

**USING MODEL THEORY IN PHONOLOGY:
A NOVEL CHARACTERIZATION OF SYLLABLE STRUCTURE AND
SYLLABIFICATION**

by

Kristina Strother-Garcia

A dissertation submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Linguistics

Summer 2019

© 2019 Kristina Strother-Garcia
All Rights Reserved

USING MODEL THEORY IN PHONOLOGY:
A NOVEL CHARACTERIZATION OF SYLLABLE STRUCTURE AND
SYLLABIFICATION

by

Kristina Strother-Garcia

Approved: _____

Benjamin Bruening, Ph.D.
Chair of the Department of Linguistics and Cognitive Science

Approved: _____

John Pelesko, Ph.D.
Dean of the College of Arts and Sciences

Approved: _____

Douglas J. Doren, Ph.D.
Interim Vice Provost for Graduate and Professional Education and
Dean of the Graduate College

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Jeffrey Heinz, Ph.D.
Professor in charge of dissertation

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Irene Vogel, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Kathryn Franich, Ph.D.
Member of dissertation committee

I certify that I have read this dissertation and that in my opinion it meets the academic and professional standard required by the University as a dissertation for the degree of Doctor of Philosophy.

Signed: _____

Adam Jardine, Ph.D.
Member of dissertation committee

ACKNOWLEDGMENTS

This dissertation would not have been possible without the support and guidance of my committee, colleagues, family, and friends. There are far too many people to thank, but I will do my best. I am especially grateful to my advisor, Jeff Heinz, for being a captivating instructor, supportive mentor, and top-notch colleague. His passion for science is contagious, and I never would have ventured into the realm of computational phonology had it not been for his seminars at UD. He also introduced me to several of the brilliant minds whose names appear below, fostering connections that have influenced my career and my personal life for the better.

Having Irene Vogel, Adam Jardine, and Katie Franich on my committee has been a privilege. I cannot overstate the value of Irene's critical eye toward this work. She challenges me to carefully consider the broader implications of my analyses, and her expertise has greatly improved this dissertation.

It has been amazing to see Adam go from a fellow student at UD to a professor at Rutgers, and I am so glad he was able to serve on my committee. Not only has he aided with the technical aspects of my analyses, but he has also helped me learn how to present my work more effectively and persuasively. He never lets me undersell my accomplishments, and I truly appreciate that.

When I asked Katie to join my committee at nearly the last minute, she agreed without batting an eye. She brought a fresh perspective to the table, and this dissertation has certainly benefited from her feedback. I would also like to thank Katie for attending the doctoral hooding ceremony to initiate me into the Ph.D. club.

To my cohort, I thank you all for walking through the fire with me: Abdullah Alghamdi, Mai Ha Vu, Myrto Grigoroglou, Ryan Rhodes, and Young-Eun Kim. I don't think I could have survived my first year at UD without you.

Extra thanks are due to Mai Ha for her endless support, as both a colleague and a beloved friend. I would also like to acknowledge Hossep Dolatian, who provided technical and moral support—along with his amazing sense of humor—and for proofreading this dissertation. There are many other students and recent graduates of the department who have contributed to my success and well-being: Hung Shao Cheng, Lexi Robbins, Olga Parshina, Taylor Miller, Hyunjin Hwangbo, Amanda Payne, and Stefan Bartell, to name just a few.

I am also grateful to Jane Creswell and Laura Edmanson for keeping all the plates spinning. I thank Nina Straitman for much wisdom, many needed laughs, and lots of lunches.

I would be remiss not to mention my colleagues and mentors at the Child's Play, Learning and Development Lab, who were instrumental in my development as a scientist and a scholar: Roberta Michnick Golinkoff, Giovanna Morini, Haruka Konishi, Natalie Brezack, Kate Ridge, Brian Verdine, Ilyse Resnick, Vinaya Rajan, and Ratna Nandakumar. Thank you for all that you taught me.

To my colleagues at the Treatment Efficacy and Language Learning Lab, I truly appreciate your encouragement and positivity: Amanda Owen Van Horne, Maura Curran, and our team of undergraduate research assistants. Thanks also to Emily Fritzson, Julie Campagna, Aquiles Iglesias, and the entire department of Communication Sciences and Disorders for welcoming and supporting me.

I would also like to acknowledge my collaborators, coauthors, and those who have brainstormed with me over the years: Kathy Hirsh-Pasek, Dani Levine, Bert Tanner, Ashkan Zehfroosh, Bill Idsardi, Jim Rogers, Eric Baković, Jane Chandlee, Thomas Graf, Kevin McMullin, Eric Raimy, Remi Eyraud, Ben Parrell, Jon Rawski, Aniello

de Santo, Alëna Aksënova, Nick Danis, and Eileen Blum. There are many others who deserve thanks, no doubt—I hope you will all forgive me for cutting the list short here.

Finally, I would have never made it this far without the support of my family and friends. I cannot thank you all enough. My parents, Carol and Todd Smith, deserve special mention for everything they have done for me. You have my undying gratitude.

To my husband, Michael: your confidence in me and your tireless encouragement kept me going when things got tough. Thank you, thank you, thank you. Go Team SG!

TABLE OF CONTENTS

LIST OF TABLES	x
LIST OF FIGURES	xi
ABSTRACT	xvi
 Chapter	
1 INTRODUCTION	1
2 THE SYLLABLE	5
2.1 Evidence for the Syllable	5
2.2 Previous Accounts of the Syllable	6
2.3 Methodological Problems of Previous Approaches	9
2.4 Moving Forward	12
3 FORMAL BACKGROUND	13
3.1 Word Models and Model Theories	13
3.1.1 The Successor Model Theory	15
3.1.2 A Brief Aside: Notational Conventions	17
3.1.3 The Precedence Model Theory	18
3.1.4 Visual Representations	19
3.2 Enriching Conventional Word Models	20
3.2.1 Enriching the Alphabet	20
3.2.2 Enriching the Structure	22
3.3 Transductions	22
3.4 Substructures	24
3.5 Formal Languages and Substructure Constraints	25
3.6 Logics and Locality	27

3.7	User-defined Formulas	29
4	SYLLABLE REPRESENTATIONS	31
4.1	Notational Equivalence	31
4.2	Representations of Syllable Structure	32
4.2.1	The Dot Model Theory	34
4.2.2	The Flat Model Theory	35
4.2.3	The Tree Model Theory	37
4.3	Comparing Different Models	39
4.3.1	L-interpretability	39
4.3.2	The Flat-to-Tree Transduction	40
4.3.3	The Tree-to-Flat Transduction	54
4.3.4	The Flat-to-Dot Transduction	58
4.3.5	The Dot-to-Flat Transduction	66
4.4	Discussion	72
5	UNIVERSAL PRINCIPLES, SONORITY SEQUENCING, AND CV TYPOLOGY	77
5.1	Structural Well-formedness Constraints	77
5.2	Sonority Sequencing	80
5.2.1	Sonority Relations	81
5.2.2	Constraints on Sonority Sequencing	83
5.3	CV Typology	86
6	CASE STUDY: IMDLAWN TASHLHIYT BERBER	90
6.1	Motivation	90
6.2	The Basics	93
6.3	Surface Well-Formedness in ITB	95
6.3.1	Structural Constraints	95
6.3.2	Sonority Sequencing Constraints	98
6.3.3	The Formal Language for ITB	100

6.3.4	Extended Example: /t-xzn-t/	101
6.4	Syllabification in Imdlawn Tashlhiyt Berber (ITB)	104
6.4.1	Sonority and Other Considerations	105
6.4.2	Identifying Syllable Constituents	107
6.4.3	The ITB Syllabification Transduction	107
6.4.4	Extended Example: /saulx/	110
7	CASE STUDY: MOROCCAN ARABIC	112
7.1	The Basics	114
7.2	Surface Well-formedness in MA	115
7.2.1	Structural Constraints	115
7.2.2	Sonority Sequencing Constraints	120
7.2.3	Exceptions to the Epenthesis Pattern	122
7.2.4	The Formal Language for Moroccan Arabic (MA)	126
7.3	Syllabification in MA	127
7.3.1	Identifying (Some) Syllable Constituents	127
7.3.2	Identifying Insertion Sites	128
7.3.3	Identifying the Remaining Syllable Constituents	132
7.3.4	The MA Syllabification Transduction	134
7.3.5	Extended Example: /krk bk/	138
8	DISCUSSION	141
8.1	Computational Complexity	141
8.1.1	Graph Transductions	141
8.1.2	Substructure Constraint Grammars	143
8.2	Other Issues in Syllable Theory	143
8.2.1	Additional Syllable Structures	144
8.2.2	Additional Processes Related to Syllable Structure	146
8.3	Other Directions for Future Work	147
9	CONCLUSION	149

LIST OF TABLES

5.1	Possible Syllable Types in Basic CV Typology	89
6.1	Alternations in ITB Perfective Verb Forms: 2ms vs. 3fs with a dative masc. object	91
6.2	Alternations in ITB Perfective Verb Forms: 3ms vs. 3fs	91
6.3	Phonemic Singleton Consonants in ITB	93
6.4	Pharyngealized Consonants in ITB	94
6.5	Unpredictable Vowel/Glide Contrasts in ITB	94
6.6	Words with GR Syllables	98
6.7	Truth Table for /saul-x/	110
7.1	Epenthesis in MA Verbs (Perfective Forms)	113
7.2	Phonemic Singleton Consonants in MA	114
7.3	Geminate CəCC Forms in MA	123
7.4	MA 1sg Past Tense Verb Forms	124
7.5	Additional Examples of Epenthesis in MA Nouns	125
7.6	Truth Table for /krkbbk/	140

LIST OF FIGURES

3.1	A visual representation of $\mathcal{M}_{ball}^{\triangleleft}$	16
3.2	A visual representation of $\mathcal{M}_{ball}^{<}$	19
3.3	$\Gamma_{b \rightarrow a}(\mathcal{M}_{ball}^{\triangleleft})$	23
3.4	ψ_{11}	25
4.1	Dot representation: a string of segments and syllable boundaries . .	32
4.2	Flat representation: a string of segments labeled with syllable constituents	33
4.3	Tree representation: hierarchical structure	33
4.4	$\mathcal{M}_{plenty}^{dot}$	35
4.5	$\mathcal{M}_{plenty}^{flat}$	36
4.6	$\mathcal{M}_{plenty}^{tree}$	38
4.7	The codomain for $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	41
4.8	Labels for Copy Set 1 in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	42
4.9	A nucleus-adjacent onset defined with the Flat Model theory	43
4.10	Labels for Copy Sets 1 and 2 in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	44
4.11	Labels for all three copy sets in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	45
4.12	The successor function in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	46

4.13	Some dominance information in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	47
4.14	Additional dominance information in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	48
4.15	An onset of length n	49
4.16	$\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$ fully specified	51
4.17	The licensed output of $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$	52
4.18	The codomain for $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$	55
4.19	Unary relations for $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$	56
4.20	$\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$ fully specified	57
4.21	The licensed output of $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$	58
4.22	The codomain for $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	59
4.23	Labeling Copy Set 1 in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	60
4.24	Labeling Copy Set 2 in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	61
4.25	Some successor information in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	63
4.26	All successor information in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	64
4.27	$\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$ fully specified	65
4.28	The licensed output of $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$	66
4.29	The codomain for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$	67
4.30	Some unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$	67
4.31	Some unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$	68
4.32	Additional unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$	70
4.33	$\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$ fully specified	71

4.34	The licensed output of $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$	72
4.35	\mathcal{M}_{CO}^{tree} , a complex onset in the Tree Model	74
4.36	The output of $\Gamma_{tf}(\mathcal{M}_{CO}^{tree})$	75
5.1	ψ_{NR} , the substructure required by κ_{NR} “Nucleus Required”	78
5.2	ψ_{NU} , the substructure banned by κ_{NU} “Nucleus Unique”	78
5.3	$\mathcal{M}_{plenty}^{tree}$	79
5.4	ψ_{OC} , the substructure banned by κ_{OC} “No Onset-Coda”	80
5.5	ψ_{right} , the substructure banned by κ_{right} “Right of Onset”	84
5.6	An onset followed by a nucleus of lesser sonority	84
5.7	\mathcal{M}_{lbik}	85
5.8	\mathcal{M}_{blik}	85
5.9	ψ_{left} , the substructure banned by κ_{left} “Left of Coda”	86
5.10	ψ_{OR} , the substructure required by κ_{OR} “Onset Required”	87
5.11	ψ_{CF} , the substructure banned by κ_{CF} “Coda Forbidden”	88
6.1	$\psi_{N IOS}$, the substructure banned by $\kappa_{N IOS}$ “No Internal Onsetless Syllable”	95
6.2	$\psi_{N FO}$, the substructure banned by $\kappa_{N FO}$ “No Final Obstruent”	96
6.3	$\psi_{N CO}$, the substructure banned by $\kappa_{N CO}$ “No Complex Onset”	97
6.4	$\psi_{N CC}$, the substructure banned by $\kappa_{N CC}$ “No Complex Coda”	97
6.5	ψ_{son} , the substructure banned by κ_{son}	99
6.6	ψ_a , the substructure banned by κ_a “No Onset [a]”	99
6.7	ψ_{left} , the substructure banned by κ_{left} “Left of Coda”	100

6.8	$\mathcal{M}_{[t_x.znt]}$	101
6.9	$\mathcal{M}_{[t_x.znt]}^{tree}$	102
6.10	$\mathcal{M}_{[t_x.z.nt]}^{tree}$	102
6.11	$\mathcal{M}_{[t_x.z.nt]}^{tree}$	103
6.12	$\mathcal{M}_{[t_x.z.n.t]}^{tree}$	103
6.13	$\mathcal{M}_{saulx}^\triangleleft$	110
6.14	$\Gamma_{ITB}(\mathcal{M}_{saulx}^\triangleleft)$	111
7.1	ψ_{NCO} , the substructure banned by κ_{NCO} “No Complex Onset” . . .	116
7.2	ψ_{NISC} , the substructure banned by κ_{NISC} “No Internal Syllabic Consonants”	117
7.3	ψ_{NIOS} , the substructure banned by κ_{NIOS} “No Internal Onsetless Syllables”	117
7.4	ψ_{N3C} , the substructure banned by κ_{N3C} “No Trisegmental Coda” .	118
7.5	$\psi_{N\partial\sigma}$, the substructure banned by $\kappa_{\partial\sigma}$ “No Open Schwa (Word-Internal)”	119
7.6	$\psi_{N\partial\kappa}$, the substructure banned by $\kappa_{\partial\kappa}$ “No Open Schwa (Word-Final)”	120
7.7	ψ_{right} , the substructure banned by κ_{right} “Right of Onset”	121
7.8	ψ_{left1} , the substructure banned by κ_{left1} “Left of Coda (Word-Internal)”	122
7.9	ψ_{left2} , the substructure banned by κ_{left2} “Left of Coda (Word-Final)”	122
7.10	The Tree Word Model for [fæk]	123
7.11	ψ_{1sg} , the substructure banned by κ_{1sg} “No Epenthesis Preceding 1sg Suffix”	124

7.12	ψ_{exc} , the substructure banned by κ_{exc} “No CCəC for Exceptional Stems”	125
7.13	The Successor Word Model for /ktf/	129
7.14	The Successor Word Model for /klb/	130
7.15	$\mathcal{M}_{[k.təb]}^{tree}$	134
7.16	The licensed output of $\Gamma_{MA}(\mathcal{M}_{[k.təb]}^{\triangleleft})$	135
7.17	$\mathcal{M}_{krkbb}^{\triangleleft}$	138
7.18	The output of $\Gamma_{MA}(\mathcal{M}_{krkbb}^{\triangleleft})$	139
8.1	The substructure corresponding to $\text{ons}_1(x)$	142
8.2	ψ_{NIOS} , the substructure banned by κ_{NIOS} “No Internal Onsetless Syllable”	144
8.3	The Rhyme Model for <i>cat</i> [kæt]	145
8.4	A tree-like representation of <i>lemon</i> [ləmən]	146

ABSTRACT

This dissertation investigates the computational properties of syllable-based phenomena using tools from Model Theory. Although the syllable has been studied by linguists for over a century, the computational complexity of syllable well-formedness and syllabification processes has not yet been investigated. After introducing the necessary formalisms from Model Theory, I present three main findings. First, I show that three types of syllable structure representations from the literature are *notationally equivalent*, meaning we can ‘translate’ between them very easily without loss of information. Second, I formalize syllable well-formedness patterns in Imdlawn Tashlhiyt Berber (ITB) and Moroccan Arabic (MA) as grammars of local, inviolable constraints. Third, I formalize syllabification processes in ITB and MA using Quantifier-Free (QF) logic, a weak logical language that can only make local computations. Taken together, these findings support the hypothesis—contrary to constraint-based paradigms which emphasize global optimization—that syllable-based phenomena are fundamentally local in nature.

Chapter 1

INTRODUCTION

This dissertation investigates the computational properties of syllable well-formedness patterns and syllabification processes using tools from Model Theory. The syllable is one of the most referenced phonological domains; it is central to economical accounts of many patterns and processes, both synchronic and diachronic. Any useful framework of phonology must therefore provide a way to represent syllable structure and to explain syllable-based phenomena. In particular, this dissertation focuses on three main questions:

1. How do we evaluate competing representations of syllable **structure**?
2. What is the nature of the syllable well-formedness **patterns** that we observe in surface forms?
3. What is the nature of the **processes** that generate syllabified Surface Representations (SRs) from unsyllabified Underlying Representations (URs)?

To answer these questions, I turn to Model Theory, which is the branch of mathematics that applies formal logic to the study of structures. Several theorists have noted the value of formal logic in phonology (e.g., [Bird et al., 1992](#); [Coleman, 1998](#); [Graf, 2010](#); [Heinz, 2011a](#); [Heinz & Idsardi, 2013](#); [Potts & Pullum, 2002](#); [Rogers et al., 2013](#); [Scobbie, 1991](#)). It gives us a way of characterizing the structures, patterns, and processes that are crucial to theoretical phonology while abstracting away from the details of any particular grammatical formalism. With formal logic, we can directly evaluate and

address expressivity problems in existing theories of phonology because the ability of a grammar to over- or under-generate is directly related to its computational *power* or *complexity*—that is, the types of statements or computations the grammar allows (Gainor et al., 2012; Graf, 2010; Potts & Pullum, 2002). In this way, computational complexity is an indicator of the types of patterns that are possible (or impossible) for a given grammar to generate.

One useful way of limiting computational complexity in phonology is to allow only *local* (as opposed to *global*) well-formedness evaluation, which constrains the grammar to a handful of highly restricted classes of patterns (Heinz, 2010b; Rogers et al., 2013; Rogers & Pullum, 2011a). These classes have already been demonstrated to encompass many patterns of local and long-distance phonotactics (Heinz, 2007, 2009, 2010a), vowel harmony (Blum, 2018; Heinz et al., 2011), tone contour well-formedness (Jardine, 2016, 2017), and stress assignment (Heinz, 2009; Rogers et al., 2013), while ruling out unattested patterns (Heinz & Lai, 2013). UR-to-SR mappings for several phonological processes (e.g., metathesis, vowel harmony, partial reduplication, dissimilation) have also been represented with mathematical formalisms that are similarly limited in terms of locality (Chandlee, 2014; Chandlee & Heinz, 2018; Heinz & Lai, 2013; Payne, 2017; Rawski, 2018).

Importantly, logical formalisms make no commitment to the real-world implementation of the grammar. In this respect, the approach here is similar to Declarative Phonology (DP) (Bird et al., 1992; Scobbie et al., 1996). However, there are two main differences. First, DP considers only SRs and not UR-to-SR maps. Second, the goal of DP is purely descriptive, unlike later model-theoretic approaches to phonology which seek to better understand characteristics of linguistic generalizations independent of grammatical formalisms (e.g., Heinz, 2011a,b; Potts & Pullum, 2002). The aim here is not to show that the brain somehow prefers formal logic over other methods of computation, but to use logic as a tool to gain insight into the nature of phonological patterns and processes. This approach tries to answer the question of what kinds of representations

and computations are *sufficient* to account for phonological generalizations.

To that end, this dissertation presents three main findings. First, I show that three types of syllable structure representations from the literature are *notationally equivalent*, meaning we can ‘translate’ between them very easily without loss of information. This is accomplished by providing effective translations between these representations and showing that it is sufficient to use First-Order (FO) logic without quantification. This restricted logic, called Quantifier-Free (QF) logic, is fundamentally related to locality. Moreover, it follows from these translations that any analysis stated in terms of one representation can be automatically translated into an analysis stated in terms of one of the others.

Second, I formalize syllable well-formedness patterns in Imdlawn Tashlhiyt Berber (ITB) and Moroccan Arabic (MA) as grammars of local, inviolable constraints. In other words, it is sufficient to account for the surface syllable structures in both languages simply by referring to connected substructures, without recourse to global evaluation.

Third, I formalize the syllabification processes themselves in ITB and MA using QF logic. Again, not only does this mean that they are fundamentally local in a strict sense, but it also implies that the UR-to-SR maps for syllabification in these languages do not require global optimization, contrary to dominant paradigms in theoretical phonology.

Together these findings support the hypothesis that local computations are sufficient to account for syllable structures and syllabification in the world’s languages.

Chapter 2 gives a brief overview of the syllable as a unit of interest in phonology. The formal mathematical background for the proposed approach is presented in Chapter 3. In Chapter 4, I discuss three types of syllable representations from the phonological literature and formalize them as mathematical objects. I then develop formalisms for translating between these representation types, and I show that they are notationally equivalent. In Chapter 5, I turn to universal principles and CV typology. Chapters 6 and 7 are case studies on syllable structure: the first on ITB and the second on MA.

Each presents its own challenges, but both turn out to be computationally simple. In Chapter 8, I discuss what these results can tell us about computational complexity in phonology and which aspects of syllable theory are compatible with the present analysis; I also suggest some directions for future work. Finally, Chapter 9 concludes with a brief summary of the findings herein.

Chapter 2

THE SYLLABLE

References to the syllable as a phonological unit abound in linguistic theory. While some phonologists have questioned the universality of the syllable (e.g., [Blevins, 2003](#); [Côté, 2000](#); [Gimson, 1970](#); [Hyman, 1983](#); [Kohler, 1966](#); [Steriade, 1998, 1999](#)), very few reject its existence outright (as discussed in [Goldsmith, 2011](#)). In this chapter, I first present support for the syllable from studies of synchronic phonological patterns, diachronic change, and articulatory gestures. I then review established theoretical approaches to the syllable and discuss some of their shortcomings.

2.1 Evidence for the Syllable

Syllable structure is central to economical descriptions of many phonological processes. For example, [Kahn \(1976\)](#) notes that many rewrite rules can be simplified by referring to the coda position rather than the otherwise unrelated environments “before/after a consonant or word boundary.” Since the 1960s, syllable-based approaches have been proposed to account for a variety of phonological processes in Spanish ([Harris, 1969](#)), Finnish ([Harms, 1964](#)), German ([Vennemann, 1968a](#)), Japanese ([McCawley, 1968](#)), Akan ([Schachter & Fromkin, 1968](#)), and Italian ([Vogel, 1977](#)), among many others. [Davis \(1985\)](#) provides additional examples of string-based descriptions of phonological processes that miss key generalizations due to their omission or dismissal of syllables. Accounts of stress assignment (e.g., [Broselow, 1976](#); [Harris, 1983](#)) and phonotactics (e.g., [Kahn, 1976](#)) typically rely on syllable structure. In general, processes of assimilation, weakening, and deletion are more common syllable-finally than syllable-initially ([Bell](#)

& Hooper, 1978; Kahn, 1976; Ohala & Kawasaki, 1984). For example, syllable-final laterals are more likely to be ‘dark’ (i.e., more backed) following a back vowel than are syllable-initial laterals (Giles & Moll, 1975; Lehiste, 1960). It is also more common for regressive nasalization to be triggered by syllable-final nasals, as opposed to syllable-initial nasals (Schourup, 1973).

In addition to these synchronic patterns, many diachronic phonological changes are easily explained if syllable structure is taken into account. Compared to syllable-initial stops, syllable-final stops are more likely to be deleted over time (Kent & Read, 1992; Locke, 1983; Manuel & Vatikiotis-Bateson, 1988). Similarly, laterals and nasals in coda position are more susceptible to diachronic weakening. Weakening of a syllable-final nasal sometimes leads to the genesis of contrastive vowel nasalization (Kawasaki, 1986), while syllable-final lateral weakening can leave a high or mid back vowel in place of a true [l] (Chen & Wang, 1975).

Studies of articulatory gestures in speech production provide additional insight. Krakow (1999, 1989) proposes that syllabic positions are associated with certain patterns of articulatory organization. She relates differences in lip-velum coordination to the greater likelihood for assimilation to affect a vowel preceding a coda /m/ than a vowel in an open syllable that precedes an onset /m/. Similarly, Browman & Goldstein (1995) show that syllable-final laterals tend to involve less tongue contact/closure at the palate and alveolar ridge than do syllable-initial laterals. This explains the backing process observed independently by Lehiste (1960), Giles & Moll (1975), and others.

Taken together, these studies strongly suggest that the syllable has explanatory value in descriptions of synchronic phonological patterns and historical sound changes, as well as a basis in articulatory phonetics.

2.2 Previous Accounts of the Syllable

The development of syllable theory began with some of the earliest works in modern linguistics. Whitney (1874), Osthoff & Burgmann (1878), Sievers (1881), and Jespersen

(1904) all discuss phonological constituents smaller than the word and larger than the segment, often directly referencing sonority peaks as the defining feature of these constituents. Making use of the notion of sonority, Saussure developed what might be the first set of ordered rules for partially syllabifying a string (Goldsmith, 2011; Laks, 2003). Saussure's focus on the syllable as the primary unit of interest in phonological theory informed many subsequent endeavors, including experimental studies of the physiological and/or acoustic correlates of syllable boundaries (e.g., Draper et al., 1959; Gay, 1978; Kent & Minifie, 1977; Kozhevnikov & Chistovich, 1965; Stetson, 1951). While these studies had mixed results at best, they paved the way for future work, both theoretical and experimental.

Perhaps in part due to the failure of early experiments to offer a definition of the syllable in physical terms, work in generative phonology at first eschewed syllable theory in favor of string-based approaches (e.g., Chomsky & Halle, 1968). In this view, phonological processes are thought to apply to strings of segments (and sometimes word and morpheme boundaries), without any notion of word-internal phonological structure. However, the shortcomings of string-based accounts led Generativists to reconsider the importance of the syllable as a phonological domain (e.g., Clements & Keyser, 1983; Davis, 1985; Hooper, 1972; Kahn, 1976; Vogel, 1977). Early generative approaches typically viewed syllabification as a predictable process defined by an ordered set of rules (e.g., Borowsky, 1990; Giegerich, 1992; Kahn, 1976; see Kenstowicz (1994) for a review), just as Saussure had originally proposed.

To illustrate a rule-based approach, consider the set of rules in (1), reproduced from Basbøll (1999). These rules encode a process for syllabifying Danish consonants and consonant clusters between a stressed short vowel and a schwa.

- (1) 1. A postvocalic C is adjoined to the left (i.e., to the syllable of the preceding V).
2. /g/ is adjoined to the left.

3. Non-adjoined C's are adjoined to the right if they form a possible onset together with all following C's (which are then also adjoined to the right), otherwise to the left.

Take, for instance, the word *entre* /'ɛntɹɛ/, 'to board.' Rule 1 applies to the /n/ because it follows a stressed short vowel, /ɛ/, and precedes a schwa. The /n/ therefore adjoins to the first syllable, occupying the coda position. Rule 2 does not apply. Rule 3 applies to /t/ because it too follows a stressed /ɛ/ and precedes a schwa. Because [tɹ] is a valid onset in Danish, both consonants adjoin to the second syllable. The end result is ['ɛn.tɹɛ].

Some later rule-based approaches make limited use of inviolable constraints. [Reiss \(2008\)](#) describes two main uses of constraints in these frameworks: to *trigger* phonological rule application to repair ill-formed structures (as in [Calabrese, 1988](#); [Paradis, 1988](#)) and to *block* rule application where it would result in an ill-formed structure (as in [Halle & Idsardi, 1995](#); [McCarthy, 1986](#)). One such framework—templatic syllabification—garnered interest alongside the development of prosodic phonology in the 1980s (e.g., [Selkirk, 1980](#); [Nespor & Vogel, 1986/2007](#)). In the templatic approach to syllabification, each language has a template for a maximal core syllable (e.g., [Duanmu, 2009](#); [Itô, 1988, 1989](#)). For example, epenthesis in Spanish can be explained as a necessary consequence of mapping URs to a maximal CCVC syllable template ([Kenstowicz, 1994](#)). Consider the loanword *esplin*, from the English *spleen*. Mapping /splɪn/ to the CCVC template from right to left, we obtain two partial syllables: s.plɪn. While the second syllable is well-formed, the first syllable needs, minimally, a nucleus. Epenthesis of [e] corrects this, resulting in the SR [es.plɪn].

The template is, in a sense, an inviolable constraint on possible syllable types. The process of syllabification can then be conceptualized as a mapping from the UR to the SR that matches this template, deleting and epenthesizing where necessary to avoid *any* deviations from the template. As [Kenstowicz \(1994\)](#) points out, carefully constructed sets of rewrite rules can encode the same information as a templatic syllabification

approach. In a sense, both treatments rely primarily on an ordered set of rules that map URs to SRs.

In stark contrast, the treatment of syllables in Optimality Theory (OT) is, of course, focused on surface well-formedness rather than the syllabification process itself. Crucially, well-formedness is evaluated globally across the entire SR. Prince & Smolensky (1993) offer several violable constraints relevant to syllable structure, two of which are given in (2) and (3). The tableaux in (4) (reproduced from Kager, 1999) shows how both of these constraints are violated if /baba/ is syllabified as [bab.a] rather than [ba.ba]. This is a universal outcome in OT; the mere presence of ONSET and NO-CODA (and the lack of constraints like NO-ONSET and CODA) guarantees that [ba.ba] will always be more optimal than [bab.a].

(2) ONSET: ‘Syllables must have onsets.’

(3) NO-CODA: ‘Syllables are open.’

(4)

/baba/	ONSET	NO-CODA
☞ a. ba.ba		
b. bab.a	*	*

Additional constraints ban syllabic nuclei of low sonority, complex syllable margins, and other marked syllable structures. The ranking of these markedness constraints with respect to faithfulness constraints such as MAX (‘Don’t delete’) and DEP (‘Don’t epenthesize’) determines the language-specific principles of syllable well-formedness.

2.3 Methodological Problems of Previous Approaches


In terms of accounting for types of phonological patterns, derivation via rewrite rules and optimization over violable constraints both overgenerate in undesirable ways (Frank & Satta, 1998; Gainor et al., 2012; Heinz, 2018; Heinz & Lai, 2013; Riggle, 2004). In other words, both approaches allow for the description of patterns that are unattested (and thought to be unnatural).

To illustrate this point, I will expound on one such pattern that is notoriously problematic for OT: Majority Rules (Baković, 1999, 2000; Lombardi, 1999). Consider a language with front-back vowel harmony. Assuming the harmonizing feature is $[\pm\text{front}]$, the two relevant constraints are $\text{AGREE}[\text{front}]$ and $\text{IDENT}[\text{front}]$. The first of these is a markedness constraint, defined in (5). It assigns one violation for every pair of consecutive vowels that does not share the same value for $[\text{front}]$. The faithfulness constraint $\text{IDENT}[\text{front}]$, defined in (6), assigns one violation for every change in the $[\text{front}]$ value of a vowel from the UR to the SR.

- (5) $\text{AGREE}[\text{front}]$: ‘Two vowels in adjacent syllables must have the same $[\text{front}]$ value.’
- (6) $\text{IDENT}[\text{front}]$: ‘Do not change the value of $[\text{front}]$.’

If $\text{AGREE}[\text{front}]$ outranks $\text{IDENT}[\text{front}]$, an odd pattern arises: the SR will harmonize to whichever type of vowel is more common in the UR (i.e., front or back). Hence the name: Majority Rules.

Abstracting away from other vocalic features and ignoring consonants altogether, let us consider sequences of $[\text{+front}]$ and $[\text{-front}]$ vowels using only the symbols $-$ and $+$. The following tableaux will help to illustrate the Majority Rules result.


	$/- - +/$	$\text{AGREE}[\text{front}]$	$\text{IDENT}[\text{front}]$
(7)	a. $- - +$	*!	
	 b. $- - -$		*
	c. $+ + +$		**!

The UR in (7) is $/- - +/$. The fully faithful candidate violates $\text{AGREE}[\text{front}]$ twice: the first and second vowels differ in value for $[\text{front}]$, as do the second and third. Because $\text{AGREE}[\text{front}]$ outranks $\text{IDENT}[\text{front}]$, this candidate is less harmonic than the other two—in fact, any candidate without complete harmony will be less harmonic than

the two candidates with complete vowel harmony. In this case, the back-harmonizing candidate incurs fewer violations of IDENT[front] and is therefore optimal.



Given the UR /+ - +/ in (8), the fully faithful candidate violates AGREE[front] twice, just as in the previous example. Among the remaining candidates, the front-harmonizing one incurs the fewest violations of IDENT[front] and is therefore optimal.

(8)

/+ - +/	AGREE[front]	IDENT[front]
a. + - +	*!*	
b. - - -		**!
 c. + + +		*

In the general case, any input with a majority of front vowels will front-harmonize completely and any input with a majority of back vowels will back-harmonize completely. One might wonder what happens when the input has an equal number of front and back vowels. In this case, the two constraints given here cannot determine a single optimal candidate. As (9) shows, the front-harmonizing and back-harmonizing candidates incur exactly the same violations.

(9)

/+ - + -/	AGREE[front]	IDENT[front]
a. + - + -	*!**	
 b. - - - -		**
 c. + + + +		**

These examples illustrate how the simple ranking of two constraints can result in an unattested, unnatural pattern (Finley, 2008; Finley & Badecker, 2008). There is nothing particular about the constraints shown here that lead to this pattern. It is *global optimization* itself that forces the Majority Rules result.

2.4 Moving Forward

The syllable has received ample attention over the past century and a half, under two main approaches: rule-based accounts of the syllabification process, and Optimality-Theoretic constraint-based accounts of input-output mappings based on surface well-formedness. While both approaches have produced adequate descriptions of many phonological processes related to syllable structure, they have major disadvantages due to the expressivity of their formalisms. This dissertation offers a third approach using a highly restricted logical language, drawing from previous work in formal language theory. The next chapter provides the formal background for the analyses to follow.

Chapter 3

FORMAL BACKGROUND

This chapter presents the mathematical tools to be used in my analyses of syllable structure and syllabification. First I introduce *word models* as a way of representing the structure of words, with examples using two commonly used *model theories*. I then present some modifications to conventional model theories in §3.2. In §3.3 I define mappings from underlying forms to surface forms using *graph transductions*. §3.4 shows how to characterize surface generalizations by referring to *substructures* and §3.5 introduces logical expressions called *substructure constraints* to define *formal languages*—sets of phonologically well-formed words. In §3.6, I discuss three different logical languages and the relationship between logical power and locality. Finally, §3.7 introduces user-defined formulas for shorthand.

3.1 Word Models and Model Theories

A word model is a type of graph used for representing relational structures such as words. Take the orthographic word *ball*, for instance. It consists of four symbols in a particular order. This can be formalized as a *string* of four ordered positions, each of which is labeled with a letter of the alphabet. For example, the second position in *ball* is labeled *a*.

There are many ways one such string can be represented. *Model theories* define classes of word models that share a common *signature*. Given an alphabet Σ (a set of symbols or labels), a model theory \mathfrak{M} has the signature $\langle \mathfrak{D}; \mathfrak{R}; \mathfrak{F} \rangle$ where \mathfrak{D} is a *domain* (a set of positions), \mathfrak{R} is a set of relations among elements of the domain (positions), and \mathfrak{F} is a set of functions.

Domain. The domain \mathfrak{D} is somewhat like the skeleton of the word model, but domain positions have no inherent properties or order. Relations and functions determine what each position means in the context of the object being modeled. By convention, the domain \mathfrak{D} is the set of *natural numbers*—the positive counting numbers (e.g., 1, 2, 3...). Although strings are sometimes represented starting at position 0, this creates confusion as to whether position 1 is the ‘first’ or ‘second’ position. I use 1 to *index* (number) the initial position, so the final position in a string of length n has the index n . In the string *ball*, for instance, the first position is 1 and the final position is 4.

Relations. The set \mathfrak{R} includes a relation R_σ for every every symbol σ in the alphabet Σ . For example, let $\Sigma = \{\mathbf{a}, \mathbf{b}, \mathbf{l}\}$. Then \mathfrak{R} includes the unary relations R_a, R_b, R_l . In the word *ball*, the first position is a member of the set R_b because it is labeled **b**. Set membership is denoted with the \in symbol, as in $1 \in R_b$, which is read “1 is in R_b ” or “1 belongs to R_b .” In contrast, the \notin symbol denotes non-membership, as in $2 \notin R_b$, meaning “2 is not in R_b ”—in other words, position 2 is not labeled **b**.

The relations described thus far are all *unary*, meaning that they only refer to a single position. Formally, a unary relation is one that takes a single *argument*. In principle, \mathfrak{R} may contain relations of higher *arity*, meaning they take more than one argument.

Functions. Unary functions in \mathfrak{F} are similar to unary relations in that they both take a single position as an argument. However, the relations used here are evaluated as *Boolean expressions*: they are either **TRUE** or **FALSE**. Position 1 is either labeled **a** or it is not. In contrast, functions *map* domain positions to domain positions.

The following examples will help to make these definitions clear.

3.1.1 The Successor Model Theory

Given an alphabet Σ , the Successor Model Theory $\mathfrak{M}^\triangleleft$ is defined in (3.1).

$$\mathfrak{M}^\triangleleft \stackrel{\text{def}}{=} \langle \mathbb{N}; \{R_\sigma \mid \sigma \in \Sigma\}; \{pred(x), succ(x)\} \rangle \quad (3.1)$$

Domain. As is tradition, the domain is \mathbb{N} .

Relations. The only relations in $\mathfrak{M}^\triangleleft$ are those corresponding to each member σ of the alphabet Σ . This set is denoted $\{R_\sigma \mid \sigma \in \Sigma\}$, where the $|$ symbol stands for “such that.” Thus the meaning of $\{R_\sigma \mid \sigma \in \Sigma\}$ is “the set of relations R_σ , such that σ is in Σ .”

Functions. Given a position x , the unary functions $pred(x)$ and $succ(x)$ pick out the immediate predecessor and successor of x , respectively.¹ For example, $succ(1) = 2$ means that the successor of 1 is 2 and $pred(2) = 1$ means the predecessor of 2 is 1. This may seem trivial, but recall that the domain has no inherent order. It could just as well be true that $succ(4) = 3$, $succ(3) = 2$, and so on—this would be similar to reading right-to-left instead of left-to-right. The successor and predecessor functions as I have defined them impose the left-to-right order tacit in English orthography. Model theories, as mathematical objects, must have all properties defined explicitly.

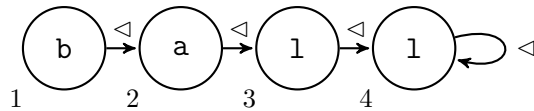
For a function to be *total*, it must yield an output for every position of its domain. In the general case, $succ(x) = x + 1$ and $pred(x) = x - 1$. But what is the predecessor of the first position? To make $pred(x)$ a total function, the initial position is defined to be its own predecessor, i.e., $pred(1) = 1$, as in [Thomas \(1982\)](#). Similarly, the final position in a string is its own successor, making $succ(x)$ total as well. Then in a string of n positions, $succ(n) = n$.

¹ The use of unary functions (rather than binary relations) for predecessor and successor is due to [Chandlee & Lindell \(2016\)](#).

The Successor Model for *ball*

The model for the string *ball* under the Successor Model Theory is denoted $\mathcal{M}_{ball}^{\triangleleft}$. Using the alphabet $\Sigma = \{a, b, 1\}$, $\mathcal{M}_{ball}^{\triangleleft}$ is defined in (3.2) and represented visually in Figure 3.1.

Figure 3.1: A visual representation of $\mathcal{M}_{ball}^{\triangleleft}$



Domain. While the domain for $\mathfrak{M}^{\triangleleft}$ is \mathbb{N} , the domain for a particular word model is a finite subset of \mathbb{N} . Domain positions in a word model are sometimes called *nodes*. In $\mathcal{M}_{ball}^{\triangleleft}$, the domain \mathcal{D} is a set of four nodes, each of which is visually represented as a circle with its index below and to the left.

Relations. Unary relations of the form R_σ are given as sets, the members of which are positions labeled σ . For example, position 1 is a member of the set R_b , so it is labeled **b**.

Functions. The successor function is illustrated by directed edges (arrows) labeled with the \triangleleft symbol. The final position is its own successor by definition, so this illustrated with a self-loop. In the remaining figures, I will generally omit these word-final self-loops for the sake of visual clarity. The predecessor function is also left out of the visual representation because this information is largely redundant, given the successor information.

Following [Chandlee & Lindell \(to appear\)](#), I use a notational extension of FO logic that includes definition by cases. As the authors note, “any extended formula is equivalent to an ordinary formula involving Boolean combinations of the terms and formulas

appearing in those cases.” Thus the use of cases does not increase the power of the logic; it is simply a convenient notation. This notation is particularly useful for defining the successor and predecessor functions, as in in (3.2). These definitions should be interpreted as follows: $succ(x) = 2$ iff $x = 1$; $succ(x) = 3$ iff $x = 2$; and so on.²

$$\begin{aligned}
 \mathcal{M}_{ball}^{\triangleleft} &\stackrel{\text{def}}{=} \langle \mathcal{D}; \{R_a, R_b, R_l\}; \{pred(x), succ(x)\} \rangle & (3.2) \\
 \mathcal{D} &\stackrel{\text{def}}{=} \{1, 2, 3, 4\} \\
 R_a &\stackrel{\text{def}}{=} \{2\} \\
 R_b &\stackrel{\text{def}}{=} \{1\} \\
 R_l &\stackrel{\text{def}}{=} \{3, 4\} \\
 succ(x) &\stackrel{\text{def}}{=} \begin{cases} 2 & \Leftrightarrow x = 1 \\ 3 & \Leftrightarrow x = 2 \\ 4 & \Leftrightarrow x \in \{3, 4\} \end{cases} \\
 pred(x) &\stackrel{\text{def}}{=} \begin{cases} 1 & \Leftrightarrow x \in \{1, 2\} \\ 2 & \Leftrightarrow x = 3 \\ 3 & \Leftrightarrow x = 4 \end{cases}
 \end{aligned}$$

3.1.2 A Brief Aside: Notational Conventions

There are several types of notation used for relations and functions. I have already used set membership notation, as in $1 \in R_b$. For unary relations like this, it is equivalent to write $R_b(1)$ or simply $b(1)$. Moving forward, I will use the latter notation for most unary relations. For the successor function, I write $succ(1) = 2$ or $1 \triangleleft 2$. Similarly, the formula $pred(2) = 1$ is equivalent to $2 \triangleright 1$.

² “Iff” is shorthand for “if and only if,” which indicates that two logical formulas are *materially equivalent*. This means that if either formula is true, the other must be true; and if either one is false, the other must be false. The \Leftrightarrow symbol denotes material equivalence.

3.1.3 The Precedence Model Theory

Another common model theory for strings is the Precedence Model. Its signature $\mathfrak{M}^<$ is given in (3.3). The key differences between (3.3) and the Successor Model Theory (3.1) are in the relations and functions.

$$\mathfrak{M}^< \stackrel{\text{def}}{=} \langle \mathbb{N}; \{R_<, R_\sigma \mid \sigma \in \Sigma\}; \emptyset \rangle \quad (3.3)$$

Domain. The domain for $\mathfrak{M}^<$ is \mathbb{N} , the same as that for $\mathfrak{M}^\triangleleft$.

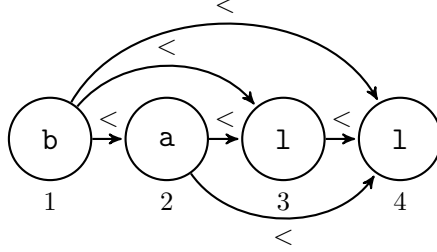
Relations. The Precedence Model Theory includes the *general precedence* relation (i.e., the order of two positions with respect to one another), denoted $R_<$. When determining general precedence, the presence or absence of intervening positions is disregarded, just as the expression $3 < 5$ is true regardless of the fact that the number 4 intervenes between 3 and 5. Because precedence relates two positions to each other, it is called a *binary relation*. To express the precedence relation between two positions, x and y , I write $R_<(x, y)$ or, equivalently, $x < y$.

Functions. Using the above definition, there are no functions needed for the Precedence Model Theory, so \mathfrak{F} is equal to \emptyset , the *empty set*. Note that precedence cannot be encoded as a function because it is a one-to-many relation. For example, position 3 is preceded by positions 2 *and* 1.

The Precedence Model for *ball*

The model for the string *ball* under this theory is denoted $\mathcal{M}_{ball}^<$. Taking the same alphabet $\Sigma = \{\mathbf{a}, \mathbf{b}, \mathbf{1}\}$, the precedence word model for *ball* is shown in (3.4). Figure 3.2 is a visual representation of $\mathcal{M}_{ball}^<$.

Figure 3.2: A visual representation of $\mathcal{M}_{ball}^<$



$$\begin{aligned}
 \mathcal{M}_{ball}^< &\stackrel{\text{def}}{=} \langle \mathcal{D}; \{R_{<}, R_a, R_b, R_l\}; \emptyset \rangle & (3.4) \\
 \mathcal{D} &\stackrel{\text{def}}{=} \{1, 2, 3, 4\} \\
 R_{<} &\stackrel{\text{def}}{=} \{\langle 1, 2 \rangle, \langle 1, 3 \rangle, \langle 1, 4 \rangle, \langle 2, 3 \rangle, \langle 2, 4 \rangle, \langle 3, 4 \rangle\} \\
 R_a &\stackrel{\text{def}}{=} \{2\} \\
 R_b &\stackrel{\text{def}}{=} \{1\} \\
 R_l &\stackrel{\text{def}}{=} \{3, 4\}
 \end{aligned}$$

Domain. Again the domain consists of four positions, 1 through 4.

Relations. Unary relations are the same as in $\mathcal{M}_{ball}^<$. Precedence relations are given as *tuples*, ordered pairs of the form $\langle x, y \rangle$. The tuple $\langle 0, 2 \rangle$ belongs to the set of precedence relations $R_{<}$, for instance, because 0 precedes 2. Precedence relations are depicted as directed edges with the $<$ label, similar to the depiction of the successor function in Fig 3.1.

3.1.4 Visual Representations

As a final caveat to this introduction to word models, it is important to consider the subtle distinction between the word model and its visual representation. A word model

is a type of graph in the mathematical, rather than colloquial, sense of the word. The visual representation of this object is exactly that—a representation: a visual aid, but not the object itself. Many different visualizations are possible for a given word model. For brevity and clarity, I caption the remaining visual representations in this dissertation with the name of the word models they represent, taking it to be understood that these are merely visual aids.

3.2 Enriching Conventional Word Models

The objects of interest in formal language theory are generally *strings* (as in Büchi, 1960; Rozenberg & Salomaa, 2012). Take for example the string *ball*, as illustrated above. Both the Successor and Precedence Word Models have a very simple type of structure: essentially, an ordered sequence of alphabetic characters. I refer to these as *conventional* word models. Although conventional word models are adequate for many purposes, linguistic descriptions of words tend to be more complex. Individual segments of a word may have multiple properties that would be better expressed as separate labels, rather than a single symbol. Furthermore, linguists discuss other types of structural relations besides precedence and successor, such as dominance in a hierarchical structure. To represent words in a manner more faithful to phonological theory, I propose two modifications to conventional word models: enriching the alphabet and enriching the structure.

3.2.1 Enriching the Alphabet

The alphabet can be conceptualized as a set of primitives—labels defined outside of the model theory itself. In conventional word models, each position has exactly one label (i.e., it belongs to a single unary relation). Formally, we say that the sets of labeling relations for any two symbols in the alphabet are *disjoint*, meaning that they have no elements in common. The set of elements shared by two sets is called the *intersection* of these sets. Intersection is denoted with the \cap symbol, as in $A \cap B$, “the

intersection of the sets A and B .” Using these terms, the mutual exclusivity of labels in conventional word models is expressed in (3.5).

$$(\forall x, y \in \Sigma)[R_x \cap R_y = \emptyset] \quad (3.5)$$

“For all symbols x, y in Σ , the intersection of the sets R_x and R_y is empty.”

In line with recent work in computational phonology (Daland et al., 2011; Heinz & Strother-Garcia, to appear; Strother-Garcia, 2018; Strother-Garcia et al., 2017; Vu et al., 2018), I permit each position to have more than one label. As Strother-Garcia et al. (2017) point out, abandoning the assumption that labeling relation sets are disjoint “allows similarities among different alphabetic symbols to be fully expressed,” opening the door to a variety of new types of word models. Non-mutually-exclusive labels can be used to reflect the well-accepted phonological description of segments as *feature bundles*, rather than wholly separate entities.

For example, say the segment [b] is composed of the features [voice, labial, stop]. The feature [voice] is the only difference between [b] and [p], but this similarity is obscured in conventional word models. I therefore use an ‘alphabet’ of phonological features rather than phonetic symbols, taking advantage of the phonological insight that phones share common features.

Let \mathcal{F} be a set of primitive phonological features. I adopt the features given in (3.6) for now.³

$$\mathcal{F} \stackrel{\text{def}}{=} \{\text{voice, cons, high, lab, alv, post, pal, vel, uv, phar, glot, stop, fric, nas, approx, lat}\} \quad (3.6)$$

³ **cons** = consonantal; **lab** = labial; **alv** = alveolar; **post** = postalveolar; **pal** = palatal; **vel** = velar; **uv** = uvular; **phar** = pharyngeal; **glot** = glottal; **approx** = approximant; **lat** = lateral.

Then the alphabet is simply $\Sigma = \mathcal{F}$. For each feature f in \mathcal{F} , there is a unary relation $R_f \in \mathfrak{R}$ that represents a particular position being labeled with that feature. Let \mathcal{R}_f be the set of such relations, defined in (3.7).

$$\mathcal{R}_f \stackrel{\text{def}}{=} \{R_f \mid \mathbf{f} \in \mathcal{F}\} \tag{3.7}$$

As with alphabetic primitives in conventional word models, $R_{\mathbf{f}}(x)$ can also be written as $\mathbf{f}(x)$ for any feature f . For example, $R_{\text{voice}}(x)$ is equivalent to $\text{voice}(x)$. Crucially, a single position may belong to several of these relations. In the word *ball*, for instance, the first position is labeled **voice**, **lab**, and **stop**.

3.2.2 Enriching the Structure

Another avenue for enriching word models is to consider additional relations and functions. While conventional word models only encode linear order (general or immediate), additional relationships among positions are important when considering syllable structure. In particular, the hierarchical structure of the syllable relies on *dominance* information, formalized in the following chapter. Because sonority plays a role in syllable well-formedness in many languages, it will also be prudent to encode sonority relations explicitly. I will discuss this in detail in Chapter 5.

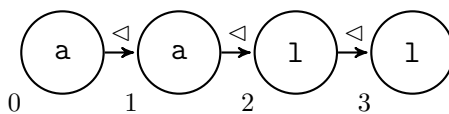
3.3 Graph Transductions

As word models are a type of graph, graph transductions can be used to represent input-output maps from one word model \mathfrak{M}^A to another \mathfrak{M}^B . A transduction is defined with a set of formulas, one for each relation R and function F in \mathfrak{M}^B , the output structure. These formulas are interpreted with respect to the input structure in \mathfrak{M}^A .⁴

⁴ See Courcelle (1994), Courcelle & Engelfriet (2012), and Engelfriet & Hoogeboom (2001) for details.

For example, consider a transduction $\Gamma_{b \rightarrow a}$ that changes every **b** in a word model to **a**. Here $\mathfrak{M}^A = \mathfrak{M}^B = \mathfrak{M}^\triangleleft$. When applied to the input $\mathcal{M}_{ball}^\triangleleft$, the transduction changes the label of the first position from **b** to **a** and leaves the remaining positions unchanged, as illustrated in Figure 3.3.

Figure 3.3: $\Gamma_{b \rightarrow a}(\mathcal{M}_{ball}^\triangleleft)$



$\Gamma_{b \rightarrow a}$ is defined by the set of formulas (3.8-3.12) where the superscript ω indicates an output position. Logical disjunction is denoted with the \vee symbol, read “or.”

$$R_a(x^\omega) \stackrel{\text{def}}{=} R_a(x) \vee R_b(x) \quad (3.8)$$

$$R_b(x^\omega) \stackrel{\text{def}}{=} \text{FALSE} \quad (3.9)$$

$$R_l(x^\omega) \stackrel{\text{def}}{=} R_l(x) \quad (3.10)$$

$$\text{succ}(x^\omega) \stackrel{\text{def}}{=} \text{succ}(x) \quad (3.11)$$

$$\text{pred}(x^\omega) \stackrel{\text{def}}{=} \text{pred}(x) \quad (3.12)$$

(3.8) states that an output position is labeled **a** iff the corresponding input position is labeled **a** or **b**. $R_b(x^\omega)$ is always **FALSE** because no output position may be labeled **b**. $R_l(x^\omega)$ is true for any input position labeled **1**, since no changes are made to input **1**s. Finally, (3.11) and (3.12) express that successor and predecessor are preserved in the output.

In this example, the input and output share the same model theory, but this need not be the case. As will be seen in the following chapter, the model theories for the input and output can differ in many ways.

3.4 Substructures

One way of conceptualizing a logical formula in the context of a word model is to consider the structure that corresponds to it. For instance, consider $\phi_{11}(x)$ defined in (3.13). Logical conjunction is denoted with the \wedge symbol, read “and.”

$$\phi_{11}(x) \stackrel{\text{def}}{=} \mathbf{1}(x) \wedge \mathbf{1}(\text{succ}(x)) \quad (3.13)$$

“ ϕ_{11} is true of position x iff x is labeled $\mathbf{1}$ and its successor is also labeled $\mathbf{1}$.”

The structure that is ‘targeted,’ in a sense, by $\phi_{11}(x)$ is the first position in a pair of adjacent ls . Thus if ll is a *substructure* of the word model, there must be some position x that satisfies $\phi_{11}(x)$. In $\mathcal{M}_{ball}^<$, $\phi_{11}(x)$ is true for $x = 3$.

Substructures are formalized as Existentially Quantified Conjunctions (EQCs). Given a domain $\mathcal{D} = \{1, \dots, n\}$ and a set of relations $\mathcal{R} = \{R_1, \dots, R_m\}$, an EQC is the *conjunction* of expressions that specify which positions belong to which relations. The general form of an EQC is given in (3.14). The set of positions belonging to a given relation may contain any finite number of nodes from zero to n . I write these sets as $\{x_a, \dots, x_b\}$ and $\{x_c, \dots, x_d\}$, but no implication of order or set size is intended. The \bigwedge symbol indicates iterative conjunction over values of i ranging from 1 to m .

$$(\exists x_1 \dots, x_n) \left[\bigwedge_{i=1}^m R_i(x_a, \dots, x_b) \right] \quad (3.14)$$

“There exist a set of positions x_1, \dots, x_n such that some relation R_1 is true of positions x_a, \dots, x_b , and some relation R_2 is true of positions x_c, \dots, x_d , and so on.”

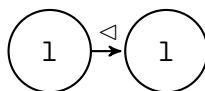
For example, the EQC corresponding to $\phi_{11}(x)$ is defined in (3.15).

$$\psi_{11} \stackrel{\text{def}}{=} (\exists x, y) [\mathbf{1}(x) \wedge \mathbf{1}(y) \wedge x < y] \quad (3.15)$$

“There exist two positions, x and y , such that x is labeled $\mathbf{1}$, y is labeled $\mathbf{1}$, and x immediately precedes y .”

An EQC can be visually represented in the same way as a word model, as shown in Figure 3.4. This visual representation drives home the fact that an EQC is a substructure of a word model.

Figure 3.4: ψ_{11}



In sum: If a formula $\phi_a(x)$ is true of some position x in model \mathcal{M} , then ψ_a , the EQC corresponding to $\phi_a(x)$, must be a substructure of \mathcal{M} . A word model \mathcal{M} *satisfies* a formula ϕ_a iff \mathcal{M} contains the substructure defined by the corresponding EQC ψ_a . Satisfaction is denoted by the \models symbol, as in $\mathcal{M} \models \phi_a$, read “ \mathcal{M} satisfies ϕ_a .” For example, it is clear from the definitions above that $\mathcal{M}_{ball}^\triangleleft \models \phi_{11}$ because $\mathcal{M}_{ball}^\triangleleft$ contains ψ_{11} .

3.5 Formal Languages and Substructure Constraints

Just as graph transductions represent UR-to-SR maps, *formal languages* can be used to represent surface generalizations. The use of the term *language* here is an unfortunate carry-over from the computer science literature. It does not refer to a natural language (e.g., English), but to a set of words—a *stringset* or, more generally, a *graphset* (as in [Jardine, 2016](#)). For my purposes, a formal language can be thought of as the set of phonologically well-formed words in a given natural language.

Given any alphabet Σ , the term Σ^* denotes the infinite set of words of arbitrary length comprised of members of Σ . For example, $\{a, b\}^*$ contains a , b , aa , ba , aba , abb , bbb , and so on. Any subset of Σ^* is a formal language.

Rather than attempting to list all the words in a formal language outright, it is useful to define the formal language using some logical formula. Take any logical formula κ ,

an alphabet Σ , and a model theory \mathfrak{M} . Let w be a word in Σ^* and \mathcal{M}_w be the model of that word under model theory \mathfrak{M} . Then the formal language defined by κ is simply the set of words in Σ^* whose word models satisfy κ . This language is denoted $L(\kappa)$, as defined in (3.16).

$$L(\kappa) \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid \mathcal{M}_w \models \kappa\} \quad (3.16)$$

“ $L(\kappa)$ is the set of words w in Σ^* whose word models \mathcal{M}_w satisfy κ .”

For example, consider the expressions $\psi_{\mathbf{d}}$ and κ_{ball} defined below.

$$\psi_{\mathbf{d}} \stackrel{\text{def}}{=} (\exists x)[\mathbf{d}(x)] \quad (3.17)$$

“There exists a position x labeled \mathbf{d} .”

$$\kappa_{ball} \stackrel{\text{def}}{=} \psi_{11} \wedge \neg\psi_{\mathbf{d}} \quad (3.18)$$

“ κ_{ball} is true iff ψ_{11} is true and $\psi_{\mathbf{d}}$ is false.”

Then $L(\kappa_{ball})$ is the simply set of all words whose models satisfy κ_{ball} . To be precise, $L(\kappa_{ball})$ consists of every word model that contains two adjacent l s and does not contain a d , as formalized in (3.19). The logical negation operator is denoted with the \neg symbol, read “not.”

$$L(\kappa_{ball}) = \{w \in \Sigma^* \mid \mathcal{M}_w^{\triangleleft} \models (\psi_{11} \wedge \neg\psi_{\mathbf{d}})\} \quad (3.19)$$

“The language of κ_{ball} is the set of words in Σ^* whose models (in the Successor Model Theory) contain ψ_{11} and do not contain $\psi_{\mathbf{d}}$.”

This includes, of course, $\mathcal{M}_{ball}^{\triangleleft}$.

A formula like κ_{ball} can be thought of as a well-formedness constraint. It mandates which substructures must or must not be present for a word to be phonologically well-formed in a given language. It takes the form of a conjunction of positive and negative *literals*, where a positive literal is an EQC (requiring that all words contain a particular substructure) and a negative literal is the negation of an EQC (banning any word from

containing particular substructure). I refer to a conjunction of one or more such literals as a *substructure constraint*.⁵ Thus κ_{ball} is a substructure constraint, and so are each of its individual *conjuncts*, ψ_{11} and $\neg\psi_a$.

I will use K to denote a logical formula that is the conjunction of *all* substructure constraints needed to define a particular pattern. K can also be called a *substructure constraint grammar*. Then $L(K)$ is the set of all words generated by the grammar K .

3.6 Logics and Locality

Three logical languages in order of decreasing power are Monadic Second Order (MSO), First-Order (FO), and Quantifier-Free (QF), which differ according to the types of allowable quantification. Here I informally discuss the differences in expressivity among these logics. For formal definitions, I refer the reader to: [Enderton \(2001\)](#), [Fagin et al. \(1995\)](#), and [Shoenfield \(1967\)](#).

Statements in FO logic can use universal (\forall) and existential (\exists) quantifiers to quantify over positions of the domain. This allows for statements like $(\forall x)[\text{voice}(x)]$ meaning “for all positions x , x is voiced.” MSO statements can also quantify over *sets* of domain positions. For example, consider the following definitions from [Jardine & Heinz \(2015\)](#). First, (3.20) defines set closure under successor is in FO logic. Then the MSO sentence (3.21) defines the general precedence relation $<$. Capital X denotes a set, while lowercase x and y denote elements of such sets. The \Rightarrow symbol represents implication in one direction—its meaning is equivalent to ‘if,’ in contrast with ‘iff.’

⁵ This comes from the *substring constraints* of [Garcia et al. \(1990\)](#), [Heinz \(2010b\)](#), [Rogers et al. \(2013\)](#), and others. Following [Jardine \(2016, 2019\)](#), I extend this notion to *subgraphs* and therefore use the more general term *substructures*. To be precise, the substructures I am concerned with are *connected subgraphs*, as in [Jardine \(2016\)](#).

$$\mathbf{closed}(X) \stackrel{\text{def}}{=} (\forall x, y)(x \in X \wedge x \triangleleft y) \Rightarrow y \in X \quad (3.20)$$

“A set X is **closed** (under successor) iff for all elements x, y , if x belongs to X and y is the successor of x , then y also belongs to X .”

$$x < y \stackrel{\text{def}}{=} (\forall X)(x \in X \wedge \mathbf{closed}(X) \Rightarrow y \in X) \quad (3.21)$$

“An element x precedes y iff for all sets X , if position x is in X and X is **closed**, then y is also an element of X .”

Because (3.21) involves universal quantification over some set X , it is strictly MSO and not FO. Because (3.20) has quantification over positions but not over sets, it is FO. Sentences of FO without quantification over elements (i.e., no quantification at all) are QF.

To see why quantification is important, compare (3.22) to (3.8), which is reproduced in (3.23). The former states that an output position x will be labeled **a** if the corresponding input position is an **a** or if there is a position labeled **b** somewhere in the input. Checking whether

$$R_a^{\omega'}(x) \stackrel{\text{def}}{=} R_a(x) \vee (\exists y)[R_b(y)] \quad (3.22)$$

$$R_a^{\omega}(x) \stackrel{\text{def}}{=} R_a(x) \vee R_b(x) \quad (3.23)$$

This example illustrates the relationship between quantification and locality. If a formula is stated with quantification, computing its truth value requires global evaluation of the string. If the formula is QF, truth evaluation must be possible over a substring of bounded size. A transduction defined entirely by QF formulas is called a QF transduction. Thus a QF transduction is one that operates locally, computing truth values for a given position by ‘looking’ only finitely many positions away. Note that $\Gamma_{b \rightarrow a}$ is QF, with all formulas referring to a single position.

3.7 User-defined Formulas

Given a model theory \mathfrak{M} and an alphabet Σ , logical formulas can be defined to make it easier to refer to certain types of information in word models. For example, writing $\text{voice}(x) \wedge \text{lab}(x) \wedge \text{stop}(x)$ to refer to a /b/ is cumbersome. Instead, I use the unary formula $\mathbf{b}(x)$, defined in (3.24).

$$\mathbf{b}(x) \stackrel{\text{def}}{=} \text{voice}(x) \wedge \text{lab}(x) \wedge \text{stop}(x) \quad (3.24)$$

“Position x is a **b** iff x has the features **voice**, **lab**, and **stop**.”

Although I write $\text{stop}(x)$ and $\mathbf{b}(x)$ similarly, note that the former is the labeling relation for a primitive of \mathfrak{M} while the latter is a formula derived from such primitives. I use `typewriter` font for primitives and `sans serif` font for user-defined formulas. IPA symbols should also be assumed to be user-defined formulas.

Natural classes can be defined similarly to $\mathbf{b}(x)$. Given the primitives in \mathcal{R}_f , I define $\text{obs}(x)$ and $\text{son}(x)$ in (3.25-3.26).⁶

$$\text{obs}(x) \stackrel{\text{def}}{=} \text{stop}(x) \vee \text{fric}(x) \quad (3.25)$$

“ x is an **obstruent** iff x is a **stop** or x is a **fricative**.”

$$\text{son}(x) \stackrel{\text{def}}{=} \neg \text{obs}(x) \quad (3.26)$$

“Position x is a **sonorant** iff x is not an **obstruent**.”

Unary formulas can also describe certain positions in the word. Initial and final positions are defined as in §3.1.1. That is, the initial position is its own predecessor and the final position is its own successor. Then a medial position is one that is neither initial nor final. These definitions are formalized in (3.27-3.29).

⁶ **obs** = obstruent; **son** = sonorant; **init** = initial; **fin** = final; **med** = medial; **pk** = peak; **prom** = prominence; **mrkd** = marked.

$$\mathbf{init}(x) \stackrel{\text{def}}{=} \mathit{pred}(x) = x \quad (3.27)$$

$$\mathbf{fin}(x) \stackrel{\text{def}}{=} \mathit{succ}(x) = x \quad (3.28)$$

$$\mathbf{med}(x) \stackrel{\text{def}}{=} \neg(\mathbf{init}(x) \vee \mathbf{fin}(x)) \quad (3.29)$$

Additional shorthand formulas will be defined in subsequent chapters.

Note that whether a formula is MSO, FO, or QF is determined by its interpretation in terms of primitives. For example, the statement $\mathbf{obs}(x) \vee \mathbf{b}(x)$ is QF because it is a conjunction of two formulas ($\mathbf{obs}(x)$ and $\mathbf{b}(x)$) that are both QF. User-defined formulas are not meant to obscure the logical nature of the description; they are just well-defined abbreviations.

Chapter 4

SYLLABLE REPRESENTATIONS

One key issue in all areas of theoretical linguistics is the best way of representing abstract structures, and syllable theory is no exception. A variety of syllable representations have been employed in the phonological literature (e.g., [Davis, 1985](#); [Duanmu, 2009](#); [Hooper, 1972](#)). A natural question is to what extent the differences between competing representations are significant or merely matters of notation—and how can we even determine whether representation types are equivalent or not? In this chapter, I first introduce the notion of *notational equivalence* and discuss how model theory can be used to compare representation types. Then I formalize three popular representations of syllable structure and show that they are notationally equivalent, in a strict sense.

4.1 Notational Equivalence

We encounter many types of abstract representations in our everyday lives without giving much thought to which aspects of notation are meaningful and which are merely stylistic. For example, intuitions are clear that font choice is irrelevant to the representation of a set of numbers: both $\{1,2\}$ and $\{\mathbf{1},\mathbf{2}\}$ describe the same set. As practitioners, we abstract away from font choice when representing such sets and confidently choose among fonts arbitrarily. In contrast, there are meaningful differences between bracket types in mathematical notation: $\{1, 2\}$ is an unordered set, while $(1, 2)$ is an ordered pair.

Are different syllabic representations interchangeable in the same way as fonts, or do they constitute truly different structures, like sets and ordered pairs? If they are notationally equivalent, it stands to reason that any constraint stated in one representation can be readily ‘translated’ and stated in another. If not, different predictions about syllable typology could be made based on the types of constraints admitted by each representation.

No previous literature addresses the question of whether different representations of syllable structure truly encode different information, but model theory and logic provide a foundation for doing just that (Danis & Jardine, 2019; Potts & Pullum, 2002; Pullum, 2007; Strother-Garcia & Heinz, 2017). The expressive power of the logic needed to ‘translate’ between two representational structures can be used as a measure of the meaningful difference between the representations. Before we can assess these differences, however, we must first define model-theoretic structures corresponding different syllable representation types.

4.2 Representations of Syllable Structure

Consider the following representations of the word *plenty*. The first simply demarcates syllable boundaries, as in Hooper (1972); Vennemann (1968b); and others. I call this the dot representation due to the popularity of marking syllable boundaries with periods or dots, as in Figure 4.1.

Figure 4.1: Dot representation: a string of segments and syllable boundaries

[plɛn.ti]

The remaining two representations refer to the syllable constituents *onset*, *nucleus*, and *coda*. Daland et al. (2011), Heinz & Strother-Garcia (to appear), and Strother-Garcia (2018) use the representation in Figure 4.2, which I call the flat representation. It

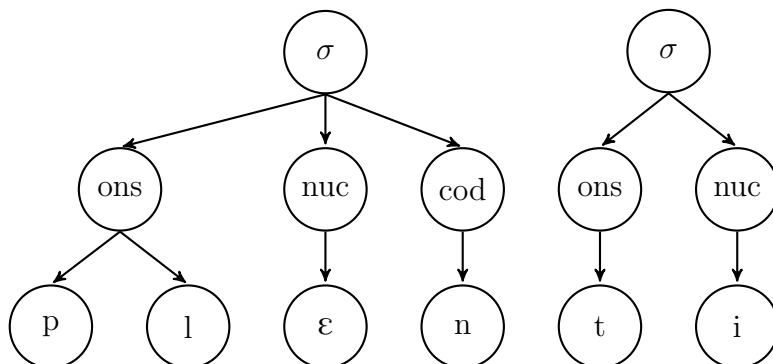
treats syllable constituents as secondary labels of each segment. For example, the p and l in *plenty* are both labeled *ons*.

Figure 4.2: Flat representation: a string of segments labeled with syllable constituents

p	l	ε	n	t	i
ons	ons	nuc	cod	ons	nuc

Finally, the representation in Figure 4.3 was introduced in Saporta & Contreras (1962) and subsequently argued for in Davis (1985) and others.¹ It represents syllables hierarchically, with syllable nodes σ dominate syllable constituent nodes, which in turn dominate segment nodes. For this reason I call it the tree representation.

Figure 4.3: Tree representation: hierarchical structure



These three examples have been illustrated here using the same conventions typically used in the literature. In the following subsections, I define formal model theories corresponding to each type of representation. This formalization is a necessary first step towards developing graph transductions between the representation types.

¹ A representation of this type is also intimated in Pike & Pike (1947), although no visual aid is provided.

4.2.1 The Dot Model Theory

The Dot Model Theory is a formalization of the dot representation. Its alphabet is the *union* of \mathcal{F} (the set of primitive phonological features) and the \bullet symbol, as in (4.1). The union of two sets is simply a new set containing all the elements of the original two sets. Then the set of relations \mathcal{R}^{dot} is simply the set of labels corresponding to members of the alphabet, as in (4.2).

$$\Sigma^{dot} \stackrel{\text{def}}{=} \mathcal{F} \cup \{\bullet\} \quad (4.1)$$

$$\mathcal{R}^{dot} \stackrel{\text{def}}{=} \{R_s \mid s \in \Sigma^{dot}\} \quad (4.2)$$

The Dot Model signature is given in (4.3). As in the traditional Successor Model, there are no binary relations, but there are two unary functions: $pred(x)$ and $succ(x)$.

$$\mathfrak{M}^{dot} \stackrel{\text{def}}{=} \langle \mathbb{N}; \mathcal{R}^{dot}; \{pred(x), succ(x)\} \rangle \quad (4.3)$$

The dot word model for *plenty* [plɛnti] is given in (4.4) illustrated in Figure 4.4. Recall that terms in **sans serif** font (e.g., **p**, **l**) are user-defined formulas used as shorthand for sets of phonological feature labels.

$$\mathcal{M}_{plenty}^{dot} \stackrel{\text{def}}{=} \langle \mathcal{D}; \mathcal{R}^{dot}; \{pred(x), succ(x)\} \rangle \quad (4.4)$$

$$\mathcal{D} \stackrel{\text{def}}{=} \{1, \dots, 7\}$$

$$\mathbf{stop} \stackrel{\text{def}}{=} \{1, 6\}$$

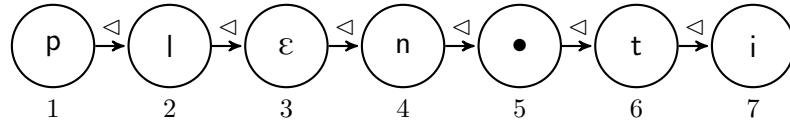
$$\mathbf{nas} \stackrel{\text{def}}{=} \{4\}$$

$$\bullet \stackrel{\text{def}}{=} \{5\}$$

⋮

$$succ(x) \stackrel{\text{def}}{=} \begin{cases} 2 & \Leftrightarrow x = 1 \\ 3 & \Leftrightarrow x = 2 \\ 4 & \Leftrightarrow x = 3 \\ 5 & \Leftrightarrow x = 4 \\ 6 & \Leftrightarrow x = 5 \\ 7 & \Leftrightarrow x \in \{6, 7\} \end{cases} \quad pred(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \Leftrightarrow x \in \{1, 2\} \\ 2 & \Leftrightarrow x = 3 \\ 3 & \Leftrightarrow x = 4 \\ 4 & \Leftrightarrow x = 5 \\ 5 & \Leftrightarrow x = 6 \\ 6 & \Leftrightarrow x = 7 \end{cases}$$

Figure 4.4: $\mathcal{M}_{plenty}^{dot}$



Comparing Figure 4.1 to Figure 4.4, it is clear that both figures illustrate the dot representation of the word *plenty*, but the latter explicitly encodes the order of segments, which is left implicit in the former. The word model also has explicit position indices, making it easy to refer to a given position and its properties. In general, formal word models encode information about positions that is only implicit in the traditional representations of phonological structure.

4.2.2 The Flat Model Theory

The Flat Model Theory is a formalization of the flat representation. Its alphabet is the union of \mathcal{F} and the set of syllable constituent labels, $\{\text{ons}, \text{nuc}, \text{cod}\}$, as in (4.5). Its set of relations and its model signature are given in (4.6) and (4.7), respectively.

$$\Sigma^{flat} \stackrel{\text{def}}{=} \mathcal{F} \cup \{\text{ons}, \text{nuc}, \text{cod}\} \quad (4.5)$$

$$\mathcal{R}^{flat} \stackrel{\text{def}}{=} \{R_s \mid s \in \Sigma^{flat}\} \quad (4.6)$$

$$\mathfrak{M}^{flat} \stackrel{\text{def}}{=} \langle \mathbb{N}; \mathcal{R}^{flat}; \{\text{pred}(x), \text{succ}(x)\} \rangle \quad (4.7)$$

Recall that, in the flat representation, words are represented as strings of segments with additional labels for onset, nucleus, and coda. The flat word model for *plenty* is given in (4.8) and illustrated in Figure 4.5.

$$\mathcal{M}_{plenty}^{flat} \stackrel{\text{def}}{=} \langle \mathcal{D}; \mathcal{R}^{flat}; \{\text{pred}(x), \text{succ}(x)\} \rangle \quad (4.8)$$

$$\mathcal{D} \stackrel{\text{def}}{=} \{1, \dots, 6\}$$

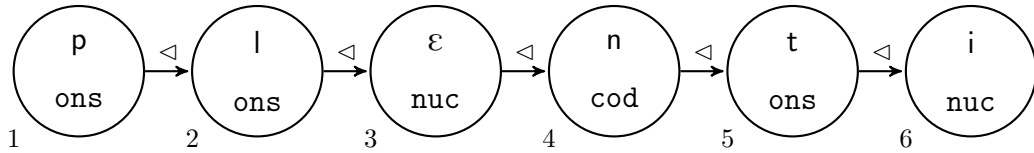
$$\text{voice} \stackrel{\text{def}}{=} \{2, 3, 4, 6\}$$

$$\text{ons} \stackrel{\text{def}}{=} \{1, 2, 5\}$$

⋮

$$\text{succ}(x) \stackrel{\text{def}}{=} \begin{cases} 2 & \Leftarrow x = 1 \\ 3 & \Leftarrow x = 2 \\ 4 & \Leftarrow x = 3 \\ 5 & \Leftarrow x = 4 \\ 6 & \Leftarrow x \in \{5, 6\} \end{cases} \quad \text{pred}(x) \stackrel{\text{def}}{=} \begin{cases} 1 & \Leftarrow x \in \{1, 2\} \\ 2 & \Leftarrow x = 3 \\ 3 & \Leftarrow x = 4 \\ 4 & \Leftarrow x = 5 \\ 5 & \Leftarrow x = 6 \end{cases}$$

Figure 4.5: $\mathcal{M}_{plenty}^{flat}$



4.2.3 The Tree Model Theory

The Tree Model Theory is a formalization of the tree representation, which is a hierarchical structure. I define hierarchical word models similarly to simple string models under the Successor Model Theory, with the addition of the function, $par(x)$. For a given position x , this function returns the *parent* of that position, i.e., the position that immediately dominates x . Thus, if position 1 immediately dominates position 3, I write $par(3) = 1$. Similar to $pred(x)$, $par(x)$ is defined such that the root node σ is its own parent, i.e., it immediately dominates itself. Note that each position is dominated by exactly one node.

The alphabet for the Tree Model Theory is the union of \mathcal{F} and the set $\{\sigma, \mathbf{ons}, \mathbf{nuc}, \mathbf{cod}\}$, as in (4.9). The set of relations and the Tree Model signature are given in (4.10–4.11).

$$\Sigma^{tree} \stackrel{\text{def}}{=} \mathcal{F} \cup \{\sigma, \mathbf{ons}, \mathbf{nuc}, \mathbf{cod}\} \quad (4.9)$$

$$\mathcal{R}^{tree} \stackrel{\text{def}}{=} \{R_s \mid s \in \Sigma^{tree}\} \quad (4.10)$$

$$\mathfrak{M}^{tree} \stackrel{\text{def}}{=} \langle \mathbb{N}; \mathcal{R}^{tree}; \{pred(x), succ(x), par(x)\} \rangle \quad (4.11)$$

The tree word model for *plenty* is given in (4.12) and illustrated in Figure 4.6.

$$\mathcal{M}_{plenty}^{tree} \stackrel{\text{def}}{=} \langle \mathcal{D}; \mathcal{R}^{tree}; \{pred(x), succ(x), par(x)\} \rangle \quad (4.12)$$

$$\mathcal{D} \stackrel{\text{def}}{=} \{1, \dots, 13\}$$

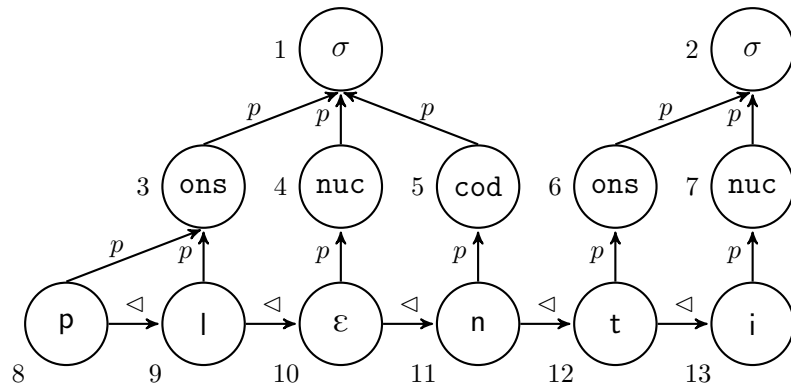
$$\mathbf{ons} \stackrel{\text{def}}{=} \{3, 6\}$$

$$\sigma \stackrel{\text{def}}{=} \{1, 2\}$$

⋮

$$\begin{array}{l}
succ(x) \stackrel{\text{def}}{=} \left\{ \begin{array}{l} 1 \Leftarrow x \in \{1, 2\} \\ 2 \Leftarrow x = 3 \\ 3 \Leftarrow x = 4 \\ 4 \Leftarrow x = 5 \\ 5 \Leftarrow x = 6 \\ 6 \Leftarrow x = 7 \\ 7 \Leftarrow x = 8 \\ 8 \Leftarrow x = 9 \\ 9 \Leftarrow x = 10 \\ 10 \Leftarrow x = 11 \\ 11 \Leftarrow x = 12 \\ 12 \Leftarrow x = 13 \end{array} \right.
\end{array}
\quad
\begin{array}{l}
pred(x) \stackrel{\text{def}}{=} \left\{ \begin{array}{l} 1 \Leftarrow x \in \{1, 2\} \\ 2 \Leftarrow x = 3 \\ 3 \Leftarrow x = 4 \\ 4 \Leftarrow x = 5 \\ 5 \Leftarrow x = 6 \\ 6 \Leftarrow x = 7 \\ 7 \Leftarrow x = 8 \\ 8 \Leftarrow x = 9 \\ 9 \Leftarrow x = 10 \\ 10 \Leftarrow x = 11 \\ 11 \Leftarrow x = 12 \\ 12 \Leftarrow x = 13 \end{array} \right.
\end{array}
\quad
\begin{array}{l}
par(x) \stackrel{\text{def}}{=} \left\{ \begin{array}{l} 1 \Leftarrow x \in \{1, 3, 4, 5\} \\ 2 \Leftarrow x \in \{2, 6, 7\} \\ 3 \Leftarrow x \in \{8, 9\} \\ 4 \Leftarrow x = 10 \\ 5 \Leftarrow x = 11 \\ 6 \Leftarrow x = 12 \\ 7 \Leftarrow x = 13 \end{array} \right.
\end{array}$$

Figure 4.6: $\mathcal{M}_{plenty}^{tree}$



The parent function is represented by directed edges (arrows) labeled p , pointing from a child to its parent. While dominance in a hierarchical structure is often represented

with edges directed in the other direction (from parent to child), I want to highlight the fact that this function points out a given position’s parent, of which there is exactly one. In contrast, a dominance ‘function’ would point from a parent to each of its children, meaning it could not be a function at all, because functions map a given input to only one output.

4.3 Comparing Different Models

On one hand, the flat and tree models may seem superior because they refer directly to syllable constituents. Yet it appears that the dot model has its own advantage over the flat model; marking syllable boundaries explicitly distinguishes two adjacent sounds that happen to be the same *type* of element from two adjacent sounds that belong to the *same* constituent of a single syllable, like the *pl* onset in *plenty*. The tree model also makes this distinction by having each syllable ultimately dominated by a single σ node. But is there a *true* difference in the information encoded in each representation—or are these apparent differences illusory?

4.3.1 L-interpretability

Word models can be compared on the basis of *L-interpretability*. A word model \mathcal{M}^1 is *L-interpretable* in terms of another, \mathcal{M}^2 , if one can write a graph transduction (in the sense of Engelfriet & Hoogeboom, 2001) from \mathcal{M}^1 to \mathcal{M}^2 using logic L. As explained in §3.3, a transduction is a way of translating information from one model to another using a logical language, L. If \mathcal{M}^1 is L-interpretable in terms of \mathcal{M}^2 and vice versa, then we say the two are *L-bi-interpretable*.

Informally, L-bi-interpretability means the two models are interchangeable with respect to logic L. It follows that the weaker the logic, the less meaningful the differences are between the models. A QF transduction is extremely restricted in the degree to which the output can differ from the input because QF is a weak logical language limited to

local operations. QF-bi-interpretability can therefore be considered an indication of notational equivalence.

In the remaining sections, I will show that the Dot, Flat, and Tree model theories are all QF-bi-interpretable, provided there is a bound on syllable size. This condition, while not universally adopted, has ample precedent in the phonological literature (e.g. [Duanmu, 2009](#)). In Sections [4.3.2](#) and [4.3.3](#), I define transductions from the Flat Model to the Tree Model and vice versa. The Flat-to-Dot and Dot-to-Flat transductions are defined in Sections [4.3.4](#) and [4.3.5](#), respectively. In each case, I illustrate an example of the transduction with the word model for *plenty* as input.

4.3.2 The Flat-to-Tree Transduction

I refer to this transduction as Γ_{ft} . It is the mapping from \mathfrak{M}^{flat} to \mathfrak{M}^{tree} , as expressed in [4.13](#).

$$\begin{aligned} & \Gamma_{ft} & (4.13) \\ & \mathfrak{M}^{flat} \xrightarrow{\Gamma} \mathfrak{M}^{tree} \\ & \langle \mathbb{N}; \mathcal{R}^{flat}; \{pred(x), succ(x)\} \rangle \xrightarrow{\Gamma} \langle \mathbb{N}; \mathcal{R}^{tree}; \{pred(x), succ(x), par(x)\} \rangle \end{aligned}$$

The crucial difference between the two lies in the sets of relations and functions. \mathcal{R}^{tree} includes everything in \mathcal{R}^{flat} , as well as the unary relation σ . \mathfrak{M}^{tree} has all the functions of \mathfrak{M}^{flat} , with the addition of the unary function $par(x)$.

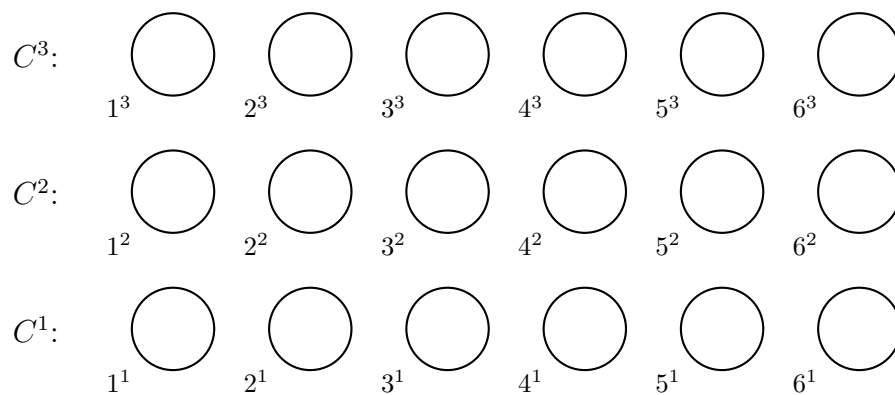
Copy Sets

Observe that the Flat Model for a given word will always have fewer positions than the Tree Model for the same word. Each segmental position must have as its parent a syllable constituent node (onset, nucleus, or coda), and each nucleus must have a parent σ node. How do we account for the fact that the output of the transduction must have more positions than the input? Put simply: copy sets.

Recall the transduction $\Gamma_{b \rightarrow a}$ from the Chapter 3, whose input and output domains were the same size. This output domain consists of a *copy set*, a set of positions identical to the input domain. While only one copy set was required for $\Gamma_{b \rightarrow a}$, transductions in general allow an arbitrary, but fixed number of copy sets. Three copy sets will be necessary for Γ_{ft} because a given segmental position is immediately dominated by a syllable constituent node, which in turn is immediately dominated by a σ node.

Given the input $\mathcal{M}_{plenty}^{flat}$ (see Figure 4.5), the output domain (also called the *codomain*) for this transduction is constructed as in Figure 4.7.

Figure 4.7: The codomain for $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



Each copy set has exactly the same number of positions as the input domain, which is 6 in this case. Instead of using ω to refer to the codomain as a whole, each output position is given a superscript indicating which copy set it belongs to. For example, output position 2^3 corresponds to position 2 in Copy Set 3, which is the third copy of input position 2. Altogether the codomain of $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$ has 18 positions.

In the remaining subsections, I will develop the transduction Γ_{ft} in a piecemeal fashion. Note that the transduction itself is not procedural; that is, formulas need not be evaluated in any particular order. The order of presentation here is arbitrary.

Unary Relations

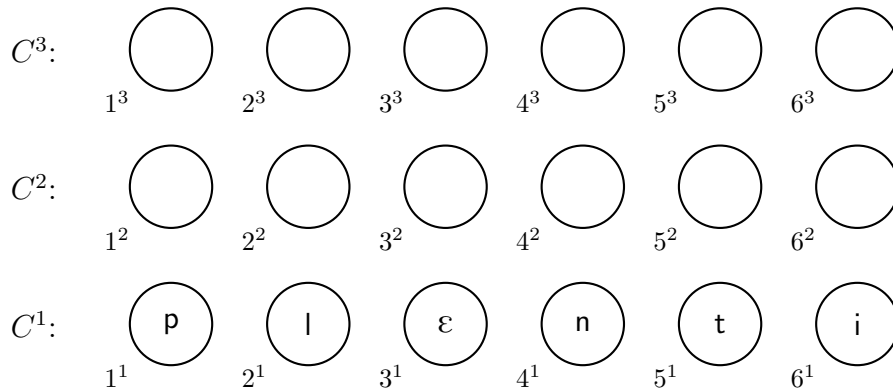
Unary Relations for Copy Set 1. While a single position in the Flat Model is labeled with segmental features **and** syllable constituents, these labels are applied to distinct positions in the Tree Model. Segmental features are labeled in Copy Set 1, as specified in (4.14).

$$(\forall \mathbf{f} \in \mathcal{R}_f)[\mathbf{f}(x^1) \stackrel{\text{def}}{=} \mathbf{f}(x)] \quad (4.14)$$

“For all features \mathbf{f} in the set \mathcal{R}_f , the first copy of x in the output is labeled \mathbf{f} iff x is labeled \mathbf{f} in the input.”

The following figure illustrates the contribution of these segmental feature formulas to the output, with user-defined formulas like \mathbf{p} standing in for feature bundles.

Figure 4.8: Labels for Copy Set 1 in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



It should be noted that, for every feature \mathbf{f} in \mathcal{F} and for every position x in the output, the formula $\mathbf{f}(x)$ evaluates to either true or false. While it is obvious that $\text{stop}(1^1) = \text{TRUE}$, it is also true that $\text{fric}(1^1) = \text{FALSE}$, $\text{stop}(2^1) = \text{FALSE}$, and so on. All of these values must be specified to fully define the transduction. Moving forward,

I focus on only those formulas that may be *true* of some positions in the output form. I will also omit position indices in figures for clarity.

Unary Relations for Copy Set 2. Syllable constituent labels are preserved from the input to the output in a similar way, but crucially in Copy Set 2. For now, I assume that each nucleus is comprised of a single segment, so the **nuc** relation is preserved exactly. However, a single **ons/cod** node in the Tree Model can dominate multiple segment nodes. In fact, contiguous segments labeled **ons/cod** in the Flat Model *should* all be dominated by the same **ons/cod** node in the Tree Model, in accordance with the syllable theory literature. To capture this generalization, it will be useful to refer directly to the onset/coda segment that is adjacent to the nucleus in the input. For reasons that will soon become apparent, I call these **ons_1** and **cod_1**, respectively. I will also refer to them as *nucleus-adjacent* segments. Note that (4.15) and (4.16) are user-defined formulas that refer to input positions only.

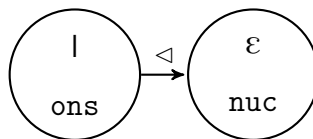
$$\mathbf{ons_1}(x) \stackrel{\text{def}}{=} \mathbf{ons}(x) \wedge \mathbf{nuc}(\mathit{succ}(x)) \quad (4.15)$$

$$\mathbf{cod_1}(x) \stackrel{\text{def}}{=} \mathbf{cod}(x) \wedge \mathbf{nuc}(\mathit{pred}(x)) \quad (4.16)$$

“Position x is nucleus-adjacent (**ons_1/cod_1**) iff x is labeled **ons/cod** and its successor/predecessor is labeled **nuc**.”

For example, Figure 4.9 illustrates a nucleus-adjacent onset which (by definition) satisfies **ons_1**.

Figure 4.9: A nucleus-adjacent onset defined with the Flat Model theory



Then the output labeling formulas for **nuc**, **ons**, and **cod** are as follows. As will be seen

in §4.3.2, labeling only the nucleus-adjacent onsets and codas in the output ensures that only nodes in Copy Set 2 that dominate a segmental node in Copy Set 1 will end up being licensed.

$$\text{nuc}(x^2) \stackrel{\text{def}}{=} \text{nuc}(x) \quad (4.17)$$

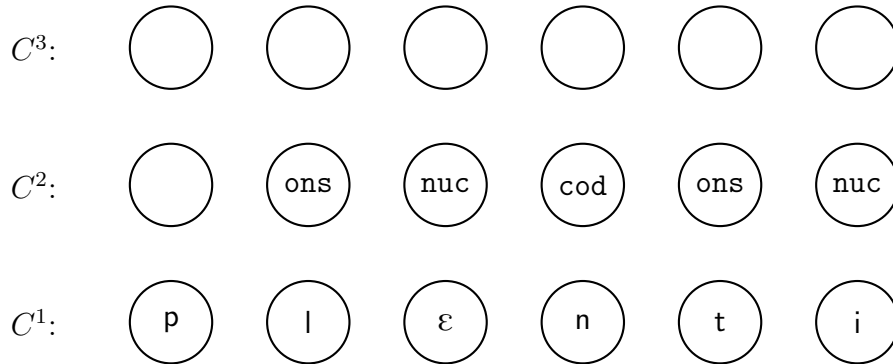
$$\text{ons}(x^2) \stackrel{\text{def}}{=} \text{ons}_1(x) \quad (4.18)$$

$$\text{cod}(x^2) \stackrel{\text{def}}{=} \text{cod}_1(x) \quad (4.19)$$

“Position x in Copy Set 2 is labeled **nuc/ons/cod** iff x satisfies **nuc/ons_1/cod_1** in the input.”

The figure below illustrates the contribution these formulas make to the output.

Figure 4.10: Labels for Copy Sets 1 and 2 in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



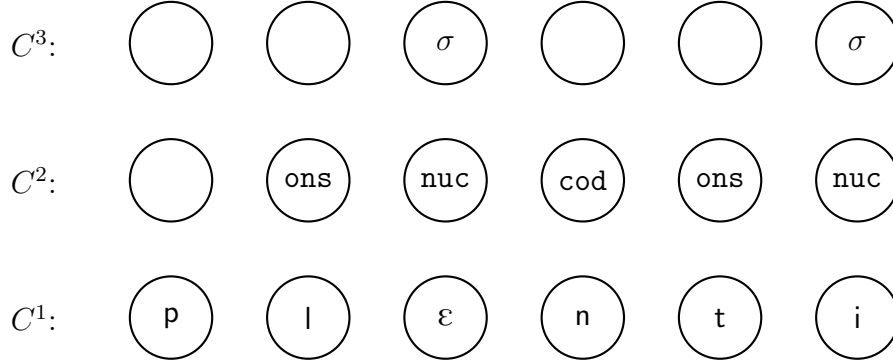
Unary Relations for Copy Set 3. The third copy set is reserved for σ nodes. Every σ node corresponds to a position labeled **nuc** in the input.

$$\sigma(x^3) \stackrel{\text{def}}{=} \text{nuc}(x) \quad (4.20)$$

“Position x in Copy Set 3 is labeled σ iff x is labeled **nuc** in the input.”

An illustration of this addition to the output is given below.

Figure 4.11: Labels for all three copy sets in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



Successor and Predecessor functions

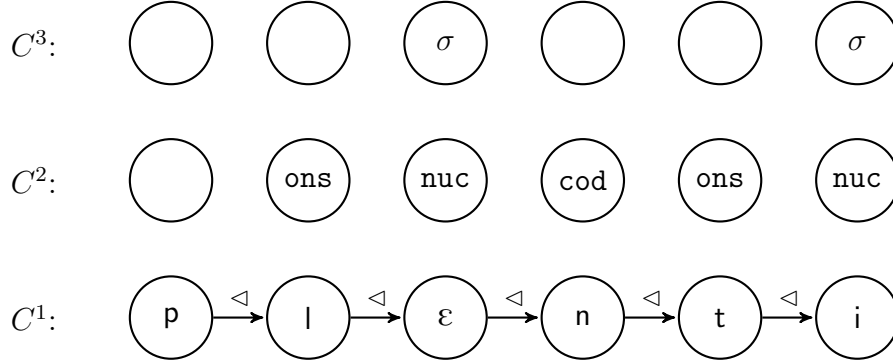
Successor and predecessor functions are also preserved in Copy Set 1 in this transduction. Recall that, while labeling relations are either true or false, functions take one position as an argument and return another position. It is therefore necessary to specify the copy set index for both the input to and output of the function.

$$succ(x^1) \stackrel{\text{def}}{=} (succ(x))^1 \tag{4.21}$$

$$pred(x^1) \stackrel{\text{def}}{=} (pred(x))^1 \tag{4.22}$$

“The successor/predecessor of the first copy of x in the output is the first copy of the successor/predecessor of x in the input.”

Figure 4.12: The successor function in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



Parent Functions

Dominance information is not explicitly encoded in the Flat Model, but it can be deduced from information that is present in the Flat Model. Unlike the successor and predecessor functions, the parent function must be defined **across** copy sets. In this section, I will first define several specific cases of parent functions. The general parent functions $par(x^2)$ and $par(x^1)$ will then be disjunctions of these cases.

Parent Functions for Nucleus Nodes. Dominance of σ nodes in Copy Set 3 over nuc nodes in Copy Set 2 is established as follows:

$$\text{nuc}(x) \Rightarrow \text{par}(x^2) = x^3 \quad (4.23)$$

“If x is labeled nuc in the input, the parent of the second copy of x is the third copy of x .”

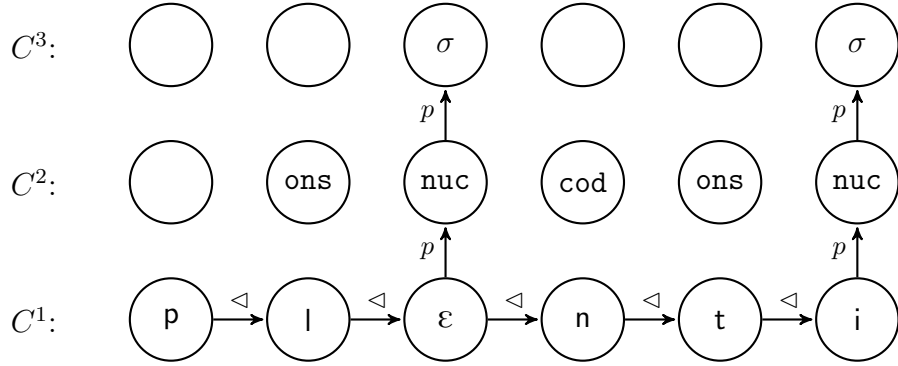
Similarly, nuc nodes in Copy Set 2 dominate segmental nodes in Copy Set 1:

$$\text{nuc}(x) \Rightarrow \text{par}(x^1) = x^2 \quad (4.24)$$

“If x is labeled nuc in the input, the parent of the first copy of x is the second copy of x .”

The figure below illustrates these additions to the output.

Figure 4.13: Some dominance information in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



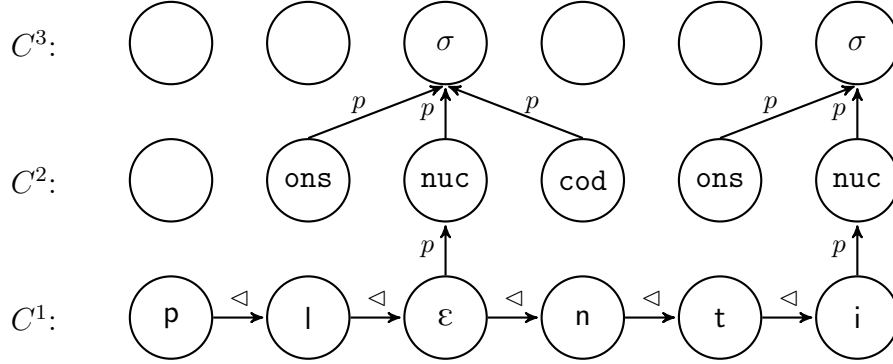
Parent Functions for Onset and Coda Nodes. Making use of the notion of nucleus-adjacency, the parent functions indicating dominance of σ nodes over **ons**/**cod** nodes are given below. Restricting dominance from a σ node to only nucleus-adjacent onsets and codas ensures that it will dominate at most one of each.

$$\text{ons.1}(x) \Rightarrow \text{par}(x^2) = (\text{succ}(x))^3 \quad (4.25)$$

$$\text{cod.1}(x) \Rightarrow \text{par}(x^2) = (\text{pred}(x))^3 \quad (4.26)$$

“If x is a nucleus-adjacent onset/coda in the input, the parent of position x in Copy Set 2 is the third copy of the successor/predecessor of x .”

Figure 4.14: Additional dominance information in $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



Parent Functions for Segment Nodes. In addition to nucleus-adjacency, it will be useful to refer to the exact distance from the nucleus in the input. For example, in *plenty*, the *l* is a nucleus-adjacent onset segment while *p* is 2 positions away from the nucleus ε . Starting with *ons_1* (defined above) as the base case, I refer to prior onset segments in increasing number from right to left (from the nucleus outward), as follows:

$$\text{ons_2}(x) \stackrel{\text{def}}{=} \text{ons}(x) \wedge \text{ons_1}(\text{succ}(x)) \quad (4.27)$$

“Position x is ‘onset-2’ (2 positions before the nucleus) iff x is labeled *ons* and its successor is ‘onset-1’ (a nucleus-adjacent onset).”

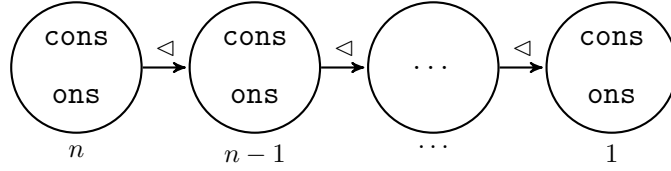
Taking n to be the maximum number of segments allowed in a single onset, there will be n such formulas. In the general case:

$$\text{ons_}i(x) \stackrel{\text{def}}{=} \text{ons}(x) \wedge \text{ons_}(i-1)(\text{succ}(x)) \text{ for } i \in \{2, \dots, n\} \quad (4.28)$$

“Position x is ‘onset- i ’ (i positions before the nucleus) iff x is labeled *ons* and its successor is ‘onset- $(i-1)$ ’, for i ranging from 2 to n .”

For example, Figure 4.15 shows an onset of length n , using indices that increase from right-to-left to illustrate the use of indices in $\text{ons_}i(x)$.

Figure 4.15: An onset of length n



Formulas analogous to $\text{ons}_i(x)$ can be written for codas, assuming a maximum length m and counting from left to right (again, outward from the nucleus). Given $\text{cod}_1(x)$ defined in (4.16) above, the general case for codas is defined in (4.29).

$$\text{cod}_i(x) \stackrel{\text{def}}{=} \text{cod}(x) \wedge \text{cod}_{i-1}(\text{pred}(x)) \text{ for } i \in \{2, \dots, m\} \quad (4.29)$$

“Position x is ‘coda- i ’ (i positions after the nucleus) iff x is labeled **cod** and its predecessor is ‘coda- $(i-1)$ ’, for i ranging from 2 to m .”

Although these formulas allow a certain type of counting, so to speak, note that they do not require counting over the entire input string. The bounds n and m enforce a finite window of look-ahead/look-back. Were this not the case, we would need to use quantifiers to identify a contiguous string of onset or coda segments with no intervening segments of a different syllable constituent type. An example of this type of formula, using the general predecessor relation $<$, is given in (4.30).

$$\begin{aligned} \text{ons}(x^2) \stackrel{\text{def}}{=} & \text{ons}(x) \wedge (\exists z)[\text{nuc}(z) \wedge x < z] \wedge \\ & \neg(\exists y)[x < y < z \wedge \neg\text{ons}(y)] \end{aligned} \quad (4.30)$$

“Position x in Copy Set 2 is labeled **ons** iff there exists a position z in the input labeled **nuc** that is preceded by x , and there does not exist a position y in the input such that $x < y < z$ and y is **not** labeled **ons** .”

Assuming no *a priori* bound on the length of a word, checking if a given position

satisfies this formula requires looking back an arbitrary number of segments to the very beginning of the string.

Given bounds n and m (or a single bound for the length of the entire word), however, parent functions from **ons** and **cod** nodes to segment nodes can be defined as follows in (4.31) and (4.32). I use the notation $succ^n(x)$ and $pred^m(x)$ to mean ‘the m^{th} successor/predecessor of x .’

$$par(x^1) = \begin{cases} (succ(x))^2 & \Leftrightarrow \text{ons_1}(x) \\ (succ(succ(x)))^2 & \Leftrightarrow \text{ons_2}(x) \\ \vdots & \vdots \\ (succ^n(x))^2 & \Leftrightarrow \text{ons_n}(x) \end{cases} \quad (4.31)$$

“The parent of position x in Copy Set 1 is the second copy of: the successor of x if input position x is ‘onset-1’; the second successor of x if input position x is ‘onset-2’; and so on, up to the n^{th} successor of x .”

$$par(x^1) = \begin{cases} (pred(x))^2 & \Leftrightarrow \text{cod_1}(x) \\ (pred(pred(x)))^2 & \Leftrightarrow \text{cod_2}(x) \\ \vdots & \vdots \\ (pred^m(x))^2 & \Leftrightarrow \text{cod_m}(x) \end{cases} \quad (4.32)$$

“The parent of position x in Copy Set 1 is the second copy of: the predecessor of x if input position x is ‘coda-1’; the second predecessor of x if input position x is ‘coda-2’; and so on, up to the m^{th} predecessor of x .”

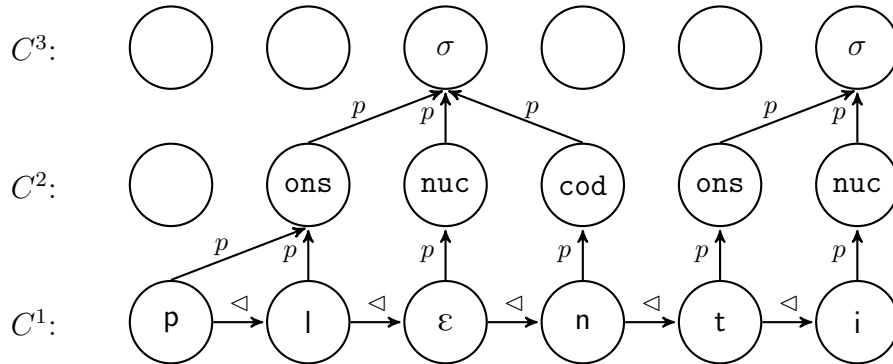
General Parent Functions. Now it is possible to write the general parent functions.

$$par(x^2) \stackrel{\text{def}}{=} \begin{cases} x^3 & \Leftrightarrow \text{nuc}(x) \\ (succ(x))^3 & \Leftrightarrow \text{ons}_1(x) \\ (pred(x))^3 & \Leftrightarrow \text{cod}_1(x) \end{cases} \quad (4.33)$$

$$par(x^1) \stackrel{\text{def}}{=} \begin{cases} x^2 & \Leftrightarrow \text{nuc}(x) \\ (succ(x))^2 & \Leftrightarrow \text{ons}_1(x) \\ (succ^i(x))^2 & \Leftrightarrow \text{ons}_i(x) \text{ for } i \in \{2, \dots, n\} \\ (pred(x))^2 & \Leftrightarrow \text{cod}_1(x) \\ (pred^i(x))^2 & \Leftrightarrow \text{cod}_i(x) \text{ for } i \in \{2, \dots, m\} \end{cases} \quad (4.34)$$

Other than these cases, no positions have parents at all. The figure below shows the fully specified output of the transduction.

Figure 4.16: $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$ fully specified



Licensing

Finally, I define a licensing function (as in Courcelle, 1994 and Engelfriet & Hoogeboom 2001), that indicates which positions in the codomain contribute meaningfully to the

output structure. Positions in the codomain that are not labeled are uninterpretable, so I license all and only those positions that have at least one label, as in (4.35).

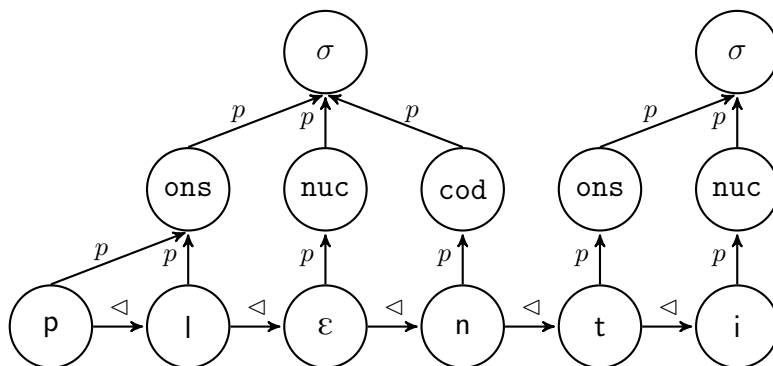
$$license(x) \stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{tree})[R(x)] \quad (4.35)$$

“Output position x is licensed iff there is some relation R in \mathcal{R}^{tree} such that $R(x)$ is true.”

Note that the definition of the licensing function is *not* QF; it uses the existential quantifier \exists . This is not a problem for my analysis because the transduction itself is QF—licensing is just a helpful way of carving out the meaningful part of the output.

Figure 4.17 shows the final output with unlicensed positions omitted. Comparing this to Figure 4.6, it is clear that $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$ is exactly $\mathcal{M}_{plenty}^{tree}$.

Figure 4.17: The licensed output of $\Gamma_{ft}(\mathcal{M}_{plenty}^{flat})$



Summary of the Flat-to-Tree Transduction and Licensing

$\Gamma_{ft}(\mathcal{M}^{flat})$ is partially defined as follows.

$$\begin{aligned}
\mathbf{f}(x^1) &\stackrel{\text{def}}{=} \mathbf{f}(x) \text{ for } \mathbf{f} \in \mathcal{F} \\
\mathbf{nuc}(x^2) &\stackrel{\text{def}}{=} \mathbf{nuc}(x) \\
\mathbf{ons}(x^2) &\stackrel{\text{def}}{=} \mathbf{ons_1}(x) \\
\mathbf{cod}(x^2) &\stackrel{\text{def}}{=} \mathbf{cod_1}(x) \\
\sigma(x^3) &\stackrel{\text{def}}{=} \mathbf{nuc}(x) \\
\mathit{succ}(x^1) &\stackrel{\text{def}}{=} (\mathit{succ}(x))^1 \\
\mathit{pred}(x^1) &\stackrel{\text{def}}{=} (\mathit{pred}(x))^1 \\
\mathit{par}(x^2) &\stackrel{\text{def}}{=} \begin{cases} x^3 & \Leftrightarrow \mathbf{nuc}(\mathbf{x}) \\ (\mathit{succ}(x))^3 & \Leftrightarrow \mathbf{ons_1}(\mathbf{x}) \\ (\mathit{pred}(x))^3 & \Leftrightarrow \mathbf{cod_1}(\mathbf{x}) \end{cases} \\
\mathit{par}(x^1) &\stackrel{\text{def}}{=} \begin{cases} x^2 & \Leftrightarrow \mathbf{nuc}(\mathbf{x}) \\ (\mathit{succ}(x))^2 & \Leftrightarrow \mathbf{ons_1}(\mathbf{x}) \\ (\mathit{succ}^i(x))^2 & \Leftrightarrow \mathbf{ons_}i(\mathbf{x}) \text{ for } i \in \{2, \dots, n\} \\ (\mathit{pred}(x))^2 & \Leftrightarrow \mathbf{cod_1}(\mathbf{x}) \\ (\mathit{pred}^i(x))^2 & \Leftrightarrow \mathbf{cod_}i(\mathbf{x}) \text{ for } i \in \{2, \dots, m\} \end{cases} \\
\mathit{license}(x) &\stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{tree})[R(x)]
\end{aligned}$$

For completeness, $\Gamma_{ft}(\mathfrak{M}^{flat})$ must also include formulas for all logically possible values for every relation in every copy set, even if no positions in that copy set have the given label. For example, (4.36) states that no position in Copy Set 1 is labeled **nuc**. Similarly, (4.37) states that no position in Copy Set 2 is labeled **voice**.

$$\text{nuc}(x^1) \stackrel{\text{def}}{=} \text{FALSE} \tag{4.36}$$

$$\text{voice}(x^2) \stackrel{\text{def}}{=} \text{FALSE} \tag{4.37}$$

...

I omit these types of formulas in the remaining transduction summaries for brevity.

4.3.3 The Tree-to-Flat Transduction

I refer to this transduction as Γ_{tf} . It is the mapping from \mathfrak{M}^{tree} to \mathfrak{M}^{flat} .

$$\mathfrak{M}^{tree} \xrightarrow{\Gamma} \mathfrak{M}^{flat} \tag{4.38}$$

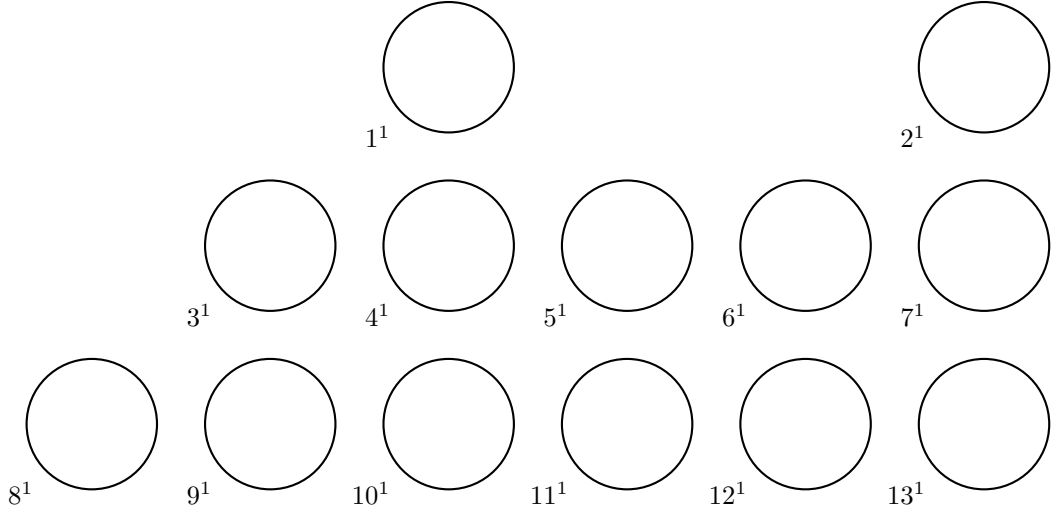
$$\langle \mathbb{N}; \mathcal{R}_{tree}; \{succ(x), pred(x), par(x)\} \rangle \xrightarrow{\Gamma} \langle \mathbb{N}, \mathcal{R}_{flat}; \{succ(x), pred(x)\} \rangle$$

Copy Set(s)

Only one copy set is required because the Tree Model for a given word will always have more positions than the Flat Model for the same word.

I illustrate this transduction using the input $\mathcal{M}_{plenty}^{tree}$ (given in Figure 4.6). Observe that every position number has subscript 1, meaning they all belong to Copy Set 1. I show these positions configured in a shape reminiscent of the tree model, but note that they are not inherently ordered or related to each other. This configuration is simply conducive to visualizing the output in a traditional way.

Figure 4.18: The codomain for $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$



Unary Relations

Just as in Γ_{ft} , segmental features are preserved in Γ_{tf} .

$$(\forall \mathbf{f} \in \mathcal{R}_f)[\mathbf{f}(x^1) \stackrel{\text{def}}{=} \mathbf{f}(x)] \quad (4.39)$$

“For all features \mathbf{f} in the set \mathcal{R}_f , the first copy of x in the output is labeled \mathbf{f} iff x is labeled \mathbf{f} in the input.”

Whereas onsets, nuclei, and codas in the Tree Model dominate corresponding segments, this information is collapsed onto a single position in the Flat Model. Thus every segmental position in the output bears the syllable constituent label of its parent in the input. This is formalized in (4.40–4.42).

$$\mathbf{ons}(x^1) \stackrel{\text{def}}{=} \mathbf{ons}(\mathit{par}(x)) \quad (4.40)$$

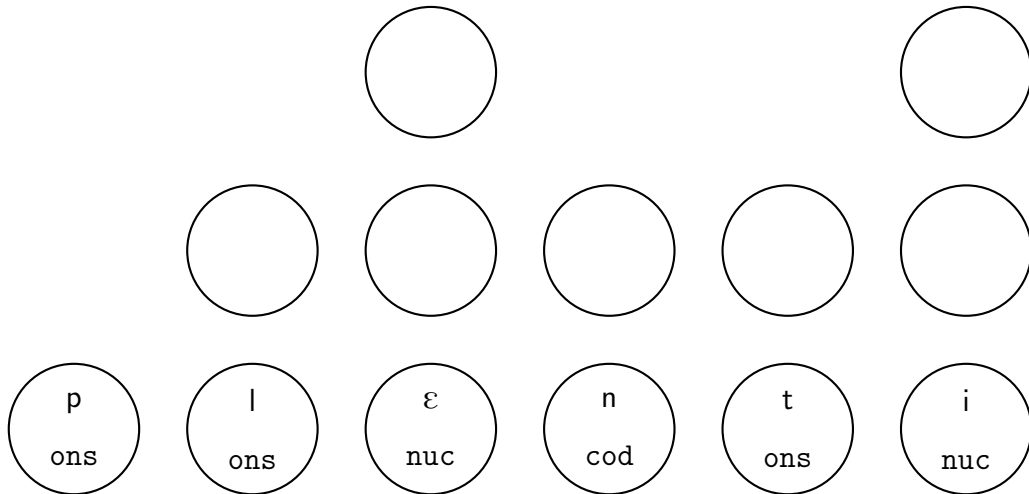
$$\mathbf{nuc}(x^1) \stackrel{\text{def}}{=} \mathbf{nuc}(\mathit{par}(x)) \quad (4.41)$$

$$\mathbf{cod}(x^1) \stackrel{\text{def}}{=} \mathbf{cod}(\mathit{par}(x)) \quad (4.42)$$

“Position x in Copy Set 1 is labeled **ons/nuc/cod** iff the parent of x in the input is labeled **ons/nuc/cod**.”

The figure below illustrates the contribution of these formulas to the output.

Figure 4.19: Unary relations for $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$



Successor and Predecessor functions

Successor and predecessor functions are preserved in Copy Set 1, just as in Γ_{ft} .

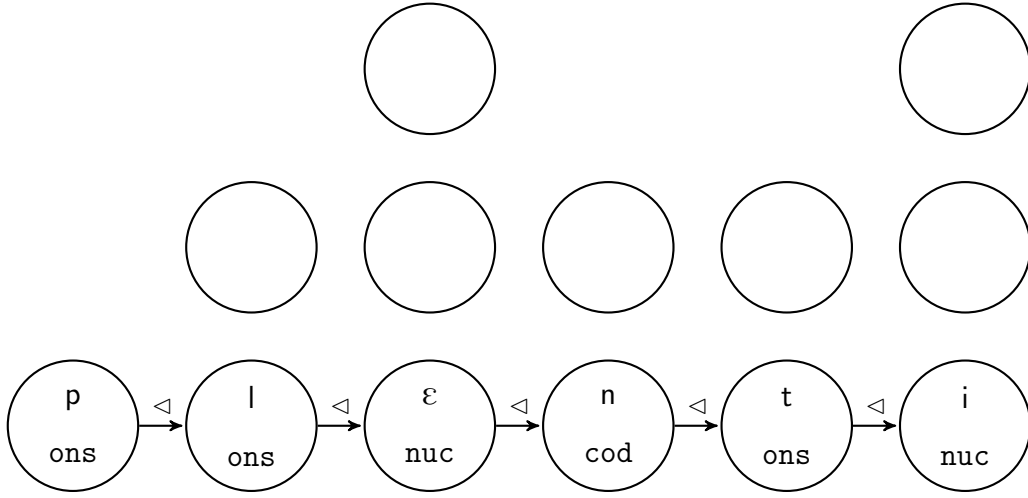
$$succ(x^1) \stackrel{\text{def}}{=} (succ(x))^1 \quad (4.43)$$

$$pred(x^1) \stackrel{\text{def}}{=} (pred(x))^1 \quad (4.44)$$

“The successor/predecessor of position x in Copy Set 1 is the first copy of the successor/predecessor of x in the input.”

The result of these formulas is illustrated in Figure 4.20.

Figure 4.20: $\Gamma_{tf}(\mathcal{M}_{plenty}^{tree})$ fully specified



Dominance information is not encoded in the output at all. All that remains is to license interpretable (labeled) nodes, as in (4.45).

$$license(x) \stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{flat})[R(x)] \quad (4.45)$$

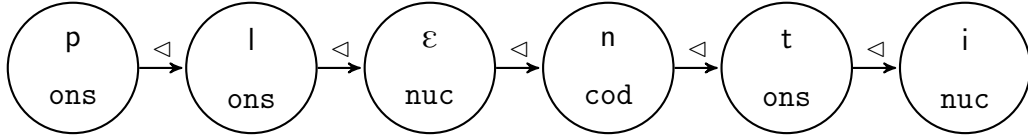
“Output position x is licensed iff there is some relation R in \mathcal{R}^{flat} such that $R(x)$ is true.”

Summary of The Tree-to-Flat Transduction and Licensing

$$\begin{aligned}
 \mathbf{f}(x^1) &\stackrel{\text{def}}{=} \mathbf{f}(x) \text{ for } \mathbf{f} \in \mathcal{F} \\
 \text{nuc}(x^1) &\stackrel{\text{def}}{=} \text{nuc}(\text{par}(x)) \\
 \text{ons}(x^1) &\stackrel{\text{def}}{=} \text{ons}(\text{par}(x)) \\
 \text{cod}(x^1) &\stackrel{\text{def}}{=} \text{cod}(\text{par}(x)) \\
 \text{succ}(x^1) &\stackrel{\text{def}}{=} (\text{succ}(x))^1 \\
 \text{pred}(x^1) &\stackrel{\text{def}}{=} (\text{pred}(x))^1 \\
 \text{license}(x) &\stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{\text{flat}})[R(x)]
 \end{aligned}$$

Comparing Figure 4.21 to Figure 4.5, it is clear that the final output of $\Gamma_{ft}(\mathcal{M}_{\text{plenty}}^{\text{tree}})$ is exactly $\mathcal{M}_{\text{plenty}}^{\text{flat}}$.

Figure 4.21: The licensed output of $\Gamma_{tf}(\mathcal{M}_{\text{plenty}}^{\text{tree}})$



4.3.4 The Flat-to-Dot Transduction

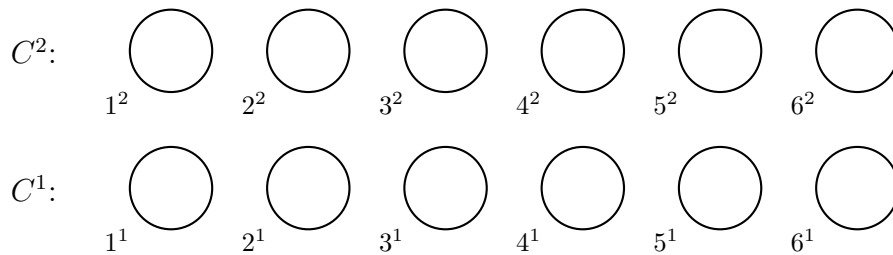
I refer to this transduction as Γ_{fd} . It is the mapping from $\mathfrak{M}^{\text{flat}}$ to $\mathfrak{M}^{\text{dot}}$.

$$\begin{aligned}
 \mathfrak{M}^{\text{flat}} &\xrightarrow{\Gamma} \mathfrak{M}^{\text{dot}} \\
 \langle \mathbb{N}; \mathcal{R}^{\text{flat}}; \{\text{pred}(x), \text{succ}(x)\} \rangle &\xrightarrow{\Gamma} \langle \mathbb{N}; \mathcal{R}^{\text{dot}}; \{\text{pred}(x), \text{succ}(x)\} \rangle
 \end{aligned} \tag{4.46}$$

Copy Set(s)

The Flat Model for a given word has exactly the number of positions as the number of segments in the word, whereas the Dot Model has additional positions for syllable boundaries (dots). I choose to mark only internal syllable boundaries and not word boundaries, so a word of length n may have up to $n - 1$ syllable boundaries (if each segment constituted its own syllable). The size of the codomain is therefore bounded at $n + n - 1$, which is equal to $2n - 1$. Thus the transduction will require two copy sets.

Figure 4.22: The codomain for $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$



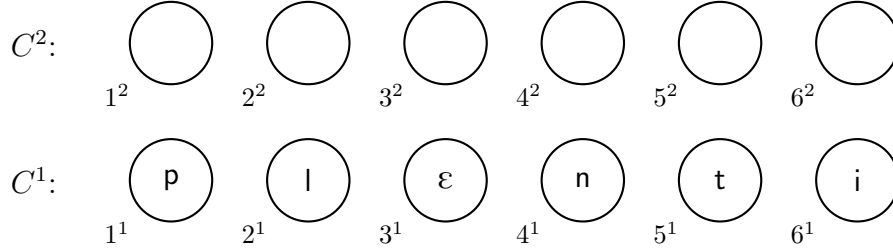
Unary Relations

The only unary relations preserved from the input in this transduction are feature labels. Additional labels are also defined over the output which are not present in the input.

Unary relations for Copy Set 1. Feature labels are mapped onto Copy Set 1, just as in the previous transductions developed in this chapter. Syllable constituents are not labeled at all in the Dot Model.

$$\mathbf{f}(x^1) \stackrel{\text{def}}{=} \mathbf{f}(x) \text{ for } \mathbf{f} \in \mathcal{F} \quad (4.47)$$

Figure 4.23: Labeling Copy Set 1 in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$



Unary Relations for Copy Set 2. Because syllable boundaries are not encoded explicitly in the Flat Model, their placement must be deduced from information in the input. In particular, a syllable boundary may occur in four environments:

1. Between a coda and the following onset.
2. Between a nucleus and the following onset.
3. Between a coda and the following nucleus.
4. Between two nuclei of adjacent syllables (hiatus).

Take the first example. The user-defined formula in (4.48) identifies the coda in a coda-onset sequence, which I refer to as a ‘c.o’ coda.

$$\text{c.o}(x) \stackrel{\text{def}}{=} \text{cod}(x) \wedge \text{ons}(\text{succ}(x)) \quad (4.48)$$

“Position x is ‘c.o’ iff x is a coda and its successor is an onset.”

formulas identifying the first segment in each of the other three environments are as follows in (4.49–4.51).

$$\mathbf{n.o}(x) \stackrel{\text{def}}{=} \mathbf{nuc}(x) \wedge \mathbf{ons}(\text{succ}(x)) \quad (4.49)$$

$$\mathbf{c.n}(x) \stackrel{\text{def}}{=} \mathbf{cod}(x) \wedge \mathbf{nuc}(\text{succ}(x)) \quad (4.50)$$

$$\mathbf{n.n}(x) \stackrel{\text{def}}{=} \mathbf{nuc}(x) \wedge \mathbf{nuc}(\text{succ}(x)) \quad (4.51)$$

A position that satisfies any of these formulas must immediately precede a syllable boundary. I therefore refer to them all as *pre-boundary* positions, defined in (4.52).

$$\mathbf{pre_bound}(x) \stackrel{\text{def}}{=} \mathbf{c.o}(x) \vee \mathbf{n.o}(x) \vee \mathbf{c.n}(x) \vee \mathbf{n.n}(x) \quad (4.52)$$

“Position x is a pre-boundary position iff it satisfies $\mathbf{c.o}(x)$, $\mathbf{c.n}(x)$, or $\mathbf{n.n}(x)$.”

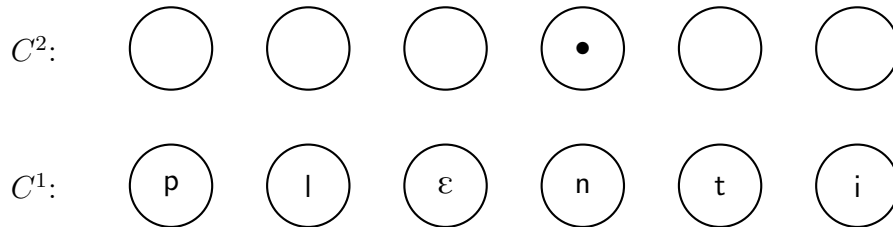
These will be labeled with a dot (\bullet) in Copy Set 2.

$$\bullet(x^2) \stackrel{\text{def}}{=} \mathbf{pre_bound}(x) \quad (4.53)$$

“Position x in Copy Set 2 is labeled \bullet iff x is a pre-boundary position in the input.”

Position 4 in the $\mathcal{M}_{plenty}^{flat}$ satisfies (4.48), so position 4^2 in the codomain is labeled \bullet . See Figure 4.24 below.

Figure 4.24: Labeling Copy Set 2 in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$



Successor and Predecessor functions

For the most part, successor and predecessor information is preserved in Copy Set 1. However, additional information about successors/predecessors between copy sets must be deduced so that the dot positions in Copy Set 2 are ordered correctly with respect to the segmental positions in Copy Set 1.

The Successor Function. In the case of a pre-boundary position, the successor function operates as follows in (4.54) and (4.55). Note that (4.54) describes a particular cases of $succ(x^1)$, but not the only case that is relevant here—the general successor function for Copy Set 1 will be defined later. In contrast, (4.55) covers the only case where the successor function is defined in Copy Set 2.

$$\text{pre_bound}(x) \Rightarrow succ(x^1) = x^2 \quad (4.54)$$

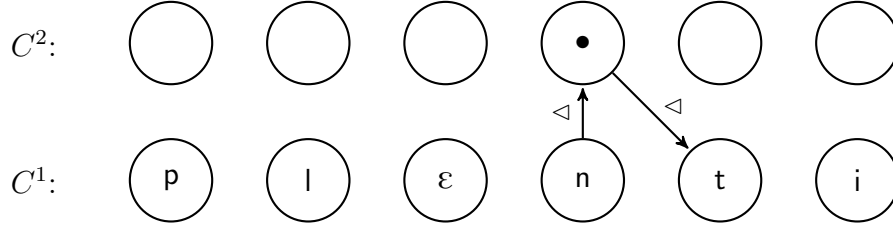
“If x is a pre-boundary position in the input, then the successor of the first copy of x is the second copy of x .”

$$succ(x^2) = (succ(x))^1 \Leftrightarrow \text{pre_bound}(x) \quad (4.55)$$

“The successor of the second copy of x is the first copy of the successor of x iff x is a pre-boundary position in the input.”

The combined effect of these two labeling formulas is that a pre-boundary position is followed by a dot, which is then followed by next the segmental position. See Figure 4.25 below.

Figure 4.25: Some successor information in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$



Excluding pre-boundary positions, the remaining successor information is preserved in Copy Set 1. Hence the other case of $succ(x^1)$, given in (4.56).

$$\neg \text{pre_bound}(x) \Rightarrow succ(x^1) = (succ(x))^1 \quad (4.56)$$

“If x is not a pre-boundary position in the input, then the successor of the first copy of x is the first copy of the successor of x .”

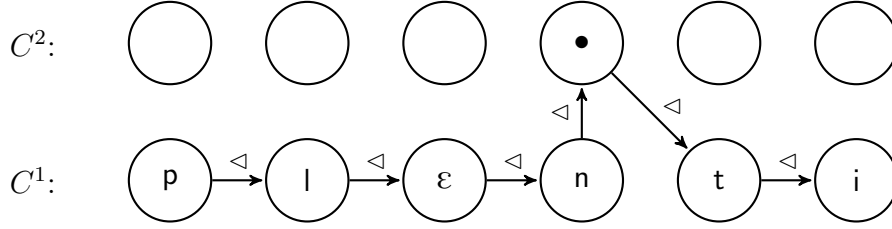
Then the general successor function for Copy Set 1 can be defined as follows in (4.57).

$$succ(x^1) \stackrel{\text{def}}{=} \begin{cases} x^2 & \Leftrightarrow \text{pre_bound}(x) \\ (succ(x))^1 & \Leftrightarrow \neg \text{pre_bound}(x) \end{cases} \quad (4.57)$$

$$(4.58)$$

Figure 4.26 shows the successor function in the output.

Figure 4.26: All successor information in $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$



The Predecessor Function. To develop the predecessor function, it will be useful to refer to post-boundary positions.

$$\text{post_bound}(x) \stackrel{\text{def}}{=} \text{pre_bound}(\text{pred}(x)) \quad (4.59)$$

“Position x is a post-boundary position iff its predecessor is a pre-boundary position.”

The predecessor of position x in Copy Set 1 is either the first or second copy of its input predecessor, depending on whether or not x is a post-boundary position. This is defined in (4.60) using cases, just like $\text{succ}(x^1)$.

$$\text{pred}(x^1) \stackrel{\text{def}}{=} \begin{cases} (\text{pred}(x))^2 & \Leftrightarrow \text{post_bound}(x) \\ (\text{pred}(x))^1 & \Leftrightarrow \neg \text{post_bound}(x) \end{cases} \quad (4.60)$$

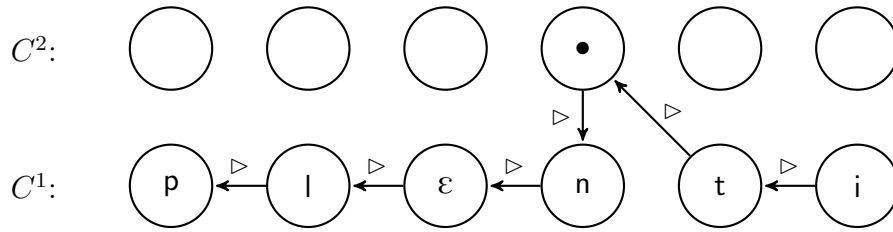
Finally, I define the predecessor function for Copy Set 2 as in (4.61).

$$\text{pred}(x^2) \stackrel{\text{def}}{=} (\text{pred}(x))^1 \Leftrightarrow \text{pre_bound}(x) \quad (4.61)$$

“The predecessor of position x in Copy Set 2 is the first copy of the predecessor of x in the input iff x is a pre-boundary position.”

Figure 4.27 illustrates the predecessor function in the output, which is essentially the inverse of the successor function.

Figure 4.27: $\Gamma_{fd}(\mathcal{M}_{plenty}^{flat})$ fully specified



Licensing

As before, all that is left is to license interpretable nodes, as in (4.62).

$$license(x) \stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{dot})[R(x)] \quad (4.62)$$

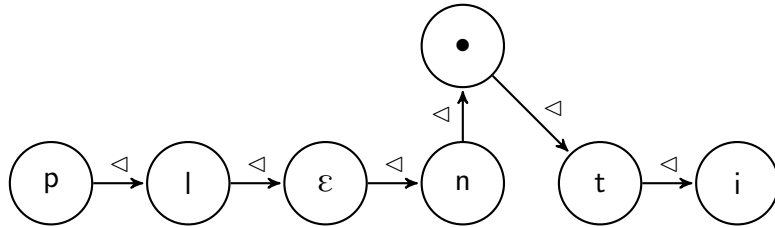
“Output position x is licensed iff there is some relation R in \mathcal{R}^{dot} such that $R(x)$ is true.”

Summary of The Flat-to-Dot Transduction and Licensing

$$\begin{aligned}
 \mathbf{f}(x^1) &\stackrel{\text{def}}{=} \mathbf{f}(x) \text{ for } \mathbf{f} \in \mathcal{F} \\
 \bullet(x^2) &\stackrel{\text{def}}{=} \text{pre_bound}(x) \\
 \text{succ}(x^1) &\stackrel{\text{def}}{=} \begin{cases} x^2 & \Leftrightarrow \text{pre_bound}(x) \\ (\text{succ}(x))^1 & \Leftrightarrow \neg \text{pre_bound}(x) \end{cases} \\
 \text{pred}(x^1) &\stackrel{\text{def}}{=} \begin{cases} (\text{pred}(x))^2 & \Leftrightarrow \text{post_bound}(x) \\ (\text{pred}(x))^1 & \Leftrightarrow \neg \text{post_bound}(x) \end{cases} \\
 \text{license}(x) &\stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{\text{dot}})[R(x)]
 \end{aligned}$$

The licensed output of this transduction is illustrated below. Note that it is mathematically identical to $\mathfrak{M}_{\text{plenty}}^{\text{dot}}$ (illustrated in Figure 4.4), although the visual representations differ slightly.

Figure 4.28: The licensed output of $\Gamma_{fd}(\mathcal{M}_{\text{plenty}}^{\text{flat}})$



4.3.5 The Dot-to-Flat Transduction

I refer to this transduction as Γ_{df} . It is the mapping from $\mathfrak{M}^{\text{dot}}$ to $\mathfrak{M}^{\text{flat}}$.

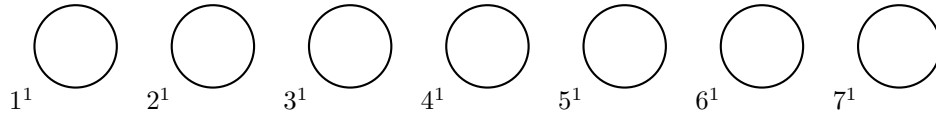
$$\mathfrak{M}^{dot} \xrightarrow{\Gamma} \mathfrak{M}^{flat} \quad (4.63)$$

$$\langle \mathbb{N}; \mathcal{R}^{dot}; \{pred(x), succ(x)\} \rangle \xrightarrow{\Gamma} \langle \mathbb{N}; \mathcal{R}^{flat}; \{pred(x), succ(x)\} \rangle$$

Copy Set(s)

The Flat Model for a given word always has an equal number or fewer positions than the Dot Model for the same word, so this transduction will require only one copy set.

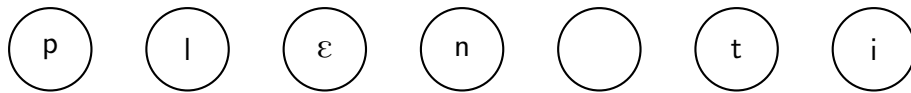
Figure 4.29: The codomain for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$



Unary Relations

Only the unary relations encoding segmental features are preserved. Note that position 5¹ does not belong to any of these relations because it is labeled with a dot in the input; this label is not available in the output model signature.

Figure 4.30: Some unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$



Additional formulas are needed for the syllable constituent labels **ons**, **nuc**, and **cod**. Unlike the Tree and Flat Models, the Dot Model provides no information about syllable-internal structure. Until this point, we have not questioned *how* segments are matched

to syllable constituents in the first place because this information was given in the word models themselves. This need not be the case, however, as there is more to a language’s phonology than just the models used to represent words.

I set aside the issue of how to syllabify underlying strings until Chapter 6. For now, it suffices to assume that every language has some principles in place to syllabify underlying strings. Minimally, the nucleus of each syllable must be identified by some combination of universal and/or language-specific principles. Then onset and coda membership can be inferred from position relative to syllable boundaries and nuclei (i.e., onset consonants are between a dot its successor labeled **nuc**, while coda consonants are between a nucleus and its successor labeled with a dot).

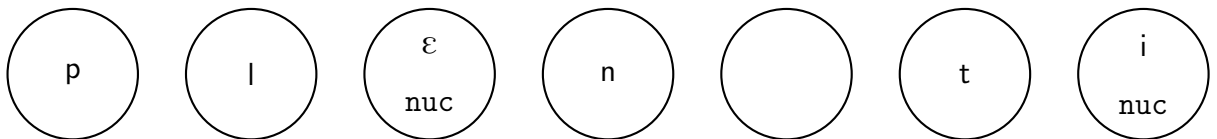
This being the case, I assume there is some relation $\text{Eng_nuc}(x)$ in the input word model that is true of syllabic nuclei in English. Then output positions will be labeled **nuc** if they satisfy $\text{Eng_nuc}(x)$.

$$\text{nuc}(x^1) \stackrel{\text{def}}{=} \text{Eng_nuc}(x) \tag{4.64}$$

“Position x in Copy Set 1 is labeled **nuc** iff x satisfies **Eng_nuc** in the input.”

These labeling relations yield the following partial output.

Figure 4.31: Some unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$



To label onset and coda segments, I again make use of formulas called **ons_1**, **cod_1**,

ons_2, cod_2, etc. Note that these differ from those defined in §4.3.2. In particular, the formulas below do not refer to **ons** and **Eng_cod** labels in the input because there are none. Instead, syllable margin membership is inferred from the absence of **nuc** or **•** labels and the ordering with respect to nucleus nodes.

$$\text{ons}_1(x) \stackrel{\text{def}}{=} \neg(\text{Eng_nuc}(x) \vee \bullet(x)) \wedge \text{Eng_nuc}(\text{succ}(x)) \quad (4.65)$$

$$\text{cod}_1(x) \stackrel{\text{def}}{=} \neg(\text{Eng_nuc}(x) \vee \bullet(x)) \wedge \text{Eng_nuc}(\text{pred}(x)) \quad (4.66)$$

“Position x is a nucleus-adjacent onset/coda iff x is not labeled **Eng_nuc** or **•**, and x immediately follows/precedes a position labeled **Eng_nuc**.”

Similarly, ons_i and cod_i for the general case are defined below. Again I take n to be the maximum number of segments allowed in a single onset and m to be the maximum number of segments allowed in a single coda.

$$\text{ons}_i(x) \stackrel{\text{def}}{=} \neg(\text{Eng_nuc}(x) \vee \bullet(x)) \wedge \text{ons}_{(i-1)}(\text{succ}(x)) \text{ for } i \in \{2, \dots, n\} \quad (4.67)$$

“Position x is ‘onset i ’ iff x is not labeled **Eng_nuc** or **•**, and its successor is ‘onset $(i-1)$ ’, for i ranging from 2 to n .”

$$\text{cod}_i(x) \stackrel{\text{def}}{=} \neg(\text{Eng_nuc}(x) \vee \bullet(x)) \wedge \text{cod}_{(i-1)}(\text{pred}(x)) \text{ for } i \in \{2, \dots, m\} \quad (4.68)$$

“Position x is ‘coda i ’ iff x is not labeled **Eng_nuc** or **•**, and its predecessor is ‘coda $(i-1)$ ’, for i ranging from 2 to m .”

It is now simple to define labeling formulas for **ons** and **cod** in the codomain.

$$\text{ons}(x^1) \stackrel{\text{def}}{=} \text{ons}_n(x) \vee \text{ons}_{(n-1)}(x) \vee \dots \vee \text{ons}_1(x) \quad (4.69)$$

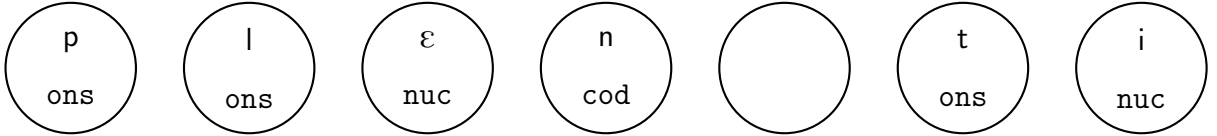
“Position x in Copy Set 1 is labeled **ons** iff x belongs to a contiguous string of segments (up to length n) in the input that are not labeled **Eng_nuc** or **•**, ending with the nucleus-adjacent onset (**ons.1**).”

$$\text{cod}(x^1) \stackrel{\text{def}}{=} \text{cod_1}(x) \vee \dots \vee \text{cod_}(m-1)(x) \vee \text{cod_}m(x) \quad (4.70)$$

“Position x in Copy Set 1 is labeled **cod** iff x belongs to a contiguous string of segments (up to length m) in the input that are not labeled **Eng_nuc** or **•**, beginning with the nucleus-adjacent coda (**cod_1**).”

These contributions to the output are illustrated in Figure 4.32.

Figure 4.32: Additional unary relations for $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$



Successor and Predecessor functions

The only caveat to the successor information in this transduction is that the position after a dot in the input must succeed the position before the dot, essentially skipping the dot position. I will use another type of a pre/post-boundary formula here.

$$\text{pre_dot}(x) \stackrel{\text{def}}{=} \bullet(\text{succ}(x)) \quad (4.71)$$

$$\text{post_dot}(x) \stackrel{\text{def}}{=} \bullet(\text{pred}(x)) \quad (4.72)$$

$$(4.73)$$

“Position x is a pre-dot/post-dot position iff its successor/predecessor is labeled with a dot.”

The output successor function for pre-dot positions is given in (4.74).

$$\text{pre_dot}(x) \Rightarrow \text{succ}(x^1) = (\text{succ}(\text{succ}(x)))^1 \quad (4.74)$$

“If x is a pre-dot position in the input, then the successor of the first copy of x is the first copy of the second successor of x .”

Conversely, the output predecessor function for post-dot positions is given in (4.75).

$$\text{post_dot}(x) \Rightarrow \text{pred}(x^1) = (\text{pred}(\text{pred}(x)))^1 \quad (4.75)$$

“If x is a post-dot position in the input, then the predecessor of the first copy of x is the first copy of the second predecessor of x .”

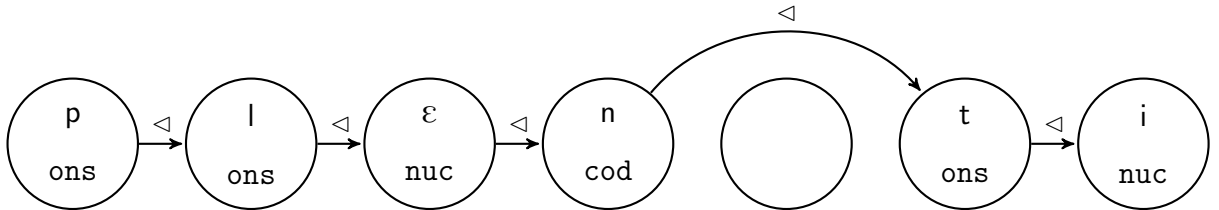
The remaining successors and predecessors are preserved from the input. Full definitions for output successor and predecessor functions are given in (4.76–4.77).

$$\text{succ}(x^1) \stackrel{\text{def}}{=} \begin{cases} (\text{succ}(\text{succ}(x)))^1 & \Leftrightarrow \text{pre_dot}(x) \\ (\text{succ}(x))^1 & \Leftrightarrow \neg \text{pre_dot}(x) \end{cases} \quad (4.76)$$

$$\text{pred}(x^1) \stackrel{\text{def}}{=} \begin{cases} (\text{pred}(\text{pred}(x)))^1 & \Leftrightarrow \text{post_dot}(x) \\ (\text{pred}(x))^1 & \Leftrightarrow \neg \text{post_dot}(x) \end{cases} \quad (4.77)$$

All output successor information is illustrated in Figure 4.33. Again I omit the predecessor information for brevity, as it is largely redundant.

Figure 4.33: $\Gamma_{df}(\mathcal{M}_{plenty}^{dot})$ fully specified

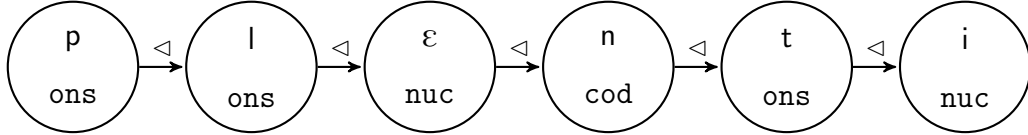


Summary of The Dot-to-Flat Transduction and Licensing

$$\begin{aligned}
 \mathbf{f}(x^1) &\stackrel{\text{def}}{=} \mathbf{f}(x) \text{ for } \mathbf{f} \in \mathcal{F} \\
 \text{nuc}(x^1) &\stackrel{\text{def}}{=} \text{Eng_nuc}(x) \\
 \text{ons}(x^1) &\stackrel{\text{def}}{=} \text{ons}_n(x) \vee \text{ons}_{(n-1)}(x) \vee \dots \vee \text{ons}_1(x) \\
 \text{cod}(x^1) &\stackrel{\text{def}}{=} \text{cod}_1(x) \vee \dots \vee \text{cod}_{(m-1)}(x) \vee \text{cod}_m(x) \\
 \text{succ}(x^1) &\stackrel{\text{def}}{=} \begin{cases} (\text{succ}(\text{succ}(x)))^1 & \Leftrightarrow \text{pre_dot}(x) \\ (\text{succ}(x))^1 & \Leftrightarrow \neg \text{pre_dot}(x) \end{cases} \\
 \text{pred}(x^1) &\stackrel{\text{def}}{=} \begin{cases} (\text{pred}(\text{pred}(x)))^1 & \Leftrightarrow \text{post_dot}(x) \\ (\text{pred}(x))^1 & \Leftrightarrow \neg \text{post_dot}(x) \end{cases} \\
 \text{license}(x) &\stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{\text{flat}})[R(x)]
 \end{aligned}$$

Figure 4.34 shows the licensed output of this transduction.

Figure 4.34: The licensed output of $\Gamma_{df}(\mathcal{M}_{plenty}^{\text{dot}})$



4.4 Discussion

I have shown that three popular types of syllable structure representations are notationally equivalent, in a strict mathematical sense. Model theory addresses the question of whether some representations are more expressive than others. In particular, I have demonstrated the QF-bi-interpretability of the Tree and Flat Models, and that of the Flat and Dot Models. But are the Tree and Dot Models also QF-bi-interpretable?

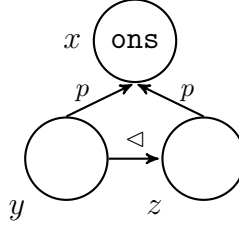
Simply put: yes. While the result has yet to be proven, QF transductions are likely closed under composition. This means that if Γ_{df} and Γ_{ft} are both QF, so is their composition, $\Gamma_{df}(\Gamma_{ft})$. Notice that this composition takes a Dot Model as input and produces the corresponding Tree Model—this is exactly the Dot-to-Tree transduction, Γ_{dt} . Similarly, because Γ_{tf} and Γ_{fd} are QF, then so their composition $\Gamma_{tf}(\Gamma_{fd})$ is likely also QF—and this is the Tree-to-Dot transduction. Thus, each pair of structures examined is QF-bi-interpretable.

Even if it turns out that QF transductions are not closed under composition, defining the remaining transductions in QF logic is trivial given the formulas previously defined in this chapter. For example, Γ_{dt} is essentially the same as Γ_{ft} , except that the syllable constituent labels must be deduced—but I have already shown in Γ_{df} that these labels *can* be deduced from information in the Dot Model using only QF formulas.

While there may be subjective advantages to using different representation types, such as highlighting certain aspects of the structure, the fact remains that they are all equally capable of encoding the same information. Furthermore, they are *not* capable of making different predictions about syllable typology. Assuming a bound on syllable size, any constraint expressed in one model can be readily translated into a constraint *with the exact same effects* in either of the other models.

For example, suppose we want to ban complex onsets. Let us first consider what a complex onset looks like in the Tree Model: two adjacent positions both dominated by a single position labeled **ons**. Figure 4.35 illustrates just that.

Figure 4.35: \mathcal{M}_{CO}^{tree} , a complex onset in the Tree Model



Recall from Chapter 3 that a *substructure* is an Existentially Quantified Conjunction (EQC) and a *substructure constraint* is a conjunction of one or more positive and/or negative EQCs. A constraint against complex onsets in the Tree Model must simply ban the substructure illustrated in Figure 4.35. This constraint is denoted κ_{NCO} , formally defined in (4.78).

$$\kappa_{NCO} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{par}(y) = x \wedge \text{par}(z) = x \wedge \text{succ}(y) = z] \quad (4.78)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **ons**, the parent of y is x , the parent of z is also x , and the successor of y is z .”

So how do we translate this constraint into an equivalent one that uses the Flat Model? We simply apply the transduction Γ_{tf} to the model \mathcal{M}_{CO}^{tree} . The relevant formulas from Γ_{tf} are reproduced below in (4.79–4.80).

$$\text{ons}(x^1) \stackrel{\text{def}}{=} \text{ons}(\text{par}(x)) \quad (4.79)$$

$$\text{succ}(x^1) \stackrel{\text{def}}{=} (\text{succ}(x))^1 \quad (4.80)$$

Because x is labeled **ons** in the input, its children will be labeled **ons** in the output, as formalized in (4.81–4.82).

$$\mathbf{ons}(x) \wedge x = \mathit{par}(y) \Rightarrow \mathbf{ons}(y^1) \quad (4.81)$$

$$\mathbf{ons}(x) \wedge x = \mathit{par}(z) \Rightarrow \mathbf{ons}(z^1) \quad (4.82)$$

“If x is labeled \mathbf{ons} and x is the parent of y/z in the input, then the first copy of y/z is labeled \mathbf{ons} in the output.”

Additionally, the successor relation is preserved as in (4.83).

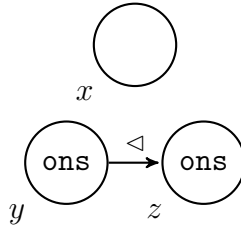
$$\mathit{succ}(y^1) \stackrel{\text{def}}{=} (\mathit{succ}(y))^1 \quad (4.83)$$

$$= z^1 \quad (4.84)$$

“The successor of the first copy of y is the first copy of the successor of y in the input. This is equal to the first copy of z .”

The resulting output is illustrated in Figure 4.36.

Figure 4.36: The output of $\Gamma_{tf}(\mathcal{M}_{CO}^{tree})$



Note that x is an unlabeled position and can therefore be ‘erased’ by the licensing function, as demonstrated previously. Considering the licensed output (positions y and z only), it is clearly a licit structure in the Flat Model Theory. Furthermore, it is exactly the structure we want to ban if complex onsets are disallowed. This example

shows how the transductions developed in this chapter can be applied not only to words, but to substructure constraints as well.

In the remaining chapters, I use the Tree Representation because it is the prevailing representation type used in the literature on Imdlawn Tashlhiyt Berber (ITB) and Moroccan Arabic (MA).

Chapter 5

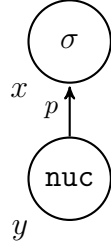
UNIVERSAL PRINCIPLES, SONORITY SEQUENCING, AND CV TYPOLOGY

Regardless of which representation of syllable structure one adopts, certain structural requirements and proclivities are generally accepted. For example, every syllable must have exactly one nucleus, and languages around the world tend to favor vowels in nucleic position over consonants. In this chapter, I first formalize some of the universal principles of syllable well-formedness as substructure constraints. Then I examine two generalizations that capture robust (although not exception-less) cross-linguistic tendencies: sonority sequencing and CV typology.

5.1 Structural Well-formedness Constraints

A variety of structural well-formedness constraints can be expressed as substructure constraints. Consider the requirement that every syllable must have exactly one nucleus. This can be captured with two substructure constraints. First, every syllable must have a sigma node dominating a nucleus node, the substructure illustrated in Figure 5.1. I call this constraint κ_{NR} for “Nucleus Required,” formally defined in (5.1).

Figure 5.1: ψ_{NR} , the substructure required by κ_{NR} “Nucleus Required”

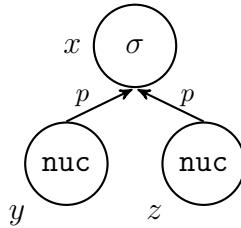


$$\begin{aligned} \kappa_{NR} &\stackrel{\text{def}}{=} (\exists x, y)[\sigma(x) \wedge \text{nuc}(y) \wedge \text{par}(y) = x] \\ &= \psi_{NR} \end{aligned} \tag{5.1}$$

“There exists a pair of nodes x, y , such that x is labeled σ , y is labeled nuc , and the parent of y is x . ψ_{NR} is exactly this structure.”

Second, we must ban a single sigma node that dominates two distinct nucleus nodes. This constraint, illustrated in Figure 5.2, is called κ_{NU} for “Nucleus Unique.” κ_{NU} is formally defined in (5.2).

Figure 5.2: ψ_{NU} , the substructure banned by κ_{NU} “Nucleus Unique”

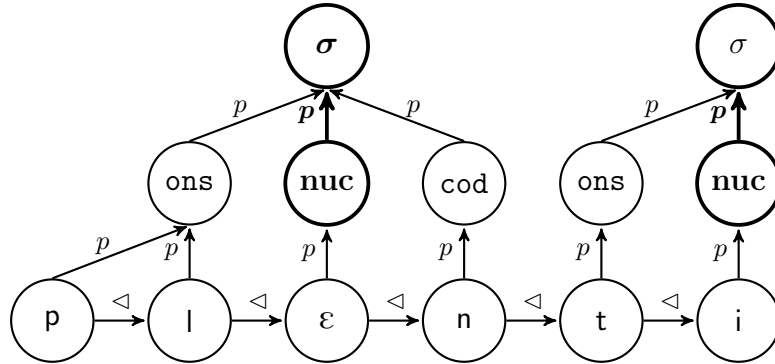


$$\begin{aligned} \kappa_{NU} &\stackrel{\text{def}}{=} \neg(\exists x, y, z)[\sigma(x) \wedge \mathbf{nuc}(y) \wedge \mathbf{nuc}(z) \wedge \mathit{par}(y) = x \wedge \mathit{par}(z) = x \wedge y \neq z] \\ &= \neg\psi_{NU} \end{aligned} \tag{5.2}$$

“There does not exist a set of three nodes x, y, z , such that x is labeled σ , y and z are both labeled \mathbf{nuc} , the parent of y is x , the parent of z is also x , and y is not equal to z . This is equivalent to $\neg\psi_{NU}$.”

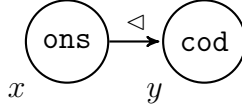
A model \mathcal{M} satisfies both of the above constraints iff it contains ψ_{NR} and does **not** contain ψ_{NU} . One model satisfying these constraints is $\mathcal{M}_{plenty}^{tree}$, reproduced in Figure 5.3 below. The substructures equivalent to ψ_{NR} are bolded. Only one is necessary to satisfy κ_{NR} . No substructure of $\mathcal{M}_{plenty}^{tree}$ includes a σ node dominating two unique nucleus nodes, so κ_{NU} is also satisfied. To express that the model satisfies both constraints, I write $\mathcal{M}_{plenty}^{tree} \models \kappa_{NR} \wedge \kappa_{NU}$.

Figure 5.3: $\mathcal{M}_{plenty}^{tree}$



Additional structural well-formedness constraints can be expressed similarly. For example, an onset may not immediately precede a coda. This substructure constraint is represented by κ_{OC} , which is illustrated in Figure 5.4 and defined in (5.3).

Figure 5.4: ψ_{OC} , the substructure banned by κ_{OC} “No Onset-Coda”



$$\begin{aligned} \kappa_{OC} &\stackrel{\text{def}}{=} \neg(\exists x, y)[\text{ons}(x) \wedge \text{cod}(y) \wedge x \triangleleft y] \\ &= \neg\psi_{OC} \end{aligned} \tag{5.3}$$

“There does not exist a pair nodes x, y , such that x is labeled **ons**, y is labeled **cod**, and the successor of x is y . This is equivalent to $\neg\psi_{OC}$ ”

Note that all of these constraints refer to subgraphs of size 3 or less. While some are positive and some are negative, well-formedness with respect to each constraint is always evaluated locally.

5.2 Sonority Sequencing

[Sievers \(1881\)](#) wrote one of the first versions of the Sonority Sequencing Principle (SSP) (also called the Sonority Sequencing Generalization or SSG), observing that sonority generally rises between a given segment and the sonority peak of its syllable. This principle has been reiterated in one form or another by many influential linguists (e.g., [Clements, 1990](#); [Hooper, 1976](#); [Saussure, 1916](#); [Selkirk, 1984](#)). While there is a great deal of disagreement regarding the psychological and physiological realizations of sonority, some principles are generally agreed upon. Namely, vowels are more sonorous than sonorant consonants (glides, liquids, and nasals), which are in turn more sonorous than obstruents (fricatives, affricates, and stops). Finer distinctions between members of these three natural classes are more contentious, but the exact details of the sonority hierarchy do not matter for the purposes of this chapter. It is also worth noting that

many languages allow SSP violations, so this is not a universal principle in its most strict interpretation.

In the following section, I define binary sonority relations that encode the relative sonority of two segments. I then introduce two substructure constraints that enforce a certain version of the SSP.

5.2.1 Sonority Relations

I assume for now that there is some known sonority hierarchy (either for a given language or more universally) such that, for every pair of phonemes, it is known whether or not one is less sonorous than the other. If segment x is less sonorous than segment y , I write $<_s(x, y)$ or, equivalently, $x <_s y$. In accordance with the traditional notion of lesser sonority, I assume that the binary relation $<_s$ is *irreflexive*, *asymmetric*, and *transitive*. The formal definitions of these terms are given in (1-3), along with examples. Before delving into those, let us clarify some notational conventions. Positions corresponding to a particular phoneme are denoted with square brackets, e.g., [t] denotes a position labeled `alv` and `stop`.

$$[t] \stackrel{\text{def}}{=} x \text{ s.t. } \text{alv}(x) \wedge \text{stop}(x)$$

“[t] is defined as a position x such that `alv`(x) = TRUE and `stop`(x) = TRUE.”

The symbol \Rightarrow represents implication in one direction. For example, the statement

$$2 < x \Rightarrow 1 < x$$

can be read as “2 is less than x implies that 1 is less than x ,” or, equivalently, “if 2 is less than x , then 1 is less than x .”¹

¹ Note that implication in the other direction is not necessarily true; e.g., $1 < x$ does not imply that $2 < x$. Let $x = 1.5$, for instance. Then $1 < x$ and $x < 2$.

With these conventions established, the properties *irreflexive*, *asymmetric*, and *transitive* are defined below.

1. The binary relation $<_s$ is *irreflexive* iff:

$$(\forall x)[<_s(x, x) = \text{FALSE}]$$

“For all x , x cannot be less sonorous than x .”

Example: $<_s([t], [t]) = \text{FALSE}$ because $[t]$ cannot be less sonorous than itself.

2. The binary relation $<_s$ is *asymmetric* iff:

$$(\forall x, y)[<_s(x, y) = \text{TRUE} \Rightarrow <_s(y, x) = \text{FALSE}]$$

“For all x, y , if x is less sonorous than y , then y cannot be less sonorous than x .”

Example: $<_s([t], [m]) = \text{TRUE} \Rightarrow <_s([m], [t]) = \text{FALSE}$

That is, $[t]$ is less sonorous than $[m]$, so $[m]$ cannot be less sonorous than $[t]$.

3. The binary relation $<_s$ is *transitive* iff:

$$(\forall x, y, z)[<_s(x, y) = \text{TRUE} \wedge <_s(y, z) = \text{TRUE} \Rightarrow <_s(x, z) = \text{TRUE}]$$

“For all x, y, z , if x is less sonorous than y and y is less sonorous than z , then x must be less sonorous than z .”

Example: $<_s([t], [m]) \wedge <_s([m], [a]) \Rightarrow <_s([t], [a])$

That is, $[t]$ is less sonorous than $[m]$ and $[m]$ is less sonorous than $[a]$, so $[t]$ is also less sonorous than $[a]$.

Given these properties of $<_s$, it is simple to define a relation $=_s$ to represent equal sonority. Recall that $\neg R(x, y)$ is synonymous with $R(x, y) = \text{FALSE}$. Moving forward, I will use the former (more condensed) notation as it aids in readability.

$$=_s(x, y) \stackrel{\text{def}}{=} \neg <_s(x, y) \wedge \neg <_s(y, x) \quad (5.4)$$

“ x and y are equally sonorous iff x is not less sonorous than y and y is not less sonorous than x .”

Similarly, I define the relation \leq_s to represent equal or lesser sonority.

$$\leq_s(x, y) \stackrel{\text{def}}{=} <_s(x, y) \vee =_s(y, x) \quad (5.5)$$

“ x is equally or less sonorous than y iff x is less sonorous than y or x and y are equally sonorous.”

Finally, I also define the relation $>_s$ to represent greater sonority.

$$>_s(x, y) \stackrel{\text{def}}{=} \neg \leq_s(x, y) \quad (5.6)$$

“ x is more sonorous than y iff x is not equally or less sonorous than y .”

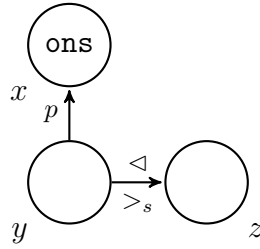
5.2.2 Constraints on Sonority Sequencing

As will be seen in Chapter 6, some languages allow equally sonorous consonants in a complex onset/coda, meaning that a strict rise in sonority from the syllable boundaries to the nucleus is not required. Instead, an interpretation of the SSP that is more compatible with these languages would forbid falling sonority from the syllable boundaries to the nucleus.

To enforce this version of the SSP, it suffices to ban two substructures: first, an onset segment must not be more sonorous than its successor; second, a coda segment must not be more sonorous than its predecessor.

The constraint κ_{right} “Right of Onset” bans a segment dominated by an **ons** node whose successor is more sonorous, the substructure illustrated in Figure 5.5. A formal definition of κ_{right} is given in (5.7).

Figure 5.5: ψ_{right} , the substructure banned by κ_{right} “Right of Onset”



$$\begin{aligned} \kappa_{right} &\stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{par}(y) = x \wedge \text{succ}(y) = z \wedge <_s(z, y)] \\ &= \neg\psi_{right} \end{aligned} \tag{5.7}$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **ons** and is the parent of y , the successor of y is z , and y is more sonorous than z . This is equivalent to $\neg\psi_{right}$.”

Making use of the notion of substructure allows this single constraint to apply to a variety of cases. For one, it forbids an onset from being more sonorous than the nucleus immediately following it. Figure 5.6 illustrates such a structure, an onset w with a nucleic s . The banned substructure ψ_{right} is bolded.

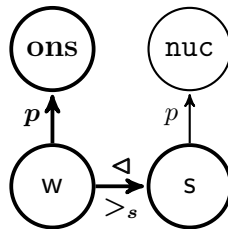
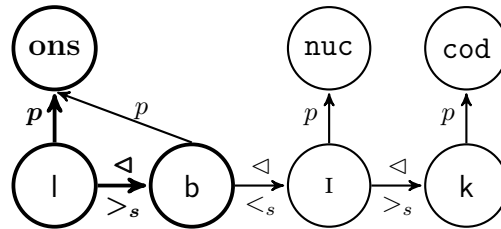


Figure 5.6: An onset followed by a nucleus of lesser sonority

Importantly, κ_{right} also bans a left-to-right fall in sonority among segments within a complex onset. Consider the nonce word *blɪk* [blɪk], which native English-speakers

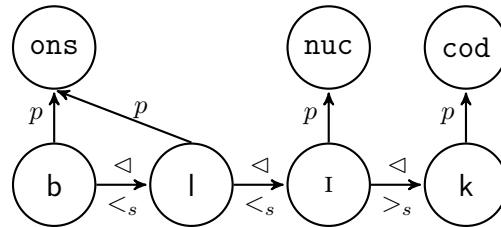
generally agree is a viable (though unattested) word in English (Chomsky & Halle, 1965). Reversing the order of the onset segments yields *lbick* [lbɪk], which is clearly not acceptable as an English word. One explanation for this is that *blick* satisfies the SSP while *lbick* does not. Figure 5.7 illustrates the tree model for \mathcal{M}_{lbik} . The bolded substructure is precisely the structure banned by κ_{right} .

Figure 5.7: \mathcal{M}_{lbik}



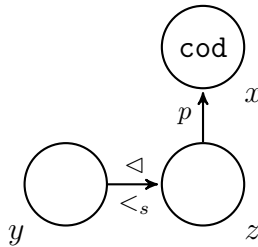
In contrast, Figure 5.8 illustrates the tree word model for *blick*. \mathcal{M}_{blik} satisfies κ_{right} because it does not contain the banned substructure. Crucially, the first segment in *blick*, [b], is less sonorous than its successor, [l].

Figure 5.8: \mathcal{M}_{blik}



Of course, the SSP does not only concern onsets. Another substructure constraint is necessary to ban a left-to-right sonority rise in complex codas and from the nucleus to the coda: κ_{left} . This constraint is defined in (5.8) and the corresponding banned substructure is illustrated in Figure 5.9.

Figure 5.9: ψ_{left} , the substructure banned by κ_{left} “Left of Coda”



$$\kappa_{left} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{cod}(x) \wedge \text{par}(z) = x \wedge \text{succ}(y) = z \wedge <_s(y, z)] \quad (5.8)$$

“There does not exist a set of three nodes, x, y, z , such that x is labeled cod and is the parent of z , the successor of y is z , and y is less sonorous than z .”

This constraint is the mirror image of κ_{right} . Conjoining the two constraints yields κ_{SSP} , a statement of the SSP in formal logic.

$$\begin{aligned} \kappa_{SSP} &\stackrel{\text{def}}{=} \kappa_{right} \wedge \kappa_{left} \\ &= \neg\psi_{right} \wedge \neg\psi_{left} \end{aligned} \quad (5.9)$$

A word model satisfies κ_{SSP} iff it contains neither ψ_{right} nor ψ_{left} .

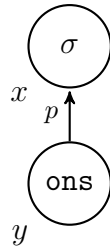
5.3 CV Typology

The key insight from previous accounts of CV phonology (e.g., [Clements & Keyser, 1983](#); [Jakobson, 1962](#)) is that, while some languages have obligatory onsets and some forbid codas, virtually no languages ban onsets or require codas. Furthermore, intervocalic consonants are thought to be universally syllabified as onsets to the following

syllable. That is, a /VCV/ sequence is syllabified as [V.CV] rather than [VC.V].²

These principles are easily describable with two substructure constraints: κ_{OR} , which requires a syllable to have an onset; and κ_{CF} , which bans a syllable from having a coda. Figure 5.10 illustrates the first of these, κ_{OR} , which is defined in (5.10). Its form is identical to that of κ_{NR} , the positive substructure constraint requiring that every syllable have a nucleus.

Figure 5.10: ψ_{OR} , the substructure required by κ_{OR} “Onset Required”



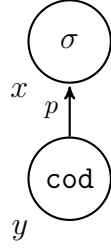
$$\begin{aligned} \kappa_{OR} &\stackrel{\text{def}}{=} (\exists x, y)[\sigma(x) \wedge \text{ons}(y) \wedge \text{par}(y) = x] \\ &= \psi_{OR} \end{aligned} \tag{5.10}$$

“There exists a pair of nodes x, y , such that x is labeled σ , y is labeled ons , and the parent of y is x . This is equivalent to ψ_{OR} .”

In contrast, κ_{CF} is a negative substructure constraint similar to κ_{NU} (which bans a syllable from having multiple nuclei). κ_{CF} is illustrated in Figure 5.11 and defined in (5.11).

² It has been claimed that Arrernte (Breen & Pensalfini, 1999), Barra Gaelic (Borgström, 1935), and Kunjen (Sommer, 1981) lack this onset preference; see Clements (1986), Blevins (1995), and Duanmu (2009) for discussion and counter-arguments.

Figure 5.11: ψ_{CF} , the substructure banned by κ_{CF} “Coda Forbidden”



$$\begin{aligned} \kappa_{CF} &\stackrel{\text{def}}{=} \neg(\exists x, y)[\sigma(x) \wedge \text{cod}(y) \wedge \text{par}(y) = x] \\ &= \neg\psi_{CF} \end{aligned} \tag{5.11}$$

“There does not exist a pair of nodes x, y , such that x is labeled σ , y is labeled cod , and the parent of y is x . This is equivalent to $\neg\psi_{CF}$.”

A given language may have one, both, or neither of these constraints in its phonology. Recall that we can define a formal language L using a logical formula that is the conjunction of all substructure constraints in the referenced natural language. Call such a formula K . If K includes κ_{OR} as one of its terms, $L(K)$ will only include words where every syllable has an onset. Further, if K includes κ_{CF} , it will not include any words with codas in them.

The logically possible combinations of κ_{OR} and κ_{CF} yield a four-way typology of basic language types (as in [Blevins, 1995](#)). If a language requires onsets and forbids codas, the only acceptable syllable type will be CV. If codas are forbidden and onsets not required, V and CV syllables are allowed. Requiring onsets and allowing codas yields CV and CVC syllables. Finally, when neither constraint is present, V, CV, VC, and CVC syllables will all be grammatical. These four language types are summarized in [Table 5.1](#).

Table 5.1: Possible Syllable Types in Basic CV Typology

	κ_{OR}	No κ_{OR}
κ_{CF}	CV	(C)V
No κ_{CF}	CV(C)	(C)V(C)

Note that the constraints defined here differ from Optimality-Theoretic constraints like ONSET and NOCODA in that they are **inviolable** and **non-universal**. The formal languages defined by combinations of κ_{OR} and κ_{CF} may be extensionally identical to those produced from a certain ranking of violable constraints. What I have demonstrated is that the four-way typological distinction can easily be derived *without* optimization; inviolable constraints work at least as well as violable ones in this regard.

Chapter 6

CASE STUDY: IMDLAWN TASHLHIYT BERBER

Imdlawn Tashlhiyt Berber (ITB) is unusual due to its tolerance of non-vocalic syllabic nuclei. Rule-based and Optimality-Theoretic (OT) accounts of ITB syllabification have been successful, but they do not directly address the question of how complex the process is or what type of formal language might capture the well-formedness conditions of ITB syllables. Model theory and formal logic allow for comparison of complexity across different theories of phonology by identifying the computational power (or expressivity) of linguistic formalisms in a grammar-independent way. With these tools, I develop mathematical formalisms for characterizing well-formedness in ITB and for representing ITB syllabification using QF logic. This result indicates that ITB syllabification is relatively simple from a computational standpoint and that grammatical formalisms could succeed with even less powerful mechanisms than are currently accepted (e.g., optimization).¹

6.1 Motivation

Accounting for syllabification in ITB has become a sort of litmus test for phonological frameworks handling syllable theory (Ridouane, 2016). Any segment can be nucleic in ITB in some environment, making words like [tɾ.gɫt] ‘you (sg.) locked’ commonplace. Notably, [tɾ.gɫt] contrasts with [tɾg.las] ‘she locked it (masc.),’ as shown in Table 6.1.²

¹ Part of this analysis appears in the Proceedings of the First Annual Meeting of the Society for Computation in Linguistics (SCiL)—see Strother-Garcia (2018).

² sg. = singular; masc. = masculine; 2ms = 2nd person masculine singular; 3fs = 3rd person feminine singular.

These examples illustrate a great deal of variety in syllabic nuclei, ranging from vowels like [a] to voiceless fricatives like [s].

Table 6.1: Alternations in ITB Perfective Verb Forms: 2ms vs. 3fs with a dative masc. object

2ms	Gloss	3fs	Gloss
[tɾ.gɫt]	‘you locked’	[tɾg.las]	‘she locked it (masc.)’
[tʂ.kɾt]	‘you did’	[tʂk.ras]	‘she did it (masc.)’

Table 6.2 shows additional alternations in perfective verb forms.³ Note the difference in the initial syllables here: masculine forms begin with [i], which forms an onset-less syllable along with a single coda consonant; feminine forms begin with [t], which serves as the onset to a syllable with a nucleic consonant.

Table 6.2: Alternations in ITB Perfective Verb Forms: 3ms vs. 3fs

3ms	Gloss	3fs	Gloss
[il.di]	‘he pulled’	[tɫ.di]	‘she pulled’
[is.ti]	‘he selected’	[tʂ.ti]	‘she selected’

Despite the unusual syllables in these words, careful study shows that syllable well-formedness depends on only a few key principles. ITB syllables follow the SSP almost perfectly, with a well-defined, language-specific class of exceptions (to be discussed further in §6.3.2). Note that the syllabic consonants in the examples above are always more sonorous than their neighboring segments.

³ 3ms = 3rd person masculine singular.

As with other phonological processes, syllabification can be thought of as a map from URs to SRs—but what is the nature of this map? Previous approaches have formalized ITB syllabification using rules (Dell & Elmedlaoui, 1985, 1988; Frampton, 2011) or rankings of OT constraints (Clements, 1997; Dell & Elmedlaoui, 2002; Prince & Smolensky, 1993). While both of these frameworks can accurately represent the syllabification process in ITB, previous research suggests they are, in a sense, more powerful formalisms than are necessary.

As explained in Chapter 3, one way to compare the power of different grammatical formalisms is to examine the kind of logic needed to express them. Under certain assumptions, rule-based and OT grammars are both demonstrably as expressive as regular relations (Frank & Satta, 1998; Johnson, 1972; Kaplan & Kay, 1994; Karttunen, 1993, 1998).⁴ Regular relations properly include regular functions (maps), which are equivalent to transductions in MSO logic (Engelfriet & Hoogeboom, 2001; Rogers & Pullum, 2011b). In contrast, I show that the transduction from URs to SRs in ITB is QF, which is strictly weaker than FO, which is in turn strictly weaker than MSO.

Recall from Chapter 3 that QF formulas are evaluated with respect to a substructure of finite size; that is, they are evaluated *locally*. Restricting the mapping to only local computations substantially constrains the predicted typology of phonological patterns and admits established learnability results (e.g., Chandlee et al., 2019; Heinz, 2010a,b; Jardine, 2016; Strother-Garcia et al., 2017). A class of mappings known as *Input Strictly Local Functions*, which rely on only local computations, is already known to characterize a variety of phonological processes (Chandlee, 2014; Chandlee & Heinz, 2018; Chandlee & Lindell, to appear), raising the question of whether all phonological mappings can be shown to be local in nature.

In the following sections, I first review the basic facts of ITB phonology relevant to the surface pattern of syllabification. I then develop a set of local substructure constraints

⁴ See Heinz (2011b) for a review of the literature on computational complexity in rule-based grammars and OT grammars.

to formalize the pattern mathematically. These constraints define the formal language corresponding to all well-formed words in ITB (with respect to syllable structure). Finally, I demonstrate that ITB syllabification can be represented as a QF graph transduction.

6.2 The Basics

The phonemic consonant inventory for ITB is given in Table 6.3.⁵ Notably, the alveolars and post-alveolars have pharyngealized counterparts, while the velars and uvulars have labialized counterparts. All consonants also contrast with their geminate forms (not represented in the table)—e.g., *imi* ‘mouth’ vs. *immi* ‘Mom.’

Table 6.3: Phonemic Singleton Consonants in ITB

	Labial	Alveolar	Palato-alveolar	Velar	Uvular	Pharyngeal	Glottal
Stop	b	t t ^ʕ d d ^ʕ		k k ^w g g ^w	q q ^w		
Fricative	f	s s ^ʕ z z ^ʕ	ʃ ʃ ^ʕ ʒ ʒ ^ʕ		χ χ ^w ʁ ʁ ^w	ħ ʕ	ʕ
Nasal	m	n n ^ʕ					
Approximant			r r ^ʕ				
Lateral		l l ^ʕ					

If a word contains an underlying pharyngeal (also called an *emphatic* consonant), pharyngealization spreads rightward throughout the word. This yields minimal pairs like

⁵ Where Dell & Elmedlaoui (2002) use nonstandard symbols, I present their IPA equivalents.

those in Table 6.4.⁶

Table 6.4: Pharyngealized Consonants in ITB

SR	Gloss	SR	Gloss
[i.zi]	‘fly’	[!i.zi]	‘gall’
[fɾd]	‘graze!’	[!fɾd]	‘clear!’

There are only three phonemic vowels: /a/, /i/, and /u/. In general, the glide/vowel distinction among the High Vocoids (HVs) [i~j] and [u~w] is predictable based on syllable position. That is, a nucleic HV is vocalic, as in the second syllable of [tag.rurt] ‘stable,’ while a non-nucleic HV surfaces as a glide, as in [sa.wɫx] ‘I spoke.’ However, there are some glides which must be specified in the UR (Dell & Elmedlaoui, 2002)—see Table 6.5.

Table 6.5: Unpredictable Vowel/Glide Contrasts in ITB

SR	Gloss	SR	Gloss
[suj]	‘let pass!’	[zwi]	‘beat down!’
[lur]	‘give back!’	[lwɾ]	‘run away!’

These can be treated as lexical exceptions by adding a labeling relation to the input word model. I will set these aside for the remainder of my analysis of ITB, but my analysis of MA in the following chapter shows that lexical exceptions do not pose a problem for inviolable substructure constraint grammars.

⁶ The ! symbol is used in SRs to indicate that pharyngealization spreads rightward from the underlying pharyngealized consonant, as in Dell & Elmedlaoui (2002).

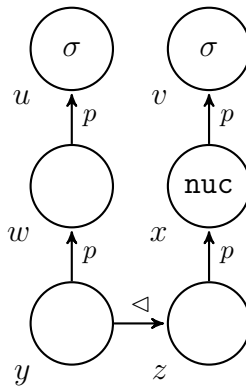
6.3 Surface Well-Formedness in ITB

In this section, I outline the inviolable constraints that govern surface well-formedness in ITB and construct the corresponding formal language.

6.3.1 Structural Constraints

Certain types of segments are banned in certain syllable positions—I refer to these restrictions as *structural constraints*, in contrast with *sonority constraints*, which will be defined in §6.3.2. In addition to universal principles of syllable well-formedness (like those described in Chapter 5), there are a few key language-specific structural constraints governing the well-formedness of ITB syllables. First, although onsets are not required in the general case, they are required in non-initial syllables. This can be formalized as a negative substructure constraint banning the nucleic segment of a non-initial syllable from immediately following a segment in another syllable. The constraint $\kappa_{N IOS}$ “No Internal Onsetless Syllable” is defined in (6.1) and illustrated in Figure 6.1.

Figure 6.1: $\psi_{N IOS}$, the substructure banned by $\kappa_{N IOS}$ “No Internal Onsetless Syllable”

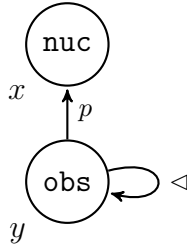


$$\kappa_{NIO S} \stackrel{\text{def}}{=} \neg(\exists u, v, w, x, y, z)[\sigma(u) \wedge \sigma(v) \wedge \mathbf{nuc}(x) \wedge \mathit{par}(w) = u \wedge \mathit{par}(x) = v \\ \wedge \mathit{par}(y) = w \wedge \mathit{par}(z) = x \wedge \mathit{succ}(y) = z] \quad (6.1)$$

“There does not exist a set of six nodes u, v, w, x, y, z , such that u and v are labeled σ , z is labeled \mathbf{nuc} , the parent of w is u , the parent of x is v , the parent of y is w , the parent of z is x , and the successor of y is z .”

Additionally, final obstruents are forbidden from being nucleic. Figure 6.2 illustrates the structure banned by κ_{NFO} (a nucleic word-final obstruent).

Figure 6.2: ψ_{NFO} , the substructure banned by κ_{NFO} “No Final Obstruent”



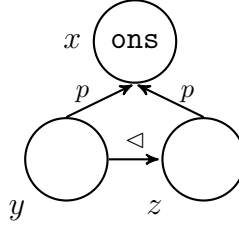
$$\kappa_{NFO} \stackrel{\text{def}}{=} \neg(\exists x, y)[\mathbf{nuc}(x) \wedge \mathbf{obs}(y) \wedge \mathbf{fin}(y) \wedge \mathit{par}(y) = x] \quad (6.2)$$

“There does not exist a set of nodes x, y , such that such that x is labeled \mathbf{nuc} , y is a final obstruent, and the parent of y is x .”

The loop labeled \triangleleft in Figure 6.2 indicates that position y is its own successor; that is, it is word-final.

Finally, we must ban complex onsets and codas. The constraint κ_{NCO} “No Complex Onset” was first defined in (4.78) and is reproduced below in (6.3). The substructure banned by the constraint is illustrated in Figure 6.3. Note that it has essentially the same form as κ_{NU} “Nucleus Unique,” defined in (5.2).

Figure 6.3: ψ_{NCO} , the substructure banned by κ_{NCO} “No Complex Onset”

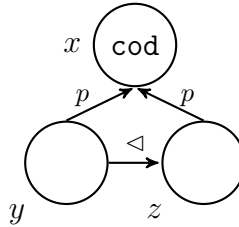


$$\kappa_{NCO} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{par}(y) = x \wedge \text{par}(z) = x \wedge \text{succ}(y) = z] \quad (6.3)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **ons**, the parent of y is x , the parent of z is also x , and the successor of y is z .”

Similarly, complex codas are banned by κ_{NCC} , defined in 6.4

Figure 6.4: ψ_{NCC} , the substructure banned by κ_{NCC} “No Complex Coda”



$$\kappa_{NCC} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{cod}(x) \wedge \text{par}(y) = x \wedge \text{par}(z) = x \wedge \triangleleft(y) = z] \quad (6.4)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **cod**, the parent of y is x , the parent of z is also x , and the successor of y is z .”

6.3.2 Sonority Sequencing Constraints

In addition to these structural constraints, we must also account for constraints on sonority sequencing. Dell & Elmedlaoui (1985, 2002) report the following sonority hierarchy for ITB.

$$\begin{aligned} & \text{voiceless stops} <_s \text{ voiced stops} <_s \text{ voiceless fricatives} \\ & <_s \text{ voiced fricatives} <_s \text{ nasals} <_s \text{ liquids} <_s \text{ high vowels} <_s [\text{a}] \end{aligned}$$

Recall that κ_{right} bans a segment dominated by an **ons** node whose successor is less sonorous. In ITB, there is one exception to this principle. If the onset is a HV, it may be followed by a sonorant of equal or lesser sonority (i.e., a nasal, liquid, or another high vowel). Syllables of this type are called Glide-Sonorant (GR) syllables.⁷ Some examples are given in Table 6.6.⁸ In GR syllables, the glide forms the onset to the nucleic sonorant, following an open syllable. Consider the UR /saul-x/. The /a/ must be nucleic because it is the most sonorous segment. If the /u/ were also nucleic, it could have no onset. Instead, the /l/ becomes a nucleus and the /u/ becomes its onset, surfacing as the glide [w].

Table 6.6: Words with GR Syllables

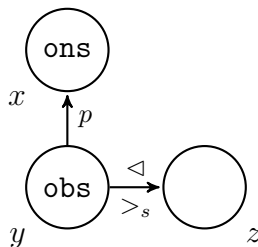
UR	SR	Gloss
/saul-x/	[sa.w x]	‘I spoke’
/t-iun-t-a-s/	[ti.ẉn.tas]	‘you climbed on him’
/i-ħaul=tn/	[i.ħa.ẉl.ṭn]	‘he made them (m.) plentiful’

⁷ As in ‘glide-resonant’ from Dell & Elmedlaoui (1985).

⁸ Following Dell & Elmedlaoui (2002), the - symbol represents word-internal morpheme boundaries and the = symbol represents the left edge of an enclitic. The difference between these types of boundaries is not relevant to the present analysis.

Thus, a modified version of κ_{right} is needed for ITB, as defined in (6.5). In particular, the SSP must only be enforced strictly if the first segment in the adjacent pair is an obstruent.

Figure 6.5: ψ_{son} , the substructure banned by κ_{son}

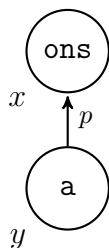


$$\kappa_{son} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{obs}(y) \wedge \text{par}(y) = x \wedge \text{succ}(y) = z \wedge <_s(z, y)] \quad (6.5)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **ons**, y is labeled **obs**, the parent of y is x , the successor of y is z , and y is more sonorous than z .”

Note that κ_{son} does not prohibit a syllable in which [a] is the onset. While HVs can be onsets in GR syllables, [a] is at the top of the sonority hierarchy and must always be nucleic. I therefore define the constraint κ_a to ban [a] in onset position. This constraint is defined in (6.6), with the banned substructure ψ_a illustrated in Figure 6.6.

Figure 6.6: ψ_a , the substructure banned by κ_a “No Onset [a]”

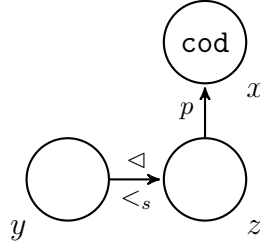


$$\kappa_a \stackrel{\text{def}}{=} \neg(\exists x, y)[\text{ons}(x) \wedge \mathbf{a}(y) \wedge \text{par}(y) = x] \quad (6.6)$$

“There does not exist a set of nodes x, y , such that x is labeled **ons**, y is an **a**, and the parent of y is x .”

On the coda side of things, ITB obeys κ_{left} , first defined in (5.8) and reproduced below in (6.7).

Figure 6.7: ψ_{left} , the substructure banned by κ_{left} “Left of Coda”



$$\kappa_{left} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{cod}(x) \wedge \text{par}(z) = x \wedge \text{succ}(y) = z \wedge <_s(y, z)] \quad (6.7)$$

“There does not exist a set of three nodes, x, y, z , such that x is labeled **cod** and is the parent of z , the successor of y is z , and y is less sonorous than z .”

6.3.3 The Formal Language for ITB

The formula K_{ITB} defined in (6.8) encapsulates all of the above constraints on syllable well-formedness in ITB.

$$\begin{aligned} K_{ITB} &\stackrel{\text{def}}{=} \kappa_{NR} \wedge \kappa_{NU} \wedge \kappa_{NIOS} \wedge \kappa_{NFO} \wedge \kappa_{NCO} \wedge \kappa_{NCC} \wedge \kappa_{son} \wedge \kappa_a \wedge \kappa_{left} \\ &= \psi_{NR} \wedge \neg\psi_{NU} \wedge \neg\psi_{NIOS} \wedge \neg\psi_{NFO} \wedge \neg\psi_{NCO} \wedge \neg\psi_{NCC} \wedge \\ &\quad \neg\psi_{son} \wedge \neg\psi_a \wedge \neg\psi_{left} \end{aligned} \quad (6.8)$$

Under the Tree Model Theory, L_{ITB} is the formal language defined by this formula, as defined in (6.9).

$$L_{ITB} \stackrel{\text{def}}{=} L(K_{ITB}) \tag{6.9}$$

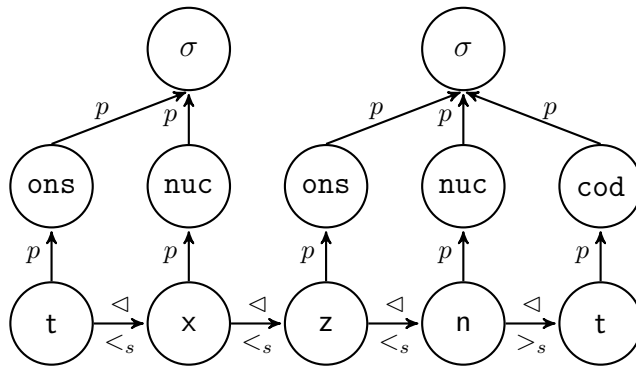
$$= \{w \in \Sigma^* \mid \mathcal{M}_w^{\text{tree}} \models K_{ITB}\} \tag{6.10}$$

Of course, L_{ITB} only accounts for basic syllable structure; a much longer formula would be required to characterize the entire phonology of ITB. For this reason, L_{ITB} is not precisely equivalent to the set of all phonologically well-formed words in ITB.

6.3.4 Extended Example: /t-xzn-t/

To illustrate how these constraints are evaluated, consider the correct tree word model for /t-xzn-t/, ‘you (sg.) stored,’ given in Figure 6.8.

Figure 6.8: $\mathcal{M}_{[t_x.znt]}$



In Figures 6.9 through 6.12, I show how the proposed inviolable constraints rule out all of the sub-optimal candidates offered by Prince & Smolensky (1993). Banned substructures are bolded.

$\mathcal{M}_{[t_x.znt]}$ violates κ_{left} because the first syllable contains ψ_{left} —in particular, because $[t] <_s [x]$. Similarly, $\mathcal{M}_{[t_xz.nt]}$ violates κ_{left} because the second syllable contains ψ_{left} ,

with $[x] <_s [z]$. Additionally, κ_{son} is violated because this word model contains ψ_{son} (because $[t] <_s [n]$), but one violation is enough to rule out this potential SR. $\mathcal{M}_{[txz.nt]}$ violates κ_{NIOs} because the first and second syllable together contain ψ_{NIOs} , the second syllable having no onset. $\mathcal{M}_{[t.x.z.n.t]}$ also violates κ_{NIOs} because all four non-initial syllables are onsetless. Again, any one violation would suffice.

Figure 6.9: $\mathcal{M}_{[tx.znt]}^{tree}$

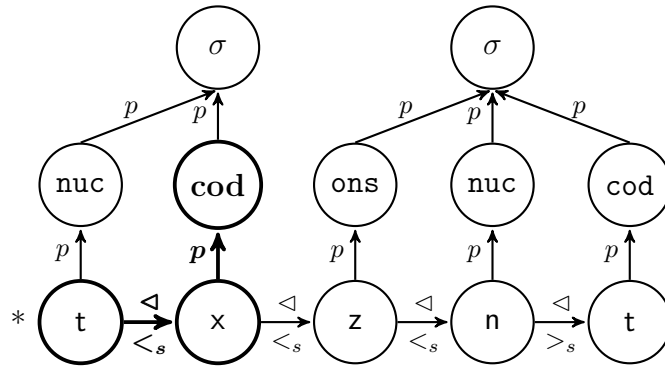
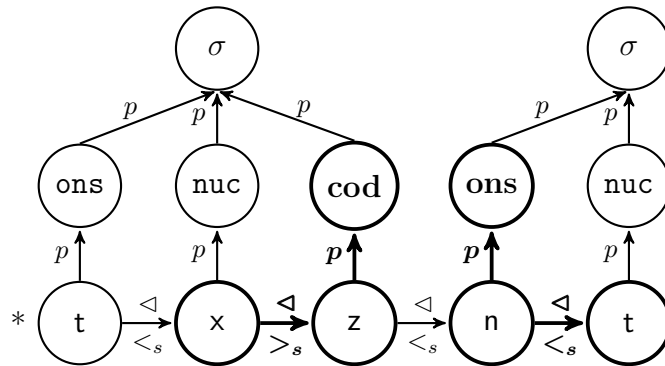


Figure 6.10: $\mathcal{M}_{[txz.nt]}^{tree}$



This extended example illustrates how the surface well-formedness of ITB syllables can be accounted for using inviolable constraints, without recourse to the optimization that OT is based on.

Figure 6.11: $\mathcal{M}_{[txz.\eta t]}^{tree}$

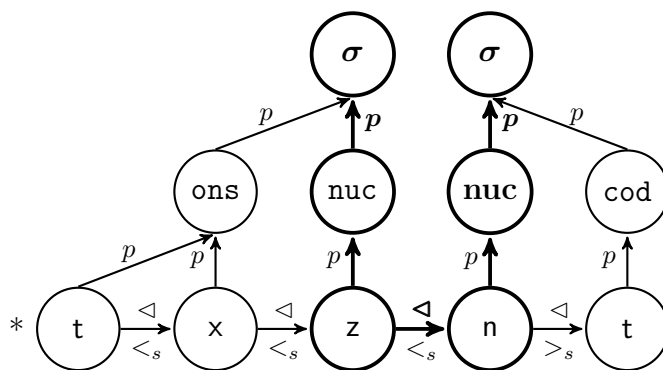
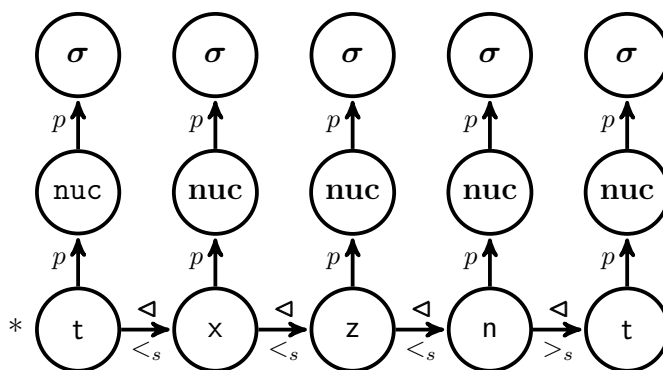


Figure 6.12: $\mathcal{M}_{[t.x.z.\eta t]}^{tree}$



6.4 Syllabification in ITB

The input to the transduction is a word model of an UR. I will use the Successor Word Model to represent this input and I will refer to several formulas first defined in Chapter 3, reproduced below. First, recall the set of primitive phonological features \mathcal{F} and the corresponding set of unary relations (labels), \mathcal{R}_f , reproduced in (6.11) and (6.12).

$$\mathcal{F} \stackrel{\text{def}}{=} \{\text{voice, cons, high, lab, alv, post, pal, vel, uv, phar, glot, stop, fric, nas, approx, lat}\} \quad (6.11)$$

$$\mathcal{R}_f \stackrel{\text{def}}{=} \{R_f \mid f \in \mathcal{F}\} \quad (6.12)$$

Additionally, I define formulas to pick out certain natural classes and certain positions in a word:

$$\text{obs}(x) \stackrel{\text{def}}{=} \text{stop}(x) \vee \text{fric}(x) \quad (6.13)$$

“ x is an **obstruent** iff it is a stop or a fricative.”

$$\text{son}(x) \stackrel{\text{def}}{=} \neg \text{obs}(x) \quad (6.14)$$

“ x is a **sonorant** iff x is not an **obstruent**.”

$$\text{init}(x) \stackrel{\text{def}}{=} \text{pred}(x) = x \quad (6.15)$$

“ x is **word-initial** iff its predecessor is itself.”

$$\text{fin}(x) \stackrel{\text{def}}{=} \text{succ}(x) = x \quad (6.16)$$

“ x is **word-final** iff its successor is itself.”

$$\text{med}(x) \stackrel{\text{def}}{=} \neg(\text{init}(x) \vee \text{fin}(x)) \quad (6.17)$$

“ x is **word-medial** iff it is neither initial nor final.”

$$\text{init_stop}(x) \stackrel{\text{def}}{=} \text{init}(x) \wedge \text{stop}(x) \quad (6.18)$$

“ x is an initial stop iff it is word-initial and a stop consonant.”

$$\text{fin_obs}(x) \stackrel{\text{def}}{=} \text{fin}(x) \wedge \text{obs}(x) \quad (6.19)$$

“ x is a final obstruent iff it is word-final and an obstruent.”

I will make use of these as I develop the ITB syllabification transduction.

6.4.1 Sonority and Other Considerations

To label syllable positions in the output, it is first necessary to identify sonority peaks, other positions that may be nucleic, and marked positions that are prohibited from being nucleic.

A word-medial sonority peak is simply a segment that is more sonorous than both its neighboring segments, as defined in (6.20). To be exhaustive, I also define word-initial and word-final ‘peaks’ as segments more sonorous than their word-medial neighbors (see (6.21) and (6.22)). Then a sonority peak (6.23) is any of these three.⁹

$$\text{med_pk}(x) \stackrel{\text{def}}{=} \text{med}(x) \wedge x >_s \text{pred}(x) \wedge x >_s \text{succ}(x) \quad (6.20)$$

“ x is a word-medial sonority peak iff it is medial, more sonorous than its predecessor, and more sonorous than its successor.”

$$\text{init_pk}(x) \stackrel{\text{def}}{=} \text{init}(x) \wedge x >_s \text{succ}(x) \quad (6.21)$$

“ x is a word-initial sonority peak iff it is initial and more sonorous than its successor.”

⁹ med = medial; init = initial; fin = final; pk = peak.

$$\text{fin_pk}(x) \stackrel{\text{def}}{=} \text{fin}(x) \wedge x >_s \text{pred}(x) \quad (6.22)$$

“ x is a word-final sonority peak iff it is final and more sonorous than its predecessor.”

$$\text{son_pk}(x) \stackrel{\text{def}}{=} \text{med_pk}(x) \vee \text{init_pk}(x) \vee \text{fin_pk}(x) \quad (6.23)$$

“ x is a sonority peak iff it is a medial peak, an initial peak, or a final peak.”

Given two adjacent segments of equal sonority, [Frampton \(2011\)](#) observes that “a slot x is ‘more prominent’ than an adjacent slot y . . . if they are equally sonorous and x is to the left of y , unless x is initial.” In other words, the leftmost segment of a sonority plateau takes prominence when it comes to assigning nucleic status, unless it is word-initial. For example, consider the /sx/ sequence in /rks-x/ ‘I hid.’ The two segments are of equal sonority, and it is the /s/ which syllabifies in the surface form [ṛ.ḳṣx] (compare to *[ṛḳ.ṣx]).

Left-prominence is captured by the formula `l_prom`, defined in (6.24).¹⁰

$$\text{l_prom}(x) \stackrel{\text{def}}{=} x =_s \text{succ}(x) \wedge \text{med}(x) \quad (6.24)$$

“ x is left-prominent iff it is of equal sonority with its successor and it is word-medial.”

Note that word-final positions are explicitly excluded. Were this left out of the definition, every final position would satisfy `l_prom` due to it being its own successor and, therefore, equally sonorous to its successor (itself). Then a prominence peak is either a sonority peak or a segment that satisfies `l_prom`, as in (6.25).

$$\text{prom_pk}(x) \stackrel{\text{def}}{=} \text{son_pk}(x) \vee \text{l_prom}(x) \quad (6.25)$$

“ x is a prominence peak iff it is a sonority peak or left-prominent.”

¹⁰ `prom` = prominence.

Prominence peaks are typically syllabic nuclei, with two exceptions. The first exception is simple: final obstruents cannot be nucleic. The second exception occurs when a prominence peak precedes a GR sequence. In this case, the glide becomes an onset despite being more sonorous than its successor. The shorthand formula $\underline{\text{GR}}$ (6.26) picks out the sonorant (resonant) in a GR syllable, which is always nucleic.

$$\underline{\text{GR}}(x) \stackrel{\text{def}}{=} \text{prom_pk}(\text{pred}(\text{pred}(x))) \wedge \text{HV}(\text{pred}(x)) \wedge \text{son}(x) \quad (6.26)$$

“ x is the nucleus of a GR syllable iff its predecessor’s predecessor is a prominence peak, x is a HV, and x is a sonorant.”

6.4.2 Identifying Syllable Constituents

Now it is easy to identify the syllable constituent for any given segment. formula (6.27) states that a segment is nucleic iff it is **a**) a prominence peak or **b**) the sonorant in a GR sequence **and** iff it is not a final obstruent. A segment is an onset iff it is not nucleic, but its successor is; this satisfies **ons**, defined in (6.28). Finally, a segment is a coda iff it is immediately preceded by a nucleus but it does not satisfy **ons** (i.e., it is not the onset for the following syllable), as in (6.29).

$$\text{nuc}(x) \stackrel{\text{def}}{=} \neg(\text{prom_pk}(x) \vee \underline{\text{GR}}(x)) \wedge \neg\text{fin_obs}(x) \quad (6.27)$$

$$\text{ons}(x) \stackrel{\text{def}}{=} \neg\text{nuc}(x) \wedge \text{nuc}(\text{succ}(x)) \quad (6.28)$$

$$\text{cod}(x) \stackrel{\text{def}}{=} \text{nuc}(\text{pred}(x)) \wedge \neg\text{ons}(x) \quad (6.29)$$

6.4.3 The ITB Syllabification Transduction

The transduction Γ_{ITB} is a set of formulas that use information in the UR (the input) to construct and label the SR (the output). For every relation R in the input word model, there is a corresponding relation R^ω defined over each copy set of the output

word model. With the formulas defined in the previous section, the transduction itself is simple to define.

Binary Relations. Sonority relations are all preserved in Copy Set 1 under Γ_{ITB} . These relations are defined in (6.30–6.31).

$$<_s(x^1, y^1) \stackrel{\text{def}}{=} <_s(x, y) \quad (6.30)$$

$$= _s(x^1, y^1) \stackrel{\text{def}}{=} = _s(x, y) \quad (6.31)$$

“The first copy of x in the output is less/equally sonorous than/to the first copy of y iff x is less/equally sonorous than/to y in the input.”

All the other sonority relations can be derived from these two, as demonstrated in Chapter 5.

Feature Labels. Recall that \mathcal{R}_f is the set of unary relations for phonological features. These feature labels are also preserved in Copy Set 1, as specified by (6.32).

$$(\forall f \in \mathcal{R}_f)[\mathbf{f}(x^1) \stackrel{\text{def}}{=} \mathbf{f}(x)] \quad (6.32)$$

“For all features \mathbf{f} in the set \mathcal{R}_f , the first copy of x in the output is labeled \mathbf{f} iff x is labeled \mathbf{f} in the input.”

Syllable Constituent Labels and Sigma Nodes. I have already done the work of identifying onsets, nuclei, and codas in the input form. All that remains is to formalize the formulas that label the syllable constituents in the *output* form—specifically, in Copy Set 2. The relevant formulas are given in (6.33–6.35).

$$\text{nuc}(x^2) \stackrel{\text{def}}{=} \text{nuc}(x) \quad (6.33)$$

$$\text{ons}(x^2) \stackrel{\text{def}}{=} \text{ons}(x) \quad (6.34)$$

$$\text{cod}(x^2) \stackrel{\text{def}}{=} \text{cod}(x) \quad (6.35)$$

“The second copy of x in the output is labeled *nuc/ons/cod* iff x satisfies *nuc/ons/cod* in the input.”

Similarly, σ nodes are simply the 3rd copy of those positions that satisfy $\text{nuc}(x)$.

$$\sigma(x^3) \stackrel{\text{def}}{=} \text{nuc}(x) \quad (6.36)$$

“The third copy of x in the output is labeled σ iff x satisfies nuc in the input.”

Functions. The successor and predecessor functions are preserved in Copy Set 1, as defined in (6.37–6.38)

$$\text{succ}(x^1) \stackrel{\text{def}}{=} (\text{succ}(x))^1 \quad (6.37)$$

$$\text{pred}(x^1) \stackrel{\text{def}}{=} (\text{pred}(x))^1 \quad (6.38)$$

“The successor/predecessor of the first copy of x in the output is the first copy of the successor/predecessor of x in the input.”

As in the Flat-to-Tree transduction Γ_{tf} , we must also ‘build’ the hierarchical structure from the flat underlying string. This is accomplished with the formulas given in (6.39–6.40).

$$\text{par}(x^2) \stackrel{\text{def}}{=} \left\{ \begin{array}{ll} x^3 & \Leftarrow \text{nuc}(x) \\ (\text{succ}(x))^3 & \Leftarrow \text{ons}(x) \\ (\text{pred}(x))^3 & \Leftarrow \text{cod}(x) \end{array} \right. \quad (6.39)$$

$$\text{par}(x^1) \stackrel{\text{def}}{=} \left\{ \begin{array}{ll} x^2 & \Leftarrow \text{nuc}(x) \\ (\text{succ}(x))^2 & \Leftarrow \text{ons}(x) \\ (\text{pred}(x))^2 & \Leftarrow \text{cod}(x) \end{array} \right. \quad (6.40)$$

$$(6.41)$$

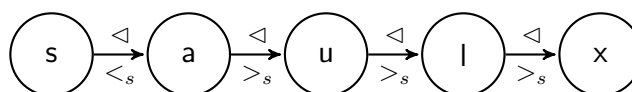
Finally, (6.42) licenses only those nodes which are labeled.

$$\text{license}(x) \stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{ITB})[R(x)] \quad (6.42)$$

6.4.4 Extended Example: /saulx/

To illustrate how the transduction works, consider the underlying form /saul-x/ ‘I spoke.’ Its word model (in the Successor Model Theory) is illustrated in Figure 6.13.

Figure 6.13: $\mathcal{M}_{saulx}^\triangleleft$



There are five positions, 1 through 5, and each position has a set of feature labels which I abbreviate with the shorthand segment formulas *s*, *a*, *u*, *l*, and *x*. Table 6.7 gives the truth values for the formulas relevant to syllable structure in the word *saulx*.

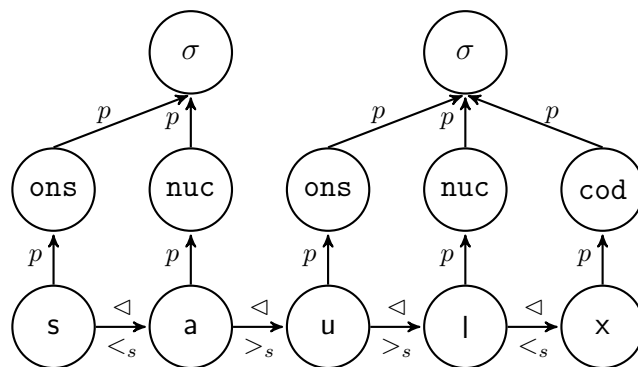
Table 6.7: Truth Table for /saul-x/

x	1	2	3	4	5
$s(x)$	✓
$a(x)$.	✓	.	.	.
$u(x)$.	.	✓	.	.
$l(x)$.	.	.	✓	.
$x(x)$	✓
$<_s(x, succ(x))$	✓
$=_s(x, succ(x))$
$>_s(x, succ(x))$.	✓	✓	✓	✓
$son_pk(x)$.	✓	.	.	.
$l_prom(x)$
$prom_pk(x)$.	✓	.	.	.
$fin_obs(x)$	✓
$GR(x)$.	.	.	✓	.
$nuc(x)$.	✓	.	✓	.
$ons(x)$	✓	.	✓	.	.
$ons(x)$	✓	.	✓	.	.
$cod(x)$	✓

The first position is less sonorous than the second, with sonority falling monotonically after that. There are no sonority plateaus, so no position may satisfy l_prom . The only prominence peak is then the single sonority peak, position 2. Because position 5 is a final obstruent, it is marked (in the sense that it cannot be nucleic). Note that position 2 is a sonority peak, position 4 is a glide, and position 5 is a sonorant. This configuration means that position 4 satisfies GR and therefore satisfies nuc , even though it is not a prominence peak. Position 2 satisfies nuc by virtue of satisfying prom_pk . Positions 1 and 3 satisfy ons because their successors are both nucleic. Finally, position 5 satisfies cod because its predecessor is a nucleus and it does not satisfy ons .

The resulting output $\Gamma_{ITB}(\mathcal{M}_{saulx}^{\triangleleft})$ is illustrated in Figure 6.14. Recall that the vowel-glide distinction is predictable from syllable constituency. Because position 2 satisfies u and ons , it surfaces as the glide $[w]$. Thus, the surface form is pronounced $[sa.w|x]$.

Figure 6.14: $\Gamma_{ITB}(\mathcal{M}_{saulx}^{\triangleleft})$



Notice that $\Gamma_{ITB}(\mathcal{M}_{saulx}^{\triangleleft}) = \mathcal{M}_{saulx}^{tree}$. The syllabification transduction has taken a flat string in the Successor Model and produced the Tree Model of the syllabified word.

Chapter 7

CASE STUDY: MOROCCAN ARABIC

While ITB is a crucial case study for syllable theory, its tolerance of obstruents as syllabic nuclei and its general adherence to the SSP actually limit the range of syllable-related phenomena to study in the language—for example, there is little to no epenthesis or deletion because well-formed syllables can be readily formed from the underlying phonemes. In Moroccan Arabic (MA), on the other hand, short vowels are frequently inserted to break up consonant clusters in certain positions. The quality of these epenthetic vowels varies somewhat from a fronted pronunciation similar to [i] to a backed pronunciation closer to [ʊ] (Heath, 1987), but I will abstract away from these distinctions and simply represent all epenthetic vowels as [ə] (in line with, e.g., Benhal-lam, 1990; Boudlal, 2001; Dell & Elmedlaoui, 2002; Keegan, 1986).¹ Some examples of epenthesis in perfective verb forms are given in Table 7.1.²

¹ Heath (1987) first assumes that [ə] and [ʊ] are phonemic, but later reanalyzes both as epenthetic, with [ʊ] being an allophone of [ə].

² As in the previous chapter, the ! symbol indicates rightward spreading of pharyngealization; the - symbol represents word-internal morpheme boundaries; and the = symbol represents the left edge of an enclitic.

Table 7.1: Epenthesis in MA Verbs (Perfective Forms)

	UR	SR	Gloss
1.	a. /d ^ɾ rb/	[!d̩.rəb]	‘he hit’
	b. /d ^ɾ rb-at/	[!d̩ər.bat]	‘she hit’
	c. /d ^ɾ rb=k/	[!d̩ər.bək]	‘he hit you’
2.	a. /krkb/	[kər.kəb]	‘he rolled’
	b. /krkb-at/	[kər.k.bat]	‘she rolled’
	c. /krkb=k/	[kər.k.bək]	‘he rolled you’

These examples are representative of where epenthesis most often occurs in MA. In a triconsonantal (CCC) root like (1a), the first consonant syllabifies and a schwa is inserted after the second consonant; that is, /CCC/ → [C.CəC]. Additional examples include /ktb/ → [k.təb] ‘he wrote’ and /ktf/ → [k.təf] ‘shoulder,’ among many others. With the addition of suffixes, as in (1b) and (1c), epenthesis does not necessarily follow the first consonant. Rather, it appears preferable to create CəC syllables when possible.

In a quadriconsonantal (CCCC) root like (2a), schwas are inserted before the second and fourth consonants; that is, /CCCC/ → [CəC.CəC]. Notice that the heteromorphic /CCCC/ sequence in (1c) behaves identically, producing two CəC syllables. Moreover, (2b) and (2c) illustrate how word-internal complex codas can form, indicating that CəC syllables may be especially preferred in word-final position. If they were always preferred word-internally, we would expect to see forms like *[kər.kabt].

As will be seen, the landscape of epenthesis in MA is even more complicated than it appears from this brief introduction. In the remainder of this chapter, I review the basic facts regarding syllable structure in MA, highlighting similarities to and differences

from ITB. Then I develop mathematical formalisms for characterizing syllable well-formedness in § 7.2 and syllabification in § 7.3.

7.1 The Basics

The Arabic and Berber languages are not phylogenetically related, but due to the history of language contact, multilingualism, and colonialism in Morocco (see [Ennaji, 2005](#) and the references therein), MA is more similar to ITB in some aspects of its phonology than it is to Arabic dialects spoken in other regions. For one, the MA and ITB consonant inventories are very similar. Table 7.2 shows the consonant inventory of MA according to [Heath, 1997](#) (compare to Table 6.3 for ITB).³

Table 7.2: Phonemic Singleton Consonants in MA

	Labial	Alveolar	Palato-alveolar	Velar	Uvular	Pharyngeal	Glottal
Stop	b	t t ^ʕ d d ^ʕ		k g	q		
Fricative	f	s s ^ʕ z z ^ʕ	ʃ ʒ		χ ʁ	ħ ʕ	h
Nasal	m	n					
Approximant		r r ^ʕ					
Lateral		l					

As in ITB, the alveolar consonants have pharyngealized counterparts that trigger rightward spreading, as in [!dər.bək] ‘he hit you’ from /d^ʕrb=k/. Consonant length is also

³ Where [Heath \(1997\)](#) uses nonstandard symbols, I present their IPA equivalents.

contrastive (e.g. [mən] ‘from’ vs. [mənn] ‘to boast’). The most notable difference between the two consonant inventories is that MA lacks the labialized velar and uvular consonants found in ITB, as well as some of the contrastive pharyngeals.

MA shares the same three phonemic vowels as ITB: /a/, /i/, and /u/.⁴ Unlike Classical Arabic (CA) and many modern Arabic dialects, MA has no contrastive vowel length. The long vowels of CA lexical items have shortened in their corresponding MA forms, and the CA short vowels have disappeared altogether (Lahrouchi, 2018). For example, compare the CA [ki.taːb] to the MA [ḳ.tab] ‘he wrote.’ The first syllable has a short [i] in CA that is absent in MA; the second syllable has a long low vowel in CA and a short low vowel in MA.

The glide/vowel distinction among the HVs [i~j] and [u~w] is sometimes predictable in MA, but not always. The glides must be considered independent phonemes because their distribution is not easily explained by allophonic variation alone (Dell & Elmedlaoui, 2002; Heath, 1987). Thus I use the feature *voc* (vocalic) to differentiate between vowels and glides: vowels are [+*voc*, -*cons*], while glides are [-*voc*, -*cons*].

7.2 Surface Well-formedness in MA

In this section, I outline the inviolable constraints that govern surface well-formedness in MA and construct the corresponding formal language L_{MA} , the set of all phonologically well-formed words.

7.2.1 Structural Constraints

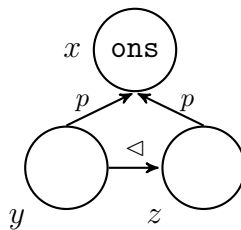
As pointed out by Shaw et al. (2009), scholars of MA phonology fall broadly into two camps regarding syllable-internal structure: those who adopt the ‘complex onset hypothesis’ (e.g., Benhallam, 1990; Heath, 1987; Keegan, 1986) and those who adopt

⁴ This holds true for native MA words. Some scholars (e.g., Heath, 1987; Sayed, 1982) have suggested that [o] might also be phonemic, but it is almost entirely restricted to loan words such as [lokal] ‘premises, site’ from the French *locale*.

the ‘simplex onset hypothesis’ (e.g., Boudlal, 2001; Dell & Elmedlaoui, 2002; Kiparsky, 2003). The former group would syllabify *skru* ‘they got drunk’ as a monosyllable [skru], while the latter would find the same string to comprise two syllables: [sk̟.ru]. Dell & Elmedlaoui (2002) present a compelling argument in favor of the simplex onset hypothesis using evidence from syllabification in verse and song; Shaw et al. (2009) and Gafos et al. (2010) offer further evidence from articulatory data. For these reasons, I adopt the simplex hypothesis. One consequence of this is that some word-initial consonants will be nucleic (as in [k̟.tab] ‘he wrote’), but they are much more restricted than in ITB.

The constraint κ_{NCO} “No Complex Onset” was first defined in (4.78) and is reproduced below in (7.1).

Figure 7.1: ψ_{NCO} , the substructure banned by κ_{NCO} “No Complex Onset”

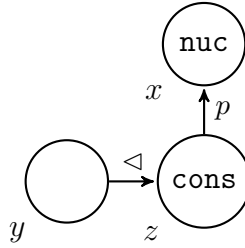


$$\kappa_{NCO} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{par}(y) = x \wedge \text{par}(z) = x \wedge \text{succ}(y) = z] \quad (7.1)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled **ons**, the parent of y is x , the parent of z is also x , and the successor of y is z .”

If we accept the simplex onset hypothesis, we must also accept that syllabic consonants are allowed, but are restricted to word-initial position. Word-internal syllabic consonants are banned by κ_{NISC} , defined in (7.2) below.

Figure 7.2: ψ_{NISC} , the substructure banned by κ_{NISC} “No Internal Syllabic Consonants”

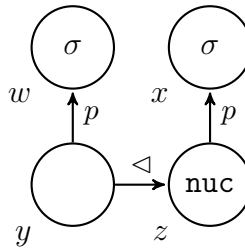


$$\kappa_{NISC} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{nuc}(x) \wedge \text{cons}(z) \wedge \text{par}(z) = x \wedge \text{succ}(y) = z \wedge y \neq z] \quad (7.2)$$

“There does not exist a set of three nodes x, y, z , such that x is labeled nuc , z is labeled cons , the parent of z is x , the successor of y is z , and $y \neq z$.”

Just as in ITB, onsets are required in all non-initial syllables. So MA has the constraint κ_{NIOS} , originally defined in (6.1) and reproduced below in (7.3).

Figure 7.3: ψ_{NIOS} , the substructure banned by κ_{NIOS} “No Internal Onsetless Syllables”



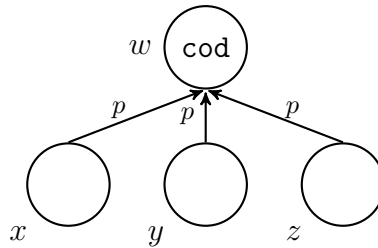
$$\kappa_{NIO\mathcal{S}} \stackrel{\text{def}}{=} \neg(\exists w, x, y, z)[\sigma(w) \wedge \sigma(x) \wedge \text{nuc}(z) \wedge \text{par}(y) = w \wedge \text{par}(z) = x \wedge \text{succ}(y) = z] \quad (7.3)$$

“There does not exist a set of four nodes w, x, y, z , such that w and x are labeled σ , z is labeled nuc , the parent of y is w , the parent of z is x , and the successor of y is z .”

The effect of this constraint is to ban any structure where the nucleus of one syllable immediately follows material in a different syllable (whether that be a nucleus or a coda).

Turning to codas, it is generally agreed that complex codas consisting of only two segments are allowed. Three-segment codas are therefore banned by κ_{N3C} , defined in (7.4) below. Note that any coda longer than three segments will also be banned by this constraint, since it would necessarily contain the substructure ψ_{N3C} .

Figure 7.4: ψ_{N3C} , the substructure banned by κ_{N3C} “No Trisegmental Coda”



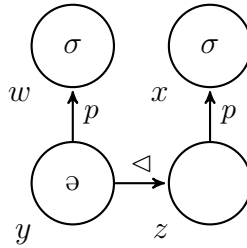
$$\kappa_{NCO} \stackrel{\text{def}}{=} \neg(\exists w, x, y, z)[\text{ons}(w) \wedge \text{par}(x) = w \wedge \text{par}(y) = w \wedge \text{par}(z) = w \wedge x \neq y \wedge x \neq z \wedge y \neq z] \quad (7.4)$$

“There does not exist a set of four nodes $w, x, y,$ and $z,$ such that w is labeled **ons**; w is the parent of $x, y,$ and $z;$ $x \neq y;$ $x \neq z;$ and $y \neq z.$ ”

In addition to these constraints on syllable margins, the position of epenthetic vowels is also restricted. Schwas are prohibited from surfacing in open syllables, meaning two substructures must be banned: 1) a schwa immediately preceding material in another syllable; and 2) a word-final schwa.

$\kappa_{\sigma\sigma}$ “No Open Schwa (Word-Internal)” is defined in (7.5) below.

Figure 7.5: $\psi_{N\sigma\sigma}$, the substructure banned by $\kappa_{\sigma\sigma}$ “No Open Schwa (Word-Internal)”



$$\kappa_{\sigma\sigma} \stackrel{\text{def}}{=} \neg(\exists w, x, y, z)[\sigma(w) \wedge \sigma(x) \wedge \theta(y) \wedge \text{par}(y) = w \wedge \text{par}(z) = x \wedge \text{succ}(y) = z] \quad (7.5)$$

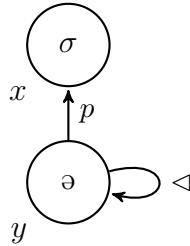
“There does not exist a set of four nodes $w, x, y, z,$ such that w and x are labeled $\sigma,$ y is a $\theta,$ the parent of y is $w,$ the parent of z is $x,$ and the successor of y is $z.$ ”

This would prohibit a form like $*[\text{k}\theta.\text{t}\theta\text{b}],$ for example.

Dell & Elmedlaoui (2002) point out that schwas can appear in word-internal open syllables in verse, but this is only after word-level syllabification is overwritten by line-level resyllabification. As I am concerned only with word-level syllabification, κ_{σ} is inviolable for the purposes of this analysis.

Similarly, $\kappa_{\sigma\varnothing}$ “No Open Schwa (Word-Final)” is defined in (7.6).

Figure 7.6: $\psi_{N_{\sigma\varnothing}}$, the substructure banned by $\kappa_{\sigma\varnothing}$ “No Open Schwa (Word-Final)”



$$\kappa_{\sigma\varnothing} \stackrel{\text{def}}{=} \neg(\exists x, y)[\sigma(x) \wedge \varnothing(y) \wedge \text{par}(y) = x \wedge \text{succ}(y) = y] \quad (7.6)$$

“There does not exist a pair of nodes x, y , such that x is labeled σ , y is a \varnothing , the parent of y is x , and y is word-final.”

This would prohibit a form like *[kət.bə]. There is no exception to this constraint, even in verse and song.

7.2.2 Sonority Sequencing Constraints

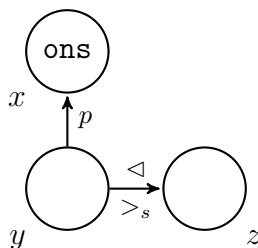
According to Dell & Elmedlaoui (2002), the sonority hierarchy of MA is much less fine-grained than that for ITB. They distinguish between only four natural classes:

$$\text{obstruents} <_s \text{ nasals} <_s \text{ liquids} <_s \text{ vocoids}$$

Here vocoids may be vowels or glides, including the low vowel which has no semivowel counterpart.

Sonority sequencing constraints in MA are slightly different from those ITB. The constraint on segments to the right of an onset, κ_{right} remains the same. This constraint was first defined in (5.7) and is reproduced below in (7.7).

Figure 7.7: ψ_{right} , the substructure banned by κ_{right} “Right of Onset”



$$\kappa_{right} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{ons}(x) \wedge \text{par}(y) = x \wedge \text{succ}(y) = z \wedge <_s(z, y)] \quad (7.7)$$

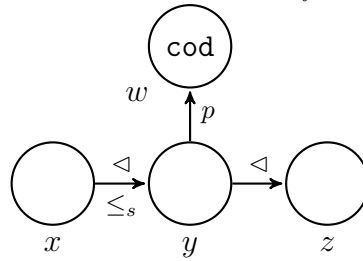
“There does not exist a set of three nodes x, y, z , such that x is labeled **ons** and is the parent of y , the successor of y is z , and y is more sonorous than z .”

In addition to this, two versions of κ_{left} are needed. For word-internal codas, a strict decrease in sonority is required, meaning that adjacent coda segments cannot have equal sonority. For example, /krkb=k/ ‘he rolled you’ syllabifies as [kærk.bæk], not *[k.ræk.bæk]; whereas /ktb-l=k/ ‘he wrote to you’ syllabifies as [k.təb.læk], not */kətəb.læk/. This generalization is captured by κ_{left1} , defined in (7.8).

$$\kappa_{left1} \stackrel{\text{def}}{=} \neg(\exists w, x, y, z)[\text{cod}(w) \wedge \text{par}(y) = w \wedge \text{succ}(x) = y \wedge x \leq_s y \wedge \text{succ}(y) = z \wedge y \neq z] \quad (7.8)$$

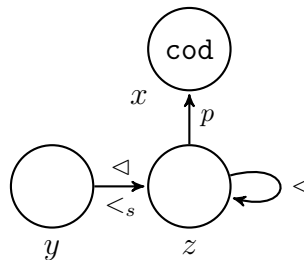
“There does not exist a set of four nodes, w, x, y, z , such that w is labeled **cod**, the parent of y is w , the successor of x is y , x is of lesser or equal sonority to y , the successor of y is z , and $y \neq z$.”

Figure 7.8: ψ_{left1} , the substructure banned by κ_{left1} “Left of Coda (Word-Internal)”



In contrast, word-final codas may have a sonority plateau, as evidenced by forms like [kædb] ‘lie’ and [ssəbt] ‘Saturday.’ This is enforced by the constraint κ_{left2} , defined in (7.9).

Figure 7.9: ψ_{left2} , the substructure banned by κ_{left2} “Left of Coda (Word-Final)”



$$\kappa_{right} \stackrel{\text{def}}{=} \neg(\exists x, y, z)[\text{cod}(x) \wedge \text{par}(z) = x \wedge \text{succ}(y) = z \wedge \text{succ}(z) = z \wedge y <_s z] \quad (7.9)$$

“There does not exist a set of three nodes, x, y, z , such that x is labeled cod and is the parent of z , the successor of y is z , z is its own successor (i.e., is word-final), and y is less sonorous than z .”

7.2.3 Exceptions to the Epenthesis Pattern

There are three types of exceptions to the generalization that /CCC/ stems syllabify as [CC.əC]. First, geminates are inseparable, so stems that end in a geminate must have

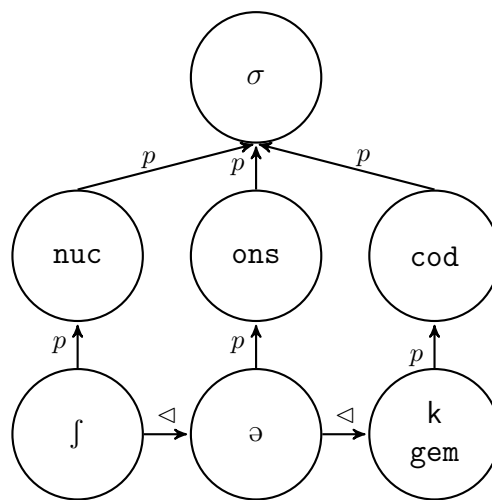
a schwa inserted before the penultimate consonant. Table 7.3 gives some examples of such stems, with both verbs and nouns represented.

Table 7.3: Geminate CəCC Forms in MA

	UR	SR	Gloss
1. a.	/ʃkk/	[ʃəkk]	‘he doubted’
b.	/ʃdd/	[ʃədd]	‘he held’
2. a.	/dmm/	[dəmm]	‘blood’
b.	/fkk/	[fəkk]	‘jaw’

Geminates could be handled in a number of ways. Perhaps the most straightforward is to add a labeling relation to the model signature. The label **gem** is true for any position corresponding to a geminate consonant. For example, Figure 7.10 illustrates the word model for [ʃəkk], with the **gem** label indicating that the third domain position is a geminate.

Figure 7.10: The Tree Word Model for [ʃəkk]



With geminates defined in this way, no constraint is needed to ban the splitting of geminates; it is simply not possible to split a single position in two.

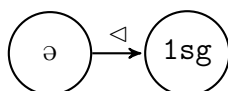
We must also make an exception for words ending in the suffix *-t*, which is the first-person singular agreement marker in the past tense. Table 7.4 shows some of these forms, alongside the forms we would expect if they were syllabified according to the typical pattern.

Table 7.4: MA 1sg Past Tense Verb Forms

	UR	SR	Expected	Gloss
1.	/ktb-t/	[k.təbt]	*[kət.bət]	‘I wrote’
2.	/krkb-t/	[kər.kəbt]	*[k.rək.bət]	‘I rolled’

To account for this exception, I will use the label **1sg** to mark the 1sg past-tense suffix in the underlying form. Then it suffices to ban an epenthetic vowel immediately preceding this suffix, as in κ_{1sg} defined in (7.10).

Figure 7.11: ψ_{1sg} , the substructure banned by κ_{1sg} “No Epenthesis Preceding 1sg Suffix”



$$\kappa_{1sg} \stackrel{\text{def}}{=} \neg(\exists x, y)[\text{ə}(x) \wedge \mathbf{1sg}(y) \wedge \text{succ}(x) = y] \quad (7.10)$$

“There does not exist a pair of nodes x, y , such that x is labeled ə , y is the 1sg suffix, and the successor of x is y .”

In addition to these systematic exceptions, there is also a class of nouns that simply do not conform to the typical epenthesis pattern. These are CCC stems that syllabify

as [C.əCC] instead of [CC.əC]. In Table 7.5, the stems in (1) conform to the typical pattern, while those in (2) do not.

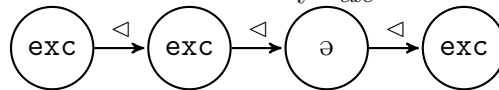
Table 7.5: Additional Examples of Epenthesis in MA Nouns

	Surface Representation	Gloss
1. a.	[k.təf]	‘shoulder’
b.	[f.təg]	‘loaf (of bread)’
2. a.	[bərd]	‘wind’
b.	[kəlb]	‘dog’

Stems that fall into the latter category will have to be marked as a lexical exception.⁵ I will use the label **exc** for this purpose. In the representations to follow, all three consonants in an exceptional stem will bear this feature. Alternatively, the feature **exc** could be reserved for initial/final position only, as a sort of tag rather than a feature of individual segments. The difference is inconsequential here, given the fact that stems of this type only ever have three consonants, and it is trivial to refer to a segment 2 positions away.

The constraint against epenthesis after the second consonant in an exceptional stem is κ_{exc} , defined in (7.11) below.

Figure 7.12: ψ_{exc} , the substructure banned by κ_{exc} “No CCəC for Exceptional Stems”



⁵ See Al Ghadi (2014) and Boudlal (2006/2007) for a sonority-based account of CəCC nouns, which is not adopted here.

$$\begin{aligned} \kappa_{exc} \stackrel{\text{def}}{=} & \neg(\exists w, x, y, z)[\mathbf{exc}(w) \wedge \mathbf{exc}(x) \wedge \mathbf{\emptyset}(y) \wedge \mathbf{exc}(z) \wedge \mathit{succ}(w) = x \\ & \wedge \mathit{succ}(x) = y \wedge \mathit{succ}(y) = z] \quad (7.11) \end{aligned}$$

“There does not exist a set of four nodes w, x, y, z , such that: w, x , and z are labeled \mathbf{exc} ; y is labeled $\mathbf{\emptyset}$; the successor of w is x ; the successor of x is y ; and the successor of y is z .”

7.2.4 The Formal Language for MA

The formula K_{MA} defined in (7.12) encapsulates all of the above constraints on syllable well-formedness in MA.

$$\begin{aligned} K_{MA} \stackrel{\text{def}}{=} & \kappa_{NR} \wedge \kappa_{NU} \wedge \kappa_{NCO} \wedge \kappa_{NISC} \wedge \kappa_{NIOS} \wedge \kappa_{N3C} \wedge \kappa_{N\mathbf{\emptyset}\sigma} \wedge \kappa_{N\mathbf{\emptyset}\mathbf{x}} \wedge \kappa_{right} \\ & \wedge \kappa_{left1} \wedge \kappa_{left2} \wedge \kappa_{1sg} \wedge \kappa_{exc} \\ = & \psi_{NR} \wedge \neg\psi_{NU} \wedge \neg\psi_{NCO} \wedge \neg\psi_{NISC} \wedge \neg\psi_{NIOS} \wedge \neg\psi_{N3C} \wedge \neg\psi_{N\mathbf{\emptyset}\sigma} \wedge \neg\psi_{N\mathbf{\emptyset}\mathbf{x}} \wedge \neg\psi_{right} \\ & \wedge \neg\psi_{left1} \wedge \neg\psi_{left2} \wedge \neg\psi_{1sg} \wedge \neg\psi_{exc} \end{aligned} \quad (7.12)$$

The alphabet of labels Σ_{MA} must include the phonological feature labels, syllable constituent labels, and the additional unary relations defined above to handle exceptions. A formal definition of Σ_{MA} is given in (7.13).

$$\Sigma_{MA} \stackrel{\text{def}}{=} \mathcal{F} \cup \{\mathbf{ons}, \mathbf{nuc}, \mathbf{cod}, \sigma\} \cup \{\mathbf{1sg}, \mathbf{exc}\} \quad (7.13)$$

Under the Tree Model Theory, the formal language L_{MA} (i.e., the set of correctly syllabified MA words) is defined in (7.14).

$$L_{MA} \stackrel{\text{def}}{=} L(K_{MA}) \quad (7.14)$$

$$= \{w \in \Sigma^* \mid \mathcal{M}_w^{\text{tree}} \models K_{MA}\} \quad (7.15)$$

As with L_{ITB} in the previous chapter, L_{MA} only accounts for basic syllable structure, not the entire phonology of MA. Nonetheless, it captures many of the surface constraints in MA that make it distinct from other dialects of Arabic.

7.3 Syllabification in MA

Just like the ITB syllabification transduction, Γ_{MA} is a transduction from a certain version of the Successor Word Model to the Tree Word Model. Throughout the remainder of this chapter, I will refer to several formulas (concerning natural classes, position within the word, sonority, etc.) that were first defined in Chapters 4–6.

7.3.1 Identifying (Some) Syllable Constituents

The easiest syllable constituents to identify are the nuclei formed from underlying vowels, as stated in (7.16). Epenthetic vowels will, of course, also be nucleic, but I will set that aside for the next section.

$$\text{nuc-V}(x) \stackrel{\text{def}}{=} \text{voc}(x) \quad (7.16)$$

“ x satisfies nuc-V iff it is labeled voc.”

A consonant is an onset if its successor is nucleic; this type of onset segment satisfies $\text{ons}_1(x)$, as defined below in (7.17).

$$\text{ons}_1(x) \stackrel{\text{def}}{=} \text{cons}(x) \wedge \text{nuc-V}(\text{succ}(x)) \quad (7.17)$$

“ x is a nucleus-adjacent onset iff it is a consonant and its successor satisfies nuc-V.”

Note that this only accounts for onsets to underlying vowels. To capture additional generalizations, we must first identify positions in the underlying form where epenthesis will occur.

7.3.2 Identifying Insertion Sites

An insertion site is the position in the word where an epenthetic vowel appears in the SR. For my purposes, the formula $\text{ins-aft}(x)$ (meaning “insert ə after x ”) will be true of any consonant present in the UR which immediately precedes an epenthetic vowel in the SR.

Let us first turn to epenthesis in the word-final syllable. For all words ending in /CCC/ that follow the typical pattern, epenthesis should occur between the final two consonants, as in /tbʔ/ → [t.bəʔ] ‘he followed.’ Epenthesis should also occur after the penultimate C if the word ends in a typical /VCC/ sequence, as in /kadb/ → [ka.dəb] ‘to be lying.’ In both of these cases, we should “insert ə after” the penultimate C—so long as the final consonant is not a member of a geminate or the 1sg suffix. The formula $\text{ins-aft}_1(x)$, defined in (7.18), identifies such insertion sites, which I will call ‘type 1.’

$$\begin{aligned} \text{ins-aft}_1(x) \stackrel{\text{def}}{=} & \text{cons}(x) \wedge \text{cons}(\text{succ}(x)) \wedge \text{fin}(\text{succ}(x)) \\ & \wedge \neg \text{gem}(x, \text{succ}(x)) \wedge \neg \text{1sg}(\text{succ}(x)) \wedge \neg \text{exc}(x) \quad (7.18) \end{aligned}$$

“ x is a type-1 insertion site iff x is a consonant, the successor of x is a consonant and is word-final, x does not form a geminate with its successor, the successor of x is not the 1sg suffix, and x is not marked as a lexical exception.”

By definition, $\text{ins-aft}_1(x)$ does not account for words ending in geminates or the 1sg suffix, as well as nouns that are marked as lexical exceptions. If any of these criteria are met in a word-final /CCC/ sequence, we should “insert ə after” the *antepenultimate*

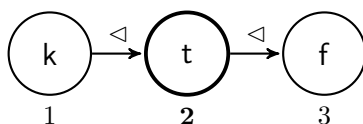
C, as in /klb/ → [kəlb] ‘dog.’ The formula $\text{ins-aft}_2(x)$ is true of the antepenultimate C in such a sequence, as defined in (7.19).

$$\text{ins-aft}_2(x) \stackrel{\text{def}}{=} \text{cons}(x) \wedge \text{cons}(\text{succ}(x)) \wedge \text{cons}(\text{succ}^2(x)) \wedge \text{fin}(\text{succ}^2(x)) \\ \wedge \left(\text{gem}(\text{succ}(x), \text{succ}^2(x)) \vee \text{1sg}(\text{succ}^2(x)) \vee \text{exc}(x) \right) \quad (7.19)$$

“ x is a type-2 insertion site iff x is a consonant, the successor of x is a consonant, the second successor of x is a consonant and is word-final, and at least one of the following is true: the successor of x forms a geminate with the second successor of x ; the second successor of x is the 1sg suffix; or x is marked as a lexical exception.”

To further illustrate the evaluation of $\text{ins-aft}_1(x)$ and $\text{ins-aft}_2(x)$, consider the Successor Word Models for the URs /ktf/ ‘shoulder’ and /klb/ ‘dog,’ illustrated in Figures 7.13 and 7.14, respectively. Insertion sites are bolded.

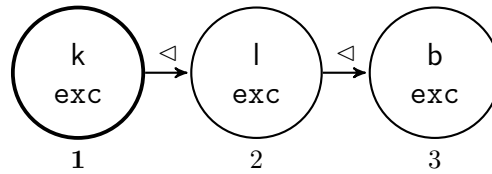
Figure 7.13: The Successor Word Model for /ktf/



In /ktf/, $\text{ins-aft}_1(2) = \text{TRUE}$ because position 2 is a consonant followed by a word-final consonant, and because it doesn’t fall into any of the exceptional categories: it is not a member of a geminate, nor does it precede the 1sg suffix, nor is it marked as a lexical exception. Hence epenthesis should occur after position 2, producing the expected SR, [k.təf].

On the other hand, position 2 in /klb/ does not satisfy ins-aft_1 because it is marked as a lexical exception. Instead, $\text{ins-aft}_2(1) = \text{TRUE}$ because position 1 is a consonant followed by a consonant, which is in turn followed by a word-final consonant, and

Figure 7.14: The Successor Word Model for /klb/



because $\text{exc}(1) = \text{TRUE}$. Epenthesis is therefore predicted to occur after position 1, which is borne out in the SR [kəlb].

Recall that epenthetic vowels are banned in open syllables and that word-internal syllables must have onsets. This entails that every epenthetic vowel must have an onset and a coda. Hence

$$\text{ins-aft}(x) \Rightarrow \text{ons}(x) \wedge \text{cod}(\text{succ}(x))$$

It may not be obvious, but we have stumbled upon a surefire way to find an onset in the final 3 segments of any word. Consider the following possibilities:

1. /...CCV/: The penultimate C is an onset to an underlying V
e.g., /kbda/ → [kəb.da] ‘liver’
2. /...CVC/: The antepenultimate C is an onset to an underlying V
e.g., /ktab/ → [k.tab] ‘book’
3. /...VCC/: The penultimate C is an onset to an epenthetic V
e.g., /wsf-at=k/ → [wəs.fa.tək] ‘she described you’
4. /...CCC/: Either the penultimate or the antepenultimate C is an onset to an epenthetic V
 - (a) e.g., /krkb/ → [kər.kəb] ‘he rolled’

(b) e.g., /krkb-t/ → [kər.kəbt] ‘I rolled’

Once we identify an onset (whether it is an insertion site or not), we can make sense of the material to its left. In particular, a consonant immediately followed by an onset must be a coda. That is,

$$\mathbf{cons}(x) \wedge \mathbf{ons}(\mathit{succ}(x)) \Rightarrow \mathbf{cod}(x)$$

When a sequence of two consonants precedes an onset, sonority plays a role in determining whether the leftmost C becomes an insertion site or becomes part of a complex coda. For instance, compare /krkb=k/ → [kər.k.bək] ‘he rolled you’ to /ktb-l=k/ → [k.təb.lək] ‘he wrote to you.’ Underlyingly, both are of the form C₁C₂C₃C₄C₅, where C₄ satisfies *ins-aft*₁. The difference is in the relative sonority of the 2nd and 3rd consonants. In /krkb=k/, C₂ is more sonorous than C₃, so they form a licit complex coda and a schwa is inserted after C₁. In cases like this, the consonant 3 positions to the left of the onset satisfies *ins-aft*₃(*x*), as defined in (7.20).

$$\begin{aligned} \mathbf{ins-aft}_3(x) \stackrel{\text{def}}{=} & \mathbf{cons}(x) \wedge \mathbf{cons}(\mathit{succ}(x)) \wedge \mathbf{cons}(\mathit{succ}^2(x)) \\ & \wedge \mathbf{ons}(\mathit{succ}^3(x) \wedge >_s(\mathit{succ}(x), \mathit{succ}^2(x))) \quad (7.20) \end{aligned}$$

“*x* is a type-3 insertion site iff *x* is a consonant, the successor of *x* is a consonant, the second successor of *x* is a consonant, the third successor of *x* is an onset, and the successor of *x* is more sonorous than the second successor of *x*.”

Put simply, *x* satisfies *ins-aft*₃ if it is the first C in a C₁C₂C₃C₄ sequence, where C₄ is an onset and C₂ >_sC₃.

In /ktb-l=k/, on the other hand, C₂ and C₃ are of equal sonority (both are obstruents), so they cannot form a licit word-internal coda. In these cases, the consonant 2 positions

to the left of the onset satisfies $\text{ins-aft}_4(x)$, as defined in (7.21), so C_2 and C_3 end up on either side of an epenthetic vowel.

$$\text{ins-aft}_4(x) \stackrel{\text{def}}{=} \text{cons}(\text{pred}(x)) \wedge \text{cons}(x) \wedge \text{cons}(\text{succ}(x)) \\ \wedge \text{ons}(\text{succ}^2(x)) \wedge \leq_s(x, \text{succ}(x)) \quad (7.21)$$

“ x is a type-4 insertion site iff the predecessor of x is a consonant, x is a consonant, the successor of x is a consonant, the second successor of x is an onset, and x is equally or less sonorous than its successor.”

In this case, x satisfies ins-aft_4 if it is the *second* C in a $C_1C_2C_3C_4$ sequence, where C_4 is an onset and $C_2 \leq_s C_3$.

Now we can define the general formula $\text{ins-aft}(x)$ as a disjunction of the four specific cases of insertion outlined above, as in (7.22).

$$\text{ins-aft}(x) \stackrel{\text{def}}{=} \text{ins-aft}_1(x) \vee \text{ins-aft}_2(x) \vee \text{ins-aft}_3(x) \vee \text{ins-aft}_4(x) \quad (7.22)$$

7.3.3 Identifying the Remaining Syllable Constituents

In the general case, an onset is either an insertion site (satisfying $\text{ins-aft}(x)$) or the predecessor of an underlying V (satisfying $\text{ons}_1(x)$). This is formalized in (7.23).

$$\text{ons}(x) \stackrel{\text{def}}{=} \text{ins-aft}(x) \vee \text{ons}_1(x) \quad (7.23)$$

This definition may appear to be circular, as $\text{ins-aft}(x)$ itself refers to $\text{ons}(x)$. However, as established above, an onset can be identified in the final three positions of any word without referring to insertion sites at all. So long as this single onset is identified, $\text{ins-aft}(x)$ can refer to this onset and $\text{ons}(x)$ can refer to insertion sites.

To identify *all* syllabic nuclei, we need to account for the syllabic consonants that sometimes appear in word-initial position, such as in [k.təb] ‘he wrote.’ As defined below in (7.24), $\text{nuc-C}(x)$ is true of a word-initial consonant whose successor is an onset.

$$\text{nuc-C}(x) \stackrel{\text{def}}{=} \text{cons}(x) \wedge \text{init}(x) \wedge \text{ons}(\text{succ}(x)) \quad (7.24)$$

Then the general formula $\text{nuc}(x)$ is defined in (7.25)

$$\text{nuc}(x) \stackrel{\text{def}}{=} \text{nuc-V}(x) \vee \text{nuc-C}(x) \quad (7.25)$$

Note that $\text{nuc}(x)$ is not satisfied by insertion sites. This is because the insertion site itself is an onset; the epenthetic vowel is nucleic, but it is not present in the input, so it cannot be referred to directly using a shorthand formula like $\text{nuc}(x)$.

Finally, codas may be identified in two ways. A consonant satisfies $\text{cod}_1(x)$ iff it does not satisfy $\text{ons}(x)$ and its predecessor is an underlying vowel or an insertion site, as in (7.26).

$$\text{cod}_1(x) \stackrel{\text{def}}{=} \neg \text{ons}(x) \wedge (\text{nuc-V}(\text{pred}(x)) \vee \text{ins-aft}(\text{pred}(x))) \quad (7.26)$$

This is exemplified by the final consonants in [k.təb] and [k.tab].

A consonant satisfies $\text{cod}_2(x)$ iff it satisfies neither $\text{ons}(x)$ nor $\text{nuc-C}(x)$ and its successor satisfies $\text{ons}(x)$. As defined in (7.27), $\text{cod}_2(x)$ is true of the first [k] in [kərk.bək], for example.

$$\text{cod}_2(x) \stackrel{\text{def}}{=} \neg \text{ons}(x) \wedge \neg \text{nuc-C}(x) \wedge \text{ons}(\text{succ}(x)) \quad (7.27)$$

Then the general case, $\text{cod}(x)$, is defined in (7.28).

$$\text{cod}(x) \stackrel{\text{def}}{=} \text{cod}_1(x) \vee \text{cod}_2(x) \tag{7.28}$$

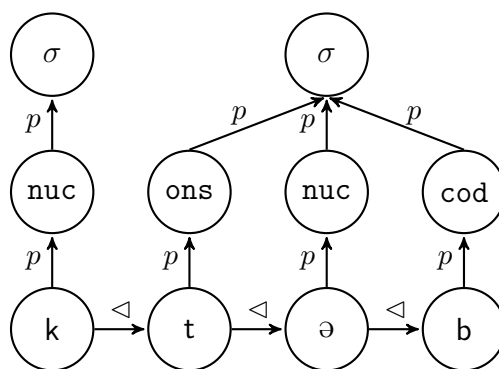
With these shorthand formulas, we are now able to define the transduction from URs to SRs in MA.

7.3.4 The MA Syllabification Transduction

In this section, I develop the transduction Γ_{MA} in a piecemeal fashion, as with previously presented transductions. Recall that, in fact, all formulas in the transduction apply to the input simultaneously.

Copy Sets. As established previously, three copy sets are needed to get from a Successor Model input to a Tree Model output. In Γ_{MA} , an additional copy set is needed for epenthetic vowels. To see why, consider the fully-specified Tree Word Model for [k.təb] ‘he wrote,’ as illustrated in Figure 7.15.

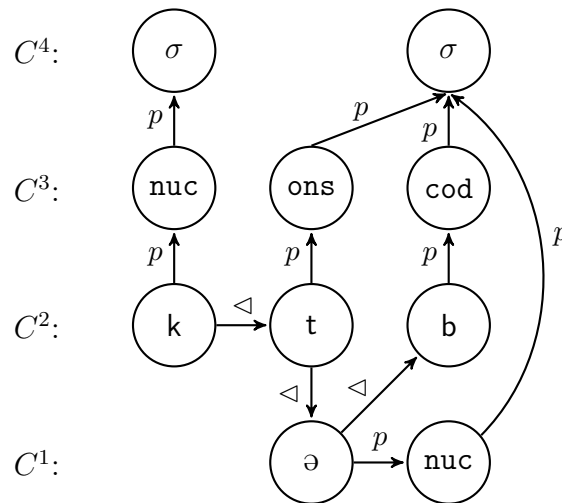
Figure 7.15: $\mathcal{M}_{[k.təb]}^{tree}$



Compare this to the input string ktb , with only 3 positions. Not only are additional positions needed for the syllable constituent nodes dominating underlying segments, but even more positions are needed to accommodate the epenthetic vowel and its

parent nucleus node. For visual clarity and consistency, I use Copy Set 1 for epenthetic vowels and their parent nucleus nodes, Copy Set 2 for underlying segments, Copy Set 3 for underlying segments' syllable constituent nodes, and Copy Set 4 for σ nodes. Figure 7.16 offers a preview of the licensed output of $\Gamma_{MA}(\mathcal{M}_{[k,t\text{ə}b]}^{\triangleleft})$, demonstrating this usage of copy sets.

Figure 7.16: The licensed output of $\Gamma_{MA}(\mathcal{M}_{[k,t\text{ə}b]}^{\triangleleft})$



Note that the graphs represented in Figures 7.15 and 7.16 are mathematically equivalent, although their visual representations appear different at face value.

Binary Relations. Sonority relations are all preserved in Copy Set 2 under Γ_{MA} , as defined in (7.29–7.30).

$$<_s(x^2, y^2) \stackrel{\text{def}}{=} <_s(x, y) \tag{7.29}$$

$$=_s(x^2, y^2) \stackrel{\text{def}}{=} =_s(x, y) \tag{7.30}$$

“The second copy of x in the output is less/equally sonorous than/to the second copy of y iff x is less/equally sonorous than/to y in the input.”

The geminate relation is also preserved in Copy Set 2; see (7.31).

$$gem(x^2, y^2) \stackrel{\text{def}}{=} gem(x, y) \quad (7.31)$$

“The second copy of x forms a geminate with the second copy of y iff x forms a geminate with y in the input.”

Feature Labels. Recall that \mathcal{R}_f is the set of unary relations for phonological features. These feature labels are also preserved in Copy Set 2, as specified by (7.32).

$$(\forall f \in \mathcal{R}_f)[\mathbf{f}(x^2) \stackrel{\text{def}}{=} \mathbf{f}(x)] \quad (7.32)$$

Syllable Constituent Labels and Sigma Nodes. The formulas defined in (7.33-7.36) are analogous to (6.33-6.36) for ITB. The interesting additions are (7.37), which gives the \mathfrak{a} label to the first copy of an insertion site, and (7.38), which gives the **nuc** label to the 1st copy of the *successor* of an insertion site. The choice of successor or predecessor is arbitrary here; the point is this: for every insertion site, we need two corresponding positions in Copy Set 1.

$$\mathbf{nuc}(x^3) \stackrel{\text{def}}{=} \mathbf{nuc}(x) \quad (7.33)$$

$$\mathbf{ons}(x^3) \stackrel{\text{def}}{=} \mathbf{ons}_1(x) \quad (7.34)$$

$$\mathbf{cod}(x^3) \stackrel{\text{def}}{=} \mathbf{cod}(x) \quad (7.35)$$

$$\sigma(x^4) \stackrel{\text{def}}{=} \mathbf{nuc}(x) \vee \mathbf{ins-aft}(\mathit{pred}(x)) \quad (7.36)$$

$$\mathfrak{a}(x^1) \stackrel{\text{def}}{=} \mathbf{ins-aft}(x) \quad (7.37)$$

$$\mathbf{nuc}(x^1) \stackrel{\text{def}}{=} \mathbf{ins-aft}(\mathit{pred}(x)) \quad (7.38)$$

Functions. Similar to the ITB transduction, the parent function must be carefully defined for each copy set, as in (7.39).

$$par(x^3) \stackrel{\text{def}}{=} \begin{cases} x^4 & \Leftarrow \text{nuc}(x) \\ (succ(x))^4 & \Leftarrow \text{ons}_1(x) \\ (pred(x))^4 & \Leftarrow \text{cod}_1(x) \end{cases} \quad (7.39)$$

$$par(x^2) \stackrel{\text{def}}{=} \begin{cases} x^3 & \Leftarrow \text{nuc}(x) \\ x^3 & \Leftarrow \text{ons}_1(x) \\ x^3 & \Leftarrow \text{cod}_1(x) \\ (pred(x))^3 & \Leftarrow \text{cod}_2(x) \end{cases} \quad (7.40)$$

$$par(x^1) \stackrel{\text{def}}{=} (succ(x))^1 \Leftarrow \vartheta(x) \quad (7.41)$$

The successor function is mostly preserved in Copy Set 2, except where insertion sites occur (similar to the Flat-to-Dot Transduction from Chapter 3). If a consonant satisfies $\text{ins-aft}(x)$, the successor of its second copy is its first copy, and the successor of its first copy is the second copy of its successor. This is formalized in (7.42).

$$succ(x^2) \stackrel{\text{def}}{=} \begin{cases} (succ(x))^2 & \Leftarrow \neg \text{ins-aft}(x) \\ x^1 & \Leftarrow \text{ins-aft}(x) \end{cases} \quad (7.42)$$

$$succ(x^1) \stackrel{\text{def}}{=} (succ(x))^2 \Leftarrow \text{ins-aft}(x) \quad (7.43)$$

$$license(x) \stackrel{\text{def}}{=} R(x) \text{ for } R \in \mathcal{R}^{tree} \quad (7.44)$$

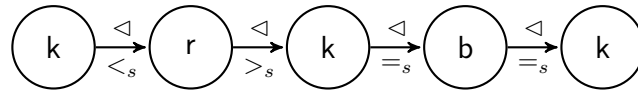
As in previous transductions, (7.45) licenses only labeled nodes.

$$license(x) \stackrel{\text{def}}{=} (\exists R \in \mathcal{R}^{MA})[R(x)] \quad (7.45)$$

7.3.5 Extended Example: /krkbbk/

To illustrate how the transduction works, consider the underlying form /krkb=k/ ‘he rolled you.’ Its word model (in the Successor Model Theory) is illustrated in Figure 7.17.

Figure 7.17: $\mathcal{M}_{krkbbk}^{\triangleleft}$



There are five positions, 1 through 5, and each position has a set of feature labels which I abbreviate with the shorthand segment formulas **k**, **r**, and **b**. Table 7.6 gives the truth values for the formulas relevant to syllable structure.

The first position is less sonorous than the second, which is more sonorous than the third. The remaining segments form a sonority plateau. Position 4 satisfies $\text{ins-aft}_1(x)$ because it is the penultimate C and it doesn’t meet any exclusionary criteria (e.g., being part of a geminate). Position 1 satisfies $\text{ins-aft}_3(x)$ because it is a consonant followed by a consonant, followed by an onset, and because it is more sonorous than its successor. Consequently, positions 1 and 4 are both onsets. Because positions 2 and 5 immediately follow insertion sites, they each satisfy $\text{cod}_1(x)$. Finally, position 3 satisfies $\text{cod}_2(x)$ because its predecessor satisfies $\text{cod}_1(x)$ and its successor satisfies $\text{ons}(x)$.

The resulting output $\Gamma_{MA}(\mathcal{M}_{krkbbk}^{\triangleleft})$ is illustrated in Figure 7.18.

Figure 7.18: The output of $\Gamma_{MA}(\mathcal{M}_{krkbbk}^{\triangleleft})$

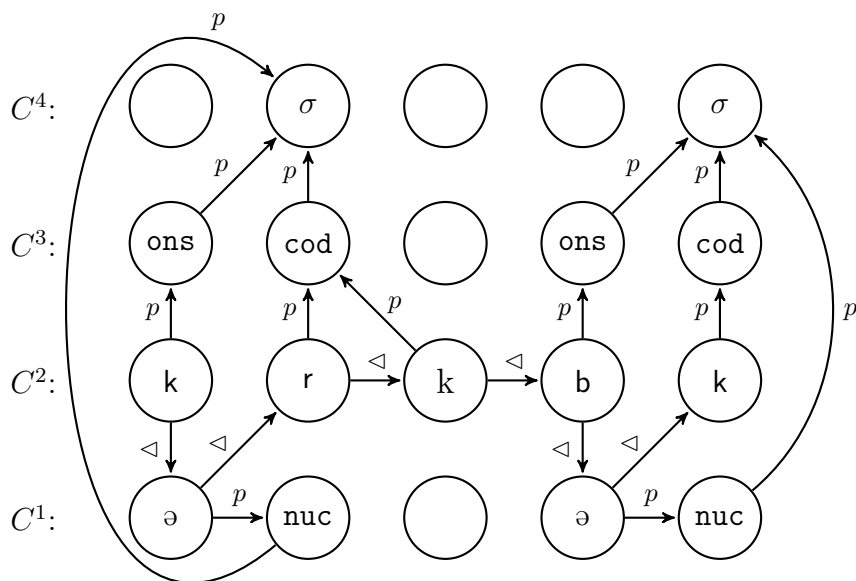


Table 7.6: Truth Table for /krkbbk/

x	1	2	3	4	5
$k(x)$	✓	.	✓	.	✓
$r(x)$.	✓	.	.	.
$b(x)$.	.	.	✓	.
$<_s(x, succ(x))$	✓
$=_s(x, succ(x))$.	.	✓	✓	✓
$>_s(x, succ(x))$.	✓	.	.	.
$gem(x, succ(x))$
$1sg(x)$
$exc(x)$
$ins-aft_1(x)$.	.	.	✓	.
$ins-aft_2(x)$
$ins-aft_3(x)$	✓
$ins-aft_4(x)$
$ins-aft_5(x)$
$nuc(x)$
$ons(x)$	✓	.	.	✓	.
$cod_1(x)$.	✓	.	.	✓
$cod_2(x)$.	.	✓	.	.

Chapter 8

DISCUSSION

In the previous chapters, I have shown how model theory provides insight into a variety of issues in syllable theory. This chapter will further discuss my findings and highlight broader implications. First, I revisit the topic of computational complexity as it relates the results established herein. I then discuss some limitations of the present work with respect to the diversity of attested syllable structures and syllable-related processes. Finally, I suggest directions for future work.

8.1 Computational Complexity

I have focused thus far on the low computational power of QF formulas in comparison with other logical languages, but there are also different degrees of complexity within the class of QF formulas. This section discusses finer distinctions to be made among QF transductions and substructure constraint grammars.

8.1.1 Graph Transductions

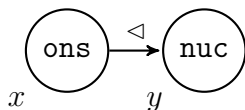
Recall from Chapters 3–4 that locality depends on the ability to compute the result of a formula by examining a connected substructure without unbounded look-ahead/look-back. The exact bound on our look-ahead/look-back window is an important parameter of the computation that has not yet been addressed; call this bound k . Every QF formula has its own k value, depending on the size of the substructure it must evaluate.

For example, consider the formula $\text{ons_1}(x)$ from the Flat-to-Tree transduction, Γ_{ft} . First defined in (4.15), $\text{ons_1}(x)$ is reproduced in (8.1) below.

$$\text{ons_1}(x) \stackrel{\text{def}}{=} \text{ons}(x) \wedge \text{nuc}(\text{succ}(x)) \quad (8.1)$$

To determine if a given position x satisfies ons_1 , we need only look at two connected positions: x and $\text{succ}(x)$. Thus the k value for the formula is 2. This is made even clearer if we consider the substructure that corresponds to $\text{ons_1}(x)$, illustrated in Figure 8.1, which consists of only two nodes.

Figure 8.1: The substructure corresponding to $\text{ons_1}(x)$



One measure of the complexity of a transduction is the largest k value needed to compute any formula in the transduction. Call such a transduction k -QF. For Γ_{ft} , the largest k value depends on the values of n and m (the upper limits on onset and coda length, respectively). All other formulas in the transduction have a k value of 2, so the largest k will be 2, m , or n , whichever is greatest. In this way, Γ_{ft} could differ in computational complexity across languages. For example, complex codas are restricted to two consonants in MA, but can be as long as four consonants in Polish (e.g., *łgarstw* [wgarstf] ‘lie (gen. pl).’ Translating from the Flat Model to the Tree Model for a Polish word is therefore more computationally costly than it is for a word in MA (though the difference is small).

In contrast, Γ_{tf} is unequivocally 2-QF because there are no additional parameters like m and n . This confirms the intuition that there truly is something simpler about

‘flattening’ a hierarchical word model rather than expanding a flat one into a multi-tiered model. It does not indicate that the Flat Model is somehow better than the Tree Model, only that the cost of translating between them is slightly cheaper in one direction than in the other.

It is worth noting that increasing the k value does not increase computational complexity in the same way as upping the power of the logical language. The former introduces a *quantitative* difference: the same method of computation can be used for all k -QF transductions, with computation time increasing with the size of k . In contrast, using a more powerful logical language like FO introduces a *qualitative* difference: the method of scanning the input word model using a window of size k simply does not work for transductions involving quantification, no matter the size of k . More complex computations are required to handle quantification and, therefore, global evaluation.

8.1.2 Substructure Constraint Grammars

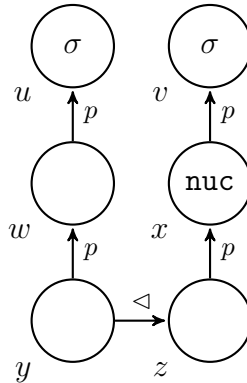
Just like the formulas in graph transductions, the formulas that make up a substructure constraint grammar also have k values: k is simply the number of connected nodes in a given substructure. Moreover, the largest k value of any substructure constraint in the grammar can be used as the overall measure of the grammar’s computational complexity.

In the grammars for both ITB and MA, the substructure constraint with the largest k value is $\kappa_{N IOS}$ “No Internal Onsetless Syllable.” This bans the substructure $\psi_{N IOS}$, first defined in (6.1) and visually represented in Figure 8.2 below. Because $\psi_{N IOS}$ has 6 nodes, its k value is 6. This means it takes a window of size 6 to scan any word model and determine if it represents a well-formed word in ITB or MA.

8.2 Other Issues in Syllable Theory

The case studies in this dissertation touch on only a handful of key topics in syllable theory: atypical nuclei (i.e., syllabic obstruents), geminates, epenthesis, and lexical

Figure 8.2: $\psi_{N IOS}$, the substructure banned by $\kappa_{N IOS}$ “No Internal Onsetless Syllable”



exceptions. Yet there are many more syllable-related structures and phenomena seen in the world’s languages. In particular, three structures not yet considered are ripe for future study: complex nuclei, rhymes, and ambisyllabic segments. Additionally, I will discuss two common phonological processes related to syllabification: deletion and stress assignment.

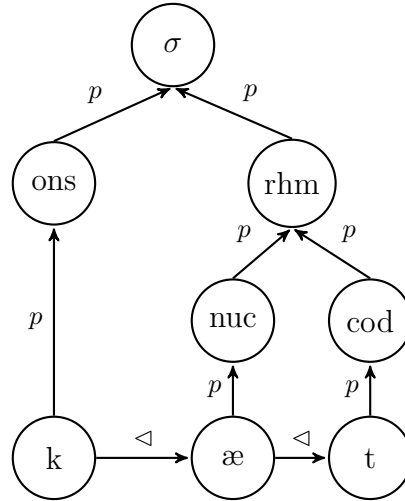
8.2.1 Additional Syllable Structures

Complex nuclei can be accounted for in the same way as I have handled complex syllable margins. No one would argue that nuclei can be of unbounded length, so this just introduces another parameter to go along with n (maximum onset length) and m (maximum coda length).

Rhymes are also straightforward to incorporate; it only takes a small change to the Tree Model to produce the Rhyme Model (a hierarchical word model with the rhyme constituent), as illustrated in Figure 8.3.¹

¹ rhm = rhyme

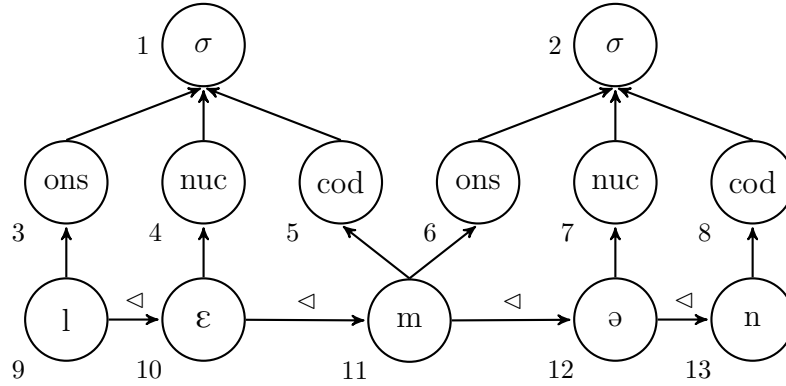
Figure 8.3: The Rhyme Model for *cat* [kæt]



Starting from a flat string, a syllabification transduction that includes the rhyme is analogous to those presented in Chapters 6–7. If my intuitions are correct, the transduction would still be QF. The main differences would be that 4 copy sets are needed instead of 3, and the k value of the transduction would likely be larger.

Handling ambisyllabicity is more challenging. The hierarchical word models used throughout this dissertation include the parent function, which picks out the unique parent of any position. If a segment is ambisyllabic, the traditional view is that it has two parents, one in either syllable (Gussenhoven, 1986; Hayes, 2009; Kahn, 1976). For instance, Figure 8.4 illustrates a tree-like representation of the word *lemon*, in which the [m] is ambisyllabic. Directed edges representing dominance are left unlabeled for the sake of clarity.

Figure 8.4: A tree-like representation of *lemon* [ləmən]



I call this representation ‘tree-like’ because it has hierarchical structure, but the individual syllables are not true trees. The issue is that position 11 has two parents, so there is more than one output for the expression $par(11)$. Whether redefining the parent function as a binary dominance relation, similar to the precedence relation, changes much in the way of computational properties remains to be shown, and I leave the details to future work.

8.2.2 Additional Processes Related to Syllable Structure

Along with epenthesis, which was covered in Chapter 7, processes of deletion and stress assignment are so intertwined with syllable structure that they must be touched upon here. The Dot-to-Flat transduction, Γ_{df} , illustrates the basic approach to deletion. This transduction essentially deletes the position in the input labeled with a dot. Deleting a phoneme instead of a dot is the same; it simply requires ‘skipping’ over an input position when computing the output successor/predecessor functions.

Stress assignment could be handled in this framework by adding the label **stress** to the output word model (or **prim** and **sec** for primary and secondary stress, respectively). Then it would simply be a question of how to determine which positions get this label,

which varies quite a lot across languages (see, e.g., [Goedemans & van der Hulst, 2009](#); [Halle et al., 1994](#); [Heinz, 2007](#); [Van der Hulst, 2014](#)). Bounded stress patterns (e.g., stress falling on the initial syllable, final syllable, penult, etc.) can easily be characterized using local computations, because they only require counting a small number of positions from the word edge ([Rogers et al., 2013](#)). Even certain types of unbounded stress patterns have been characterized with local substructure constraint grammars (e.g., [Strother-Garcia et al., 2017](#)), which suggests that the corresponding stress assignment transduction would be QF. It is unclear whether or not stress patterns that fall outside of these categories could also be captured with QF graph transductions.

8.3 Other Directions for Future Work

One obvious next step would be to develop learning algorithms to learn the patterns and processes described in the previous chapters. Learners have already been established for several types of substructure constraint grammars (e.g., [Chandlee et al., 2019](#); [Heinz, 2010b](#); [Jardine & Heinz, 2016](#); [Strother-Garcia et al., 2017](#)), which could be straightforwardly extended to learn the types of constraints presented here. A more substantial challenge would be to develop a learning algorithm for QF graph transductions. [Chandlee et al. \(2014\)](#) present a learner for Input Strictly Local (ISL) functions, a class of mappings which includes QF graph transductions for conventional string models. How to adapt this learning method to QF transductions over syllabic structures (or arbitrary data structures) is an area of future study.

Another extension of this work would involve writing a program to implement the ITB syllabification transduction on strings and testing its accuracy using data from CoTaSS, the Corpus of Tashlihyt Semi-spontaneous Speech ([Bruggeman & Roettger, 2017](#)). CoTaSS currently consists of 39 minutes of audio, along with time-aligned transcription in Praat’s TextGrid format. This could be readily analyzed (and reformatted as necessary) using Python or another programming language.

It would also be valuable to model a whole language phonology as a graph transduction,

rather than focusing on 1–2 related processes (e.g., syllabification and epenthesis) as I have done here. If a transduction accounting for all phonological processes in a given language could be shown to be QF, it would be a compelling addition to the evidence which suggests that computation in phonology is fundamentally local.

Chapter 9

CONCLUSION

This dissertation presents three main results. I have demonstrated that three popular types of syllable structure representations are notationally equivalent, in a strict mathematical sense—that is, they are QF-bi-interpretable. I have also shown that syllable well-formedness in ITB and MA can be characterized with grammars of local, inviolable substructure constraints. Furthermore, I have shown that the syllabification process in either language can be represented by a QF graph transduction, a formalism that allows for substantially lower computational complexity than those previously proposed. It is worth noting that these findings hold true regardless of which syllable representation type is used, due to the results established in Chapter 4. Taken together, these findings strongly suggest that syllable-based phenomena can be formalized in Model Theory using only local computations.

It is surprising that QF logic, being fundamentally **local**, is sufficient to account for a variety of syllabification processes, because the dominant paradigm in phonology (OT) uses **global** optimization. The results of this dissertation indicate that syllabification is not evidence for global optimization, contrary to the argument made in [Prince & Smolensky \(1993\)](#) on the basis of ITB. The case studies in this dissertation highlight an insight that is obfuscated by grammatical formalisms: the local nature of computing syllable constituency is exactly what precludes the need for quantification (and, therefore, global optimization).

So which aspects of complexity are genuinely linguistic and which ones are by-products of the chosen formalism? Model Theory provides a foundation for studying this question. The minimal power of the logical language needed to define a transduction over

representational structures is a measure of the computational complexity of the mapping. Similarly, the minimal power of the logic used to define a substructure constraint grammar is a measure of how computationally complex it is to decide whether a given word is grammatical. A fruitful avenue of future research in rule-based and OT frameworks alike would be to identify which properties of these grammatical formalisms are responsible for their relatively high computational complexity. It is possible that careful modifications may increase restrictiveness in a way that makes these formalisms equivalent to QF transductions or substructure constraint grammars.

REFERENCES

- Al Ghadi, Abdellatif. 2014. *Moroccan Arabic plurals and the organization of the lexicon*: University Mohammed V, Faculty of Letters and Human Sciences dissertation.
- Baković, Eric. 1999. Assimilation to the unmarked. *Penn Working Papers in Linguistics* 6(1).
- Baković, Eric. 2000. *Harmony, dominance and control*: Rutgers University dissertation.
- Basbøll, Hans. 1999. Syllables in Danish. In Harry van der Hulst & Nancy A. Ritter (eds.), *The syllable: Views and facts*, New York: Mouton de Gruyter.
- Bell, Alan & J. Bybee Hooper. 1978. Issues and evidence in syllabic phonology. In Alan Bell & J. Bybee Hooper (eds.), *Syllables and segments*, 3–22. Amsterdam: North Holland.
- Benhallam, Abderrafi. 1990. Moroccan Arabic syllable structure. *Langues et littératures* 8. 177–191.
- Bird, Steven, John S Coleman, Janet Pierrehumbert & James M Scobbie. 1992. Declarative Phonology. In *Proceedings of the XVth International Congress of Linguists*, Université Laval, Québec.
- Blevins, Juliette. 1995. The syllable in phonological theory. In John A. Goldsmith (ed.), *The handbook of phonological theory*, 206–244. Cambridge, Mass: Blackwell.
- Blevins, Juliette. 2003. The independent nature of phonotactic constraints: an alternative to syllable-based approaches. In Caroline Féry & Ruben van de Vijver (eds.), *The syllable in Optimality Theory*, 375–404. Cambridge University Press.

- Blum, Eileen. 2018. On the locality of vowel harmony over multi-tiered autosegmental representations. Unpublished manuscript.
- Borgstrøm, Carl Hjalmar. 1935. The dialect of Barra in the Outer Hebrides. *Norsk Tidsskrift for Sprogvidenskap* 8. 71–242.
- Borowsky, Toni Jean. 1990. *Topics in the lexical phonology of English*. New York: Garland.
- Boudlal, Abdelaziz. 2001. *Constraint interaction in the phonology and morphology of Casablanca Moroccan Arabic*: Mohammed V University dissertation.
- Boudlal, Abdelaziz. 2006/2007. Sonority-driven schwa epenthesis in Moroccan Arabic. *Languages and Linguistics* 18/19. 59–81.
- Breen, Gavan & Rob Pensalfini. 1999. Arrernte: A language with no syllable onsets. *Linguistic Inquiry* 30(1). 1–25.
- Broselow, Ellen I. 1976. *The phonology of Egyptian Arabic*: University of Massachusetts, Amherst dissertation.
- Browman, Catherine P & Louis Goldstein. 1995. Gestural syllable position effects in American English. *Producing speech: Contemporary issues* 19–33.
- Bruggeman, Anna & Timo Benjamin Roettger. 2017. CoTaSS: Corpus of Tashlhiyt Semi-spontaneous Speech. Online database. cotass.uni-koeln.de/.
- Büchi, J. Richard. 1960. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly* 6(1-6). 66–92.
- Calabrese, Andrea. 1988. *Towards a theory of phonological alphabets*: Massachusetts Institute of Technology dissertation.
- Chandlee, Jane. 2014. *Strictly local phonological processes*: University of Delaware dissertation.

- Chandlee, Jane, Rémi Eyraud & Jeffrey Heinz. 2014. Learning strictly local subsequential functions. *Transactions of the Association for Computational Linguistics* 2. 491–503.
- Chandlee, Jane, Remi Eyraud, Jeffrey Heinz, Adam Jardine & Jonathan Rawski. 2019. Learning with partially ordered representations. *arXiv preprint arXiv:1906.07886* .
- Chandlee, Jane & Jeffrey Heinz. 2018. Strict locality and phonological maps. *Linguistic Inquiry* 49(1). 23–60.
- Chandlee, Jane & Steven Lindell. 2016. Local languages. Paper presented at the 4th Workshop on Natural Language and Computer Science, in affiliation with LICS at Columbia University, NY.
- Chandlee, Jane & Steven Lindell. to appear. A logical characterization of input strictly local functions. In Jeffrey Heinz (ed.), *Doing computational phonology*, Oxford University Press.
- Chen, Matthew Y & William S-Y Wang. 1975. Sound change: actuation and implementation. *Language* 255–281.
- Chomsky, Noam & Morris Halle. 1965. Some controversial questions in phonological theory. *Journal of linguistics* 1(2). 97–138.
- Chomsky, Noam & Morris Halle. 1968. *The sound pattern of English*. New York: Harper & Row.
- Clements, George. 1990. The role of the sonority cycle in core syllabification. In John Kingston & Mary Beckmann (eds.), *Papers in laboratory phonology*, vol. 1, 283–333. Cambridge: Cambridge University Press.
- Clements, George N. 1986. Syllabification and epenthesis in the Barra dialect of Gaelic. In K. Bogers, M. Maus & H. van der Hulst (eds.), *The phonological representation of suprasegmentals*, 317–336. Dordrecht: Foris.

- Clements, George N. 1997. Berber syllabification: derivations or constraints? In *Derivations and constraints in phonology*, 289–330. Clarendon Press Oxford.
- Clements, George N & Samuel Jay Keyser. 1983. *CV phonology: A generative theory of the syllable*. Cambridge, Mass: MIT Press.
- Coleman, John. 1998. *Phonological representations: their names, forms and powers*. Cambridge University Press.
- Côté, Marie-Hélène. 2000. *Consonant cluster phonotactics: a perceptual approach*: Massachusetts Institute of Technology dissertation.
- Courcelle, Bruno. 1994. Monadic second-order definable graph transductions: a survey. *Theoretical Computer Science* 126(1). 53–75.
- Courcelle, Bruno & Joost Engelfriet. 2012. *Graph structure and monadic second-order logic: a language-theoretic approach*, vol. 138. Cambridge University Press.
- Daland, Robert, Bruce Hayes, James White, Marc Garellek, Andreas Davis & Ingrid Normann. 2011. Explaining sonority projection effects. *Phonology* 28(197). 197–234.
- Danis, Nick & Adam Jardine. 2019. Q-theory representations are logically equivalent to autosegmental representations. *Proceedings of the Society for Computation in Linguistics* 2(1). 29–38.
- Davis, Stuart Michael. 1985. *Topics in syllable geometry (phonology)*: The University of Arizona dissertation.
- Dell, François & Mohamed Elmedlaoui. 1985. Syllabic consonants and syllabification in Imdlawn Tashlhiyt Berber. *Journal of African Languages and Linguistics* 7. 105–130.
- Dell, François & Mohamed Elmedlaoui. 1988. Syllabic consonants in Berber: Some new evidence. *Journal of African Languages and Linguistics* 10(1). 1–17.

- Dell, François & Mohamed Elmedlaoui. 2002. *Syllables in Tashlhiyt Berber and in Moroccan Arabic*, vol. 2. Dordrecht, The Netherlands: Berlin: Springer.
- Draper, MH, Peter Ladefoged & David Whitteridge. 1959. Respiratory muscles in speech. *Journal of Speech, Language, and Hearing Research* 2(1). 16–27.
- Duanmu, San. 2009. *Syllable structure: The limits of variation*. Oxford University Press.
- Enderton, Herbert B. 2001. *A mathematical introduction to logic*. Academic Press 2nd edn.
- Engelfriet, Joost & Hendrik Jan Hoogeboom. 2001. MSO definable string transductions and two-way finite-state transducers. *ACM Transactions on Computational Logic (TOCL)* 2(2). 216–254.
- Ennaji, Moha. 2005. *Multilingualism, cultural identity, and education in Morocco*. Berlin: Springer.
- Fagin, R., L.J. Stockmeyer & M.Y. Vardi. 1995. On monadic NP vs monadic co-NP. *Information and Computation* 120(1). 78 – 92.
- Finley, Sara. 2008. *Formal and cognitive restrictions on vowel harmony*: Johns Hopkins University dissertation.
- Finley, Sara & William Badecker. 2008. Analytic biases for vowel harmony languages. In *Wccfl*, vol. 27, 168–176.
- Frampton, John. 2011. GDE syllabification: A generalization of Dell and Elmedlaoui’s syllabification algorithm. *The Linguistic Review* 28(3). 241–279.
- Frank, Robert & Giorgio Satta. 1998. Optimality Theory and the generative complexity of constraint violability. *Computational Linguistics* 24(2). 307–315.

- Gafos, Adamantios I, Philip Hoole, Kevin Roon, Chakir Zeroual, Cécile Fougeron, Barbara Kühnert, Mariapaola D'Imperio & Nathalie Vallée. 2010. Variation in overlap and phonological grammar in Moroccan Arabic clusters. *Laboratory Phonology X, Mouton de Gruyter, Berlin/New York* 657–698.
- Gainor, Brian, Regine Lai & Jeffrey Heinz. 2012. Computational characterizations of vowel harmony patterns and pathologies. In *The proceedings of the 29th west coast conference on formal linguistics*, 63–71.
- Garcia, Pedro, Enrique Vidal & José Oncina. 1990. Learning locally testable languages in the strict sense. In *Proceedings of the workshop on algorithmic learning theory*, 325–338.
- Gay, Thomas. 1978. Articulatory units: Segments or syllables. *Syllables and segments* 121–132.
- Giegerich, Heinz J. 1992. *English phonology: An introduction*. Cambridge: Cambridge University Press.
- Giles, Stephen B & Kenneth L Moll. 1975. Cinefluorographic study of selected allophones of English /l/. *Phonetica* 31(3-4). 206–227.
- Gimson, Alfred Charles. 1970. *An introduction to the pronunciation of English*. London: Arnold.
- Goedemans, Rob & Harry van der Hulst. 2009. Stresstyp: A database for word accentual patterns in the world's languages. *The use of databases in cross-linguistics research* 235–282.
- Goldsmith, John. 2011. The syllable. In John A. Goldsmith, Jason Riggle & Alan C. L. Yu (eds.), *The blackwell handbook of phonological theory*, vol. 2, 164–196. Wiley-Blackwell.

- Graf, Thomas. 2010. Comparing incomparable frameworks: A model theoretic approach to phonology. *University of Pennsylvania Working Papers in Linguistics* 16(2). Article 10.
- Gussenhoven, Carlos. 1986. English plosive allophones and ambisyllabicity. *Gramma* 10(2). 119–141.
- Halle, Morris & William Idsardi. 1995. General properties of stress and metrical structure. In John A. Goldsmith (ed.), *The handbook of phonological theory*, 403–443. Cambridge, Mass: Blackwell.
- Halle, Morris, William Idsardi & John A Goldsmith. 1994. General properties of stress and metrical structure. In Eric Sven Ristad (ed.), *Language computations: DIMACS workshop on human language, march 20-22, 1992* Center for Discrete Mathematics and Theoretical Computer Science New Brunswick, NJ: DIMACS series in discrete mathematics and theoretical computer science, American Mathematical Society.
- Harms, Robert Thomas. 1964. *Finnish structural sketch*. Bloomington: Indiana University.
- Harris, James W. 1969. *Spanish phonology*. Cambridge, MA: Massachusetts Institute of Technology Press.
- Harris, James W. 1983. Syllable structure and stress in Spanish. a nonlinear analysis. *Linguistic Inquiry Monographs Cambridge, Mass.* 8. 1–158.
- Hayes, Bruce. 2009. *Introductory phonology*. Wiley-Blackwell.
- Heath, Jeffrey. 1987. *Ablaut and ambiguity: Phonology of a Moroccan Arabic dialect*. SUNY Press.
- Heath, Jeffrey. 1997. Moroccan Arabic phonology. In Alan S. Kaye (ed.), *Phonologies of Asia and Africa (including the Caucasus)*, vol. 1, 205–217. Eisenbrauns.

- Heinz, Jeffrey. 2007. *The inductive learning of phonotactic patterns*: University of California, Los Angeles dissertation.
- Heinz, Jeffrey. 2009. On the role of locality in learning stress patterns. *Phonology* 26(2). 303–351.
- Heinz, Jeffrey. 2010a. Learning long-distance phonotactics. *Linguistic Inquiry* 41(4). 623–661.
- Heinz, Jeffrey. 2010b. String extension learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, 897–906. Uppsala, Sweden: Association for Computational Linguistics.
- Heinz, Jeffrey. 2011a. Computational phonology part I: Foundations. *Language and Linguistics Compass* 5(4). 140–152.
- Heinz, Jeffrey. 2011b. Computational phonology part II: Grammars, learning, and the future. *Language and Linguistics Compass* 5(4). 153–168.
- Heinz, Jeffrey. 2018. The computational nature of phonological generalizations. In Larry Hyman & Frans Plank (eds.), *Phonological typology*, 126–195. Mouton.
- Heinz, Jeffrey & William Idsardi. 2013. What complexity differences reveal about domains in language. *Topics in cognitive science* 5(1). 111–131.
- Heinz, Jeffrey & Regine Lai. 2013. Vowel harmony and subsequentiality. In Andras Kornai & Marco Kuhlmann (eds.), *Proceedings of the 13th meeting on the mathematics of language (mol 13)*, 52–63. Sofia, Bulgaria.
- Heinz, Jeffrey, Chetan Rawal & Herbert G. Tanner. 2011. Tier-based strictly local constraints for phonology. In *Proceedings of the 49th annual meeting of the association for computational linguistics*, 58–64. Portland, Oregon, USA: Association for Computational Linguistics.

- Heinz, Jeffrey & Kristina Strother-Garcia. to appear. Cluster reduction in Tibetan. In Jeffrey Heinz (ed.), *Doing computational phonology*, Oxford University Press.
- Hooper, Joan B. 1972. The syllable in phonological theory. *Language* 525–540.
- Hooper, Joan B. 1976. *An introduction to natural generative phonology*. New York: Academic Press.
- Van der Hulst, Harry. 2014. *Word stress: Theoretical and typological issues*. Cambridge University Press.
- Hyman, Larry M. 1983. Are there syllables in gokana. *Current approaches to African linguistics* 2. 171–179.
- Itô, Junko. 1988. *Syllable theory in prosodic phonology*. New York: Garland Press.
- Itô, Junko. 1989. A prosodic theory of epenthesis. *Natural Language & Linguistic Theory* 7.
- Jakobson, Roman. 1962. *Selected writings*. The Hague: Mouton.
- Jardine, Adam. 2016. *Locality and non-linear representations in tonal phonology*: University of Delaware dissertation.
- Jardine, Adam. 2017. The local nature of tone-association patterns. *Phonology* 34(2). 363–384.
- Jardine, Adam. 2019. The expressivity of autosegmental grammars. *Journal of Logic, Language and Information* 28(1). 9–54.
- Jardine, Adam & Jeffrey Heinz. 2015. A concatenation operation to derive autosegmental graphs. In *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, 139–151. Chicago, USA.
- Jardine, Adam & Jeffrey Heinz. 2016. Learning tier-based strictly 2-local languages. *Transactions of the Association for Computational Linguistics* 4. 87–98.

- Jespersen, Otto. 1904. *Lehrbuch der Phonetik*. Leipzig and Berlin: B. G. Teubner.
- Johnson, C Douglas. 1972. *Formal aspects of phonological description*. The Hague: Mouton.
- Kager, René. 1999. *Optimality Theory*. Cambridge: Cambridge University Press.
- Kahn, Daniel. 1976. *Syllable-based generalizations in English phonology*: Massachusetts Institute of Technology dissertation.
- Kaplan, Ronald M & Martin Kay. 1994. Regular models of phonological rule systems. *Computational linguistics* 20(3). 331–378.
- Karttunen, Lauri. 1993. Finite-state constraints. *The last phonological rule* 173–194.
- Karttunen, Lauri. 1998. The proper treatment of optimality in computational phonology: plenary talk. In *Proceedings of the international workshop on finite state methods in natural language processing*, 1–12. Association for Computational Linguistics.
- Kawasaki, Haruko. 1986. Phonetic explanation for phonological universals: The case of distinctive vowel nasalization. *Experimental phonology* 81–103.
- Keegan, John M. 1986. The role of syllabic structure in the phonology of Moroccan Arabic. *Current Approaches to African Languages and Linguistics* 3(6). 209–226.
- Kenstowicz, M. J. 1994. *Phonology in generative grammar*. Cambridge, Mass.: Blackwell.
- Kent, Raymond D & Fred D Minifie. 1977. Coarticulation in recent speech production models. *Journal of Phonetics* 5(2). 115–133.
- Kent, Raymond D & Charles Read. 1992. *The acoustic analysis of speech*. San Diego: Singular Publishing Group.
- Kiparsky, Paul. 2003. Syllables and moras in Arabic. In *The syllable in optimality theory*, 147–182. Cambridge University Press.

- Kohler, Klaus J. 1966. Is the syllable a phonological universal? *Journal of Linguistics* 2(2). 207–208.
- Kozhevnikov, Valerii Aleksandrovich & Liudmila Andreevna Chistovich. 1965. *Speech: Articulation and perception*. Washington: Joint Publications Research Service.
- Krakow, Rena A. 1999. Physiological organization of syllables: a review. *Journal of Phonetics* 27(1). 23–54.
- Krakow, Rena Arens. 1989. *The articulatory organization of syllables: A kinematic analysis of labial and velar gestures*: Yale University dissertation.
- Lahrouchi, Mohamed. 2018. Syllable structure and vowel/zero alternations in Moroccan Arabic and Berber. In *The routledge handbook of african linguistics*, 168–180. New York: Routledge. doi:10.4324/9781315392981.
- Laks, Bernard. 2003. Saussure’s phonology. *Take Danish for instance. Linguistic studies in honour of Hans Basbøll* 199–211.
- Lehiste, Ilse. 1960. An acoustic–phonetic study of internal open juncture. *Phonetica* 5(1). 5–54.
- Locke, John L. 1983. *Phonological acquisition and change*. Academic Press.
- Lombardi, Linda. 1999. Positional faithfulness and voicing assimilation in Optimality Theory. *Natural Language & Linguistic Theory* 17(2). 267–302.
- Manuel, Sharon Y & Eric Vatikiotis-Bateson. 1988. Oral and glottal gestures and acoustics of underlying /t/ in English. *The Journal of the Acoustical Society of America* 84(S1). S84–S84.
- McCarthy, John J. 1986. OCP effects: Gemination and antigemination. *Linguistic inquiry* 207–263.

- McCawley, James D. 1968. *The phonological component of a grammar of Japanese 2*. The Hague: Mouton.
- Nespor, Marina & Irene Vogel. 1986. *Prosodic phonology*, vol. 28. Dordrecht, Holland: Foris.
- Ohala, John J. & Haruko Kawasaki. 1984. Prosodic phonology and phonetics. *Phonology* 1. 113–127.
- Osthoff, Hermann & Karl Burgmann. 1878. *Morphologische Untersuchungen auf dem Gebiete der indogermanischen Sprachen*. Leipzig: Hirzel.
- Paradis, Carole. 1988. On constraints and repair strategies. *The Linguistic Review* 6. 71–97.
- Payne, Amanda. 2017. All dissimilation is computationally subsequential: Supplemental material. *Language* 93(4).
- Pike, Kenneth L & Eunice Victoria Pike. 1947. Immediate constituents of Mazateco syllables. *International Journal of American Linguistics* 13(2). 78–91.
- Potts, Christopher & Geoffrey K. Pullum. 2002. Model theory and the content of OT constraints. *Phonology* 19. 361–393.
- Prince, Alan & Paul Smolensky. 1993. *Optimality Theory: Constraint interaction in generative grammar*. Malden, Mass: Blackwell.
- Pullum, Geoffrey K. 2007. The evolution of model-theoretic frameworks in linguistics. In James Rogers & Stephan Kepser (eds.), *Model-theoretic syntax at 10*, 1–10. Dublin, Ireland.
- Rawski, Jon. 2018. Subregular complexity across speech and sign. *Proceedings of the Society for Computation in Linguistics* 1(1). 225–226.

- Reiss, Charles. 2008. Constraining the learning path without constraints, or the OCP and NOBANANA. In Bert Vaux & Andrew Nevins (eds.), *Rules, constraints, and phonological phenomena*, 252–302. Oxford University Press.
- Ridouane, Rachid. 2016. Leading issues in tashlhiyt phonology. *Language and Linguistics Compass* 10(11). 644–660.
- Riggle, Jason Alan. 2004. *Generation, recognition, and learning in finite state Optimality Theory*: University of California, Los Angeles dissertation.
- Rogers, James, Jeffrey Heinz, Margaret Fero, Jeremy Hurst, Dakotah Lambert & Sean Wibel. 2013. Cognitive and sub-regular complexity. In Glyn Morrill & Mark-Jan Nederhof (eds.), *Formal grammar*, vol. 8036 Lecture Notes in Computer Science, 90–108. Berlin: Springer.
- Rogers, James & Geoffrey K Pullum. 2011a. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20(3). 329–342.
- Rogers, James & Geoffrey K. Pullum. 2011b. Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20. 329–342.
- Rozenberg, Grzegorz & Arto Salomaa. 2012. *Handbook of formal languages: Word language grammar*, vol. 1. Berlin: Springer.
- Saporta, Sol & Heles Contreras. 1962. *A phonological grammar of Spanish*. University of Washington Press.
- Saussure, Ferdinand de. 1916. *Cours de linguistique générale*. Paris: Payot.
- Sayed, Abdelrahman Ahmed. 1982. *The phonology of Moroccan Arabic: A generative phonological approach*: University of Texas, Austin dissertation.

- Schachter, Paul & Victoria Fromkin. 1968. A phonology of Akan: Akuapem, Asante, Fante. *UCLA Working Papers in Phonetics* 9.
- Schourup, Lawrence C. 1973. A cross-language study of vowel nasalization. *Ohio State University Working Papers in Linguistics* 15. 190–221.
- Scobbie, James M. 1991. Towards Declarative Phonology. *Edinburgh Working Papers in Cognitive Science* 7. 1–27.
- Scobbie, James M, John S Coleman & Steven Bird. 1996. *Key aspects of Declarative Phonology*. Salford: European Studies Research Institute.
- Selkirk, Elisabeth O. 1980. The role of prosodic categories in English word stress. *Linguistic inquiry* 11(3). 563–605.
- Selkirk, Elisabeth O. 1984. On the major class features and syllable theory. In M. Halle, M. Aronoff & R. T. Oehrle (eds.), *Language sound structure: Studies in phonology*, Cambridge, Mass: MIT Press.
- Shaw, Jason, Adamantios Gafos, Philip Hoole & Chakir Zeroual. 2009. Syllabification in Moroccan Arabic: evidence from patterns of temporal stability in articulation. *Phonology* 26(1). 187–215.
- Shoenfield, Joseph R. 1967. *Mathematical logic*, vol. 21. Reading: Addison-Wesley.
- Sievers, Eduard. 1881. *Grundzuge der Phonetik*. Leipzig: Breitkopf and Hartel.
- Sommer, Bruce. 1981. The shape of Kunjen syllables. In D. L. Goyvaerts (ed.), *Phonology in the 1980s*, Ghent: E. Story-Scientia.
- Steriade, Donca. 1998. Phonetics in phonology: the case of laryngeal neutralization. *UCLA Working Papers in Phonology* 3. 25–146.
- Steriade, Donca. 1999. 9 alternatives to the syllabic interpretation of consonantal phonotactics. In O. Fujimura, B. Joseph & B. Palek (eds.), *Proceedings of the 1998 linguistics and phonetics conference*, 205–242. The Karolinum Press.

- Stetson, RH. 1951. *Motor phonetics: a study of speech movements in articulation*. Amsterdam: North Holland.
- Strother-Garcia, Kristina. 2018. Imdlawn Tashlhiyt Berber syllabification is quantifier-free. In *Proceedings of the society for computation in linguistics*, vol. 1, .
- Strother-Garcia, Kristina & Jeffrey Heinz. 2017. Logical foundations of syllable representations. Poster presented at the 5th Annual Meeting on Phonology, New York University, New York City.
- Strother-Garcia, Kristina, Jeffrey Heinz & Hyun Jin Hwangbo. 2017. Using model theory for grammatical inference: A case study from phonology. In Sicco Verwer, Menno van Zaanen & Rick Smetsers (eds.), *Proceedings of the 13th International Conference on Grammatical Inference (ICGI)*, vol. 57 Proceedings of Machine Learning Research, 66–78. Delft: PMLR.
- Thomas, Wolfgang. 1982. Classifying regular events in symbolic logic. *Journal of Computer and System Sciences* 25(3). 360–376.
- Vennemann, Theo. 1968a. *German phonology*: University of California, Los Angeles dissertation.
- Vennemann, Theo. 1968b. On the use of paradigmatic information in a competence rule of modern German phonology. In *Proceedings of the 30th annual LSA summer meeting, Urbana, IL*, .
- Vogel, Irene. 1977. *The syllable in phonological theory: with special reference to Italian*: Stanford University dissertation.
- Vu, Mai H, Ashkan Zehfroosh, Kristina Strother-Garcia, Michael Sebok, Jeffrey Heinz & Herbert G Tanner. 2018. Statistical relational learning with unconventional string models. *Frontiers in Robotics and AI* 5. 76.

Whitney, William Dwight. 1874. *Oriental and linguistic studies: The east and west; religion and mythology; orthography and phonology; Hindu astronomy*, vol. 2. New York: Scribner, Armstrong, and Co.