeled) phonological, syntactic, semantic, pragmatic, stylistic, and historical properties of the (infinite) set of available stems.
Figure 7.3 plots the productivity of a number of English suffixes as predicted by the five models studied in this thesis. To reduce the effect of noise in the training data, I included only those suffixes which occurred in more than ten word types in the input corpus. The $x$–axis represents the (log) potential productivity of each suffix (i.e., $\log(\hat{\Pi}_p)$), and the $y$–axis represents the (log) expanding productivity using the unnormalized score $\log(\Pi_p)$.

By comparing the plots for the five models, we can explore the range of productivity scores which each model assigns to various suffixes (variances for $\hat{\Pi}_p$ and $\Pi_p$ are included in Table 7.1). All of the models produce a fair amount of variability with respect to expanding productivity ($y$–axis). This score measures the relative rate at which different suffixes contribute new words to the vocabulary. Because, for any suffix, all models allow productive computation, they all show variability on this dimension.

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expanding Productivity (y–axis)</td>
<td>4.434</td>
<td>3.029</td>
<td>1.818</td>
<td>3.873</td>
<td>3.105</td>
</tr>
<tr>
<td>Potential Productivity (x–axis)</td>
<td>5.408</td>
<td>0.0</td>
<td>2.491</td>
<td>0.641</td>
<td>0.036</td>
</tr>
</tbody>
</table>

Table 7.1: Variances in Productivity Measures: The variances in the two productivity measures plotted in Figure 7.3 for each model.

This situation is quite different for the potential productivity ($x$–axis) which measures the rate at which individual suffixes add new words with respect to the set of total words using the suffix. FG (inference–based) produces a broad spread in this dimension, indicating that it has learned that some suffixes are nearly always parsed (high productivity), while others are nearly always stored together with their base (low productivity). AG (full–listing) produces some variability in this dimension, but much less than FG (see Table 7.1). Because FG can explicitly represent productive computations in stored fragments, it is unsurprising that FG assigns higher productivity to some suffixes than AG. Surprisingly, however, FG also assigns lower absolute productivity scores (min value: $-49.28$) to some forms than any score assigned by the AG model (min value: $-47.89$). Because AG is a full–listing model, which stores all suffixes together with their bases, it might seem that it should produce the lowest values for the productivity of individual suffixes. However, these results show that based on the distribution of forms in the data, FG is able to infer more extreme values for productivity that the full–listing model.

The remaining models produce much lower variability in potential productivity. MDPCFG (full–parsing) produces no variability in this dimension. This is a natural consequence of the fact that in this model all computation is productive. The two exemplar–based models both show small amounts of variability in potential productivity, with DOP1 producing slightly more variability than GDMN.

To summarize, all of the models produce variability in the rates in which they predict suffixes to be used in forming novel words (expanding productivity). However, only FG (inference–based) and AG produce substantial variability in the rates at which they predict that new words will be produced given a particular suffix (potential productivity), and FG shows much higher range and variability in this dimension. Therefore, although all models adopt a scalar view of productivity, FG produces the largest variability on the scale, inferring highly productive suffixes, highly unproductive suffixes, and intermediate cases.

---

5On average FG (inference–based) produces the lowest absolute scores on this dimension, indicating it is assigning low probability to novel forms. This may be a result of FG fitting the dataset more tightly than the other models, and, therefore, placing less probability mass overall on new forms.
Figure 7.3: **Distributions of Productivity Scores**: This plot shows the distribution of potential productivity ($\Pi_p$; $x$-axis) and expanding productivity ($\Pi_p$; $y$-axis) for suffixes for each model.
Correlation with Productivity Scores from Hay and Baayen (2002)

In Section 6.3.3, I discussed Baayen’s estimators $P^*$ and $P$, which can be understood as estimators of expanding (i.e., $\Pi^*_p$) and potential (i.e., $\hat{\Pi}_p$) productivity (respectively). As I noted in Section 6.3.3, there are a number of reasons to believe that these statistics provide reasonable estimators for some aspects of productivity. First, there has been a long discussion in the literature about the intuitive plausibility and interpretation of Baayen’s statistics, the conclusion of which has generally been positive (e.g., Baayen, 1993, 2006; Baayen and Lieber, 1991; van Marle, 1992). Second, these statistics correlate well across corpora (Baayen and Renouf, 1996). Third, they have a strong theoretical underpinning, being derived from the theory of Good–Turing estimation. Fourth, they correlate well with other phenomena, such as ordering and parsability, which are predicted to be related to productivity (e.g., Hay and Baayen, 2002; Plag and Baayen, 2009). Fourth, and finally, there is a small amount of experimental evidence supporting their use (Baayen, 1994). These estimators, therefore, represent a theory which has withstood a large amount of empirical scrutiny, and which is quite different from the theory proposed in this thesis. It is therefore useful to test whether these estimators and the models in the thesis make convergent predictions about productivity.

Hay and Baayen (2002) report the values of $P$ and $P^*$ for a large number of suffixes computed from a corpus based on CELEX morphological analyses. Table 7.2 shows the Pearson correlations between these values and corresponding values from the models, as plotted in the last section (i.e., estimates of $\hat{\Pi}_p$ and $\Pi^*_p$).

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>0.907</td>
<td>-0.003</td>
<td>0.692</td>
<td>0.346</td>
<td>0.143</td>
</tr>
<tr>
<td>$P^*$</td>
<td>0.662</td>
<td>0.480</td>
<td>0.568</td>
<td>0.402</td>
<td>0.500</td>
</tr>
</tbody>
</table>

Table 7.2: Correlation with Productivity Measures: The correlation between quantities computed from the trained models and empirical estimates of Baayen’s $P$ and $P^*$ given in Hay and Baayen (2002).

FG (inference–based) provides the best fit for both sets of values, with AG (full–listing) also providing reasonable fit. Thus, FG and Baayen’s corpus–based, Good–Turing estimators provide strongly converging predictions about the productivity of suffixes. Critically, the measures computed by Hay and Baayen (2002) are independent of, and distinct from the simulations reported here: They are based on a different underlying theory and are computed in a different manner.

Examples of Individual Suffixes

Although there are no wide–coverage datasets available for quantitative evaluation of the productivity predictions made by each model, the linguistic and psycholinguistic literatures have discussed the productivity of many specific suffixes. Three of the most commonly cited examples of highly productive suffixes are: adverbial -ly (e.g., Di Sciullo and Williams, 1987), agentive -er (e.g.,

---

6Hay and Baayen (2002) also extend this set with a heuristic parser to more accurately sample low–frequency forms with high–productivity suffixes. However, they use a different parser, and their analyses did not involve any of the modifications of the corpus described in Section 7.2.2, nor did they include extensive hand–correction of forms.

7Note that Hay and Baayen (2002) only provide only the string versions of the various suffixes, rather than their input and output categories. These correlations are therefore paired on the string value of each suffix for the subset of suffixes for which $P$ and $P^*$ values were available.

8Due the large amounts of automatic processing and hand correction that was performed to create the training data for these simulations, the data sets used to compute $P/P^*$ and $\Pi_p/\Pi^*_p$ also (likely) differed substantially.
Fabb, 1988), and -ness (e.g., Aronoff, 1976). To ground intuitions and to illustrate the behavior of the models, it is useful to examine the most productive suffixes predicted by each model.

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ly:Adj&gt;Adv</td>
<td>quickly</td>
</tr>
<tr>
<td>er:V&gt;N</td>
<td>talker</td>
</tr>
<tr>
<td>ness:Adj&gt;N</td>
<td>tallness</td>
</tr>
<tr>
<td>y:N&gt;Adj</td>
<td>mousey</td>
</tr>
<tr>
<td>er:N&gt;N</td>
<td>prisoner</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ion:V&gt;N</td>
<td>regression</td>
</tr>
<tr>
<td>ly:Adj&gt;Adv</td>
<td>quickly</td>
</tr>
<tr>
<td>ate:BND&gt;V</td>
<td>segregate</td>
</tr>
<tr>
<td>ment:V&gt;N</td>
<td>development</td>
</tr>
<tr>
<td>er:V&gt;N</td>
<td>talker</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ion:V&gt;N</td>
<td>regression</td>
</tr>
<tr>
<td>ly:Adj&gt;Adv</td>
<td>quickly</td>
</tr>
<tr>
<td>ate:BND&gt;V</td>
<td>segregate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ion:V&gt;N</td>
<td>regression</td>
</tr>
<tr>
<td>ly:Adj&gt;Adv</td>
<td>quickly</td>
</tr>
<tr>
<td>ate:BND&gt;V</td>
<td>segregate</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>ion:V&gt;N</td>
<td>regression</td>
</tr>
<tr>
<td>ly:Adj&gt;Adv</td>
<td>quickly</td>
</tr>
<tr>
<td>ate:BND&gt;V</td>
<td>segregate</td>
</tr>
</tbody>
</table>

Table 7.4 shows the 5 most productive suffixes predicted by each model (measured using \( \Pi_p \); the joint probability of the suffix and a novel stem). Interestingly, several frequently-discussed productive suffixes—notably agentive -er and adverbial -ly—are among the most productive suffixes under all models, indicating that the productivity of these suffixes is evidenced under all approaches to productivity and reuse studied here. However, there are two cases where FG (inference-based) produces predictions that diverge from the other models. First, FG is the only model to rate -ness—perhaps the most commonly-cited productive English suffix—among the top 5. Second, all four competing models (but not FG, inference-based) rate -ion as the most, or second most, productive suffix in English. Although I know of no direct, experimental evidence regarding the productivity of this suffix, there are a number of reasons to believe that it is relatively unproductive. First, this affix is a classical example of a phonologically non-transparent formative-boundary affix (Chomsky and Halle, 1968), triggering idiosyncratic sound changes in the base (e.g., transmit/transmission). Since non-transparency is typically correlated with non-productivity, this provides evidence that -ion is not a highly productive suffix. Second, in a lexical decision study, Bradley (1978) found that, unlike a number of other suffixes which she examined (e.g., -ness), base-frequency manipulations did not change latencies for -ion when whole-form frequencies were controlled, suggesting that -ion is unparsed in most words. Since parsability is typically assumed

---

9This suffix is the 15th most productive suffix in FG (inference-based).
to be correlated with productivity, this is further evidence that this suffix is not highly productive. Third, Wheeler and Schumsky (1980) found that forms using this suffix were rated as “uncertain suffix” words in a metalinguistic classification task. Thus, its appearance alongside such clearly productive suffixes as -ly and -er in the four competing models is puzzling.

One possible explanation for the differing behavior of the models follows from the way in which they use type and token frequencies. The suffix -ion occurs in a large number of frequent words, and, thus, has high type and token frequencies. The full–parsing model (MDPCFG) bases productivity inferences on the token frequency of suffixes, while the full–listing model (AG) bases productivity inferences on the type frequency of suffixes. Therefore, in both cases, these models may infer that -ion has high productivity, based on these properties of the suffix. The two exemplar models (DOP1 and GDMN) can be understood as interpolating between type and token frequencies in different ways, with GDMN behaving more like MDPCFG and DOP1 behaving more like AG. Therefore, they also infer that -ion is a productive suffix. In contrast, FG (inference–based) bases its inferences about productivity on a tradeoff in the distribution of tokens over types. Even suffixes with high type frequency can be unproductive in the model if too many individual words which use these suffixes are stored.

Whether or not this is the correct explanation for these data is difficult to determine based on a single affix, and, therefore, this proposal must remain speculative until more empirical evidence is available (and other, similar examples are discovered). However, cases such as -ion—where an suffix has both high type and token frequencies, but is, nevertheless, unproductive—may provide empirical leverage to distinguish between theories of productivity and reuse.

Parsability and the Relative Frequency Hypothesis

A closely related property to the productivity of a suffix is its parsability—the rate with which existing words using that suffix are parsed. In this section, I examine the parsing and retrieval of individual forms, and the relationship of the models to Hay’s relative frequency hypothesis (RFH; see Section 6.3.4). The RFH states that the parsability of a suffix is an increasing function of its base to whole–form frequency ratio. When the frequency of the base is greater than the frequency of a derived word, the suffix will tend to be composed. When the frequency of the derived word is greater than the frequency of the base, the suffix will tend to be stored whole. A second prediction of the RFH is that the baseline rate of parsing should vary from suffix to suffix, depending on the proportion of forms which contain the suffix and are parsed. A highly productive suffix (e.g., -ness) should show overall higher rates of parsing than a highly unproductive one (e.g., -th). As I reviewed in Section 6.3.4, these two predictions of the RFH have been supported in a number of empirical studies (e.g., Hay, 2001).

In order to examine parsability and the RFH, for each suffixed form in the training corpus, I computed the probability that the final suffix was composed or retrieved from memory (together with the base). Because words can have more than one derivation, this probability can fall anywhere between 0 and 1. Figure 7.5 plots the probability that the final suffix for each word in the training corpus was parsed, as a function of the training word’s whole–form (x–axis) and base (y–axis) frequencies. Words for which this value was less than 0.5 (i.e. which tended towards retrieval of the base together with the suffix) are plotted in shades of blue, with darker shades of blue indicating lower probability of parsing. Words for which the probability of parsing was greater than 0.5 are plotted in red, with deeper shades of red indicating words for which the probability of the final suffix being parsed are highest. Lighter blue and more orange points are closer to the 0.5 boundary value and are, therefore, more ambiguous with regard to parsing versus retrieval.

The plots in Figure 7.5 reveal several patterns. First, unsurprisingly, the MDPCFG (full–parsing) model produces only parsing behavior. This is a logical necessity: This model provides no
Figure 7.5: Parsing and Retrieval for Individual Suffixed Forms: This figure shows the probability of parsing for the final suffix of words in the training set. Deeper blue points represent words whose final suffix was more likely to be retrieved together with its base. Deeper red points represent words whose final suffix was likely to be composed. Lighter blue and more orange points represent ambiguous words whose retrieval/parsing probability was closer to the 0.5 boundary.
mechanism for whole–form storage. Also, unsurprisingly, AG (full–listing) produces nearly deterministic retrieval of the final suffix of words. In this case, the pattern is not a logical necessity: AG has the option of choosing to generate a form compositionally using the underlying base system, but, for existing words, the probability of this possibility is very low.

Three of the models, FG (inference–based), DOP1 (exemplar–based), and GDMN (exemplar–based) allow more variability in parsing and retrieval. There are two important observations which apply to all three of these models. First, all three models appear to show a preference for retrieval over parsing. This result accords with the intuition that derivational morphology is a domain where most structure is stored. Table 7.3 confirms this impression. The first row shows the mean probability that an existing form is parsed under each model. The second row shows the weighted mean of this value, taking into account the probability of each word (as computed by each model). Because the weighted mean takes into account the probability of each word, and, because more probable words are more likely to be stored as wholes, the weighted mean of parsability tends to be significantly lower than the unweighted mean. Nevertheless, even for the unweighted mean, there is a clear global preference for retrieval over parsing in every model that allows for retrieval.

The second important observation is that all three of these models (i.e., FG, DOP1, and GDMN) are consistent with the RFH. The x–axis in the plots in Figure 7.5 represents the whole–form frequency of each suffixed word, and the y–axis represents the base frequency of each suffixed word. Therefore, base frequency is greater than whole–form frequency for points which fall above a line where x = y. If parsability depended only on whole–form frequency, then red and blue points would tend to fall on either side of a vertical line. If parsability depended on base frequency, then red and blue points would tend to be separated by a horizontal line in these plots. However, for all three models, red points tend to appear above a line with positive slope, showing that parsability depends on the ratio of base to whole–form frequency, as predicted by the RFH.

The consistency of the three models which allow for variability in parsability is further confirmed by Figure 7.6, which show that final–suffix parsing probabilities for individual forms for three specific suffixes: -ness:Adj>N, -ity:Adj>N, and -th:Adj>N. Because MDPCFG (full–parsing) and AG (full–listing) show no variability in parsability, their plots are not included here.

For all three suffixes and models shown in Figure 7.6, the parsability of individual forms depends on the ratio of base to whole–form frequency. These plots also show that the three models are strongly consistent with the second prediction of the RFH mentioned above: More productive suffixes show a higher baseline rate of parsing than less productive suffixes. For all three models, -ness:Adj>N shows the highest rate of parsing, ity:Adj>N shows an intermediate rate, and -th:Adj>N shows the lowest rate.

Although both FG (inference–based) and the exemplar–based models are consistent with Hay’s relative frequency hypothesis, there is one striking difference between them. Examining the plots in Figures 7.5 and 7.6, we see that, while FG tends to produce points which are strongly
Figure 7.6: Parsing and Retrieval for Forms Ending in Particular Suffixes: This figure shows the probability of parsing for the final suffix of words ending in \(-ness \text{ Adj} > N\), \(-ity \text{ Adj} > N\), and \(-th \text{ Adj} > N\). Deeper blue points represent words whose final suffix was more likely to be retrieved together with its base. Deeper red points represent words whose final suffix was likely to be composed. Lighter blue and more orange points represent ambiguous words whose retrieval/parsing probability was closer to the 0.5 boundary.
red (parsed with near probability 1) or strongly blue (retrieved with near probability 1), DOP1 (exemplar–based) and GDMN (exemplar–based) produce a number of light blue or orange points. This indicates that while FG tends to treat each word as either parsed or retrieved, the exemplar-based models predict that many words will be represented as intermediate mixtures of these modes of computation.

The data in Table 7.4 illustrate this point. Like Table 7.3, Table 7.4 shows the weighted and unweighted mean final–suffix parsing rates, but in 7.4 these have been divided into two subsets. The first subset, labeled Parsed, are those forms whose rate of final–suffix parsing is greater than 0.5. The second, labeled Unparsed, are those forms whose rate of final–suffix parsing is less than 0.5. The FG (inference–based) model produces a sharp dichotomy between these two subsets. For the exemplar–based models, by contrast, the dichotomy is less sharp. For the GDMN (exemplar–based) model, in particular, the rates of parsing or retrieval for the two subsets are near the logical mean points of the two subsets: 0.25 for forms less than 0.5 and 0.75 for forms greater than 0.5.

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parse Probability Parsed</td>
<td>0.990</td>
<td>1.0</td>
<td>NA</td>
<td>0.680</td>
<td>0.720</td>
</tr>
<tr>
<td>Parse Probability Parsed</td>
<td>0.972</td>
<td>1.0</td>
<td>NA</td>
<td>0.719</td>
<td>0.770</td>
</tr>
<tr>
<td>Parse Probability Unparsed</td>
<td>0.0001</td>
<td>NA</td>
<td>0.00165</td>
<td>0.0588</td>
<td>0.243</td>
</tr>
<tr>
<td>Parse Probability Unparsed</td>
<td>0.00007</td>
<td>NA</td>
<td>0.00017</td>
<td>0.0140</td>
<td>0.212</td>
</tr>
</tbody>
</table>

Table 7.4: Probability of Parsing Broken down by Group: This table shows the parsing rates for the various models broken down by forms which have > 0.5 probability of parsing and forms which have ≤ 0.5 probability of parsing. FG, unlike the other models, produces a sharp dichotomy between these two sets.

To my knowledge, no empirical work has ever tried to assess the degree to which parsability is a property which can vary for individual word forms. Although most processing experiments have tended to treat parsability as a fixed, inherent property of individual words, it is unclear whether this is just an unexamined assumption or reflects a deeper intuition. Therefore, this property of FG (inference–based) should be taken as an untested prediction of the model. Nevertheless, as I will discuss in the next section, this property is crucial for understanding the differences between FG and the other models studied in this thesis.

Discussion of Productivity and Parsability

In the preceding sections, I compared and contrasted the predictions made by each of the five models for the phenomena of productivity and parsability. First, in an exploratory analysis of the distribution of predicted productivity scores, FG (inference–based) produced the widest variance in two standard measures of productivity, and surprisingly, assigned lower productivity to some suffixes than even the full–listing model (AG). Second, FG also showed strong convergence in its predictions with two theoretical, corpus–based measures of productivity: Baayen’s \( P \) and \( P^* \). With the exception of AG, which produced reasonably high correlations with these values, the other models did not fare well in this test. Third, although the results must be interpreted cautiously due to lack of experimental data and small number of suffixes involved, the five most productive suffixes under FG appeared to be more intuitively plausible than the same predictions for the other four models. Only FG placed the highly productive suffix -ness in the top 5, and, unlike the other four models, it did not infer that -ion was one of the most productive suffixes in English.

Turning to parsability, I showed that the three models which predicted variability for parsing and retrieval for existing forms (i.e., FG, inference–based, and the two exemplar–based models,
DOP1 and GDMN) were all consistent with Hay’s relative frequency hypothesis—a theory of the relationship between parsability and frequency, with accumulating empirical support. This exploratory analysis of parsability also revealed a way in which FG differed from the two exemplar–based models. While the exemplar–based models predicted that many existing words should show an intermediate degree of parsability, FG predicted that words should be either strongly parsed or strongly retrieved.

This last point is crucial for understanding the other analyses of productivity and parsability. As I described in Section 2.4.4, the distributions used to define FG’s generative model lead to a rich–get–richer dynamic. As a result, FG (inference–based) exhibits attractor–like inference behavior. Once the model has decided to parse or retrieve a form, it prefers to parse and retrieve that form in the future. Because, the productivity of a suffix on novel forms is related to the proportion of existing forms in which it was composed, this attractor–like behavior also explains why FG produces the largest variance in productivity scores (and, likely, why its predictions correlate highly with Baayen’s measures). The prior distribution defined by the model prefers to treat suffixes as highly productive, or highly unproductive, thus preferring to avoid the ambiguous middle ground.

These preferences may shed light on a problem which I discussed in Chapter 1. In Section 1.3, I discussed the representational approach to productivity and reuse, which assumes that linguistic structures can be categorized into subsets (based on properties independent from storage and computation), some of which are computed productively and some of which are stored in memory. For example, a representational approach might propose that words (defined phonologically) are stored in memory, while larger linguistic units, such as phrases, are computed productively. In this thesis, I have discussed several additional examples of this approach. In dual–mechanism theories of the past tense, the relevant property for categorization is regular versus irregular: Regulars are computed productively, while irregulars are retrieved from memory. In stratal theories of lexical organization, the relevant property is the phonological transparency of various affixation processes. One important advantage of the representational approach is that it is often (approximately) right. For example, converging evidence indicates that many (or most) regular past–tense forms are composed (see Section 4.3.4). The attractor–like dynamic of FG (inference–based) may help explain why this is the case. When a set of forms share a substructure, FG prefers to treat this substructure as either usually composed, or usually reused (within that set of forms). The prior distribution disfavors intermediate mixtures of the two modes of computation. While this is just a tendency, and (crucially) FG allows for exceptions, such as the storage of high–frequency regular items, this tendency may shed light on why productivity and reuse cluster so strongly with different categories of linguistic structures.

I conclude this section with a discussion of the relationship between Fragment Grammars and Hay’s relative frequency hypothesis.

**Fragment Grammars and the Relative Frequency Hypothesis** In the preceding sections, we saw that both FG (inference–based) and the two exemplar–based models produced predictions which were consistent with Hay’s relative frequency hypothesis (RFH). In this section, I discuss the relationship between this hypothesis and the FG model.

Inference in the the Fragment Grammar model is based on optimizing a tradeoff between productivity and reuse, finding the granularity of units to store which best explains the distribution of structures in an input dataset. Intuitively, when a base is shared across many different affixed forms, each with low frequency, FG (inference–based) will tend to treat these forms as composed—representing the base as an independent reusable unit. Conversely, when some particular affixed form is very frequent, and its subparts are not shared with many other forms, FG will tend to store the form as a whole. This is the same intuition captured by the RFH’s comparison of whole–form and base frequencies. Also, like the RFH, Fragment Grammars predict that the parsability of an affix will depend on the proportion of forms in which it was composed in the past, not the absolute
type or token frequencies of the affix. However, as a fully-specified formal model, there are several respects in which FG (inference-based) goes beyond the RFH. First, although it is clear from Hay’s discussions that the frequency of all subparts of a word are meant to be compared to its whole-form frequency when determining if the word is composed (Hay, 2003, p. 60), the RFH does not provide an explicit formula for how this should be done. In particular, most empirical studies of the RFH (e.g., Hay, 2001, 2003; Hay and Baayen, 2002) have focused on a single locus of affixation, examining the parsability of a set of forms while varying whole-form and base frequencies. By contrast, because FG is defined over arbitrary, hierarchical systems of rules, it provides a formula for predicting the complex set of tradeoffs which arise when words have more than two subparts. Second, although in some studies parsability is assumed to be linear in the ratio of whole-form and base frequencies (e.g., Hay and Baayen, 2002), in general, the RFH is consistent with any monotonic increasing function of this ratio. FG, by contrast, defines a specific, non-linear functional relationship between frequencies and the probability that a form is composed. As I discussed in Section 2.4.4, the model can exhibit phase-shifts as the relative frequencies of forms and their parts change during learning. Nevertheless, the RFH organizes a large number of theoretical proposals from the processing literature, and captures an important empirical insight—FG can be viewed as a formalization and extension of this theory.

7.3.2 Ordering and Base-Driven Selectional Restriction

Out of the many thousands of logically possible sequences of English suffixes, only a tiny fraction are actually attested in English words, and these occur in very specific orders (affix-ordering). Furthermore, the suffixes already present in a base often govern which suffixes can further attach to the base (base-driven selectional restrictions). Unlike productivity and parsability, affix orderings are directly observable in corpora of English words, and base-driven selectional restrictions have been the subject of a number of experiments, affording, in both cases, the ability to directly evaluate the five models studied here. In this section, I examine these issues.

I first show that, of the five models, Fragment Grammars are most consistent with the main prediction of CBO: More productive affixes tend to appear later in affix sequences. I then evaluate the performance of the models against empirical ordering measures reported by Plag and Baayen (2009). I show that FG (inference-based) provide the best fit to the summary mean rank statistic, but that AG (full-listing) provide the best fit to raw corpus frequency. However, I also discuss a way in which FG capture structure which AG cannot: In some cases, Fragment Grammars learn that specific combinations of suffixes (e.g., -able+ity, i.e., -ability) are generalizable. I examine this phenomenon in detail in the next section, showing that Fragment Grammars capture the pattern of -ity and -ness generalization found in the experimental literature (Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978; Cutler, 1980). Finally, I conclude by presenting examples of other affix sequences which FG predicts to be generalizable as units.

Complexity-Based Ordering

In Section 6.3.5, I discussed the phenomenon of suffix ordering, and various theories that have been proposed to account for it. In particular, I discussed Jennifer Hay’s complexity-based ordering (CBO) hypothesis, which proposes that more parsable (and therefore more productive) suffixes should appear outside of less parsable suffixes. In this section, I examine the degree to which each model is consistent with CBO.

To examine CBO, it necessary to generate suffix ordering predictions from each model. To do this, I considered the subset of all forms with exactly two suffixes (i.e., words of the form stem -suffix -suffix) and computed the marginal probability that each suffix occurred in the first or second
suffix position. Using these values, the log odds of observing each suffix in the second position was calculated. These values can be understood as measuring the (log) number of times more likely it is that a particular suffix will appear in the second position as opposed to the first position (in forms with exactly two suffixes).

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation</td>
<td>0.323</td>
<td>-0.023</td>
<td>0.173</td>
<td>-0.273</td>
<td>-0.064</td>
</tr>
</tbody>
</table>

Table 7.5: Correlation of Parsability of Suffixes and Ordering Preferences: The Pearson correlation between the productivity of a suffix and the log odds that it will appear second in a sequence of two suffixes.

Table 7.5 shows the results of correlating the log odds of a suffix appearing in second position with its productivity, $\Pi_p$ (i.e., the joint probability of the suffix and a novel stem). FG (inference-based) produces the strongest positive relationship between these two measures, showing that, for this model, the productivity of each suffix is predictive of the order in which the suffix appears in morphologically complex words. There are two points to note about these correlations. First, CBO doesn’t provide any predictions about the magnitude of the relationship between productivity and ordering, therefore, all that can be concluded based on this analysis is that FG is most consistent with the CBO. Second, the values in Table 7.5 are correlations between two model internal quantities. Therefore, although they show that FG is the model which is most consistent with Hay’s CBO, they do not measure how well any of the models are accounting for the empirical ordering facts.

Correlation with Mean Ranks from Plag and Baayen (2009)

In this section, I consider the degree to which each of the five models is able to capture the empirical pattern of ordering relations between English derivational suffixes. In Section 6.3.5, I discussed the empirical measure of suffix ordering presented in Plag and Baayen (2009), called the mean rank statistic. Intuitively, the mean rank is a summary statistic which captures how often, on average, a particular suffix appears after other suffixes in a set of words in some corpus. In Plag and Baayen (2009), mean ranks are provided for a number of English suffixes.

Table 7.6 shows the Spearman rank correlations between the mean rank values for suffixes reported in Plag and Baayen (2009), and the log odds that each suffix appears in second position as computed against the five models (see the preceding section).

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation</td>
<td>0.568</td>
<td>0.275</td>
<td>0.424</td>
<td>0.452</td>
<td>0.431</td>
</tr>
</tbody>
</table>

Table 7.6: Correlation of Suffix Ordering Probabilities and Mean Ranks: The Spearman rank correlation between the log odds that a particular suffix will appear second in a sequence of two suffixes and the mean rank statistic for that suffix as given in Plag and Baayen (2009).

Table 7.6 shows that FG (inference-based) is best able to predict (in summary) the pattern of suffix orderings in the corpus studied by Plag and Baayen (2009). Critically, although the mean rank is a summary statistic, it is computed based on observed data. Therefore, unlike the analyses discussed above for productivity and parsability, this result shows that FG can account for empirical patterns of affix orders.
Correlation with Raw Frequencies from Plag and Baayen (2009)

Although the correlation with mean rank reported in the last section provides empirical support that FG (inference–based) can correctly predict ordering information, the mean rank statistic has an important limitation. Because it is a summary statistic, it collapses ordering information over many different attested combinations of suffixes. Therefore, in this section, I examine the ability of the models to predict specific sequences of suffixes attested in English.

In their paper, Plag and Baayen (2009) provide the raw frequencies of the suffix combinations that appeared in their sample corpus. One way to assess the performance of each model in learning ordering restrictions on a more fine–grained scale is by seeing the degree to which each model can predict these frequencies. The appropriate quantity for predicting the (relative) frequency of different sequences of suffixes is the total probability assigned to that suffix combination by each model. There are two ways that this can be computed. First, we can compute the marginal probability of each sequence, which can be understood as the probability that a randomly sampled word will end in the sequence. Because this quantity sums over all bases, it includes the probability of all forms which were attested in the training sample. Second, we can compute the wug probability of each sequence, which can be understood as the probability that each suffix combination generalizes to novel bases. This latter quantity allows us to assess the degree to which English words contain sequences which can be generalized as combinations.

<table>
<thead>
<tr>
<th>Model</th>
<th>FG</th>
<th>MDPCFG</th>
<th>AG</th>
<th>DOP1</th>
<th>GDMN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation (Marginal Probability of Pair v. Frequency of Pair)</td>
<td>0.671</td>
<td>0.367</td>
<td>0.695</td>
<td>0.456</td>
<td>0.422</td>
</tr>
<tr>
<td>Correlation (Wug Probability of Pair v. Frequency of Pair)</td>
<td>0.501</td>
<td>0.367</td>
<td>0.434</td>
<td>0.434</td>
<td>0.407</td>
</tr>
</tbody>
</table>

Table 7.7: Correlation of Suffix Ordering Probabilities and Ranks: The Pearson correlation between the marginal probability of each pair of suffixes and the log frequency of that pair as reported in Plag and Baayen (2009). The first row represents the correlation between these log frequencies and the marginal probability of the pair of suffixes, while the second row represents the correlation between the log frequencies and the probability of the suffix pair generalizing to a novel wug stem.

Table 7.7 shows the correlations between model predictions of these two quantities and the logarithm of the sequence frequencies from Plag and Baayen (2009). Because most suffix combinations are unattested in the Plag and Baayen (2009) sample—some for principled, and some for accidental reasons—I have excluded frequency 0 forms from these correlations. The table shows that when compared with marginal probability, AG (full–listing) provides the best predictions of suffix ordering, with FG (inference–based) as a close second. The two exemplar–based models (i.e., DOP1 and GDMN) do substantially worse, and the MDPCFG (full–parsing) model does extremely poorly, as has been true elsewhere.

Because it is a full–listing model, AG stores all of the words in the training set, and, because marginal probability takes into account these stored forms, AG is able to provide the best predictions about attested suffix combinations. FG (inference–based) stores many, but not all, forms and, therefore, provides the second best predictions for the frequencies of suffix combinations, based on marginal probability. These results reflect the fact that many affix combinations in English are not generalizable, only occurring in existing, stored forms.

When evaluated against the wug–probability of each sequence, all of the models do more poorly (or the same in the case of MDPCFG); however, now FG (inference–based) best fits the data. This pattern of correlations suggests an interesting possibility. Because these quantities were computed using a completely novel wug stem, AG accounted for them entirely using its underlying PCFG base system. Recall that the probabilities of rules in the AG base system are estimated (approximately) from the type frequency of each suffix. Therefore, the (wug–probability) correlation for AG (0.434)
can be considered a baseline representing the amount of structure that can be captured in suffix combinations using only a fully-compositional system whose probabilities are estimated based on type frequencies. Similarly, because MDPCFG is a full-parsing model which estimates the probability of each suffix based on its token frequency, the \( \text{wug-probability} \) correlation for MDPCFG (i.e., 0.367) can be considered a baseline representing the amount of structure that can be captured in suffix combinations using a fully-compositional system whose probabilities are estimated based on token frequencies.

The \text{wug-probability} correlations for AG (full-listing) and MDPCFG (full-parsing) show the amount of structure that can be captured based on the assumption that attested combinations are fully-compositional (for type and token estimates of rule probability, respectively). However, the \text{wug-probability} correlation for FG (inference-based; i.e., 0.501) is higher than both of these values. What additional structure is FG exploiting? One possibility is that some combinations of derivational suffixes in English are generalizable as single units, and that this is reflected in some attested combinations. In the next section, I examine generalizable suffix combinations in more detail.

**Base-Driven Selectional Restrictions**

In this section, I consider one particular kind of selectivity which is common in English derivational morphology: base-driven selectional restrictions. As I reviewed in section 6.3.1, there are many different kinds of selectional restrictions which constrain the distribution of English derivational suffixes; these include various conditions on phonological, semantic, and pragmatic structure. The assumptions made in modeling English derivational morphology make it impossible to capture these restrictions directly. However, there is one kind of selectivity, base-driven selectional restrictions, which can be captured in the present framework. A classic and well-studied example of this class of restrictions concerns the distribution of \text{ness}\text{Adj}>\text{N} and \text{ity}\text{Adj}>\text{N}. In general, \text{ness}\text{Adj}>\text{N} is a far more productive suffix than \text{ity}\text{Adj}>\text{N}. However, after certain suffixes, this basic pattern is reversed. For example, three affixes that show this reversal of preferences and which have been discussed extensively in the literature are -\text{al} (e.g., \text{minimality}), -\text{ic} (e.g., \text{electricity}), and -\text{able} (e.g., \text{walkability}) (Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978; Cutler, 1980; Embick and Marantz, 2008).

A particularly important pair of affixes which contrast in whether they prefer \text{ness}\text{Adj}>\text{N} or \text{ity}\text{Adj}>\text{N} are -\text{able} and -\text{ive}. While there are many existing English words which end in both -\text{ability} (e.g., \text{mutability}) and -\text{ivity} (e.g., \text{selectivity}), experimental studies have found that, although subjects prefer to generalize -\text{ity} to words ending in -\text{able}, they prefer to generalize -\text{ness} to words ending in -\text{ive} (Anshen and Aronoff, 1981; Aronoff and Schvaneveldt, 1978).10 In the next section, I compare the predictions of the five models with respect to context-dependent -\text{ity} and -\text{ness} generalization.

**Comparison of -\text{ity} and -\text{ness}** For each of the five models, I computed the preference for -\text{ity} versus -\text{ness} after all suffixes which occurred before both of them in the training corpus. I computed these preferences in two ways. First, I computed the log odds in favor of -\text{ity} versus -\text{ness} with respect to the marginal probability of each suffix combination. Because the marginal probability sums over all stems, these values include all forms which appeared in the training data. Second, I computed the preference for -\text{ity} versus -\text{ness} with respect to the \text{wug-probability} of each suffix combination. This values measure the willingness of each model to generalize the combination of affixes to novel words.

---

10Anshen and Aronoff (1981) examines the suffix -ible rather than -\text{able}, however, because these were collapsed in the input data for the present simulations, I frame my discussion in terms of the latter.
Figure 7.7 plots the results for the 9 most probable preceding suffix contexts, and the null (i.e., bare stem) context. Bars which extend to the left of the center line signify a preference for -ity:Adj>N in the context of the suffix labeling the bar. Bars which extend to the right of the center line signify a preference for -ness:Adj>N in the context of the suffix labeling the bar. The left–hand column shows the results for marginal probabilities, and the right–hand columns shows the results for the wug probabilities.

The column of marginal–probability plots reflects the evidence for each suffix combination that was present in the training data. Notably, the suffixes -(i)an, -ic, -ive, -able, -al, and the null context have high marginal likelihood in combination with -ity in all models which can represent such combinations at all (i.e., with the exception of MDPCFG, full–parsing). This reflects the large number of English words which attest these combinations. As is reflected in the marginal likelihood plots, the other suffixes (i.e., -ful, -ent, -y, and -ous), occur more often in combination with -ness in the training data.

When examining how each model chooses to generalize -ity and -ness based on the preceding context (i.e., the wug–probabilities), we see stark differences between the models. Consistent with the higher overall productivity of -ness, FG (inference–based) prefers to generalize this affix in most cases. In precisely three cases (all discussed in the linguistic literature), -ic, -able, and -al, however, FG prefers to generalize with -ity, indicating that it assigns high probability to these combinations as a unit with a variable in the base position. In two cases, -(i)an and -ive, FG prefers -ness despite the fact that, in the training data, most forms ending in these suffixes used -ity (as reflected in the marginal likelihood plots). In these cases, FG is effectively ignoring the high token frequency of -ivity and -(i)anity in the input. For two cases, -ability/-ableness and -ivity/-iveness, there is experimental evidence showing that the predictions of FG are consistent with native speaker preferences. This pair of cases is especially important because both affix pairs are frequent in the training data; however, their generalization preferences with respect to -ity and -ness go in opposite directions.

Turning to the generalization performance of the other models, we see that MDPCFG (full–parsing) and AG (full–listing) do not demonstrate variability between affix combinations. Because MDPCFG estimates probabilities using token frequencies, it shows a small overall preference for -ity. Because AG estimates the probability of rules in its underlying PCFG using type frequencies, it shows an overall preference for -ness. This latter result is of special importance. In several place in this thesis, we have seen that AG provides reasonably accurate predictions about generalization for single affixes. It performed nearly as well as FG (inference–based) when generalizing the regular past tense rule (see Section 5.3.2) and in its correlations with Baayen’s measures of productivity (see Section 7.3.1). However, AG’s ability to generalize correctly depended on the fact that these cases concerned a single affix that could be represented as a single rule in AG’s base system. Because it cannot represent more complex structures which are also generalizable, AG cannot learn the pattern of base–driven selectional restrictions examined here.

Finally, the two exemplar–based models (DOP1 and GDMN) generalize in a way that mirrors the marginal probabilities that they assign to suffix combinations. Importantly, they cannot capture the crucial difference between forms ending in -ive and -ian and other forms which frequently occur with -ity in the input.

Examples of Other Suffix Combinations While the theoretical literature on affix ordering has focused primarily on a few affix combinations (such the -ity and -ness example discussed above), there are, of course, millions of potential suffix sequences in English. The next analysis is a preliminary attempt to generate predictions about affix combinations for future empirical work.

I performed a systematic search over the space of all possible suffix sequences up to length 4, scoring the wug–probability of each possible combination as given by FG (inference–based). Since AG (full–listing) and MDPCFG (full–parsing) cannot represent generalizable combinations of suffixes, it
Figure 7.7: Base–driven Selection of $\text{ity:Adj} > \text{N}$ versus $\text{ness:Adj} > \text{N}$: These plots show the preferences for $\text{ity:Adj} > \text{N}$ versus $\text{ness:Adj} > \text{N}$, after various preceding suffix contexts, for the five models. The left–hand column of plots shows these values computed against the marginal probability of each suffix combination, and the right–hand plot shows these values computed against the wug–probability of each suffix combination.
did not perform search for these models. Furthermore, the large size of the grammars for the two exemplar–based models (DOP1 and GDMM) makes any such systematic search infeasible. Figure 7.8 displays the sequences of suffixes length 2, 3, and 4 which are predicted to be most generalizable under the FG model.

<table>
<thead>
<tr>
<th>Length 2</th>
<th>Length 3</th>
<th>Length 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>Category</td>
<td>Sequence</td>
</tr>
<tr>
<td>-ate-ion</td>
<td>N</td>
<td>-ology-ic-al</td>
</tr>
<tr>
<td>-ic-al</td>
<td>A</td>
<td>-ate-ion-al</td>
</tr>
<tr>
<td>-ate-ice</td>
<td>A</td>
<td>-ment-ate-ion</td>
</tr>
<tr>
<td>-al-ity</td>
<td>N</td>
<td>-ify-ate-ion</td>
</tr>
<tr>
<td>-al-ize</td>
<td>V</td>
<td>-ize-ate-ion</td>
</tr>
<tr>
<td>-ology-ist</td>
<td>N</td>
<td>-ate-ion-al</td>
</tr>
<tr>
<td>-ment-al</td>
<td>A</td>
<td>-ic-al-ly</td>
</tr>
<tr>
<td>-able-ity</td>
<td>N</td>
<td>-ate-ion-ary</td>
</tr>
<tr>
<td>-ist-ic</td>
<td>A</td>
<td>-al-ist-ic</td>
</tr>
<tr>
<td>-ous-ity</td>
<td>N</td>
<td>-ate-ion-ist</td>
</tr>
</tbody>
</table>

Figure 7.8: Suffix Sequences with High Generalizability: This table shows suffix sequences with high generalizability that resulted from a systematic search with the FG (inference–based) model.

Contained in the examples in Table 7.8 are a number of suffix combinations which are of particular interest. In Section 7.2.2, I discussed the decision on the part of the designers of the CELEX database to use a single suffix, -ation, to represent the combination -ate+ion in cases where the -ate base did not exist (e.g., form/*formate/formation). I argued that this decision reflects an intuition that this combination of suffixes is generalizable as a unit. Reassuringly, this combination is the most highly predicted of all length two combinations by the FG (inference–based) model. In essence, FG has learned the morphological theory implicit in the CELEX analysis scheme.

Several other combinations in the table, which have not been discussed previously, are interesting because they correspond to non–transparent collocations of suffixes. For example, -ify -ic is pronounced -ific, -ance -al is pronounced -antial, and -ify -ate-ion is pronounced -ification. These are important because they emphasize the tight correlation between non–productivity and non–transparency. Stored units can accumulate phonological and semantic non–transparencies. Although I have not represented the necessary phonological (or orthographic) structure to capture these non–transparencies, the system has learned that these combinations must appear together as units, using only the distributional information in the training data.

Many other intuitive combinations also appear on these lists. For example, there seems to be an entire paradigm devoted to various forms of the names of sciences (i.e., -ology, -ologist, -ological, etc.). There is also the common sequence -ical: The productivity of this combination may help explain differences in usage between dialects of English (e.g., genetic in American English versus genetical in British English).

Because the combinations in Table 7.8 are the result of systematic search, they should provide a rich source of hypotheses for future experimental work.

Discussion

In the preceding sections, I have examined the predictions of the five models with respect to suffix ordering and base–driven selectional restrictions. First, I showed that, of the five models studied in this thesis, FG (inference–based) was most consistent with Hay’s complexity–based ordering
(CBO) hypothesis, which states that more parsable (and productive) affixes should appear outside of less parsable affixes. Second, I showed that FG was able to provide the best predictions for the mean rank summary statistic of ordering, and the second best predictions for raw frequencies of attested suffix orders (after AG, full–listing). Crucially, unlike parsability and productivity, suffix ordering is a directly observable property of English words, and, therefore, the second set of results reflect the ability of the models to account for empirical data.

I also argued that the discrepancy between AG (full–listing) and FG (inference–based)’s ability to predict suffix combination frequencies, when examining wug–generalization probabilities, was the result of FG’s representation of some suffix combinations as reusable single units. I then showed that for the case of -ity and -ness, which has been intensively studied in the literature, only the FG model was able to provide an adequate account of the observed empirical pattern of base–driven selectional restrictions. I now discuss in more detail why FG is able to account for the phenomena just presented. First, I discuss the relationship between Fragment Grammars and CBO, and, then, I return to a more detailed discussion of base–driven selectional restrictions.

**Fragment Grammars and Complexity–Based Ordering**  In this section, I consider the question of why FG (inference–based) predicts a correlation between the parsability and productivity of affixes, and the order in which they appear in complex words.

Consider a partial computation containing a variable like that shown in Figure 7.9. As described in detail in Chapter 2, in the present framework the variable Adj is interpreted as an instruction to sample some adjective to complete the computation. From the perspective of this partial computation, which adjective is irrelevant. The adjective will be sampled from the distribution associated with the category Adj, independent of the fact that it will be used ultimately in a word ending in -ness.

![Figure 7.9: Partial Computation](image)

As I discussed in Section 1.4.3, the probability distribution over computations declines geometrically in the number of computational choices that must be made to construct a form. In general, this means that a variable like Adj will usually be completed by a single random choice, less often, by two random choices, even less often by three, etc. The probability of completing the computation of Adj with larger sampled structures drops off very quickly in the size of those structures.

However, as I discussed in Chapters 1, storage of large derivation tree fragments by the FG (inference–based) model allows sequences of computation to be completed with a single random choice. All else being equal, the chance of sampling a complex, but stored, base versus a stem with no internal structure depends only on the frequency of the corresponding forms in the data. Thus, when sampling a structure to fill the Adj variable above, the system will tend to use bare stems or morphologically complex bases which are stored in their entirety. However, whether an suffix tends to be stored together with its base (e.g., like -th) or composed with its base (e.g., like -ness), is precisely the parsability of the suffix. For English derivational morphology, more productive suffixes will occur with a greater variety of bases, including complex bases involving less productive suffixes. Thus, on average, the system will sample forms where less parsable suffixes appear inside of more parsable suffixes. This is precisely the prediction of CBO.
Base–Driven Selectional Restrictions  The ability of FG (inference–based) to predict base–driven selectional restrictions follows from its ability to represent structures like that shown in Figure 7.10, and the inability of AG (full–listing) and MDPCFG (full–parsing) to account for these facts followed from their inability to represent such structures.

![Diagram](image)

Figure 7.10: Greater than Depth–one Generalizable Structure: The FG (inference–based) model represents the greater productivity of -ity after -able with a derivation tree fragment like that shown above.

However, the difference between FG (inference–based) and the two exemplar–based models (DOP1 and GDMN) is more subtle, and highlights an important facet of FG inference. Recall from Section 7.3.2, that, for the two exemplar–based models, generalization of -ity and -ness followed directly from the probability of various suffix combinations in the marginal distribution over forms. Because, the combination -ivity had higher token frequency in the training data than the combination -iveness the two exemplar models preferred -ity to -ness after -ive—contrary to experimental findings (Anshen and Aronoff, 1981). Why did FG avoid this error? The FG model balances a tradeoff between productivity and reuse. A structure with high token or type frequency can still be judged unproductive if it occurs mostly in forms for which there is evidence of storage. Although -ivity had high token frequency in the training data, FG learned to store most words ending in this combination as wholes. Therefore, when faced with a novel form ending in -ive the -ivity structure was not independently reusable, and, so, the model defaulted to the more productive -ness. For bases ending in -al, -able, and -ic the situation was different. Although these suffixes frequently appeared with -ity, their own bases varied considerably, causing FG to learn a generalizable rule which allowed the addition of these pairs as single units.

7.4 Conclusion

In this chapter, I have presented a number of analyses examining the ability of the five models studied in the thesis to account for productivity, parsability, ordering, and base–driven selectional restrictions. The results for productivity showed that FG (inference–based) produced the greatest variability in productivity scores between affixes, and the tightest convergence with Baayen’s corpus–based measures of morphological productivity: $P$ and $P^*$. I also presented preliminary evidence showing that FG may produce more plausible predictions about productivity in the case of specific English suffixes. The parsability analysis showed that, of the five models, both FG and the two exemplar–based models were consistent with Hay’s relative frequency hypothesis. However, FG was different from these two models in predicting a strong preference for either storage or retrieval of individual forms, rather than a mixture of both modes of computation. FG was also the model most consistent with Hay’s complexity–based ordering hypothesis. FG produced the best correlation with the empirical mean rank summary measure of ordering patterns reported in (Plag and Baayen, 2009), and the second best correlation with raw corpus frequencies of affix pairs (after AG, full–listing). Finally, I showed that FG was able to reproduce the exact pattern of base–driven generalization of -ity and -ness reported in the literature. It was able to reproduce this pattern because, by balancing a tradeoff between productivity and reuse, it could successfully distinguish between highly frequent
combinations which do not generalize (e.g., -ivity) and highly frequent combinations which do (e.g., -ability).
Chapter 8

Conclusion

In this chapter, after a summary of the earlier chapters of the thesis, I consider two additional topics. First, I will show that Fragment Grammars make an unintuitive prediction: Languages in which high-frequency forms are irregular are more probable under the prior distribution defined by the model. This fact may shed light on the pervasive correlation between irregularity, frequency, and (non)productivity in morphological systems. Second, I will consider a persistent idea from the language acquisition literature: the belief that the availability of an abstract generalization implies that the generalization must be used productively by children. I will argue that, contrary to this assumption, an inference-based approach to productivity and reuse predicts that (in general) unproductive forms will be produced by children before they employ productive generalizations. I conclude the chapter with a discussion of future directions and a brief wrap-up.

8.1 Summary of the Thesis

This dissertation has examined the mirror-image problems of productivity and reuse, focusing on two empirical domains: the English past tense and English derivational morphology. English words can be formed using a variety of morphological processes. For example, adjectives can be turned into abstract nouns using one of several suffixes: -ness (e.g., coolness), -ity (e.g., verticality), and -th (e.g., warmth). These suffixation processes vary greatly in their productivity as well as their phonological and semantic regularity. The suffix -ness is highly productive as well as semantically and phonologically regular. The suffix -ity is typically unproductive, but can be generalized in certain contexts, such as after the suffix -able. It also sometimes triggers phonological changes in the bases to which it joins and is generally less semantically transparent than -ness. The suffix -th is highly unproductive, only occurring in a small set of reusable, highly-frequent forms. These -th words exhibit a number of semantic and phonological idiosyncrasies. The variability in productivity exhibited by cases such as -ness/-ity/-th raises three difficult problems for linguistic and psycholinguistic theories. First, what kinds of representations can account for this mixture of productivity and reuse? Second, how can the pattern of productivity and reuse be learned from the input data? And, third, why are productivity, regularity, and frequency correlated?

In Chapter 1, I introduced two traditional accounts of productivity and reuse: The representational approach attempts to reduce computation and storage to independently definable properties of linguistic structures. The Bloomfieldian approach argues that only idiosyncratic structure is stored; all structure that can be captured by regular rules is computed. I argued that both approaches were empirically inadequate and proposed an alternative theory. This theory (and its formal instantiation, Fragment Grammars) is built on two ideas. First, it treats productivity and
reuse as an inference that is based on the distribution of structures in the input data. Under this approach, the only evidence that counts in favor of the productivity of a word–formation process is the frequency with which it is used to form novel words. The only evidence that a word–formation process can only be reused in a restricted set of words is the repetition of those words. Second, the theory is built upon a set of formal techniques—drawn from Bayesian generative modeling and probabilistic programming (Goodman et al., 2008; Johnson et al., 2007a)—which allow (probabilistic) generative models to compile and reuse frequently recurring combinations of computations as single units.

I introduced three alternative approaches to productivity and reuse: full-parsing (implemented by Multinomial–Dirichlet Context–Free Grammars, Johnson et al., 2007b), full-listing (implemented by Adaptor Grammars, Johnson et al., 2007a), and exemplar–based productivity and reuse (implemented by two variants of Data–Oriented Parsing, Bod et al., 2003). In Chapters 2 and 3, I showed how the models can be formalized. In particular, I showed how Fragment Grammars, a generalization of Adaptor Grammars (Johnson et al., 2007a), can be understood in terms of stochastic variants of two techniques from computer science: stochastic memoization and stochastically lazy evaluation. Using these techniques, I then showed how Fragments Grammars could again be generalized to allow the reuse of subcomputations in arbitrary Bayesian generative models, and tentatively suggested that this perspective may provide a way of extending this work to other domains of cognition, such as vision.

Chapters 4–7 applied Fragment Grammars and the alternative models to two empirical domains: the English past tense and English derivational morphology. These two domains are especially interesting for evaluating models of productivity and reuse because each domain exhibits very different distributions of these two properties. Derivational morphology is characterized by a broad cline of affixes of differing levels of productivity (e.g., -ness/-ity/-th), while the English past tense is characterized by a single highly productive regular rule (i.e., +ed) and a number of irregular classes of much more limited productivity (e.g., sing/sang, sleep/slept, etc.). An adequate model of productivity must be able to capture both kinds of linguistic system.

Chapter 4 reviewed the empirical and theoretical literatures on the English past tense, arguing that any empirically adequate theory of this domain must provide both the means for abstract generalization and a quantitative account of competition between generalizations. In Chapter 5, I showed that, of the five models considered in the thesis, Fragment Grammars provided the most accurate account of competition between the regular past–tense rule and irregular inflectional classes. The model was able to learn that the regular rule generalizes to novel stems and, thus, functions as a default, while the irregular classes generalize much more rarely, but block the regular rule when they are available. I also showed that Fragments Grammars used frequency information in a way which is consistent with the results of naturalness judgment and processing experiments. Finally, I showed that only Fragment Grammars provided an account of past tense development which was consistent with empirical findings, producing substantial, but low levels of overregularization when trained on a series of datasets designed to idealize the increasing input data available to the child learner.

Chapter 6 reviewed the literature on English derivational morphology, focusing on five properties of the system: selectivity, transparency, productivity, parsability and ordering. Critically, these five properties are highly correlated. More productive affixes are also more parsable, show fewer selectional restrictions on their distribution, are more phonologically and semantically transparent, and tend to appear outside of less productive affixes. Less productive affixes, by contrast, show more evidence of whole-form retrieval, are less phonologically and semantically transparent, and tend to appear inside of more productive affixes in complex forms.

In Chapter 7, I evaluated the five models studied in this thesis on the domain of English derivational morphology. I showed that Fragment Grammars produce predictions about the pro-
ductivity of individual English suffixes which were convergent with corpus–based measures from the literature. I also showed that, in several specific cases (i.e., -ness and -ion), the productivity scores predicted by Fragment Grammars were more consistent with discussions in the linguistic and psycholinguistic literatures than the predictions produced by the other models. Fragment Grammars produced the best fit to a summary empirical measure of affix–ordering facts, and the second best fit (after Adaptor Grammars) to the raw corpus frequencies of suffix combinations. Finally, I showed that only Fragment Grammars could correctly predict the pattern of base–driven generalization of -ity and -ness which has been found in the experimental literature.

8.2 Productivity, Irregularity, and Frequency

In both empirical domains examined in this thesis—the English past tense and English derivational morphology—there is a strong correlation between (non)productivity, (ir)regularity, and frequency. Productive processes, such as the regular past tense +/d/ rule, and -ness affixation, are characterized by phonological and semantic transparency as well as a word frequency distribution which includes a large number of relatively infrequent words. Unproductive processes, such as the irregular inflectional classes and -th affixation, are characterized by phonological and semantic idiosyncrasy, greater restrictions on the phonological make–up of the bases to which they can apply, and a word–frequency distribution with a smaller number of more frequent forms.

In Chapters 5 and 7, I showed how the Fragment Grammars model uses the distribution of word tokens over word types to infer the pattern of productivity and reuse in both morphological systems. However, the simplifications made in modeling these domains—in particular, the absence of phonological and semantic structure in the underlying representations—meant that these simulation results could not shed light on the systematic correspondence between productivity and regularity, on one hand, and nonproductivity, irregularity and frequency, on the other.

In this section, I consider this problem: Why do highly frequent forms exhibit higher levels of irregularity? After reviewing several earlier approaches to this question, I demonstrate an unintuitive prediction of the Fragment Grammar model. Under the inference–based approach to productivity and reuse, systems in which the most frequent forms do not share structure with the long tail of productively computed forms have higher prior probability than completely regular systems. Therefore, systems where high–frequency forms are irregular are more easily learned by the model and are predicted to be favored over the cultural evolution of language.

8.2.1 Earlier Accounts

Many researchers have observed that irregular forms are usually highly frequent and that highly–frequent forms have a greater chance of being irregular. In the literature, several theories have been proposed to explain this correspondence between irregularity and frequency. By far the most common account is the memory–limit-hypothesis (MLH) (e.g., Baayen and Lieber, 1996; Bybee, 2006; Lieberman et al., 2007). The MLH assumes that there is an overall preference for regularity in language. However, processes of language change introduce variation in the way in which particular meanings can be expressed. Because the learner’s ability to acquire particular forms depends on memory, and because high frequency items are easier to memorize, as the language changes, older, less productive structures will be removed from the language except when they have particularly high frequency. For example, Baayen and Lieber (1996, p. 283) proposed that non–compositionality in the meaning of words is related to frequency via such a memory mechanism.

1There are many potential reasons for a general preference for regularity: a general simplicity bias, ease of learning, etc.
A high frequency of use guarantees that their [semantically non-transparent forms] opaque reading can be retained in memory. Thus it is only to be expected that opaque formations show a strong tendency to appear in the highest ranges of the frequency spectrum.

The key assumption underlying the MLH is clearly correct: Highly frequent items are easier to memorize. Moreover, the MLH appears to provide an accurate account of historical language change. For example, some irregular past tense inflectional classes were productive in earlier stages of the English (e.g., /i/ → /æ/, sing/sang). Over time, these classes lost their productivity, while the regular +/d/ rule became productive. As a result, many lower-frequency irregulars have been (and continue to be) regularized (e.g., bide/bode → bide/bided), while the most frequent irregulars have tended to resist regularization (Lieberman et al., 2007).

However, the MLH has an important shortcoming (at least in its simplest form). As noted by Hay (2003), language learners have little trouble learning thousands of monomorphemic words from very few examples. Many of these monomorphemic words are less common than words which have regularized over historical time. For example, the word help was once irregular (Lieberman et al., 2007), but appears nearly 7000 times in the CELEX sample used in Chapter 7—much more frequently than thousands of monomorphemic words in this sample. These facts show that regularization cannot simply result from irregular forms having insufficient frequency to retain in memory. At the very least, the MLH must be augmented with some notion of a tradeoff between the strength of a memory-trace for each irregular word and the degree to which it is attracted into the regular class.

Such a tradeoff-based account is offered by Prasada and Pinker (1993) who observe that irregular forms tend to be phonotactically more natural than irregular forms, and are, therefore, more easily processed than regulars (on average). Because high-frequency forms have a larger effect on average processing time, there is more pressure to ensure that they are easier to process. Furthermore, because irregulars are often more phonotactically natural than their corresponding regular would be (if it existed), there is pressure for high-frequency forms to remain (or become) irregular. They note, however, that some forms must be regular because the existence of a productive rule is a necessity in order to allow past-tense marking to generalize to new forms. Thus, the correspondence between irregularity and frequency follows from a tradeoff between the need for productivity and a general pressure to increase average processability.

In the following section, I show that the Fragment Grammars model also gives rise to a tradeoff whereby there is a preference for irregularity for high-frequency forms. Unlike Prasada and Pinker’s account, however, this tradeoff does not involve a processing-cost component. It arises purely from considerations of productivity and reuse.

8.2.2 The Prior Probability of Irregularity under Fragment Grammars

The simulations presented in Chapters 5 and 7 focused on the question: What can be inferred about productivity and reuse from the distribution of morphemes in the domains of the English past tense and derivational morphology. The present discussion is focused on a different kind of question: Why might there be a general correspondence between regularity and frequency? To answer this latter we must examine the prior distribution over possible languages defined by the Fragment Grammars model.

The prior probability of a data set is known as its marginal likelihood—the probability of the data, summed over all of the possible ways that it could have been generated. To examine the marginal likelihood of irregularity, I conducted a set of simple artificial simulations. I generated a series of data sets of artificial words with the structure stem -suffix. Each data set contained 1000
potential stems. Each stem was assigned a rank, \( r \), and datasets of word tokens were generated with frequencies following a power law distribution with exponent 1.8 (based on the estimate of the power law exponent for natural language word–frequency distributions from Goldwater et al., 2009).

\[
P(r) \propto \frac{1}{r^{1.8}}
\]

Each stem appeared with one of two suffixes: the regular and the irregular suffix. In each dataset, the irregular suffix appeared with exactly one stem type. I systematically varied the rank of this irregular item between datasets. Thus, for some datasets, the irregular stem was a highly frequent item; for others, it was an item of intermediate frequency; and, for others, it was a low–frequency item. I also included a training data set for which all forms were regular.

Marginal likelihoods are notoriously hard to compute because they involve a sum over all possible ways that the data could have been generated. For these simulations, I used an approximation which summed the top twenty (distinct) posterior modes discovered by the Fragment Grammars sampler. Although this is a rather crude approximation, the small size of the training data sets means that it is highly likely that each sampler found several grammars which were near the maximum a posteriori (i.e., best) grammar for each data set in posterior space. The posteriors in these simulations appear to be steeply peaked, and, thus, these high–probability solutions should represent a considerable proportion of the mass in the complete marginal likelihood sums.

The underlying CFG was identical for all simulations and included rules for each possible stem and for both the regular and irregular suffix. Therefore, the only factor which varied between simulations was the frequency distribution of regular and irregular forms in the input.

The results of these simulations are plotted in Figure 8.1. The y–axis of the plot in Figure 8.1 represents the marginal likelihood (i.e., prior probability) of each artificial language as the rank of the irregular stem is varied along the x–axis. The black line represents the values for the languages that have a single irregular form. I have also included the marginal likelihood of the completely regular language as a red line extending across the plot. Note, however, that this language does not vary with the x–axis; I have extended it across the plot to ease comparison to the languages with irregular forms.

The important thing to note about this plot is the relation between the black and red lines. At the right side of the plot, the irregular form has very low frequency and the resulting language has low marginal likelihood. For these languages the black line appears below the red line, indicating that Fragment Grammars prefer a perfectly regular language to one in which very low–frequency irregular forms appear. However, as the rank and, therefore, the frequency of the irregular form increases, the prior probability of the resulting language sharply increases, becoming more probable than the completely regular language. In sum, Fragment Grammars prefer high–frequency forms to be irregular, and this preference is stronger for more frequent forms.

There is a simple intuition behind this behavior. Under the Fragment Grammar model, novelty is the only form of evidence for productivity. Repetition is evidence for reuse. This means that, after the first time it is used, each regular form counts against the productivity of the regular suffix. If a regular form is repeated a sufficient number of times, it will be stored and no longer contribute to the productivity of the regular suffix. If a sufficient number of regular forms are stored in this way, they will begin to detract from the productivity of the regular rule. However, when high–frequency forms are irregular, they cannot be analyzed using the regular rule, and, therefore,

\( ^2 \)The exact strength of this effect depends on the precise details of the word–frequency distribution and CFG base system. Therefore, the word–frequency rank at which irregularity becomes preferred in these simulations is not necessarily reflective of predictions for natural–language word frequency distributions and grammars. What is important is that there is a point at which some irregular high–frequency forms will be preferred to a fully regular language.
they cannot detract from the rule’s productivity. Put another way, since high-frequency forms are likely to be stored regardless of whether they are regular or irregular, it is preferable that they do not share structure with the regular rule. Moreover, an irregular form requires correspondingly less evidence to store than a form which has an alternate compositional analysis. Thus there are two ways in which high-frequency irregulars make the system more probable. First, they reduce the degree to which forms—which were going to be stored anyway—detract from the productivity of the regular rule. Second, they make it easier to learn that those forms should be stored: Irregularity serves as an additional signal to the system that a form should be stored.

8.2.3 Discussion

In the preceding section, I showed that Fragment Grammars predict a correlation between frequency and irregularity that arises from the way in which they balance the tradeoff between productivity and reuse. For the highest probability languages (under the model), high-frequency forms will not share structure with the productively–computed forms in the long tail of the word frequency distribution. From the perspective of learning, the highest prior probability languages require the least evidence to learn correctly, and, therefore, languages in which high-frequency
forms are irregular will be more learnable by the model. In the remainder of this section, I consider a number of additional questions which arise from the prediction described above. First, I consider whether the acquisition of fully–regular morphological processes, such as the English progressive marker (e.g., *-ing*), provides evidence against the prediction in the last section. Second, I discuss two related issues: what counts as irregularity, and why do irregulars often share structure? Third, I review some recent mathematical work which provides a connection between the prior distribution over data defined by Bayesian models (i.e., the marginal likelihood) and the cultural evolution of language. Fourth, and last, I discuss the issue of why natural language frequency distributions are often characterized by a small number of highly–frequent forms and large number of infrequent forms, that is, why they often have the form of a power law.

**Fully–Regular Rules** The results in the preceding section predict that linguistic systems where high–frequency forms are irregular will be easier to learn than fully–regular systems. This prediction appears to be at odds with evidence that children acquire some fully–regular generalizations earlier than they acquire generalizations which include exceptional cases. An example of such a case is the English progressive marker, *-ing*, which is completely exceptionless, and appears to be acquired earlier than the past tense (which contains exceptions) by children (see Brown, 1973; Pinker, 1981). This example may present a serious challenge to the prediction in the last section.

One possible resolution to this problem may lie in the difference between identification of potential generalizations and inferences about productivity. Under Fragment Grammars, when high–frequency forms are irregular, it will be easier to learn which forms should be stored and which should be computed by rule. Clearly, however, to acquire a language, children must learn much more than which forms are stored and which are computed. In particular, they must learn the structure of the specific rules which their language uses. When determining that some phonological sequence, such as *-ing*, is a candidate for a suffixation rule, regularity is likely to be highly advantageous. The early emergence of the progressive marker may be due to the ease with which the candidate generalization can be identified, rather than to the inference that this generalization is productive. If this account is correct, then inferences about productivity and identification of generalizations may represent opposing biases in language learning. Identification of generalizations will favor systems where all forms are regular, whereas inferences that a generalization is productive will favor systems where the highest frequency forms are irregular. Whether or not such an account can reconcile the prediction made above with cases such as the English progressive is a question for future work.

**What Counts as Irregularity?** A second question which arises from the prediction outlined in the last section is: What ways can high–frequency forms be irregular and, thus, lead to more probable languages? The prediction in the preceding section is extremely general. Any kind of irregularity which prevents sharing of structure with the long–tail of productive forms will suffice. This could mean complete phonological idiosyncrasy (e.g., *goed*/*went*), or the use of rules whose selectional restrictions are different from the productive rule (e.g., */i/ → */w/ in the context of */ŋ/). In either case, the high–frequency form will not share structure with the productive generalization, and, thus, will not detract from its productivity.

---

3 This account is similar to the tradeoff–based account of Prasada and Pinker (1993), except that the advantage for high–frequency irregulars arises as the result of a learning bias, rather than a processing bias. It should be noted, however, that the two accounts are not mutually exclusive. Irregular high–frequency forms can obviously be both more learnable and more processable.
The Structure of Irregulars  A related question is: Why do irregular forms sometimes share structure? For example, why do irregular past tense forms exhibit family resemblance structure (see Sections 4.3 and 4.4.3)? The prediction above seems to imply that the system would assign high probability to a language where each high–frequency form was irregular in its own way. In the absence of a model which incorporates phonological and semantic structure, a solution to this problem must be tentative, but one possibility is as follows.

Recall that Fragment Grammars can be understood as being composed of two layers. The top layer consists of stored, reusable derivation fragments, and the bottom layer is the underlying PCFG which is used when constructing new stored fragments and which is estimated based on the type frequencies of subparts of stored structures. Once the derivation of an irregular form is stored, its internal structure no longer has an effect in the overall operation of the top–layer of the system. However, an irregular form must be analyzed correctly the first time it is encountered. This will be easiest when the irregular shares structure with other stored forms. So long as this shared structure is not shared with productively computed forms, it will not compete with the productive rule, but will make learning irregulars easier.

Notice, also, that there is a tradeoff involved in these observations. If too many irregular forms share the same structure, they may begin to provide evidence for a second productive rule; if too few share structure, the set of irregular forms will be more difficult to learn. Therefore, it could be that more probable languages (under the model) organize high–frequency forms into a number of irregular subclasses. Although this prediction is speculative, if it proves correct, it could provide an explanation for the family resemblance structure among irregulars in systems such as the English past tense.

Connections to the Cultural Evolution of Language  Over the last fifteen years there has been much interest in the evolutionary dynamics of language change on a cultural time scale (e.g., Berwick and Niyogi, 1996; Griffiths and Kalish, 2007; Komarova and Niyogi, 2004; Niyogi and Berwick, 2009; Nowak et al., 2001, 2002; Plotkin and Nowak, 2000; Reali and Griffiths, 2009). A natural question which arises in this context is how does a prior bias, like that described above, relate to such dynamics? A series of recent mathematical results have shed light on this problem. It can be shown that when learners are Bayesian agents, as the linguistic system is repeatedly learned and relearned, the properties of learners’ prior biases become properties of the distribution over languages.

Imagine that a Bayesian agent constructs a posterior distribution over possible languages given the input received from the preceding generation. Griffiths and Kalish (2007) showed that, in the simple case where each agent learns from a single agent in the preceding generation, if the learner samples a language from its posterior distribution, and, then, uses this sampled language to generate data for the next generation, the system will converge in the limit to the starting prior distribution over languages. If, instead, each agent chooses the most probable language from the posterior distribution (rather than sampling), Griffiths and Kalish (2007) showed that the system will converge to a state which exaggerates prior biases: More probable languages become more probable and less probable languages become less probable over time. Nevertheless, in both cases, the distribution over languages after iterated learning depends on the prior distribution over languages possessed by each learner. These results therefore provide a link between the prior distributions over languages defined by the Fragment Grammars model and the evolutionary dynamics of language change.4

4 These results apply to the simplified scenario of a single chain of learners; however, ongoing work is exploring the space of evolutionary dynamics with these assumptions relaxed in many different ways (see e.g., Burkett and Griffiths, 2010; Ferdinand and Zuidema, 2009; Griffiths et al., 2006; Griffiths and Kalish, 2007; Kirby et al., 2007; Smith, 2009; Smith and Kirby, 2008). Dynamics can change significantly in population settings, but appear to still depend (in
The Origins of Skewed Word–Frequency Distributions  The model–based prediction that frequency should be correlated with irregularity is predicated on two assumptions. First, there must be a large number of infrequently observed forms so that a productive rule is advantageous. Second, some forms must be sufficiently frequent to be stored by the system. In other words, the word–frequency distribution must be characterized by a steep peak and a long tail: a power law.

Why do natural language word–frequency distributions (often) follow such a power law? There are two classes of (non–circular) explanations. First, these distributions might reflect the structure of the environment. Perhaps the set of meanings that humans need to express involve a small number of frequently repeated messages and a long tail of infrequent, but useful messages. Such a scenario is not implausible: Power–law distributions arise as the consequence of a large number of stochastic processes (Biemann, 2007; Egghe, 2005; Evert, 2004; Gisiger, 2001; Goldwater et al., 2009; Howes, 1968; Mitzenmacher, 2003; Niyogi and Berwick, 1995; Simon, 1955). If aspects of the natural environment can be described by such processes, then it may be plausible that the power–law distribution of natural language structures have such an external explanation.5

A second possibility is that power–law distributions arise as a consequence of the properties of computation in the brain. Under one version of this class of theories, this is accidental. However, a more interesting possibility is that power–law distributions arise as the best, or most natural (in some relevant mathematical sense) solutions to the problem of simultaneously balancing the ability to handle a large number of novel observations, while accounting for reuse. Ferrer I Cancho and Solé (2003), for example, argue that power–law distributions can be derived mathematically from Zipf’s principle of least effort.

Whatever the correct explanation of the power–law structure of (many) natural language frequency distributions, when such distributions do occur, Fragment Grammars predict that high–frequency forms will be tend to be irregular.

8.3 The Emergence of Productive Generalizations during Development

There has been a persistent belief in language acquisition that the availability of an abstract generalization implies that the generalization must be used productively by children. For example, McClelland and Patterson (2002) quote Brown (1973, p. 275).

There is always a considerable period in which production–when–required is probabilistic. This is a fact that does not accord well with the notion that the acquisition of grammar is a matter of the acquisition of rules, since the rules either apply or do not apply. One would expect rule acquisition to be sudden.

The intuition behind this idea is that if a child possesses combinatorial rules which abstract over classes of inputs then he or she will immediately begin to produce utterances that make use of such rules. The fact that this prediction sometimes fails is often taken to mean that the child does not possess abstract representations. For example, Braine (1976) extensively explored the failure of this assumption in the context of phrase–structure rules. More recently, similar reasoning has driven much research in usage–based models of language acquisition (Tomasello, 2000).

complex ways) on the relative probability of possible languages in the prior distribution.

5Anderson and Schooler (1991) explicitly pursue the argument that the structure of human memory arises out of power–law distributions in the environment. However, the data which they use to support this claim is natural language data, and, therefore, renders their argument circular from the point of view of the present discussion.
I argued extensively in the introduction of this dissertation that adult knowledge of language requires an account of how different linguistic processes can have differential productivity—and that this account can be orthogonal to the issue of representation. In subsequent chapters, I have argued for the value of such an account in the domains of the English past tense and English derivational morphology. In fact, such an account may be a logical necessity given that the learner must move from a state where any generalization present in the world’s languages can be represented in principle, to a state where only productive generalizations licensed in a particular language are represented.

Any abstract generalization which is productive in some language must be available (possibly latently) in the initial hypothesis space of every learner. What does a theory that treats productivity as an inference predict about how the learner will use these potential abstract generalizations? Under such an approach the purpose of productive rules is to be able to produce or comprehend novel utterances. This means that the kind of evidence which is most diagnostic of productivity is—by its nature—rarer than the kind of evidence available for reuse. Each particular observation in the primary linguistic data can only be novel once, and, of course, because the system is combinatorial, most structure is not novel, but, rather, is reused. Thus, it will take time for a learner to see enough instances of a process in novel contexts to infer that the process is productive. The mere existence of a potential abstraction does not predict that it will be immediately used.

There are many more generalizations consistent with the data than will be available productively in the adult state. Suppose that the learner did not require substantial evidence that a process was productive, but merely required it to be witnessed once in the data, and then immediately began using it productively. Because only a tiny fraction of the available productive generalizations are correct, most of the hypotheses that the learner considered would be wrong, and there would be no way to recover from these errors. Conversely, since the evidence is finite, only a small subset of structures potentially available in all human languages will be actually witnessed in the primary linguistic data, and the primary linguistic evidence available to a child is always highly unlikely a priori. Therefore, the learner is warranted in taking evidence of reuse at face value.

The English past tense, considered in Chapters 4 and 5, provides a clear illustration of how even a rule–based theory should predict conservativity during learning. English language learners must eventually arrive at a state where they have available an abstract and productive regular rule (i.e., +/d/) and a number of irregular inflectional classes of lesser productivity (e.g., /i/ → /æ/).

Under an inference–based theory of productivity and reuse, it will be the case that evidence for the lack of productivity of irregulars accrues more quickly than evidence for the productivity of regulars. This is a necessary fact about the data given the assumptions that productivity is inferred from novelty and that the regular rule is more productive. If this were not the case then, by assumption, the regular rule would not be productive. Hearing repeated instances of went is strong evidence that this form can be reused in the future, but to infer that +/d/ can apply to novel stems, the child must hear it used together with many stems whose past tense form was not previously encountered. We can expect, therefore, that there will be a lag between the correct use of irregular forms and the productive use of the regular rule, regardless of when the child gains the resources necessary to represent the abstract generalization underlying the latter.

8.4 Future Directions

The preceding chapters suggest two natural extensions to the work presented in this thesis. The first is empirical. In Chapter 5, I showed that Fragment Grammars use the whole-form and stem frequencies of past tense forms in a way which is consistent with the results of naturalness judgment, speeded naming, and lexical decision experiments (e.g., Alegre and Gordon, 1999a; Beck, 1997; Gordon and Alegre, 1999; Prasada et al., 1990; Seidenberg and Bruck, 1990; Shenkman, 1994;
Chapter 8: Conclusion

Ullman, 1993, 1999). However, I was unable to correlate model predictions directly with empirical data because of the lack of wide-coverage datasets of such experimental measures (especially naturalness judgments). Similarly, in Chapter 7, with the exception of a few specific examples, my discussion of productivity and parsability was limited to exploratory analyses and correlation with Baayen’s corpus-based measures $P$ and $P^*$. In both cases, the availability of large-scale, wide-coverage datasets of naturalness judgments (or wug-tests) would make formal, empirical evaluation of the models possible. Such datasets would also be valuable to the psychology community at large. The collection of such wide-coverage corpora of naturalness judgments is made more feasible by the possibility of web-based data collection.

A second natural extension to the work in this thesis would be the representation (and learning) of phonological structure associated with individual morphemes. In my developmental evaluation of the past tense in Chapter 5, I noted that, because Fragment Grammars could not represent monomorphemic words and did not model the process of acquiring the segmentations and phonological content of past-tense allomorphs, they were unable to predict a robust period of early correct performance or a correlation between reliable regular marking and overregularization. I also noted that it was impossible to evaluate the degree to which Fragment Grammars predicted that phonological similarity protected irregulars from overregularization, for similar reasons. In Chapter 7, I noted that the problem of constructing an appropriate corpus for the domain of derivational morphology might be circumvented by the use of an unsupervised model which combined Fragment Grammars and a rule-induction model. Finally, in this chapter, I tentatively suggested that Fragment Grammars’ bias in favor of irregularity for high-frequency forms might predict phenomena such as the family resemblance structure of irregular past tenses. Extension of Fragment Grammars to include phonological structure—along the lines sketched in Appendix A—would extend the empirical phenomena which could be covered by the model.

8.5 Conclusion

At its heart, the theory presented in this thesis is based on a simple idea: The learner infers the productivity of a linguistic process by observing novel forms which use the process (much like the language scientist), and infers the reusability of combinations of linguistic structures by observing their repeated use. Although this idea is simple, when combined with the machinery of traditional rule-based linguistic representations, it has surprisingly broad implications. A formal model built on this idea, Fragment Grammars, was able to provide accounts of two very different subsystems of English morphology. On one hand, it could capture the pattern of defaultness and blocking in the English past tense, a system characterized by a highly productive default rule and a number of (mostly) unproductive irregular classes. On the other hand, it could capture phenomena such as suffix ordering and base-driven selectional restrictions in English derivational morphology, a system characterized by a broad array of processes of differing levels of productivity and regularity.

Fragment Grammars make (approximately) optimal use of the information available in the distribution of morphemes in the input data. Their success suggests that it is plausible that human learners also make (approximately) optimal use of this information during language acquisition. Of course, the results in this thesis have focused on a very specific set of phenomena involving productivity and reuse. Whether human learning appears to be optimal when other learning problems are considered is a question for future research. More broadly, however, the inference-based perspective seems to be consistent with a fundamental problem solved by the linguistic system: This system must provide the flexibility to produce and comprehend new thoughts while maintaining the ability to specialize for commonly re-encountered situations.
Bibliography


Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press. 95


Daugherty, K. G. and Hare, M. L. (1993). What’s in a rule? The past tense by some other name might be called a connectionist net. In *Proceedings of the 1993 Connectionist Models Summer School*. 95


vii, 148, 149, 155, 156, 160, 161, 168, 174, 182


Bibliography


Integrating Phonology

Consider a context–free grammar (CFG) rule like $A \rightarrow B \ C$. This rule specifies that a constituent of category $A$ consists of a constituent of category $B$ concatenated to a constituent of category $C$. To make this more clear we might write the rule as $A \rightarrow \text{concatenate}(B, C)$. This notation makes the concatenation of the outputs of $B$ and $C$ explicit. It also makes clear that some other function can substituted for concatenate: $A \rightarrow f(B, C)$. This function, $f$, could perform more elaborate operations. For instance, it could change the order of the subconstituents, delete a subconstituent, or change the internal structure of one or both subconstituents.

A version of such a system could be used to integrate phonological structure into the models used in this thesis. For example, a number of derivational affixes cause stress shift in their bases (e.g., -ity: normality v. normalness). This phenomenon could be represented as the composition of two functions, concatenate and shift-stress, applied to the right–hand side of the rule which appends -ity to words: $N \rightarrow \text{shift-stress}(\text{concatenate}(\text{Adj}, \text{-ity}))$. By starting with an inventory of primitive functions, such as concatenate, shift-stress, lax-stem-vowel, delete-final-consonant, and so on, a generative model can be defined over possible values for $f$ in rules of the form: $A \rightarrow f(B, C)$. Such a generative model could replace the underlying CFG used by Fragment Grammars, and, thus, open the possibility of learning phonological structure simultaneously with the pattern of productivity and reuse.

---

1This perspective lies behind many mildly-context sensitive formalisms for syntax. Weir (1988) showed how a number of formalisms proposed for syntax were equivalent to linear context-free rewrite systems (LCFRS). A LCFRS is a context–free grammar where each production is equipped with an additional function, like $f$ above, which can operate on the string representations of the constituents sampled for each element of the right–hand side. If $f=\text{concatenate}$ then an LCFRS reduces to a CFG. However, in general the class of LCFRS grammars is much richer than simple CFGs. Precisely speaking, LCFRSs generalize the strings generated by CFGs to tuples of strings and allow for linear, non–deleting operations on those tuples. This leads to an infinite hierarchy of languages—in terms of weak generative capacity—which is given by the maximum arity of the tuples. This formalism has in turn been proven equivalent to a large array of proposed linguistic generative models, such as Derivational Minimalist Grammars, Set-Local Tree Adjoining Grammar, Multiple Context-free Grammars, certain restrictions of Abstract Categorial Grammars, the class of languages describable by weak monadic second-order logic with more than one successor relation, and (when restricted to 2-tuples) Tree Adjoining Grammars, Head Grammars, and Combinatory Categorial Grammars (de Groote, 2001; Rogers, 2003; Stabler, 2009; Steedman, 2000).
Appendix B

Past Tense Inflectional Classes

In this Appendix, I show the regular and irregular rules used for forms in the simulations presented in Chapter 5.
<table>
<thead>
<tr>
<th>Tense</th>
<th>Stem Change</th>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>VBD</td>
<td>/u/ → /œ/</td>
<td>+/t/</td>
<td>lost</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /æ/</td>
<td>0</td>
<td>rang</td>
</tr>
<tr>
<td>VBD</td>
<td>/æ/ → /æ/</td>
<td>0</td>
<td>hung</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/d/</td>
<td>redid</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>led</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>took</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>fought</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>gave</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>ate</td>
</tr>
<tr>
<td>VBD</td>
<td>go/went</td>
<td>go/went</td>
<td>went</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /æ/</td>
<td>0</td>
<td>ran</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>taught</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/t/</td>
<td>brought</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>grew</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>lay</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>lit</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/d/</td>
<td>said</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>beat</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>foresaw</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>withheld</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>got</td>
</tr>
<tr>
<td>VBD</td>
<td>delete–final–consonant</td>
<td>+/d/</td>
<td>made</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>won</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>flew</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>experimented</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>understood</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>shot</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>fell</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/d/</td>
<td>sold</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/t/</td>
<td>bought</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>wound</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>came</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>struck</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/t/</td>
<td>caught</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/t/</td>
<td>left</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>+/d/</td>
<td>heard</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>stole</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>wrote</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>drew</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>chose</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>sent</td>
</tr>
<tr>
<td>VBD</td>
<td>/œ/ → /œ/</td>
<td>0</td>
<td>woke</td>
</tr>
</tbody>
</table>

Table B.1: Inflectional Classes from SWBD Simulations:
### Appendix B: Past Tense Inflectional Classes

<table>
<thead>
<tr>
<th>Tense</th>
<th>Stem Change</th>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>VBN</td>
<td>/aj/ → /au/</td>
<td>∅</td>
<td>wound</td>
</tr>
<tr>
<td>VBN</td>
<td>/ej/ → /ou/</td>
<td>+/n/</td>
<td>worn</td>
</tr>
<tr>
<td>VBN</td>
<td>/e/ → /oo/</td>
<td>+/-d/</td>
<td>sold</td>
</tr>
<tr>
<td>VBN</td>
<td>∅</td>
<td>+/-d/</td>
<td>incurred</td>
</tr>
<tr>
<td>VBN</td>
<td>/e/ → /a/</td>
<td>+/-n/</td>
<td>gotten</td>
</tr>
<tr>
<td>VBN</td>
<td>/œn/ → /o/</td>
<td>∅</td>
<td>understood</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /x/</td>
<td>∅</td>
<td>won</td>
</tr>
<tr>
<td>VBN</td>
<td>/ou/ → /x/</td>
<td>∅</td>
<td>withheld</td>
</tr>
<tr>
<td>VBN</td>
<td>/œf/ → /a/</td>
<td>+/-t/</td>
<td>taught</td>
</tr>
<tr>
<td>VBN</td>
<td>/aj/ → /a/</td>
<td>∅</td>
<td>fought</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /x/</td>
<td>+/-t/</td>
<td>left</td>
</tr>
<tr>
<td>VBN</td>
<td>/u/ → /a/</td>
<td>∅</td>
<td>shot</td>
</tr>
<tr>
<td>VBN</td>
<td>/œ/ → /x/</td>
<td>∅</td>
<td>hung</td>
</tr>
<tr>
<td>VBN</td>
<td>/œf/ → /a/</td>
<td>+/-t/</td>
<td>caught</td>
</tr>
<tr>
<td>VBN</td>
<td>/ej/ → /x/</td>
<td>+/-d/</td>
<td>said</td>
</tr>
<tr>
<td>VBN</td>
<td>/u/ → /ou/</td>
<td>+/-n/</td>
<td>chosen</td>
</tr>
<tr>
<td>VBN</td>
<td>delete–final–consonant</td>
<td>+/-d/</td>
<td>made</td>
</tr>
<tr>
<td>VBN</td>
<td>/ou/ → /a/</td>
<td>+/-n/</td>
<td>gone</td>
</tr>
<tr>
<td>VBN</td>
<td>/aj/ → /i/</td>
<td>+/-n/</td>
<td>ridden</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /œ/</td>
<td>∅</td>
<td>sat</td>
</tr>
<tr>
<td>VBN</td>
<td>/d/ → /œ/</td>
<td>∅</td>
<td>rebuilt</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /œ/</td>
<td>∅</td>
<td>led</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /œ/</td>
<td>+/-d/</td>
<td>heard</td>
</tr>
<tr>
<td>VBN</td>
<td>/aj/ → /ou/</td>
<td>+/-n/</td>
<td>floun</td>
</tr>
<tr>
<td>VBN</td>
<td>/u/ → /œ/</td>
<td>+/-t/</td>
<td>brought</td>
</tr>
<tr>
<td>VBN</td>
<td>/i/ → /oo/</td>
<td>+/-n/</td>
<td>spoken</td>
</tr>
<tr>
<td>VBN</td>
<td>/u/ → /œ/</td>
<td>+/-n/</td>
<td>overdone</td>
</tr>
<tr>
<td>VBN</td>
<td>∅</td>
<td>+/-n/</td>
<td>drawn</td>
</tr>
<tr>
<td>VBN</td>
<td>∅</td>
<td>∅</td>
<td>run</td>
</tr>
<tr>
<td>VBN</td>
<td>/aj/ → /œ/</td>
<td>∅</td>
<td>struck</td>
</tr>
<tr>
<td>VBN</td>
<td>/aj/ → /œ/</td>
<td>+/-t/</td>
<td>bought</td>
</tr>
<tr>
<td>VBN</td>
<td>∅</td>
<td>+/-t/</td>
<td>burnt</td>
</tr>
<tr>
<td>VBN</td>
<td>/u/ → /œ/</td>
<td>+/-t/</td>
<td>lost</td>
</tr>
</tbody>
</table>

Table B.2: **Inflectional Classes from SWBD Simulations**:

<table>
<thead>
<tr>
<th>Tense</th>
<th>Stem Change</th>
<th>Suffix</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>VB</td>
<td>∅</td>
<td>∅</td>
<td>drive</td>
</tr>
<tr>
<td>VBG</td>
<td>∅</td>
<td>+/-u/</td>
<td>continuing</td>
</tr>
<tr>
<td>VBP</td>
<td>∅</td>
<td>∅</td>
<td>follow-up</td>
</tr>
<tr>
<td>VBZ</td>
<td>/ej/ → /œ/</td>
<td>+/-z/</td>
<td>says</td>
</tr>
<tr>
<td>VBZ</td>
<td>∅</td>
<td>+/-z/</td>
<td>favors</td>
</tr>
</tbody>
</table>

Table B.3: **Inflectional Classes from SWBD Simulations**:
Appendix C

Derivational Suffixes

This Appendix contains a list of English derivational suffixes which appeared in at least 10 words types in the simulations reported in Chapter 7.
### Table C.1: Derivational Suffixes 1

<table>
<thead>
<tr>
<th>String Form</th>
<th>Input Category</th>
<th>Output Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>-able</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-able</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-able</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-age</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-age</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-al</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-al</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-al</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-al</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-al</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-al</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-al</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-al</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-an</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-an</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-an</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-an</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-an</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-an</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ance</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ance</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ance</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ant</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ant</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ant</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ant</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ant</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ary</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ary</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ary</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ary</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ary</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ate</td>
<td>A</td>
<td>V</td>
</tr>
<tr>
<td>-ate</td>
<td>BND</td>
<td>V</td>
</tr>
<tr>
<td>-ate</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ate</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ate</td>
<td>N</td>
<td>V</td>
</tr>
<tr>
<td>-ate</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ate</td>
<td>V</td>
<td>V</td>
</tr>
</tbody>
</table>
### Table C.2: Derivational Suffixes 2

<table>
<thead>
<tr>
<th>String Form</th>
<th>Input Category</th>
<th>Output Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ation</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ation</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-dom</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-dom</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ed</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ee</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ee</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-eer</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-en</td>
<td>A</td>
<td>V</td>
</tr>
<tr>
<td>-en</td>
<td>BND</td>
<td>V</td>
</tr>
<tr>
<td>-en</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-en</td>
<td>N</td>
<td>V</td>
</tr>
<tr>
<td>-en</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ence</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ence</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ence</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ent</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ent</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ent</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ent</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-er</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-er</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-er</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-er</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ery</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ery</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ese</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ese</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ess</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ette</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ety</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-fold</td>
<td>Q</td>
<td>A</td>
</tr>
<tr>
<td>-ful</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ful</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ful</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-hood</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ic</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ic</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ic</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ic</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ic</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ier</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ier</td>
<td>V</td>
<td>N</td>
</tr>
</tbody>
</table>
### Table C.3: Derivational Suffixes 3

<table>
<thead>
<tr>
<th>String Form</th>
<th>Input Category</th>
<th>Output Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ify</td>
<td>A</td>
<td>V</td>
</tr>
<tr>
<td>-ify</td>
<td>BND</td>
<td>V</td>
</tr>
<tr>
<td>-ify</td>
<td>N</td>
<td>V</td>
</tr>
<tr>
<td>-ion</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ion</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ion</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ion</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ish</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-ish</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ism</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ism</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ism</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ist</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ist</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ist</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ist</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-itis</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ity</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ity</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ity</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ity</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ive</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-ive</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ive</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ive</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ize</td>
<td>A</td>
<td>V</td>
</tr>
<tr>
<td>-ize</td>
<td>BND</td>
<td>V</td>
</tr>
<tr>
<td>-ize</td>
<td>N</td>
<td>V</td>
</tr>
<tr>
<td>-less</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-less</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-let</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-let</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-like</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ling</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ly</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-ly</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>-ly</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ly</td>
<td>N</td>
<td>B</td>
</tr>
<tr>
<td>-ly</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ly</td>
<td>V</td>
<td>B</td>
</tr>
<tr>
<td>-ment</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ment</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ment</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-most</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>String Form</td>
<td>Input Category</td>
<td>Output Category</td>
</tr>
<tr>
<td>-------------</td>
<td>----------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>-ness</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ness</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-oid</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-oid</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ology</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-ology</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-or</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-or</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-or</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-or</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ory</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ory</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ous</td>
<td>BND</td>
<td>A</td>
</tr>
<tr>
<td>-ous</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ous</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-proof</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ry</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-ry</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ship</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-some</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-some</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-some</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-ster</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ster</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-th</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-th</td>
<td>Q</td>
<td>Q</td>
</tr>
<tr>
<td>-th</td>
<td>V</td>
<td>N</td>
</tr>
<tr>
<td>-ty</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-ty</td>
<td>Q</td>
<td>Q</td>
</tr>
<tr>
<td>-ward</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>-ward</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-ward</td>
<td>N</td>
<td>B</td>
</tr>
<tr>
<td>-wise</td>
<td>N</td>
<td>B</td>
</tr>
<tr>
<td>-y</td>
<td>A</td>
<td>A</td>
</tr>
<tr>
<td>-y</td>
<td>A</td>
<td>N</td>
</tr>
<tr>
<td>-y</td>
<td>BND</td>
<td>N</td>
</tr>
<tr>
<td>-y</td>
<td>N</td>
<td>A</td>
</tr>
<tr>
<td>-y</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>-y</td>
<td>V</td>
<td>A</td>
</tr>
<tr>
<td>-y</td>
<td>V</td>
<td>N</td>
</tr>
</tbody>
</table>

Table C.4: Derivational Suffixes 4