To appear (2023) in: Syntax-Prosody in Optimality Theory: Theory and Analyses
Edited by Jennifer Bellik, Junko Ito, Nick Kalivoda, Armin Mester
Equinox Publishing
https://www.equinoxpub.com/home/syntax-prosody/

1 Syntax-Prosody in Optimality Theory (SPOT)

Jennifer Bellik, Junko Ito, Nick Kalivoda, and Armin Mester

(jbellik@ucsc.edu, ito@ucsc.edu, nick.kalivoda@gmail.com, mester@ucsc.edu)

1. Introduction

In syntax-prosody mapping, a syntactic tree S maps to a prosodic tree P, and the constituent structure of P defines the domains of phonological processes. While P is typically based on S, the two trees are not always isomorphic; mismatches can occur, and these are often attributable to purely phonological pressures like eurhythmy and length. Syntax-prosody mapping is therefore particularly amenable to analysis within Optimality Theory (OT; Prince and Smolensky 1993/2004), with mapping constraints (analogous to faithfulness constraints) favoring perfect syntax-prosody isomorphism, and prosodic markedness constraints that sometimes demand deviation from perfect matching. Syntactic trees serve as inputs, and are mapped to outputs in the form of prosodic trees. The study of syntax-prosody in OT has proved extremely fruitful, and this book continues in this research tradition.

As our understanding of Optimality Theory has advanced, increasing emphasis has been placed on the need to consider all candidates admitted by GEN, and to evaluate them against all constraints contained in CON (Prince 2017a,b among many others). In syntax-prosody mapping, achieving this level of rigor has long proved daunting, since the number of possible prosodic outputs for a given input can be immense. Depending on the details of GEN, there may be thousands of candidates for how to prosodify even a simple sentence.

The papers in this book rise to meet this challenge by using an open source JavaScript application called SPOT ("Syntax-Prosody in Optimality Theory"), developed as part of a collaborative research project with the same name in the Linguistics Department at the University of California, Santa Cruz, funded by National Science Foundation Grant #1749368. The SPOT application allows the user to rigorously investigate mapping from syntactic to prosodic structure by automating the generation of syntactic and prosodic trees according to customizable GEN specifications, and by automating constraint violation counting, given a customizable constraint set CON. SPOT news and information are available at the SPOT website (http://spot.sites.ucsc.edu), which links to the web application and codebase. SPOT produces violation tableaux which can be viewed in the browser or downloaded and imported into OTWorkplace (Prince, Merchant, and Tesar 2007-2020) or other OT tools for further analysis.

This chapter is structured as follows: §2 provides background information on syntax-prosody mapping; §3 introduces Optimality Theory; and §4 provides an overview of the book.

2. Syntax-Prosody background

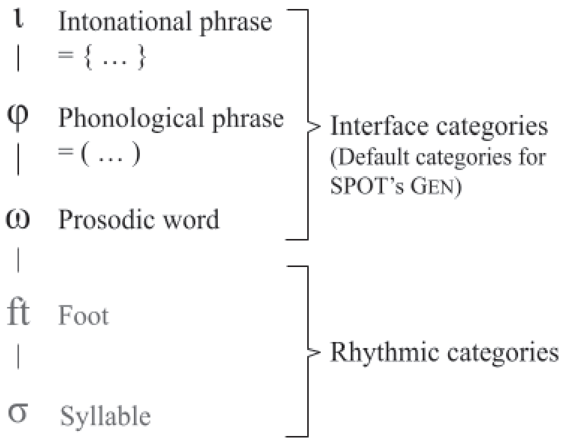
This section discusses the theoretical backdrop of syntax-prosody mapping, covering Prosodic Hierarchy Theory (§2.2) and Align and Match Theory (§2.2). It also introduces tree vocabulary and labeled diagrams in §2.3, and forms of syntax-prosody mismatches in §2.4.

2.1. *Prosodic Hierarchy Theory*

There are two major approaches to the syntax-phonology interface: direct and indirect reference theories (Inkelas 1990, see Elordieta 2008 for detailed summary and discussion). According to direct reference theories, phonological processes are directly sensitive to syntactic structure. Indirect reference theories, by contrast, maintain that the effect of syntax on phonology is mediated by purely prosodic structures derived from syntactic trees, but not necessarily isomorphic to them, and impoverished of much information such as lexical category and other purely syntactic features.

The papers in this book assume indirect reference, and in particular Prosodic Hierarchy Theory (Selkirk 1980 and subsequent work, see Elordieta 2008). The main claim of this theory is that phonological representations are organized in hierarchical tree structures built from a small set of prosodic categories. While the precise inventory of prosodic categories is a matter of debate, we follow Ito and Mester (2009a) in restricting them to the intonational phrase (ι), the phonological phrase (φ), the prosodic word (ω), the foot (ft), and the syllable (σ). The first three of these, ι, φ, and ω, are interface categories which reflect aspects of syntactic structure, while the foot and the syllable are sub-word rhythmic categories. The papers in this book focus almost exclusively on the interface categories.

(1)



As its name implies, the Prosodic Hierarchy imposes a hierarchical structure on the inventory of prosodic categories: the intonational phrase ι is higher than the phonological phrase φ, which in turn is higher than the prosodic word ω. According to Layeredness, which we take to be an inviolable principle of phonological representation, a prosodic category cannot dominate a category that is higher on the prosodic hierarchy. This rules out structures in which an ω dominates a φ or ι, or a φ dominates an ι.

Layeredness is one component of what Selkirk (1984) dubbed the Strict Layer Hypothesis (SLH; also known as Strict Layering), which imposes additional requirements on prosodic trees. According to Strict Layering, the principle of Exhaustivity requires every node of level *i* of the prosodic hierarchy to be dominated by a node of level *i*+1 (unless *i* is itself the highest level in the hierarchy, in which case a node of level *i* is the root node of the tree). With a hierarchy ι > φ > ω > ft > σ, Exhaustivity requires that σ be contained in a ft; every ft in an ω; every ω in a φ; and every φ in an ι. A second component of the SLH is a ban on prosodic recursion, Nonrecursivity. This principle demands that no node dominate another node of the same prosodic category; no φ may dominate a φ, etc.

While Layeredness is still broadly assumed, Exhaustivity and Nonrecursivity have more recently been demoted from their status as inviolable conditions on representations, and have instead become violable constraints (Truckenbrodt 1999, Ito and Mester 2009a). In Optimality-Theoretic (OT) terms, they are no longer taken to be conditions on GEN, but constraints in CON, the universal constraint set. Thus, while a prosodic category of level *i* is still unable to dominate one of a category greater than *i* (Layeredness), it may immediately dominate one at a level less than *i*–1 (violating Exhaustivity) or at level *i* (violating Nonrecursivity). For instance, an ω may be immediately dominated by an ι instead of a φ; a φ may dominate another φ; etc. The proposal that such prosodic trees are admissible is known as the Weak Layer Hypothesis, or Weak Layering (Ito and Mester 2003).

Many papers in this volume adopt some form of Weak Layering when defining GEN for the OT systems they consider. Details of implementation vary, but are always stated precisely in the definition of GEN. Cao, Bibbs, and Bellik (Chapter 9) are a notable exception, as they explore several systems with Strict Layering in their analysis of Xiamen Chinese. Tarlov (Chapter 3) also studies two systems with Strict Layering, comparing them to minimally different systems with Weak Layering. Shingler and Bellik (Chapter 2) consider several gradations between Strict and Weak Layering, which are possible settings for GEN in the SPOT application (Bellik, Bellik and Kalivoda 2015-2021).

While Prosodic Hierarchy Theory posits a universal hierarchy of prosodic categories (or at least *available* prosodic categories), these categories have different phonological reflexes in different languages. According to Selkirk (1980), prosodic constituents are domains for phonological processes, which can be classified as domain span rules, domain juncture rules, and domain limit rules. These are phonological processes which may occur throughout a domain, at the border between two domains of the same category, or simply at the left or right edge of a domain. Each category of the prosodic hierarchy has been identified as the domain for a wide range of phonological phenomena in diverse languages. For instance, Tarlov (Chapter 3) follows Bickmore (1989, 1990) in identifying the φ as the domain of high tone deletion in Kinyambo, and Cao, Bibbs, and Bellik (Chapter 9) follow Chen (1987) in treating the φ in Xiamen Chinese as the domain of tone sandhi. Sometimes prosodic structure is diagnosed not by an overt segmental or tonal process like these, but by the alignment of some element to a domain edge. This is exemplified by Bibbs' (Chapter 8) treatment of weak pronoun placement in Chamorro, based on work by Chung (2003).

The advent of Weak Layering, and in particular the loosening of the Nonrecursivity requirement of Strict Layering, opened the door to analyses in terms of recursive prosodic domains. In a structure like $(_{\varphi}\ (_{\varphi}\ \omega_1\ \omega_2)\ \omega_3)$, a phonological process that marks the left edge of a φ will apply to ω_1, while a process that marks the right edge of a φ will apply both to ω_2 and to ω_3. With Strict Layering, such behavior would be deeply mysterious; in a strictly layered structure, ω_2 and ω_3 could not both be at right φ-edges without a left φ-edge between them as well. Strong evidence for such structures is rare, since for many languages, researchers have only identified evidence for one edge of a given prosodic constituent: left or right. However, several languages indicate that such structures are indeed necessary. On Truckenbrodt's (1995, 1999) reanalysis of Odden's (1987) data from Kimatuumbi, a φ-final ω is immune to vowel shortening, which applies elsewhere, and the left edge of every non-utterance-initial φ is preceeded by a high tone, which docks to the last word of the preceding φ. With this interpretation, Kimatuumbi displays phonological phrases of the form $(_{\varphi}\ (_{\varphi}\ \omega_1\ \omega_2)\ \omega_3)$. Structures of this type are also proposed by Selkirk (2011) for Xitsonga (with data from Kisseberth 1994 as well as Cassimjee and Kisseberth 1998), with evidence from High Tone Spread within the φ.

Truckenbrodt's reanalysis of Kimatuumbi and Selkirk's reanalysis of Xitsonga are instances of the simplest assumption about the interpretation of recursive prosodic categories: every category of level *i* has the same phonological consequences, regardless of whether it contains or is contained by another category of level *i*. But Ito and Mester (2007, 2009a, 2009b) have pointed out that we can differentiate prosodic categories based on their minimality and maximality. A minimal prosodic category π is one that does not dominate another π, and a maximal π is one that is not dominated by another π. Thus, in trees containing prosodic recursion, nodes may be identified not only by their prosodic category, but also by the features [±minimal] and [±maximal], which freely cross-classify. Ito and Mester observe that if phonological processes can be sensitive not only to prosodic category labels, but also to these features, then we can attribute divergent phonological phenomena to prosodic categories of the same level on the prosodic hierarchy, but at different levels in the tree. For instance, Ito and Mester (2009b) propose that in Japanese, the maximal φ is the domain of downstep, while the left edge of every φ is the locus of initial rise. Taking up this notion, Elfner (2012, 2015) proposes that the left edge of every non-minimal φ is marked by an LH tonal complex in Irish, while the right edge of every φ is marked by HL. Both of these analyses are discussed by Kalivoda, in chapters 7 and 4, respectively.

Prosodic recursion, plus minimality and maximality features, allows Ito and Mester to propose a prosodic hierarchy with only three interface categories: ι > φ > ω. This is a significant reduction in the number of prosodic categories when compared to earlier proposals (Selkirk 1980, Nespor and Vogel 1986) assuming an utterance category above ι and a clitic group above ω. In Japanese, the domain of downstep has been identified with a Major Phrase and the domain of initial rise with a Minor Phrase (McCawley 1968, Haraguchi 1977, Poser 1984, Kubozono 1988, Pierrehumbert and Beckman 1988, Selkirk and Tateishi 1988, etc.). All of these categories can be reduced to ι, φ, and ω with different values for minimality and maximality. The utterance can be reinterpreted as the maximal ι; the clitic group as the maximal ω; the Major Phrase as the maximal φ; and the Minor Phrase as the minimal φ. The papers in this book all employ Ito and Mester's prosodic hierarchy, though the main theoretical points of each chapter would remain intact if more prosodic categories turned out to indeed be needed, as argued for instance by Vogel (2009).

2.2. *Align Theory and Match Theory*

As mentioned above, prosody mirrors syntax imperfectly. Under Align Theory (Chen 1987, Selkirk 1986, Truckenbrodt 1999), the mapping process itself yields mismatches, and there is no pressure for prosody to be perfectly isomorphic to syntax. By contrast, under Match Theory (Selkirk 2011), the mapping process demands that the prosodic structure be isomorphic to the syntactic structure, and mismatches come from purely phonological factors. Align Theory and Match Theory are the dominant OT approaches to syntax-prosody mapping, and we introduce them here because they feature prominently in most of the chapters of this volume.

Align Theory grew out of Selkirk's (1986) pre-OT end-based theory, in which phonology is sensitive only to the ends of syntactic constituents. This edge-oriented approach to syntax-prosody mapping is implemented in Optimality Theory through constraints like ALIGNLEFT(X_{Lex}, ω) (WdCon in Selkirk 1995), ALIGNRIGHT(XP, φ), and ALIGNRIGHT(Focus, φ) (Truckenbrodt 1999), connecting to Generalized Alignment (McCarthy and Prince 1993). These constraints typically require the *d* (direction: left or right) edge of a syntactic constituent *s* to be mapped to a corresponding *d* edge of a prosodic constituent *p* of the correct category. Multiple syntactic edges can be mapped to a single prosodic edge. In this framework, languages are understood to typically prioritize mapping of one edge over the other. For example, in Zulu, right edges of CPs are generally mapped to right edges of ι, but left edges of CPs are not mapped to left edges of ι (Cheng and Downing 2007); this can be modeled with a constraint ranking ALIGNRIGHT(CP, ι) >> X >> ALIGNLEFT(CP, ι), where X is a constraint that conflicts with mapping both edges simultaneously, such as NONRECURSIVITY. Other work in this edge-based framework (Truckenbrodt 1995, 1999) also introduced a cohesional constraint WRAP(XP), which demands that each XP be contained in a phonological phrase. It conflicts with ALIGN(XP) when the syntactic input contains recursive XPs and the prosodic output does not. For example, WRAP(XP) objects to the non-recursive parse $[_{XP1}\ XP_2\ X_1] \rightarrow (XP_2)(X_1)$, which satisfies ALIGNRIGHT(XP, φ), but not to the isomorphic recursive parse $[_{XP1}\ XP_2\ X_1] \rightarrow ((XP_2)\ X_1)$, which also satisfies ALIGNRIGHT(XP, φ).

It is common for analyses employing Align Theory to consider only non-recursive and exhaustively parsed prosodic representations (Hale and Selkirk 1986, Selkirk 2000, Cheng and Downing 2007)—that is, it is common for them to adopt Strict Layering, explicitly or implicitly. Strictly layered structures can fully satisfy both ALIGNLEFT and ALIGNRIGHT without being isomorphic to the syntax, as in (2), where every left XP edge has a corresponding left φ-edge, and likewise for right edges.

(2) [$_{\text{VP}}$ g [$_{\text{NP1}}$ wakial] g [$_{\text{NP2}}$ wisilo] cepos] Truckenbrodt 1999 (24d)
(   ) (   ) (   )

Selkirk (2011) introduced a new indirect reference theory of the relationship between syntax and prosody: Match Theory. According to Match Theory, prosody is sensitive not only to syntactic edges, but to syntactic constituents. Therefore, constraints on syntax-prosody mapping are fully satisfied only when prosody is isomorphic to syntax. Selkirk's (2011) original formulation of MATCH constraints referred simultaneously to left and right edges, similar to a combination of ALIGNLEFT and ALIGNRIGHT. Elfner (2012) reformulated MATCH constraints to refer to the set of terminals contained in a constituent, rather than referring to edges. We adopt Elfner's definitions of MATCH throughout this volume; these versions of the MATCH constraints ensure that reordering terminals in prosody (see Bennett et al. 2016) does not alter the correspondence between a syntactic and prosodic constituent. In (3), for example, a MATCH(XP, φ) constraint that referred to both left and right edges would be violated for the vP, because there is no φ that has *é* at its left edge and *scoile* at its right edge. But a terminal-based MATCH(XP, φ) implementation is satisfied for vP, because there is a φ that contains all and only the set of terminals {*é, fhad, le, teach, na, scoile*}, even though they are not in the same order in the prosody as they are in the syntax.

(3) [$_{\Sigma\text{P}}$ thug [$_{\text{DP}}$ mo mháthair] [$_{\text{vP}}$ [$_{\text{DP}}$ é] [$_{\text{PP}}$ fhad le teach na scoile]]]
(thug mo mháthair) ((fhad le teach na scoile) é)
Bennett, Elfner, and McCloskey 2016 (70b)

Align Theory has been argued to have been superseded by Match Theory (Selkirk 2011), but has since been shown to still be needed in order to capture asymmetries in syntax-prosody mapping (Bellik et al. to appear).

Match Theoretic analyses normally include one syntax-to-prosody mapping constraint, and one prosody-to-syntax mapping constraint. Align Theoretic analyses, however, normally include two (or three, if WRAP is included) syntax-to-prosody mapping constraints and no prosody-to-syntax mapping constraints (though see Cheng and Downing 2016 for a counter-example). Since having more constraints inevitably increases the size of the predicted typology, Align systems tend to generate larger typologies than Match systems, as will be illustrated in Chapter 3.

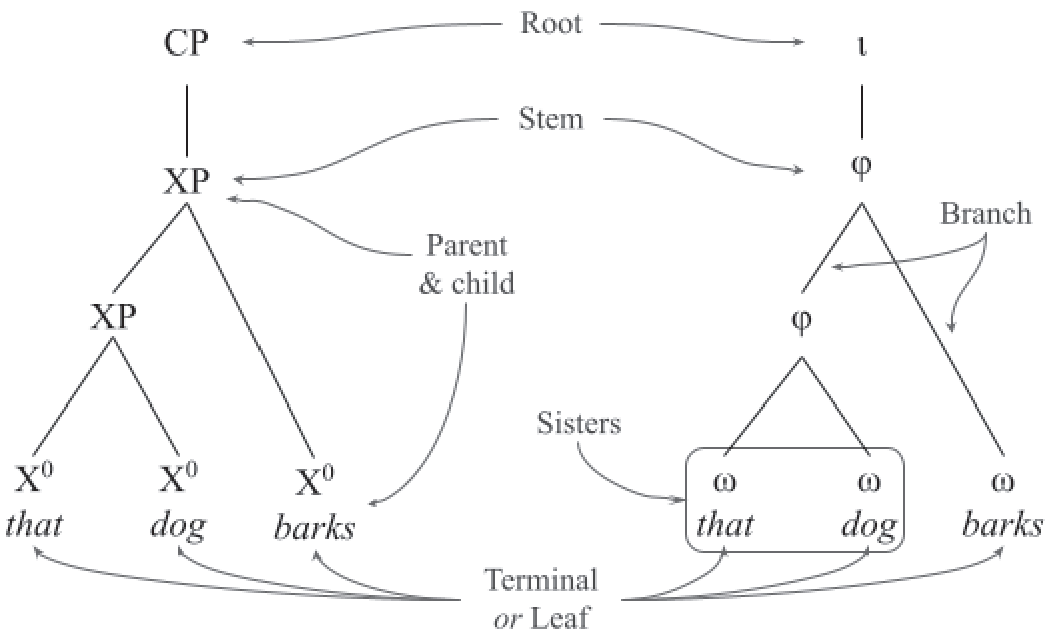
2.3. *Tree vocabulary and labeled diagrams*

In the domain of syntax-prosody mapping, inputs are syntactic trees and outputs are prosodic trees. A candidate is a pair <s-tree, p-tree>. This book frequently discusses trees and their structure, so we will briefly introduce several relevant terms that occur throughout the volume. The terms are illustrated below in (4). A tree, in the sense employed in linguistic representations, is a graph-theoretic object, consisting of labeled vertices ("nodes") that are connected by edges (*branches*), in such a way that the structure contains no cycles. The trees we will be concerned with are rooted trees, in which the top node in the tree is designated as its *root*, and contains the rest of the tree. In the trees below, CP is the root of the syntactic tree on the left, and ι is the root of the prosodic tree on the right. Every branch in a rooted tree connects a *parent* node (closer to the root) with a *child* node (farther from the root). Parents will always be depicted above their children. Children in a tree that share an immediate parent are called *sisters*, as is standard in linguistics, though not in graph theory. No child node will have multiple parents, due to the requirement that a tree contain no cycles; there is only one path between any two nodes in the tree.

The trees we will be concerned with represent syntactic or prosodic structures. Nodes in syntactic trees are labeled with their syntactic category; the categories relevant to syntax-prosody mapping are CP, XP, and X^0. They may also be labeled with additional information, such as whether the projection is lexical or functional, or what type of syntactic phrase an XP is (noun, verb, adjective, etc.). Nodes in prosodic trees are labeled with a prosodic category drawn from the prosodic hierarchy in (1).

The *terminals* in a tree are the nodes that have no children, in this case, the X^0s and ωs. We sometimes also refer to the *terminal string*, as a way to reference all the segmental content of a tree or trees; the sample trees in (4) have the same terminal string, *that dog barks* (non-branching XPs are depicted here as X^0s for visual simplicity). Sometimes we extend the botanical analogy of the tree by calling the terminals in a prosodic tree *leaves*; this term is standard in graph theory, though not widely used in linguistics. Finally, we refer to a non-root node that is the sole child of the root as the tree's *stem*. Unlike the other terms introduced here, "stem" is not standard vocabulary in either linguistics or graph theory.

(4)



2.4. *Forms of syntax-prosody mismatches*

A key theme in this book is mismatches between syntactic and prosodic structure. There are numerous ways for the prosodic structure to diverge from the syntactic structure that it corresponds to. In (5) and (6), we list particular forms of syntax-prosody mismatch that recur throughout the book.

(5) *Strong mismatch* (cf. Kalivoda 2018, p. 39), or *rebracketing*:
A strong mismatch occurs when the syntactic tree contains two terminals *a* and *b* that belong to a syntactic constituent to the exclusion of some additional terminal *z*, while the corresponding prosodic tree instead contains a prosodic constituent containing *b* and *z* to the exclusion of *a*, or *a* and *z* to the exclusion of *b*. The prosodic tree in this scenario contains a correspondent-less constituent whose brackets would cross those of the missing syntactic constituent.

For example, the mapping [[[a b] c] d] → ((a b)(c d)), as in Japanese (Chapter 4), is an example of a strong mismatch, where we find crossing brackets: [$_{\text{XP}}$ a b ($_{\varphi}$ c $_{\text{XP}}$] d $_{\varphi}$). The syntax contains a constituent [a b c], which excludes *d*, while the prosody instead contains the constituent (c d), which parses c and *d* together while excluding *a* and *b*.

We refer to mismatches that are not strong, or do not involve rebracketing, as weak mismatches. This term is employed descriptively in Kalivoda (2018, p. 39) without a formal definition. Weak mismatches typically come in three flavors, as in (6).

(6) *Weak mismatches* (cf. Kalivoda 2018, p. 39)
a. *Category promotion* occurs when a syntactic constituent is mapped to a prosodic constituent that is too high in the PH to satisfy Match.
b. *Category demotion* occurs when syntactic constituent is mapped to a prosodic constituent that is too low in the PH to satisfy Match. For example, when an XP is mapped to an ω.
c. *Flattening* occurs when the prosodic tree is shallower than the syntactic tree.

For example, in [[[a b] c] d] → (a b c d), the syntactic tree is three levels deep, but the prosodic tree has been flattened and has only one level. It lacks the substructure present in the syntactic tree. A less extreme version is the mapping [[[a b] c] d] → ((a b) c d), which we refer to as *partial flattening*, or sometimes *squishing*. These are all mismatches in which one or more syntactic constituent has no prosodic correspondent.

3. Optimality Theory

This section introduces OT systems and Property Theory in §3.1 and §3.2, respectively, and the computational tools SPOT and OTWorkplace in §3.3.

3.1. *OT Systems*

The papers in this volume all deal with precisely defined Optimality-Theoretic *systems* in the sense of Alber et al. (2016) and Prince (2017a,b). An OT system S is a formal object ⟨S.Gen, S.CON⟩, in which S.GEN determines a set of candidate sets (csets) and S.CON is a constraint set (i.e., a set of functions from the candidates of S.GEN to the non-negative integers). Every system S gives rise to a factorial typology S.Typ, which can be viewed both extensionally as a set of languages and intensionally as the grammars associated with those languages. While S.CON is necessarily finite, S.GEN need not be; a system can admit finitely or infinitely many csets, and each cset may include finitely or infinitely many candidates.

Explicitly spelling out GEN and CON for an OT system is necessitated by the definition of optimality itself. As Prince (2017a) points out, for a candidate to be optimal according to a constraint hierarchy, it must win out against *all* of the competitors in its candidate set. To determine whether a candidate beats all of its competitors, we need to know exactly what all of its competitors are, i.e., we must have a fully explicit GEN function (or simply an exhaustive list of candidates). Further, as Prince (2017a) puts it, "Comparison polls the judgment of all constraints," meaning that every constraint in CON must be specified, and must assign a violation count to every candidate. Without these pieces in place, the concept of optimality in OT loses meaning. These assumptions are at least tacitly in the background of all work in OT, dating back to Prince and Smolensky (1993/2004), but only more recent work has embraced full explicitness. The papers in this book strive for perfect clarity in system definitions.

While the details of S.GEN and S.CON vary across systems, all of the systems in this book share certain characteristics. On the GEN side, all candidates are input-output mappings in which the input is a syntactic tree made up of syntactic clauses, phrases, and words, and the output is a prosodic tree made up of prosodic categories like intonational phrase, phonological phrase, and phonological word, as in (1) above.

In terms of CON, every system in this volume involves syntax-to-prosody and/or prosody-to-syntax *mapping constraints* (analogous to faithfulness constraints) as well as purely prosodic *markedness constraints.* Mapping constraints include MATCH constraints and ALIGN constraints that refer both to syntactic and prosodic categories. Markedness constraints include constraints like various version of STRONGSTART, EQUALSISTERS, and BINARITY. Several chapters (Tarlov Chapter 3, Kalivoda Chapter 4, Bibbs Chapter 8, Cao et al. Chapter 9) compare the predictions of the two main approaches to syntax-prosody mapping in OT: Match Theory and Alignment Theory (see section 1.2). The contrast between these two theories arises only in the realm of mapping constraints; both share basic assumptions about GEN and markedness constraints. Other chapters (Bellik Chapter 10, Chapter 11) explore systems that differ only in terms of the markedness constraints included in CON, while Van Handel et al. (Chapter 5) examine systems that employ different types of MATCH constraints.

In all chapters, the concept of the *factorial typology* is central, even when the authors have the narrower aim of accounting for the facts from one particular language. For each system S, the authors have calculated the typology S.Typ using OTWorkplace (Prince, Merchant, and Tesar 2007-2020), a software for OT calculations. Given the importance of factorial typologies in this book, it is worth elucidating the concept in more detail.

Following Prince (2017a), we define a *language* as the collection of candidates that emerge as optimal under a given constraint hierarchy. Extensionally speaking, S.Typ is the set of languages admitted by S. Which languages make up S.Typ is determined by considering all possible rankings of the constraints in S.CON. Each linear order of the constraints in S.CON selects a particular set of optima, selecting one or more optima from each cset given by S.GEN. Selection of optima according to a constraint hierarchy proceeds by the usual process of OT candidate filtration. For details, see Prince and Smolensky (1993/2004) and Prince (2017a).

Each language in a factorial typology has an associated *grammar*. OT grammars can be stated in two different ways (Merchant and Prince to appear). On the one hand, grammars are sets of all constraint rankings that deliver the same language. Merchant and Prince call the individual rankings within a ranking grammar *linear extensions of the grammar*, or *legs*. Grammars stated as sets of legs are called *ranking grammars*. Alternatively, an OT grammar can be viewed as a collection of *elementary ranking conditions* (ERCs; Prince 2002ab), i.e. as an *ERC grammar*. The ERCs of an ERC grammar delineate the exact criteria for a leg's membership in a language's ranking grammar. A system's factorial typology, at an intensional level, is its set of grammars, whether viewed as ranking grammars or ERC grammars.

3.2. *Property Theory*

In many chapters, the interest in calculating a system's factorial typology lies (at least partially) in proving that the typology contains or does not contain a pattern associated with a particular language, such as Kinyambo (Tarlov Chapter 3), Irish (Kalivoda Chapter 4), Japanese (Kalivoda Chapter 7), Chamorro (Bibbs Chapter 8), or Xiamen Chinese (Cao et al. Chapter 9). The authors then go on to analyze the language's grammar, discussing all of the crucial rankings involved by presenting elementary ranking conditions (or if the language is not present in the typology, they discuss why this is so).

Several chapters, however, treat factorial typologies as objects of study in themselves, using the tools of Property Theory (Alber et al. 2016, DelBusso 2018, Alber and Prince 2021), a theory of the structure of OT typologies. The authors that use Property Theory are Tarlov (Chapter 3), Kalivoda (Chapter4, Chapter 7), and Bellik (Chapter 10, Chapter 11).

The basic claim of Property Theory is that an OT typology can be analyzed using *properties*, which are ranking conditions of a certain narrowly defined type. The grammars of a typology can be characterized by the *values* they take for the properties, and ideally, these are then associated with extensional *traits* in the corresponding languages. A *property analysis* of a typology is a set of properties which, taken together, completely characterize the grammars of the typology.

The most basic form of a property is A<>B, where A and B are individual constraints. Call this property P.a/b, where "a" and "b" are the values of the property. A grammar has the value P.a if A dominates B in all of its legs, and P.b if B dominates A in all of its legs. If A dominates B in some of the grammar's legs, while B dominates A in others, the property P.a/b is *moot* for the grammar. As an example, consider the typology of Tarlov's (Chapter 3) system Match.Rec. The system's constraint set, Match.Rec.CON, contains three constraints: sp.MATCH, m.BINMIN, and m.BINMAX. (The constraint prefix "sp" stands for "syntax-to-prosody," and "m" stands for "markedness".) The system's typology, Match.Rec.Typ, contains two languages: one in which unary φs are permitted (L.1), and one in which every φ must be branching (L.2). Expressed as ERCs, the grammars of the two languages are presented in the following comparative tableaux.

(7) Comparative tableau for L.1 of Tarlov's (Chapter 3) Match.Rec

Input	Winner	Loser	sp.MATCH	m.BINMIN	m.BINMAX
[[a][b]]	((a)(b))	(a b)	W	L	e

(8) Comparative tableau for L.2 of Tarlov's (Chapter 3) Match.Rec

Input	Winner	Loser	sp.MATCH	m.BINMIN	m.BINMAX
[[a][b]]	(a b)	((a)(b))	L	W	e

The comparative tableau in (7) shows that in L.1 of Match.Rec, sp.MATCH dominates m.BINMIN. The 'W' under sp.MATCH indicates that this constraint favors the L.1 winner, [[a][b]]→((a)(b)), over the L.1 loser, [[a][b]]→(a b). The 'L' under m.BINMIN indicates that this constraint favors the loser over the winner. The 'e' under m.BINMAX indicates that this constraint does not favor either the winner or the loser. The comparative tableau in (8) indicates that the opposite ranking holds in L.2, i.e., that m.BINMIN dominates sp.MATCH.

The two ERC grammars in (7) and (8) are equivalent to the following ranking grammars:

(9) Ranking grammar for L.1 of Tarlov's Match.Rec
{[sp.MATCH >> m.BINMIN >> m.BINMAX],
[sp.MATCH >> m.BINMAX >> m.BINMIN],
[m.BINMAX >> sp.MATCH >> m.BINMIN]}

(10) Ranking grammar for L.2 of Tarlov's Match.Rec
{[m.BINMIN >> sp.MATCH >> m.BINMAX],
[m.BINMIN >> m.BINMAX >> sp.MATCH],
[m.BINMAX >> m.BINMIN >> sp.MATCH]}

The legs in (9) are the three legs compatible with the ERC in (7), while those in (10) are the legs compatible with the ERC in (8). In every leg in (9), sp.MATCH dominates m.BINMIN, and in every leg in (10), the opposite holds. This fact allows Tarlov to establish the property p.MapUnary:

(11) Property for Tarlov's Match.Rec
p.MapUnary: sp.MATCH <> m.BINMIN
Values: unary.allowed / unary.banned

L.1 has the property value *unary.allowed*, because sp.MATCH dominates m.BINMIN in every leg of its ranking grammar, while L.2 has the property value *unary.banned*, because m.BINMIN dominates sp.MATCH in every leg of its ranking grammar. Tarlov associates each of the intensional property values with an extensional trait. In a language with the value *unary.allowed*, a φ can contain a single word, while in a language with the value *unary.banned*, a φ must be branching.

Tarlov's p.MapUnary is an example of an elementary property, consisting of two constraints in opposition to one another, but properties can also be more complex, with one or more side consisting of a constraint class suffixed with an operator. The two possible operators are '.dom' and '.sub'. Suppose K is a set of constraints $\{C_1, C_2, ..., C_n\}$. For an individual leg λ, K.dom returns the highest-ranked element of K in λ, while K.sub returns the lowest-ranked element of K in λ. For example, take a leg λ_1 = [A >> B >> C] and a constraint set J = {A, B}. J.dom(λ_1) = A, and J.sub(λ_1) = B. By allowing either side of a property to take the form of a constraint class with a suffixed operator, we generalize the notion of property to allow for more complex typological analysis.

3.2.1. The .dom operator

As a concrete example of the usefulness of the '.dom' operator, consider a property from Kalivoda's system SB_4 in Chapter 7. This system's constraint set includes four constraints, which we abbreviate here as sp.MATCH, sp.ALIGN-L, sp.ALIGN-R, and m.BINMAX. The system's typology consists of four languages called Iso, Bal, EP.L, and EP.R. The grammars of these languages are shown in the following *skeletal bases*, special comparative tableaux which condense ERC grammars into their bare essentials (Brasoveanu and Prince 2011). The *e*-values are omitted for clarity.

(12) Skeletal basis for Iso in Kalivoda's SB_4

sp.MATCH	sp.ALIGN-L	sp.ALIGN-R	m.BINMAX
W		W	L
W	W		L

(13) Skeletal basis for EP.L in Kalivoda's SB_4

sp.ALIGN-L	m.BINMAX	sp.MATCH	sp.ALIGN-R
W	L		
	W	L	L

(14) Skeletal basis for EP.R in Kalivoda's SB_4

sp.ALIGN-R	m.BINMAX	sp.MATCH	sp.ALIGN-L
W	L		
	W	L	L

(15) Skeletal basis for Bal in Kalivoda's SB$_4$

m.BinMax	sp.Match	sp.Align-L	sp.Align-R
W	L	L	L

In the Iso language of SB$_4$, all syntactic inputs map to perfectly matching prosodic outputs, while the other three languages all exhibit some amount of syntax-prosody non-isomorphism. Among the three languages that display non-isomorphism, EP.L and EP.R always preserve what Kalivoda calls their *main edge*, mapping the main edge of every XP to the main edge of some φ. In EP.L, this is the left edge, while in EP.R it is the right edge. These *edge-preserving* languages differ from the fourth language Bal, in that Bal maps syntactic structures to wildly mismatching prosodic structures, sometimes failing to align both the left and right edges of XPs. To distinguish the edge-preserving EP.L and EP.R on the one hand from the non-edge-preserving Bal on the other, Kalivoda introduces a property p.MainEdge.pres/del.

(16) p.MainEdge.pres/del
{sp.Align-L, sp.Align-R}.dom <> m.BinMax

EP.L and EP.R both have the value p.MainEdge.pres. In both, the *dominant* alignment constraint of the class {sp.Align-L, sp.Align-R} outranks m.BinMax, but which alignment constraint is dominant depends on the language. In EP.L, {sp.Align-L, sp.Align-R}.dom = sp.Align-L, and in EP.R, sp.Align-L, sp.Align-R}.dom = sp.Align-R.

The ranking grammar of Bal in SB$_4$ contains all legs in which m.BinMax dominates the other three constraints. In some of these legs, sp.Align-L outranks sp.Align-R, while in others, the reverse holds. Bal has the value p.MainEdge.del because regardless of which alignment constraint is dominant in any one of its legs, m.BinMax dominates that alignment constraint.

The constraint class {sp.Align-L, sp.Align-R} in the above property is not simply arbitrary; it consists of the system's two constraints from the sp.Align family. And in fact, in Kalivoda's property analysis of SB$_4$, there is an additional property p.Align.L/R, which has the form sp.Align-L <> sp.Align-R. Since the constraints in the class {sp.Align-L, sp.Align-R} are exactly those from the property p.Align.L/R, they are what Alber and Prince (2021) call a *public class*, and in the property definition of p.MainEdge.pres/del we can write {sp.Align-L, sp.Align-R}.dom as c.Align.dom, underscoring the fact that these constraints face off in a separate property.

3.2.2. The .sub operator

Finally, to demonstrate the use of the ".sub" operator, consider the third property from Kalivoda's property analysis of SB$_4$: p.Match.iso/mis.

(17) p.Match.iso/mis
{sp.Match, c.Align.sub}.dom <> m.BinMax

This separates the language Iso, with the value p.MATCH.iso, from the other three languages, with the value p.MATCH.mis. The left side of the property is a complex class: the *dominant* constraint in the class containing sp.MATCH and the *subordinate* alignment constraint. In other words, in every leg of Iso's ranking grammar, either sp.MATCH or *both* alignment constraints dominate m.BINMAX. Iso's ranking grammar contains fourteen legs, among them [sp.MATCH >> m.BINMAX >> sp.ALIGN-L >> sp.ALIGN-R], in which {sp.MATCH, c.ALIGN.sub}.dom = sp.MATCH, and [sp.ALIGN-L >> sp.ALIGN-R >> m.BINMAX >> sp.MATCH], in which {sp.MATCH, c.ALIGN.sub}.dom = sp.ALIGN-R.

Additional information about Property Theory is provided in the chapters that make use of it. For a complete introduction to the theory, the interested reader is referred to Alber and Prince (2021).

3.3. *SPOT and OTWorkplace*

Defining an OT system for an analysis of syntax-prosody mapping entails defining a candidate set consisting of pairs of trees. For the sake of illustration, let us consider the candidate sets of two very simple OT systems, SL for Strict Layering and WL for Weak Layering. Both SL and WL will have a single input syntactic tree, for presentational simplicity. Let this input be a uniformly left-branching, binary syntactic tree with four terminals, as in (18).

(18) Input to GEN in SL and WL: [$_{XP}$ [$_{XP}$ [$_{XP}$ a b] c] d]

In the interest of simplicity, we define GEN for SL and for WL to differ in the output structures that they admit. The first system, SL, will admit any prosodic tree that conforms to Strict Layering and does not add, remove, or reorder any terminals relative to the input syntactic tree.

(19) Gen.SL(*s*) = {All ⟨*s, p*⟩ pairs such that *p* fulfills the conditions in (a-f)}

  a. *p* is a tree rooted in an intonational phrase (ι).
  b. All non-root, non-terminal nodes in *p* are phonological phrases (φ).
  c. All terminal nodes in *p* are prosodic words (ω).
  d. Each terminal in *p* corresponds to a unique terminal in *s*, and each terminal in *s* corresponds to a unique terminal in *p*, with linear precedence relations between terminals being identical in both trees.
  e. Exhaustivity: No child of ι is an ω or lower.
  f. Non-Recursivity: No child of a φ is also a φ.

It is reasonably possible to build all the candidates in Gen.SL by hand. There are eight possible prosodic trees with four terminals that fit the conditions in (19); they are listed in (20). Curly braces { } stand for ι boundaries, and parentheses ( ) stand for φ boundaries. Each letter stands for a terminal ω. Despite the relatively small number of possible prosodic trees here, however, it would be easy to accidentally omit one or more of them when constructing the list by hand, particularly if the analyst doing the tree construction has not calculated out how many such trees are possible.

(20) All possible values of p for Gen.SL, given that s = [[[a b] c] d]

a. {(a b c d)}
b. {(a b c)(d)}
c. {(a b)(c d)}
d. {(a b) (c) (d)}
e. {(a)(b c d)}
f. {(a) (b c) (d)}
g. {(a) (b) (c d)}
h. {(a) (b) (c) (d)}

The difficulty of constructing all the trees in the candidate set becomes much more acute when we consider a second system, WL. WL admits not only the strictly layered trees, but also prosodic trees that contain recursive phonological phrases (level doubling) and/or prosodic words that are non-exhaustively parsed directly into the root intonational phrase (level skipping).

(21) Gen.WL(s) = {All $\langle s, p\rangle$ pairs such that p fulfills the conditions in (a-e)}

a. p is a tree rooted in an ι.
b. All non-root, non-terminal nodes in p are φs.
c. All terminal nodes in p are ωs.
d. Each terminal in p corresponds to a unique terminal in s, and each terminal in s corresponds to a unique terminal in p, with linear precedence relations between terminals being identical in both trees.
e. Headedness: Every ι contains at least one φ.

The cardinality of Gen.WL is dramatically larger than that of Gen.SL: There are 351 trees that satisfy the conditions in (4). For a discussion of how we can know that this is the correct number of trees, see Chapter 2, "Counting Tree Parses" (Shingler and Bellik, this volume). A small subset of the prosodic trees admitted by Gen.WL is listed in (22); these are the first eight and last five trees in the list SPOT generates.

(22) Some possible values of p for Gen.WL(s), given that s = [[[a b] c] d]

a. {(a b c d)}
b. {((a b c) (d))}
c. {((a b c) d)}
d. {(((a b) (c)) (d))}
e. {(((a b) (c)) d)}
f. {(((a b) c) (d))}
g. {(((a b) c) d)}
h. {((((a) (b)) (c)) (d))}
i. …
j. {a (b) (c (d))}
k. {a (b) c d}
l. {a (b) (c) (d)}
m. {a (b) (c) d}
n. {a (b) c (d)}

Generating the full candidate set for Gen.WL by hand would be difficult, time-consuming, and impractical. Instead, most syntax-prosody analyses manually generate a handful of prosodic structures that appear, to the analyst's intuition and reasoning, to be all the relevant possible optima. There is some justification to this approach, since most of the 351 candidates above will turn out to be harmonically bounded in many systems. For instance, if the constraint set consists of the four constraints on mapping and BINARITY in (23), only the two structures in (24) are possible optima; the other 349 are harmonically bounded.

(23) Con1
   a. MATCH(XP, φ): Assign a violation for every XP in the input that lacks a corresponding φ in the output.
   b. MATCH(φ, XP): Assign a violation for every φin the output that lacks a corresponding XP in the input.
   c. BINMAX(φ, branches): Assign a violation for every φ that has more than two branches (children).
   d. BINMAX(φ, ω): Assign a violation for every φ that dominates more than two ωs (at any level).

(24) Possible optima in WL, where Con.WL = Con1
   a. Isomorphic: {(((a b) c) d)}
   b. Squished: {(a b) c d}

The problem with this intuitive approach, however, is that it is not at all easy to correctly identify which particular outputs are possible optima, when the full candidate set is not being considered. In fact, the Squished candidate (22b) is not the competitor to the isomorphic parse (22a) that is considered in Selkirk (2011), which instead contrasts (22a) with a recursive balanced parse {((a b)(c d))}. Ishihara (2014) identifies another candidate that harmonically bounds the balanced parse, which is the non-recursive balanced parse: {(a b)(c d)}. The squished parse is not considered in either analysis, but Kalivoda (2018) shows that it harmonically bounds the balanced parse under both Selkirk's and Ishihara's constraint sets (see Bellik et al. to appear for further details on the problem of candidate omission for this input).

The SPOT app (Bellik, Bellik, and Kalivoda 2015–2021) offers an alternative. All analyses in this book were developed using SPOT to generate the violation tableaux, and the Excel extension OTWorkplace (Prince, Merchant, and Tesar 2007-2020) to generate and analyze the resulting typology. OTWorkplace can calculate the typology generated by a violation tableau (that is, a candidate set, or cset, and the violation profiles of each candidate), as well as provide constraint ranking information for each language in the typology. It also identifies which output structures are possible optima, and which are harmonically bounded, as well as identifying a support (a subset of the input structures that is sufficient to generate the entire typology). We have also used OTWorkplace in developing and validating the Property Analyses in this volume. The use of SPOT and OTWorkplace ensures that no candidates were accidentally omitted, and all possible optima were taken into account.

4. Overview of the book

We now turn to an overview of this volume, with a chapter-by-chapter summary in §4.1, and a discussion of recurring themes across chapters in §4.2.

4.1. *Chapter-by-chapter summary*

This book consists of four parts, dealing with Gen settings (Part I), Match Theory (Part II), Alignment (Part III), and Prosodic Wellformedness (Part IV), followed by a SPOT Tutorial (Part V). We here give a brief description of each part to guide the reader regarding the theoretical and descriptive focus of individual chapters.

Part I contains two chapters focused on the Gen settings for syntax-prosody mapping. A key component of an OT system is the Gen(erator) function, which defines the candidate set for that system. In the domain of syntax-prosody mapping, the candidates are <syntactic tree, prosodic tree> pairs. The goal of Chapter 2 (by Shingler and Bellik) is to develop a deeper understanding of how to construct the outputs in syntax-prosody mapping candidate sets, and of how the number of candidates grows with the addition of terminals. This chapter describes the options that the SPOT app makes available for defining Gen, and establishes through mathematical reasoning that SPOT's Gen functions are generating the correct number of candidates for each set of parameter values. Even though much of Optimality-theoretic research is concerned with the interaction of constraints, OT analyses require a well-defined Gen in addition to a well-defined Con and Eval. Chapter 3 (by Tarlov) shows how different definitions of Gen can interact with constraints in ways that are not immediately obvious. Different approaches are pursued in the analysis of phrasing issues arising in the Bantu language Kinyambo, and a property analysis of the resulting typologies yields interesting results.

Three chapters comprise Part II, which focuses on Match Theory, as seen from the perspective of SPOT. Chapter 4 (by Kalivoda) takes up Irish phonological phrasing, which has been the subject several influential analyses in recent years, in particular, by Elfner (2012, 2015) and Bennett, Elfner, and McCloskey (2016, 2019). However, a ranking paradox first noticed by Elfner (2012) remains unsolved in the context of standard OT with strict domination. The problem finds a solution by introducing a Match constraint sensitive only to overtly headed XPs, and by altering the definition of StrongStart to refer specifically to the left edge of the intonational phrase. The resulting analysis furthermore makes predictions for longer sentences of Irish. Chapter 5 (by Van Handel, Brodkin, and Eischens) examines Match constraints that are sensitive to subcategories, such as that of Ishihara's (2014) sp.Match(XP$^{[+\text{max}]}$, $\varphi^{[+\text{max}]}$). Admitting subcategory-sensitive Match constraints leads to two theoretical problems: first, a proliferation of possible Match constraints, and second, the emergence of Match constraints which enforce syntax-prosody non-isomorphisms, e.g., sp.Match(XP$^{[-\text{max}]}$, $\varphi^{[+\text{max}]}$). Chapter 5 presents the results of a SPOT investigation showing that two sets of subcategory-sensitive Match constraints drive non-isomorphism: (i) those in which only the second argument has a feature specification for its subcategory, e.g., sp.Match(XP, $\varphi^{[+\text{max}]}$), and (ii) those in which the first and second arguments have conflicting specifications for subcategory, e.g. sp.Match(XP$^{[+\text{max}]}$, $\varphi^{[-\text{max}]}$). A ban on the existence of these constraints is proposed, and it is argued that this ban follows from the integration of Match constraints into the theory of Faithfulness. Finally, Chapter 6 (by Van Handel) raises an important question in Match Theory concerning which syntactic constituents are visible to Match constraints. Based on data from Italian, Irish, and Xitsonga, and focusing on three proposals at the phrasal level: (i) Match(XP$_{\text{Lexical}}$, φ), which sees only lexical XPs; (ii) Match(XP$_{\text{OvertlyHeaded}}$, φ), which sees only XPs with phonologically overt heads; and (iii) Match(XP, φ), which sees all XPs. It is argued that Match(XP$_{\text{OvertlyHeaded}}$, φ) usually can and sometimes must be used instead of MATCH(XP$_{\text{Lexical}}$, φ). This raises the question of whether the lexical/functional distinction is actually needed: although the lexical/functional distinction is a useful heuristic, because it often correlates with the silent/overt head distinction, it may not be necessary to capture the full range of phrasing data.

While Match Theory was intended to replace Align/Wrap Theory, Part III contains two chapters arguing that ALIGN is a necessary ingredient in the analyses of Japanese phrasing asymmetries (Chapter 7) and Chamorro clitic movement (Chapter 8). Bellik, Ito, Kalivoda, and Mester (to appear) already showed (i) that MATCH and markedness alone cannot handle the left/right phrasing asymmetry in Japanese, while ALIGN can, whereas (ii) ALIGN cannot account for prosodic recursion of sufficient depth, and so MATCH is needed, as well. Chapter 7 (by Kalivoda) analyzes the resulting hybrid theory with MATCH, ALIGN, and BINARITY constraints, by defining an OT system with these constraints, generating its factorial typology, and elucidating the relation between intensional properties of its grammars and extensional traits of its languages in terms of Property Theory (Alber and Prince 2021). Clitic movement is usually considered to be syntactically driven, but previous work has shown that there are cases where this movement is prosodically motivated, e.g. by prosodic subcategorization (Chung 2003), or via constraints on prosodic well-formedness (Bennett, Elfner, and McCloskey 2016). Chapter 8 (by Bibbs) argues that for Chamorro, clitic movement does not require prosodic subcategorization, and instead can be motivated through the interaction of syntax-prosody mapping constraints and markedness constraints on prosodic well-formedness. Furthermore, detailed investigations with SPOT reveal that only ALIGN constraints on syntax-prosody relations can motivate clitic movement, while MATCH constraints are insufficient. Somewhat different conclusions are reached in Chapter 9 (by Cao, Bibbs, and Bellik), where tone sandhi in Xiamen Chinese can be properly accounted by either MATCH or ALIGN constraints, so long as the mapping constraints are subcategorized to the non-minimal syntactic phrase.

Moving beyond the faithfulness aspects of syntax-prosody (MATCH and ALIGN) in the previous chapters, Part IV focuses on the markedness constraints that play a role in determining the optimal prosodic form. Analyses of syntax-prosody mapping rely on notions of purely phonological well-formedness, and yet the constraints defining this well-formedness are not as clearly defined as the mapping constraints, in part because the space of possible prosodic mismatches has not been fully explored. Chapter 10 (by Bellik and Van Handel) provides a formal examination of the commonly used BINARITY constraints to capture size effects, that is, the tendency for longer strings to be parsed into more prosodic constituents. In some implementations, BINARITY is assessed locally by counting immediate children (= branch-counting); in others, BINARITY is assessed globally by counting all descendants of some category (= leaf-counting). Branch-counting BINARITY motivates size-driven prosodic recursion, and operates as a special case of MATCH(XP). In contrast, leaf-counting BINARITY motivates size-driven category promotion, and conflicts with MATCH(XP), leading to larger typology sizes. A constraint on UNIFORMITY is shown to be able to derive size-driven mismatches as well. Chapter 11 (by Bellik) examines several ways to define the prosodic well-formedness constraints EQUALSISTERS (Myrberg 2013) and STRONGSTART (Elfner 2012; Bennett, Elfner, and McCloskey 2016), and the consequences of these definitions for the predicted typology, with a focus on stringency interactions between them. Results are argued to support the use of categorical, parent-oriented definitions of both EQUALSISTERS and STRONGSTART.

The final chapter of this volume is the SPOT tutorial Chapter 12 (by Bellik and Kalivoda), which contains a step-by-step guide to how to use SPOT to build violation tableaux, as well as information on how to further analyze these tableaux in OTWorkplace and other OT software.

4.2. *Recurring themes across chapters*

Three major themes concerning constraint interactions recur in multiple chapters in this book. The first is the comparison and sometimes interaction of MATCH and ALIGN constraints; Chapters 3, 8, and 9 compare systems that differ only in their choice of mapping constraint, while Chapter 4 combines MATCH and ALIGN in the same system. Chapter 3 tries to derive subject-branchingness sensitivity (SBS) using either MATCH or ALIGN, and finds that either theory of mapping constraints can derive a less stringent version of SBS, but that only MATCH, used in a recursive system, can model a more stringent version of SBS. Furthermore, MATCH functions in a comprehensible way with a GEN function that permits prosodic recursion, while driving complex mismatch patterns under Strict Layering. ALIGN, in contrast, is compatible with either recursive or Strict Layering GEN. Similarly, Chapter 9 finds that either MATCH or ALIGN, when subcategorized appropriately, can account for the syntax-prosody mapping of Xiamen Chinese tone groups in a strict layering system. ALIGN predicts a larger typology due to the inclusion of five mapping constraints (WRAP plus four different ALIGN constraints); the equivalent MATCH system has only two constraints and two languages. In Chapter 8, in contrast, a similar comparison of MATCH and ALIGN systems finds that only ALIGN can yield the desired leftward clitic movement in Chamorro weak pronoun placement; MATCH does not work. Yet another situation obtains in Chapter 4, where MATCH and ALIGN must combine to account together to yield asymmetric rebracketing that is still sensitive to depth of recursion, as observed in Japanese.

The second major theme is the interaction of mapping constraints (primarily MATCH but also ALIGN) with BINARITY. This issue is explored in the greatest depth in Chapter 10 (by Bellik), which defines branch-counting and leaf-counting BINARITY. Branch-counting BINARITY functions as a special case of MATCH, while leaf-counting BINARITY conflicts with MATCH. In Chapter 3, Tarlov includes both minimal and maximal BINARITY, which interact with both ALIGN and MATCH, in two different candidate sets. Minimal BINARITY conflicts with ALIGN and MATCH in all systems; the conflict is more complex in ALIGN systems, where it defines two separate properties. Maximal BINARITY, in contrast, only conflicts with mapping constraints in non-recursive systems; all optima satisfy BINMAX in the recursive systems. In fact, the non-recursive systems involve more complex interactions overall than the recursive systems, because they do not allow BINMAX to be satisfied by building prosodic recursive structure, so that BINMAX and BINMIN are pitted against each other. Chapter 4 (by Kalivoda) takes up this theme from a different angle: only binary branching candidates are considered, so this chapter examines the interaction of leaf-counting maximal BINARITY with mapping constraints when branch-counting maximal BINARITY is inviolable.

Limitations on MATCH's ability to promote isomorphism form a third theme in this volume. Under some circumstances, contrary to their originally envisioned purpose, MATCH constraints favor syntax-prosody mismatches—what we call anti-match effects. This is documented in Chapters 3, 5, and 9. In the non-recursive MATCH system in Chapter 3, strong mismatches are optimal for certain inputs. MATCH is surprisingly unable to distinguish between [a [b [c]]] → (a)(b c), a weak mismatch, and [a [b [c]]] → (a b)(c), a strong mismatch. These are optimal in some languages of this typology because the structures that better fulfill MATCH are excluded from GEN. Next, Chapter 5 reveals that some ways of subcategorizing MATCH constraints produce anti-match effects even when GEN is unrestricted. In particular, when the specifications on the first and second argument differ, or when only the second argument is subcategorized, then the subcategorized MATCH conflicts with the unrestricted MATCH. The authors suggest that anti-match effects can and should be avoided by restricting the range of possible subcategorizations. We see in Chapter 9, however, that the concept of an anti-match effect needs to be refined when other factors, such as a restricted candidate set, prevent MATCH from selecting an isomorphic output. When GEN is restricted to non-recursive candidates, as in Chapter 9, general MATCH prefers to match the greatest number of XP, which means matching minimal XPs in preference to matching non-minimal ones. In the absence of any prosodic well-formedness constraints, this does not result in any rebracketed optima, but it does produce many weak mismatches with non-minimal XPs having no prosodic correspondent, when MATCH is undominated. General MATCH in this system conflicts directly with the subcategorized constraint MATCHNONMIN, which demands that non-minimal XPs be matched in the prosody, but ignores minimal XPs. This constraint does not have anti-match effects in Chapter 5, where recursive candidates are included and are optima, but does have anti-match effects in Chapter 9, where the candidate set is restricted to strictly layered structures. The term anti-match may actually be misleading in this context, however, since matching the non-minimal XPs actually produces a better category-blind MATCH—meaning it has fewer mismatches if we disregard category promotion and demotion, and focus on whether every syntactic constituent has some prosodic correspondent (of any category). Having MATCHNONMIN dominate the general MATCH produces a prosody that better reflects the overall constituency of the syntax.

5. References

Alber, Birgit, Delbusso, Natalie, and Prince, Alan (2016). From intensional properties to universal support. *Language* 92: e88–e116.

Alber, Birgit and Prince, Alan (2021). *The Structure of OT Typologies.* Chapter 1: Introduction to Property Theory. Unpublished manuscript. ROA 1381, Rutgers Optimality Archive, http://roa.rutgers.edu.

Bellik, Jennifer, Bellik, Ozan, and Kalivoda, Nick (2015-2021). Syntax-Prosody in Optimality Theory (SPOT). Javascript application. http://spot.sites.ucsc.edu. Codebase at https://github.com/syntax-prosody-ot.

Bellik, Jennifer, Ito, Junko, Kalivoda, Nick and Mester, Armin (to appear). Matching and alignment. In Haruo Kubozono, Junko Ito, and Armin Mester (eds.) *Prosody and Prosodic Interfaces*. Oxford University Press.

Bennett, Ryan, Elfner, Emily, and McCloskey, James (2016). Lightest to the right: An apparently anomalous displacement in Irish. *Linguistic Inquiry* 47: 169–234.

Bickmore, Lee (1989). *Kinyambo Prosody*. PhD dissertation, University of California, Los Angeles.

Bickmore, Lee (1990). Branching nodes and prosodic categories: evidence from Kinyambo. In Sharon Inkelas and Draga Zec (eds.) *The Phonology-Syntax Connection* 1-18. Chicago: University of Chicago Press.

Brasoveanu, Adrian and Prince, Alan (2011). Ranking and necessity: the Fusional Reduction Algorithm. *Natural Language and Linguistic Theory* 29: 3–70.

Cassimjee, Farida and Charles W. Kisseberth (1998) Optimal domains theory and Bantu tonology: A case study from IsiXhosa and Shingazidja, in Larry M. Hyman and Charles W. Kisseberth (eds) *Theoretical Aspects of Bantu Tone.* Stanford, CA: CSLI. 33–132.

Chen, Matthew (1987). The syntax of Xiamen tone sandhi. *Phonology Yearbook* 4: 109–149.

Cheng, Lisa Lai-Shen and Downing, Laura J. (2016). Phasal syntax = cyclic phonology? *Syntax* 19: 156–191.

Chung, Sandra (2003). The syntax and prosody of weak pronouns in Chamorro. *Linguistic Inquiry* 34: 547–599.

DelBusso, Natalie R. (2018). *Typological Structure and Properties of Property Theory.* PhD dissertation, Rutgers University.

Delbusso, Natalie (2020). The final-over-final condition, stringency, and typological structure. *Linguistic Inquiry* 51: 765–784.

Elfner, Emily (2012). *Syntax-Prosody Interactions in Irish*. PhD dissertation, University of Massachusetts Amherst.

Elfner, Emily (2015). Recursion in prosodic phrasing: evidence from Connemara Irish. *Natural Language and Linguistic Theory* 33: 1169–1208.

Elordieta, Gorka (2008). An overview of theories of the syntax-phonology interface. *International Journal of Basque Linguistics and Philology* 42: 209–286.

Haraguchi, Shosuke (1977) *The Tone Pattern of Japanese: An Autosegmental Theory of Tonology.* Tokyo: Kaitakusha.

Inkelas, Sharon (1990). *Prosodic Constituency in the Lexicon*. New York: Garland.

Ishihara, Shinichiro (2014). Match theory and the recursivity problem. In Shigeto Kawahara and Mika Igarashi (eds.) *MIT Working Papers in Linguistics 73: Proceedings of Formal Approaches to Japanese linguistics 7* 69–88. Cambridge, MA.

Ito, Junko and Mester, Armin (1992/2003). Weak layering and word binarity. In Takeru Honma, Masao Okazaki, Toshiyuki Tabata and Shin-ichi Tanaka (eds.) *A New Century of Phonology and Phonological Theory: A Festschrift for Professor Shosuke Haraguchi on the Occasion of His Sixtieth Birthday* 26-65. Tokyo: Kaitakusha.

Ito, Junko and Mester, Armin (2007). Prosodic adjunction in Japanese compounds. In Yoichi Miyamoto and Masao Ochi (eds.) *MIT Working Papers in Linguistics 55: Proceedings of Formal Approaches to Japanese Linguistics 4* 97-111. Cambridge, MA.

Ito, Junko and Mester, Armin (2009a). The extended prosodic word. In Janet Grijzenhout and Barış Kabak (eds.) *Phonological Domains: Universals and Deviations* 135-194. Berlin: Mouton de Gruyter.

Ito, Junko and Mester, Armin (2009b). The onset of the prosodic word. In Steve Parker (ed.), *Phonological Argumentation: Essays on Evidence and Motivation* 227–260. London: Equinox.

Kalivoda, Nick (2018). *Syntax-Prosody Mismatches in Optimality Theory*. PhD dissertation, University of California, Santa Cruz.

Kisseberth, Charles W. (1994). On domains, in *Perspectives in Phonology*, ed. By Jennifer Cole and Charles Kisseberth, 133 – 166, Stanford, CA: CSLI.

Kubozono, Haruo (1988). The Organisation of Japanese Prosody. University of Edinburgh dissertation.

McCarthy, John J. and Prince, Alan (1993). Generalized alignment. In Geert Booij and Jaap van Marle (eds.) *Yearbook of Morphology* 79-153. Dordrecht: Kluwer.

Merchant, Nazarré and Prince, Alan (to appear). *The Mother of All Tableaux: Order, Equivalence, and Geometry in the Large-scale Structure of Optimality Theory*. Equinox Press.

Myrberg, Sara (2013). Sisterhood in prosodic branching. *Phonology* 30: 73–124.

Nespor, Marina and Vogel, Irene (1986). *Prosodic Phonology.* Dordrecht: Foris.

Odden, David (1987) Kimatuumbi phrasal phonology. *Phonology Yearbook* 4, 13 – 36.

Pierrehumbert, Janet and Beckman, Mary (1988). *Japanese Tone Structure.* Cambridge, MA: MIT Press.

Poser, William J. (1984). The Phonetics and Phonology of Tone and Intonation in Japanese. MIT, dissertation.

Prince, Alan (2002a). Arguing optimality. In Angela Carpenter, Andries Coetzee, and Paul de Lacy (eds.) *Papers in Optimality Theory II* (University of Massachusetts Occasional Papers in Linguistics 26) 269-304. Amherst, MA: GLSA.

Prince, Alan (2002b). Entailed ranking arguments. Unpublished manuscript. ROA 500, Rutgers Optimality Archive, http://roa.rutgers.edu.

Prince, Alan (2007). The pursuit of theory. In Paul de Lacy (ed.) *The Cambridge Handbook of Phonology* 33–60. Cambridge: Cambridge University Press.

Prince, Alan (2017a). What is OT? ROA 1271, Rutgers Optimality Archive, http://roa.rutgers.edu.

Prince, Alan (2017b). OT Checklist. ROA 1306, Rutgers Optimality Archive, http://roa.rutgers.edu.

Prince, Alan and Smolensky, Paul (1993/2004). *Optimality Theory: Constraint Interaction in Generative Grammar*. Malden, MA: Blackwell Publishing.

Prince, Alan, Merchant, Nazarré, and Tesar, Bruce (2007–2020). OTWorkplace. http://sites.google.com/site/otworkplace

Selkirk, Elisabeth (1980). Prosodic domains in phonology: Sanskrit revisited. In Mark Aronoff and Mary-Louise Kean (eds.) *Juncture 7* 107–129. Saratoga, CA: Anma Libri.

Selkirk, Elisabeth (1984). *Phonology and Syntax: the Relation between Sound and Structure*. Cambridge, MA: MIT Press.

Selkirk, Elisabeth (1996). The prosodic structure of function words. In James L. Morgan and Katherine Demuth (eds.) *Signal to Syntax: Bootstrapping from Speech to Grammar in Early Acquisition* 187-213. Mahwah, NJ: Erlbaum.

Selkirk, Elisabeth (2000). The interaction of constraints on prosodic phrasing. In Merle Horne (ed.) *Prosody: Theory and Experiment. Studies Presented to Gösta Bruce* 231-261. Dordrecht: Kluwer.

Selkirk, Elisabeth (2011). The syntax–phonology interface. In John A. Goldsmith, Jason Riggle, and Alan C. L. Yu (eds.) *The Handbook of Phonological Theory* 435–484. Blackwell Publishing.

Selkirk, Elisabeth O. and Koichi Tateishi (1988). Constraints on minor phrase formation in Japanese, in Proceedings of the 24th Annual Meeting of the Chicago Linguistics Society, ed. by Gary Larson and Diane Brentari, 316 – 336, Chicago, IL: Chicago Linguistics Society.

Truckenbrodt, Hubert (1995). *Phonological Phrases: Their Relation to Syntax, Focus, and Prominence*. PhD dissertation, Massachusetts Institute of Technology.

Truckenbrodt, Hubert (1999). On the relation between syntactic phrases and phonological phrases. *Linguistic Inquiry* 30: 219–255.

Vogel, Irene (2009). Universals of Prosodic Structure. In Sergio Scalise, Elisabetta Magni, Antonietta Bisetto (eds.) *Universals of Language Today.* Amsterdam, Springer. pp. 59-82.