Input- and output-oriented generalizations in Iny ATR harmony

Adam McCollum (corresponding author)
Department of Linguistics, Rutgers University
ORCID ID, adam.mccollum@rutgers.edu

Adam Jardine
Department of Linguistics, Rutgers University,
ORCID ID, adam.jardine@rutgers.edu

September 22, 2022

Abstract

A number of fundamental concerns have motivated linguistic theory within the generative tradition. We consider three such concerns: explanation, typology, and computation. Significantly, these three goals are not equally well captured in the various phonological formalisms developed in the last half century. In this paper, we develop a computational analysis of ATR harmony in Iny (Ribeiro 2002, 2012) using boolean monadic recursive schemes (BMRS; Bhaskar et al. 2020) to account for the behavior of *icy targets* (Jurgec 2011) – elements that undergo but do not further propagate the harmonic feature. We argue that direct reference to input and output features values in BMRS enables a superior explanation for the data than is possible in other formalisms. Compared to its derivational predecessors, access to surface-oriented patterns is a key component of Optimality Theory's analytical and typological successes. However, its inability to directly reference input-oriented (opaque) patterns is a persistent problem for the theory. BMRS is able to directly reference both inputs and outputs, allowing analysis of opaque patterns. Further, the intensional characterizations available in the formalism allow more familiar predictions than compuational work using automata-theoretic tools.

Keywords: computational phonology, vowel harmony, opacity, boolean monadic recursive schemes

1 Introduction

As developed in early work by Chomsky (Chomsky 2013) and Halle (Halle 1959), and more clearly in (Chomsky and Halle 1968, *SPE*), the generative approach to phonological theory attempts to generate surface representations from underlying representations supplied by the lexicon and morphology. This grammatical architecture is used to model the linguistic knowledge of humans, thus attempting to predict the grammatical sound sequences in a language, as well as the more general structure of the human language faculty.

Within the generative tradition, a number of fundamental concerns have motivated the development of linguistic theory. We consider three such concerns: (1) explanation, (2) typology, and (3) computation. Significantly, these three goals are not equally well captured in the various formalisms developed in the last half century. For example, in terms of explanation, a major argument for Optimality Theory (Prince and Smolensky 2004; henceforth OT) over earlier *SPE*-derived formalisms was that it could explain conspiracies through the interaction of a single markedness constraint with multiple faithfulness constraints. From the typological point of view, both OT and the parametric theories are attempts to predict the typological space of linguistic patterns with greater clarity than is available in the rewrite rule formalism developed in earlier work. Computationally, the recent advancement of formal language-theoretic work on the structure of phonological generalizations (Heinz 2018) follows partially from the realization that OT predicts a range of strange, "pathological" patterns, even from the interaction of simple constraints (Frank and Satta 1998, Gerdemann and Hulden 2012), thus calling into question OT's claim that the central computational mechanism in phonological grammars is optimization. Heinz and colleagues instead argue for a view of phonology that is motivated by independent notions of computational complexity, which they show makes strong claims about typology and learnability in phonology. Developing theories that offer significant explanatory insight, predict the breadth of typological patterns attested, and are computationally restrictive has been part and parcel of of linguistic research.

We explore these issues, in particular explanation, through an analysis of ATR harmony in Iny, an underdescribed language of central Brazil (Ribeiro 2002, 2012). In (1) the basics of the pattern are evident. To see these, we need only to observe the activity of Iny vowels as triggers, targets, and propagators (derived triggers) of leftward ATR harmony. In (1a), mid vowels trigger, undergo, and propagate [+ATR] leftward. In (1b-d), high vowels trigger and undergo harmony, while low vowels do not undergo harmony. The most interesting aspect of the pattern is evident in (1e,f), derived [+hi,+ATR] vowels do not further propagate [+ATR] leftward. In these examples, the word-initial mid vowels surface as [-ATR] despite the fact that they precede a surface [+ATR] vowel. Thus, while mid vowels trigger, undergo, and propagate harmony leftward, high vowels trigger and undergo, but do not propagate the harmonic feature. They are what Jurgec (2011) dubs *icy targets*.

(1) ATR harmony in Iny

a.	/r-ɛ-rɔ=r-e/	[rerore]	'CTFG-1+TRANS-eat.solids- CTFG-IMPERF'
b.	/brɔrɛ-dĩ/	[broreni]	'deer-similar.to'
c.	/hãdɪ-dĩ/	[hãdini]	'jacu (fish species)-similar.to'
d.	/b-Ø-ɪ-krɔ=kre/	[bikrokre]	'2-CTFG-TRANS-cut= FUT'
e.	/kɔɗʊ-dĩ/	[kɔɗuni]	'turtle-similar.to'
f.	/r-ɛ-hɪ=r-e/	[rɛhire]	'CTFG-1+TRANS-drive.away-CTFG-IMPERF'

We present an analysis of Iny in a recently developed computational formalism *boolean monadic recursive schemes* (BMRS; Bhaskar et al. 2020, Oakden 2021, Chandlee and Jardine 2021a). In this formalism, the key generalizations are stated simply – underlying [-lo, -ATR] vowels will surface as [+ATR] if either of the following is true: the following vowel is [+hi, +ATR] in the input or the following vowel is [-hi, +ATR] in the output. Although the generalizations are relatively simple, previous analyses of Iny have required a number of representational and architectural modifications, and despite the machinery invoked, are still incapable of accounting for the pattern. Specifically, we demonstrate that the metrical structures developed in McCollum (2022) are extraneous; string-based representations suffice to explain the pattern. Additionally, the need for output-output correspondence paired with anti-faithfulness in Ozburn and Leduc (2022) is avoided by BMRS' ability to directly refer to input as well as output structure. This ability to reference inputs and outputs is a core component of BMRS, and is crucial to the success of our analysis.

The paper is structured as follows. In order to establish our criteria for a successful phonological analysis, §2 discusses issues of explanation, typology, and computation, and their significance for an adequate theory of phonology. In §3 we lay out the basic empirical facts in Iny. §4 we introduce the BMRS formalism and use it to analyze the data from Iny. In §5 we lay out a range of alternative analyses, discussing the various ways in which they fall short. In §6 we conclude the paper.

2 Computational theories and the goals of phonological analysis

Robust cross-linguistic generalizations pervade human sound patterns. As Kager (1999, 1) notes, “The central goal of linguistic theory is to shed light on the core of grammatical principles that is common to all languages.” This goal – the pursuit of analytical insight into the shared properties of human language – is one thing that differentiates linguists from engineers. Thus, while generative theories of phonology vary widely, they all share the central goal of understanding the nature of sound patterns in human language through the grammatical systems that represent them. Within this general pursuit, (Pater 2019, 351-352) writes “The original goal of OT is that of its predecessor, the principles and parameters framework . . . It aims to provide a formal framework that allows for the specification of grammatical systems that generate all and only the attested language types in some empirical domain." Although approaching the goals of theory from a computational perspective, (Heinz 2018, 135) frames his discussion in terms of this same goal: "A good theory must be both *sufficiently expressive* to accurately describe the actual phonologies in the world’s language and *maximally restrictive*."

The ways that phonologists have pursued an expressive yet restrictive theory generally reflect theoretical and methodological priorities. To see some of these differences, consider Anderson’s (1974) critique of Johnson’s (1972) computational approach to theory construction. Johnson (1972) compares the generative capacity of various modes of rule application, showing that grammars using simultaneuous rule application generate a superset of patterns derivable from a grammar using linear rules. Thus, while both are sufficiently expressive, he contends that a theory employing simultaneous application is too powerful. Johnson’s concerns are largely typological and computational – generating attested sound patterns while maintaining the computational restrictiveness of the theory. Johnson states:

> A phonological theory must characterize precisely the form of phonologies and the way they process strings. Among the results of such a theory will be a prediction as to what sorts of mappings can be affected by the phonologies of natural languages. (Johnson 1972, p. 9)

However, Anderson (1974) objects to this method of theory construction:

> Any adequate theory of phonology will have to put very strong substantive constraints on the notion of “possible rule” . . . and these constraints will surely be much more restrictive than any formal constraint of the sort discussed by Johnson. Since the grammars constructed in Johnson’s proof contain rules which seem highly implausible as part of the phonology of a natural lanugage, they would undoubtedly be ruled out by substantive considerations, and not on formal grounds. . . . In any case, the correct procedure in this area [construction of a restrictive theory] is surely not to establish such limitations a priori; rather, the structure of individual rules must be established from simple, noniterative cases, and then the operation of the rule in more complex circumstances investigated. Only on such a basis can we decide what sort of principle should be followed in applying a rule. (Anderson 1974, 225-226)

Anderson thus argues for a bottom-up approach to rule application, and more generally to typology. He indicates that the operation of simple devices, and then the interaction of multiple simple devices is more likely to yield desireable results. This logic, which is employed in parametric and Optimality Theoretic work, is intended to guarantee clear, simple analytical generalizations. On Anderson’s view, a top-down set of “a priori” restrictions are not necessarily able to capture these same basic generalizations.

However, it is not necessarily the case that the bottom-up approach guarantees a superior characterization of the regularities in phonological generalizations. This is particularly clear in the case of OT, as discussed in Jardine (2019)’s review of the difficulties of capturing spreading in OT. He concludes that “global optimisation makes it difficult, if not impossible, to make any coherent statement about the structural properties of phonological processes in OT” (Jardine 2019, p. 348), instead arguing for a top-down theory that states the least expressive classes of functions to which phonological processes belong.

2.1 Substantive and formal explanation

Thus, there is a tension between formal and substantive views of phonological theory. From the substantive point of view, understanding the nature of the individual generalizations in the grammar is an indispensible part of the phonological endeavor, for without such understanding the best we can hope for is an inscrutable “black box” that captures all and only the attested patterns but provides no meaningful understanding of human language or cognition. *SPE*-style rules were capable of describing individual generalizations, but were unable to account for functional unity

across different rules. Thus, conspiracies exposed deficiencies in the explanatory value of rewrite rule-based theories. Moreover, it proved difficult to restrict the notion of a "possible rule" beyond the broadly-defined formalism of *SPE*. As a consequence, there are few straightforward ways to probe the extensional consequences of a given device (cf. Johnson's (1972) discussion of simultaneous vs linear rules). This interest in explanation at both the local (rule, constraint, etc) and global (derivation, factorial typology) level motivates parametric and constraint-based analysis, and is central to traditional generative theories of phonology.

Yet, interest in local generalizations is de-emphasized in some recent formal work. One critique of such work is that it focuses on broad categories of functions rather than the more fine-grained generalizations to be found in the data. For instance, Jardine (2016) and Chandlee and Heinz (2018) view phonological patterns in terms of *subsequential* functions and *input strictly local* functions, respectively. These classes of functions are defined in terms of formal language theory, a subfield of computer science that classifies the complexity of patterns. The goal of formal language-theoretic analyses of phonological patterns is to reveal extensional properties of phonological patterns themselves, independent of how they may be cast in a particular intensional theory. Chandlee et al. (2018, 173) remarks, "This focus on the maps themselves means the conclusions we draw are not about the nature of the phonological rules or phonological constraint-based grammars, but the nature of the input-output map that all such formalisms aim to describe." Thus, while these categories reveal much about the nature of phonological patterns, they do not capture some of the fine-grained distinctions that phonologists are used to making. For example, much of the analysis in Jardine (2016) and Chandlee and Heinz (2018) uses *finite-state automata*, which are abstract machines that process symbols, instead of the features, autosegments, etc., that are the *lingua franca* of phonological theory. Thus, for some phonologists, this approach does not look familiar as a linguistic theory. Pater (2019, 353) expresses his skepticism thusly: "[I]t seems to me unlikely that [formal language-theoretic] formalisms will give phonologists the means to transparently express the regularities that hold of phonological systems, in ways comparable with existing rule- or constraint-based formalisms."

However, it is a misconception that the computational perspective ignores linguistically significant generalizations. To the contrary, computational formalisms do capture linguistically significant generalizations, including ones that are difficult to capture in other formalisms. Furthermore, through *logical* – as opposed to automata-theoretic – formalisms for describing phonological patterns, computationally-grounded theories indeed do have ways to transparently express substantive aspects of phonological patterns. The BMRS analysis of Iny in the subsequent sections illustrates this well. First, we outline the linguistically significant *computational* generalizations this analysis aims to capture.

2.2 Some linguistically significant computational generalizations

Computational analysis distills phonological processes to their fundamental properties as functions. This can reveal aspects of phonological phenomena that are otherwise difficult to state. For example, many phonological processes have been found to be *input strictly local* (ISL; Vaysse 1986, Chandlee 2014, Chandlee and Heinz 2018). ISL functions are exactly those for which the output of any input symbol is computed entirely based on the local environment of that symbol *in the input*. For example, consider the function described by the a non-iterative rule in (2a), in which an *a* is mapped to a *b* before an input *b*. Example mappings from this function are given in (2b).

(2) a. $a \rightarrow b \,/\, \_\_\, b$ (non-iterative)

b. $aaa \mapsto aaa$
$aba \mapsto bba$
$aab \mapsto abb$
$aaab \mapsto aabb$
$\vdots \mapsto \vdots$

This function is ISL because whether a target *a* is changed into a *b* is determined entirely on its local context in the input – namely, whether that *a* is followed by a *b* in the input. This is highlighted schematically in (3), which compares how the outputs are computed for the two input *a*s in the string *aab*.

(3) a. / *a* [*a*] *b* /
[*a* [*b*] *b*]

b. / [*a*] *a* *b* /
[[*a*] *b* *b*]

In both (3a) and (b), the target *a* and its corresponding output are enclosed in boxes. The right context in both is highlighted in green. In (3a), the target *a* is immediately followed by a *b* in the input, so it is output as *b*. In (3b), the target *a* is followed by an *a* in the input, so it is output as *a*, even though that following *a* becomes a *b* in the output. Thus, as it is specifically the input that determines the function, it is ISL.

The corresponding class of functions that is dependent entirely on the local context in the output is *output strictly local* (OSL; Chandlee et al. 2015, Chandlee and Heinz 2018). An example OSL function (that is minimally different example than the function in (2)) is the one in (4) that changes an *a* to a *b* when it is followed by a *b in the output*.

(4) a. $a \rightarrow b \,/\, \_\_\, b$ (iterative)
b. $aaa \mapsto aaa$
$aba \mapsto bba$
$aab \mapsto bbb$
$aaab \mapsto bbbb$
$\vdots \mapsto \vdots$

That this is OSL is illustrated by the diagram in (5), again using an input *aab* – which for this function is output as *bbb*.

(5) a. / *a* **a** *b* /
[*b* **b** *b*]

b. / **a** *a* *b* /
[**b** *b* *b*]

In (5a), the target *a* is immediately followed by an output *b*, and so it is output as *b*. However, as this *b* is also a *b* in the input, it is not clear that it is its output value that matters. This is made clear in (5b), in which the target *a* becomes a *b* because it is followed by an output *b*, even though that *b* was an *a* in the input. Thus, as it is specifically the output that determines the function, it is OSL.[1]

The ISL and OSL functions are important as a tool for understanding phonology for several reasons. First, a theory of phonology that states processes are ISL or OSL is extremely restrictive. Most functions are ISL or OSL, and so stating that phonological processes must be ISL or OSL excludes a great deal of pathological processes. Regardless, such a theory has wide empirical coverage. Chandlee and Heinz (2018) report that 95% of the processes described in the P-Base database of phonological processes (Mielke 2004) are ISL. Furthermore, these notions of input- and output-locality can be extended to tier-based representations to encompass long-distance processes (Burness et al. 2021). Finally, the ISL and OSL functions are defined based on independent considerations of computational complexity. This allows us to apply techniques from theoretical computer science to address empirical problems in phonological theory. For example, Chandlee and Heinz (2018) gives an overview of how computational learning theory has been leveraged to produce procedures that learn ISL and OSL functions from positive data.

As such, it is desirable for a phonological theory to be able to directly state such properties of phonological processes. To return to the empirical example at hand, ATR spreading in Iny is particularly interesting because it simultaneously exhibits ISL and OSL properties.

First, ATR spread in Iny as initated by high vowels is ISL. To see this, compare the outputs of the underlying /ɛ/ vowels in (1b) /brɔrɛ-dĩ/ [broreni] 'deer-similar.to' and (1f) /r-ɛ-hɪ=r-e/ [rɛhire] 'CTFG-1+TRANS-drive.away-CTFG-IMPERF'. This is shown schematically in (6). Note that the below assumes that vowels are phonologically adjacent, á la Gafos (1996). Equivalently, we could assume they are adjacent on a tier without consonants—formally, as a *tier-based* ISL function (Burness et al. 2021). Regardless, what is important is the dependence of the function on the input.

(6) a. / ɔ **ɛ** **ĩ** /
[o **e** i]

b. / **ɛ** **ɪ** e /
[**ɛ** i e]

The examples in (6) parallel those in (3). In (6a), the /ɛ/ is output as [e] because the following vowel in the input—highlighted in green—is [+high, +ATR]. In contrast, in (6b) the target /ɛ/ is output as [ɛ] because its following input vowel is [+high, −ATR]. Thus, because the output of the target can be computed entirely based on local information to the input, ATR spread via high vowels is ISL. Note that the output [±ATR] value of the following high vowel is irrelevant: in both cases, it is [+high].

Conversely, spread triggered by mid vowels ignores the input and is dependent on the output value of the triggering vowel. It is thus OSL. This is shown schematically in (7), using different target vowels from the same examples (1b) /brɔrɛ-dĩ/ [broreni] 'deer-similar.to' and (1f) /r-ɛ-hɪ=r-e/ [rɛhire] 'CTFG-1+TRANS-drive.away-CTFG-IMPERF'.

[1] Technically, it is *right*-OSL, as the conditioning context is to the right of the target. OSL is divided into two classes of functions, *right*-OSL and *left*-OSL, depending on which direction the computation operates. ISL, in contrast, has no such split. For more details, see Chandlee (2014), Chandlee et al. (2015), Chandlee and Heinz (2018).

(7) a. / ɔ ɛ ĩ /
[o e i]

b. / ɛ ɪ e /
[ɛ i e]

In (7b), the highlighted target [−ATR] vowel /ɪ/ changes to the [+ATR] [i] preceding an output [e]. However, as this [e] is also underlyingly /e/, it is ambiguous as to whether it is the input [e] or the output /e/ that is triggering the spread. This is disambiguated in (7a), in which the highlighted target [−ATR] vowel /ɔ/ changes to the [+ATR] [o] preceding an output [e]. Importantly, this output [e] is underlyingly [−ATR] /ɛ/ and only became [+ATR] as the result of ATR spread (as described above). Thus, it is the *output* value of the mid vowel that triggers the change of the target /ɔ/ to [o]. Thus, mid-triggered ATR spread in Iny is OSL.

To review, ATR spread in Iny has both ISL and OSL aspects: high-triggered ATR spread is ISL, whereas mid-triggered ATR spread is OSL. We can be even stronger, in fact. High-triggered ATR spread is *properly* ISL because the input is relevant while the output is irrelevant, and conversely mid-triggered ATR spread is properly OSL because the *output* is relevant while the input is irrelevant.

Throughout the paper, we draw on these facts of Iny, further described in §3 and analyzed in §4, to highlight BMRS' ability to directly capture both computational and substantive aspects of the pattern. We contrast this with OT and rule-based theories of phonology. In particular, OT has difficulty capturing properly ISL functions (Chandlee and Jardine 2021b), as the OT analyses of Iny in §5 show. Rule-based theories like *SPE* can capture both patterns by tagging rules as either iterative or non-iterative. However, as we show in §5, to fully the capture ATR spread in Iny it needs to appeal to extrinsic ordering despite the fact the environments that trigger harmony are disjoint – input [+hi,+ATR] vowels and output [-hi,+ATR] vowels. This obscures the OSL nature of the pattern, as an *SPE*-style analysis crucially depends on intermediate forms. In contrast, below we show that BMRS both captures both mid- and high-triggered ATR spread in Iny in a way that directly expresses their computational properties as well.

3 Iny

3.1 Background

Iny (ISO: kpj) is a Macro-Jê language spoken by approximately 3,000 along the Araguaia River in central Brazil (Ribeiro 2012). The language is also known as Karajá, which as noted in (Ribeiro 2012, 7) is most likely a pejorative exonym. We thus use the endonym, Iny [i.ˈnə̃]. All data were taken from Ribeiro (2002) and Ribeiro (2012).

3.2 Inventory

Iny has an inventory of sixteen contrastive vowel qualities – twelve oral vowels and four nasal vowels, shown in (8). In addition, Ribeiro (2012) notes an additional surface vowel in the language, [ə].

(8) Vowel inventory

	Oral			Nasal	
High	i	ɨ	u	ĩ	
	ɪ	ɨ̇	ʊ		
Mid	e	ɘ	o		õ
	ɛ	(ə)	ɔ	ə̃	
Low		a		ã	

In an earlier description, Rodrigues (1999) suggests only nine oral vowels and three nasal vowels exist in the language; Rodrigues' description does not differentiate ATR pairings for the high vowels. ATR contrasts for the high peripheral vowels were first recognized in Ribeiro (2000), though the ATR distinction among the high central vowels was not identified until Ribeiro (2002, 2012). Although no phonetic work has been done on the full set of central vowels described in more recent work, an acoustic study (Fulop and Warren 2014) supports an ATR distinction among the mid and high peripheral vowels. We draw the vast majority of our examples from the peripheral vowels since their status is most firmly established.

3.3 Data

In (9) we see that [+ATR] mid vowels trigger the assimilation of preceding mid and high vowels. In (9a-d), a final [+ATR] mid vowel triggers leftward assimilation of all preceding vowels. In (9e,f) we see that low vowels do not assimilate to [+ATR], blocking further spreading of [+ATR].

(9) Harmony triggered by mid vowels

a.	/r-ɛ-rɔ=r-e/	[rerore]	‘CTFG-1+TRANS-eat.solids- CTFG-IMPERF’
b.	/r-ɛ-rɔ=r-e/	[rerore]	‘CTFG-1+TRANS-eat.solids- CTFG-IMPERF’
c.	/b-Ø-ɪ-krɔ=kre/	[bikrokre]	‘2-CTFG-TRANS-cut= FUT’
d.	/Ø-d-ɪ-wɛ=d-e/	[diwede]	‘3-CTPT-TRANS-penetrate- CTPT-IMP’
e.	/wa-ritʃɔrɛ-boho/	[waritʃoreboho]	‘1-offspring- PL’
f.	/r-ɛ-hãɗɛ=r-e/	[rɛhãɗere]	‘CTFG-1+TRANS-hit-CTFG-IMPERF’

ATR harmony is not only triggered by mid vowels. The high vowels also participate in ATR harmony (cf. Fulop and Warren (2014) for the central vowels). Consider the data in (10). In these examples a high vowel initiates leftward harmony, triggering ATR alternations among preceding mid vowels. Several things are worth noting here. First, the triggering [+ATR] vowel may occur in any morphological position – root, affix, or clitic. Second, in (10b,c) we see the non-alternation of low vowels again, further supporting the case the low vowels, oral or nasal, do not participate in ATR harmony. Third, we see that harmony is strictly regressive, targeting preceding vowels only. In (10c,d) all mid vowels following the [+ATR] trigger do not exhibit any ATR alternations, and surface faithfully as [-ATR].

(10) Harmony triggered by high vowels

a.	/brɔrɛ-d$\tilde{i}$/	[broreni]	‘deer-similar.to’
b.	/hãlɔkɔɛ=ʧi/	[hãlokoeʧi]	‘jaguar=LOC’
c.	/wa-θɛ-ritʃɔrɛ/	[waθeritʃɔrɛ]	‘1-mother-offspring’
d.	/Ø-r-ɔ-ɗuhɔ=r-ɛrɪ/	[roʧuhɔrɛrɪ]	‘3-CTGF-ANTI-curse-CTFG-PROG’

Further evidence for purely regressive directionality is found within morphemes (11). In polysyllabic morphemes, non-low [-ATR] vowels may follow [+ATR] vowels (11g-i) but non-low [-ATR] vowels may not precede [+ATR] vowels. This suggests that harmony operates from right to left.

(11) Harmony within morphemes

[-ATR][-ATR]	a.	dɔrɛ	‘parrot’
	b.	bɛrɔ	‘puba (a type of manioc flour)’
	c.	kɔrʊ	‘forehead’
[+ATR][+ATR]	d.	hedə	‘smoke (noun)’
	e.	boho	‘PL’
	f.	kube	‘palm’
[+ATR][-ATR]	g.	bədɛ	‘land’
	h.	ʧuʃɔ	‘quati (a type of mammal)’
	i.	ritʃɔrɛ	‘offspring’
[-ATR][+ATR]	Disallowed; harmony applies		

We saw in (9-10) that low vowels do not participate in the harmony pattern, always occurring as [-ATR]. The role of /a ã/ as blockers is further evident in (12). In addition to the two low vowels, (12) illustrates the behavior of two other non-participating vowels, /ə õ/. Of interest, though /õ/ is [+ATR] it does not trigger harmony on preceding vowels /õ/, as seen in (12g,h).

A second [+nasal, +ATR] vowel is present in the inventory, /$\tilde{i}$/. Concerning this vowel, we have not been able to find any examples where surface [$\tilde{i}$] is in a context to trigger harmony, so we cannot be certain of its behavior. In some examples, underlying /$\tilde{i}$/ is denasalized and harmony occurs (10a), but without examples of [$\tilde{i}$] in the relevant contexts it is impossible to determine whether this vowel is an active trigger of harmony or not. In the absence of more informative data, we assume that /$\tilde{i}$/ behaves just like /i/, triggering harmony on preceding vowels.

(12) Non-participation of /a ã ə õ/

/a/

a.	/bɛra=ʧi/	[bɛra=ʧi]	'water-LOC'
b.	/r-ɛ-ka=r-e/	[rɛhãɗere]	'CTFG-1+TRANS-dig-CTFG-IMPERF'

/ã/

c.	/hãbu/	[hãbu]	'man'
d.	/r-ɛ-hãɗɛ=r-e/	[rɛhãɗere]	'CTFG-1+TRANS-hit-CTFG-IMPERF'

/ə̃/

e.	/a-r-ɪ-rɔ=rɛdə̃=kre/	[arɪrɔrɛdə̃kre]	'1-CTFG-TRANS-eat-CTFG-PL=FUT'
f.	/r-ɛ-bə̃=r-e/	[rɛmə̃re]	'CTFG-1+TRANS-take-CTFG-IMPERF'

/õ/

g.	/Ø-r-ɪ-rɔ=kõ=kre/	[rɪrɔkõkre]	'3-CTFG-TRANS-eat-NEG=CTFG-IMPERF'
h.	/r-ɛ-õ=r-e/	[rɛõre]	'CTFG-1+TRANS-give-CTFG-IMPERF'

We have established several key facts: (1) harmony is regressive, (2) oral mid vowels trigger and undergo harmony, (3) high vowels trigger and undergo harmony, and (4) the low vowels and nasal non-high vowels do not participate in harmony, either as triggers or undergoers. The remaining and most important facet of the pattern for our analysis is the differential behavior of the high and mid vowels. We saw that mid vowels trigger, undergo, and further propagate [+ATR] in (9). In (10) we saw that high vowels can trigger harmony, and in (9c,d) we saw that high vowels can undergo harmony. However, in (9c,d) the high vowels were word-initial, and thus could not further propagate the harmonic feature. Consider the examples in (13) – in each of these, an underlying [+hi, -ATR] vowel undergoes harmony but fails to propagate it any further leftward in the word, leaving the preceding mid vowels as [-ATR]. In (13a-c), the high vowel target immediately precedes the triggering vowel, but in (13d) two intervening mid vowels harmonize between the trigger and high vowel undergoer. As a final comment on the behavior of high vowels, Ribeiro (2012, 108-109) indicates that in some lexically-specific contexts, derived high vowels may optionally propagate [+ATR] further leftward. We will return to this point in §5.2.

(13) Medial high vowels

a.	/kɔɗʊ-dĩ/	[kɔɗuni]	'turtle-similar.to'
b.	/krɔbɪ-dĩ/	[krɔbini]	'monkey-similar.to'
c.	/r-ɛ-hɪ=r-e/	[rɛhire]	'CTFG-1+TRANS-drive.away-CTFG-IMPERF'
d.	/r-ɛ-hʊkɔɗɛ=r-e/	[rɛhukoɗere]	'CTFG-1+TRANS-lend-CTFG-IMPERF'

We can conclude from these data that the extent of harmony is not definable in terms of syllables, since harmony iteratively assimilates all vowels in the word in examples like (9a,b) while in (13d) harmony spans over four syllables, but only two in (13a-c). Furthermore, we cannot define the domain of harmony in terms of morphology, since clitics, affixes, and roots may all trigger and undergo harmony. Thus, no syllable- or morphology-based analysis of Iny is tenable. Instead, we demonstrate that the analysis depends on two factors, inputs vs output feature specifications, and vowel height.

4 Analysis

We analyze Iny harmony using the *boolean monadic recursive schemes* of Bhaskar et al. (2020). We first introduce this formalism before launching into the analysis.

4.1 Boolean monadic recursive schemes

Boolean monadic recursive schemes (Bhaskar et al. 2020, Chandlee and Jardine 2021a) is a logical formalism defined as a restriction on a more general theory of computation known as recursive program schemes (Moschovakis 2019). The *boolean* in the name refers to the fact that BMRS only define boolean properties—for example, the feature [±ATR] can be modeled as a boolean function [ATR](x) that returns $\top$ (true) when x is an [+ATR] segment and $\bot$ (false) when x is a [−ATR] segment. *Monadic* means that these functions can only target individual elements in a representation—that is, BMRS defines properties like [ATR](x) that target individual segments but *not* properties like $P(x, y)$ that hold between x and y. We can and will, however, use monadic *successor* and *predecessor* functions, $s(x)$ and $p(x)$, respectively, to refer to the immediately following and immediately preceding segment of x. This allows us to refer to the local context around a segment while adhering to the monadic nature of the functions. For example, [ATR]($s(x)$), for example, is a property that holds or does not hold of the segment following x.

A *scheme* is thus a series of definitions of boolean, monadic properties. That these schemes are *recursive* refers to the fact that these properties can be invoked in their own definitions.

A scheme can thus specify a mapping by defining the properties of the output in terms of the input structure. To explain with an example, recall the two abstract rules (2) and (4) from Section 2.2. First, consider the non-iterative, ISL rule in (2), repeated below in (14).

(14) a. $a \rightarrow b \,/\, \_\_\, b$ (non-iterative)
b. $aaa \mapsto aaa$
$aba \mapsto bba$
$aab \mapsto abb$
$aaab \mapsto aabb$
$\vdots \mapsto \vdots$

We can represent the simple inventory $\{a, b\}$ with a single feature [±B], where a is [−B] and b is [+B]. In turn, this feature can be represented with the boolean, monadic property $[\mathrm{B}]_{\mathrm{i}}(x)$, where $[\mathrm{B}]_{\mathrm{i}}(x) = \top$ when x is a [+B] segment (i.e., a b) and $[\mathrm{B}]_{\mathrm{i}}(x) = \bot$ for a [−B] segment (i.e., an a). The subscript i here indicates that $[\mathrm{B}]_{\mathrm{i}}(x)$ holds in the input. An example with the string *aab* given in (15).

(15) The string *aab* and its truth values for $[\mathrm{B}]_{\mathrm{i}}(x)$

index	1	2	3
segment	*a*	*a*	*b*
$[\mathrm{B}]_{\mathrm{i}}(x)$	$\bot$	$\bot$	$\top$

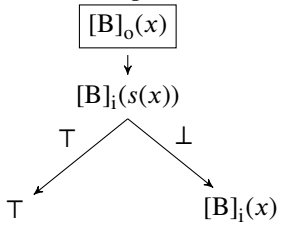
The table in (15) represents *aab* explicitly with each position in the string given an index. For example, position 1 in *aab* is an *a*; thus, $[\mathrm{B}](1) = \bot$; in contrast, $[\mathrm{B}](3) = \top$, as position 3 is a *b*. In this way, any distinct string of *a*s and *b*s will have a distinct truth table for $[\mathrm{B}](x)$. Thus, we can represent any string of *a*s and *b*s with a sequence of indices and truth values for $[\mathrm{B}](x)$. A brief note: equating boolean values $\top, \bot$ with the feature values $+, -$, respectively, is just a notational choice. Equivalently, we could consider a mix of functions that return either $\bot, \top$ or $+, -$ values. However, as we will see, it will be simpler to use boolean functions.

To define new functions, BMRS use an `if ... then ... else ...` syntax that explicitly models the logic of a computation. For example, the expression in (16) defines an *output* feature $[\mathrm{B}]_{\mathrm{o}}(x)$ given the non-iterative rule in (14).

(16) $[\mathrm{B}]_{\mathrm{o}}(x) =$ `if` $[\mathrm{B}]_{\mathrm{i}}(s(x))$ `then` $\top$ `else` $[\mathrm{B}]_{\mathrm{i}}(x)$

This expression specifies how to determine the output of $[\mathrm{B}]_{\mathrm{o}}(x)$. It reads exactly as one might expect: if $[\mathrm{B}](s(x))$ is $\top$—that is, if the successor of x is [+B] in the input—then return $\top$; otherwise return $[\mathrm{B}]_{\mathrm{i}}(x)$—that is, return the input value of [±B] for x. The `if ... then ... else ...` syntax suggests an order of evaluation, which we assume here; this evaluation is illustrated schematically in (17).

(17) Flow of computation of (16)

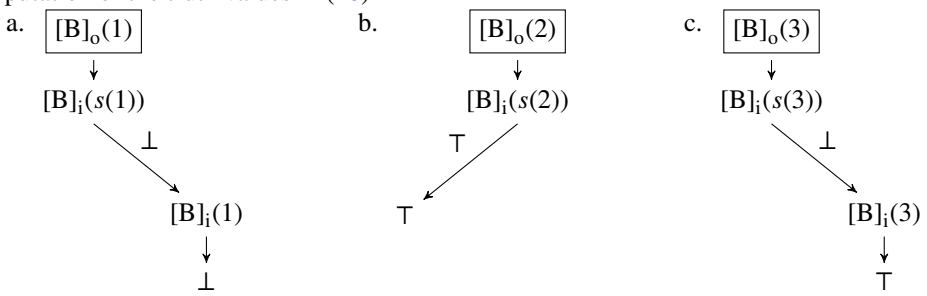


The truth values for $[\mathrm{B}]_{\mathrm{o}}(x)$ are given for the same example string *aab* below in (18) and (19) shows the flow of evaluation for each segment in the string, following the structure of the schematic in (16).

(18) The string *aab* and its truth values for $[\mathrm{B}]_{\mathrm{o}}(x)$

index	1	2	3
segment	*a*	*a*	*b*
$[\mathrm{B}]_{\mathrm{o}}(x)$	$\bot$	$\top$	$\top$

(19) Computation of the truth values in (18)



To compute the value of $[\mathrm{B}]_\mathrm{o}(x)$ for each segment in *aab*, we simply replace x with the index of that segment in the definition in (16) and then follow the logic of the computation. For segment 1 (the first *a* in *aab*), we thus evaluate

$$[\mathrm{B}]_\mathrm{o}(1) = \texttt{ if } [\mathrm{B}]_\mathrm{i}(s(1)) \texttt{ then } \top \texttt{ else } [\mathrm{B}]_\mathrm{i}(1).$$

How $[\mathrm{B}]_\mathrm{o}(1)$ is computed is illustrated in (19a). First, the computation checks $[\mathrm{B}]_\mathrm{i}(s(1))$. As $s(1) = 2$, and $[\mathrm{B}]_\mathrm{i}(2) = \bot$ (as 2 is an *a*), the `if` condition fails and the computation returns the value of the `else` statement. In terms of the diagram in (19a), the computation follows the arrow labeled $\bot$ leading out of $[\mathrm{B}]_\mathrm{i}(s(1))$. As the `else` statement is $[\mathrm{B}]_\mathrm{i}(1)$, which evaluates to $\bot$, the entire computation returns this value. Thus, $[\mathrm{B}]_\mathrm{o}(1) = \bot$.

For the computation of $[\mathrm{B}]_\mathrm{o}(2)$, in contrast, $[\mathrm{B}]_i(s(2))$ evaluates to $\top$, as $s(2) = 3$ is an input *b*. Thus, the `if` condition is satisfied and the computation returns the value of the `then` statement. Diagrammatically, the computation follows the arrow labeled $\top$ leading out of $[\mathrm{B}]_\mathrm{i}(s(2))$. As is clear in the diagram, the `then` statement simply returns $\top$, so the entire computation returns $\top$. Thus, $[\mathrm{B}]_\mathrm{o}(2) = \bot$.

Finally, the computation of $[\mathrm{B}]_\mathrm{o}(3)$, the evaluation proceeds like that with $[\mathrm{B}]_\mathrm{o}(1)$, but with a slight twist. When $[\mathrm{B}]_\mathrm{o}(s(3))$ is evaluated, there is a technical issue: as 3 is the last index of the string, $s(3)$ does not exist! In formal terms, $s(3)$ is *undefined*. For cases like this, we adopt the (innocuous) semantic convention that $P(s(x))$ is $\bot$ whenever $s(x)$ is undefined. For this reason, $[\mathrm{B}]_\mathrm{o}(s(3))$ evaluates to $\bot$. As with $[\mathrm{B}]_\mathrm{o}(1)$, the evaluation then goes to the `else` statement $[\mathrm{B}]_\mathrm{i}(3)$. As this returns $\top$ (segment 3 is a *b*), the entire computation thus returns $\top$.

a.

index	1	2	3
UR	*a*	*a*	*b*
$[\mathrm{B}]_\mathrm{i}(x)$	$\bot$	$\bot$	$\top$
$[\mathrm{B}]_\mathrm{o}(x)$	$\bot$	$\top$	$\top$
SR	*a*	*b*	*b*

b.

index	1	2	3	4	5	6
UR	*b*	*a*	*a*	*a*	*b*	*a*
$[\mathrm{B}]_\mathrm{i}(x)$	$\top$	$\bot$	$\bot$	$\bot$	$\top$	$\bot$
$[\mathrm{B}]_\mathrm{o}(x)$	$\top$	$\bot$	$\bot$	$\top$	$\top$	$\bot$
SR	*b*	*a*	*a*	*b*	*b*	*a*

Figure 1 Truth values and mappings based on (16) in the example strings (a) *abb* and (b) *baaaba*.

We have now arrived at each of the truth values for $[\mathrm{B}]_\mathrm{o}(x)$ in (18): segment 1 (*a*) is $\bot$, segment 2 (*b*) is $\top$, and segment 3 (*b*) is $\top$. If we interpret these as the truth values for the feature [±B] in a string of *a*s and *b*s, we get the output string [*abb*]. Thus, as explicitly shown in Figure 1, we can interpret the definition in (16) of $[\mathrm{B}]_\mathrm{o}(x)$ as describing a *map* which changes an /*a*/ into a [*b*] immediately preceding an input /*b*/. To show this, Figure 1 gives the truth values for $[\mathrm{B}]_\mathrm{o}(x)$ for another example input string, /*baaaba*/ and shows how it is output by this map to [*baabba*].

In general, the structure of a BMRS definition for an output property P_o is as follows in (20).

(20)
$$\begin{aligned} P_\mathrm{o}(x) = \ & \texttt{if } S_1(x) \texttt{ then } \top/\bot \texttt{ else} \\ & \texttt{if } S_2(x) \texttt{ then } \top/\bot \texttt{ else} \\ & \texttt{if } S_3(x) \texttt{ then } \top/\bot \texttt{ else} \\ & \ldots \\ & \texttt{if } S_{n-1}(x) \texttt{ then } \top/\bot \texttt{ else} \\ & S_n(x) \end{aligned}$$

In (20), each $S_i(x)$ is some property that is evaluated for x, and each (save for the last, S_n) is associated with a boolean value $\top$ or $\bot$. If $S_i(x)$ evaluates to $\top$, then $P_\mathrm{o}(x)$ returns the boolean value associated with that property.

Following the terminology in Chandlee and Jardine (2021a), we call $S_o(x)$ a *licensing structure* if its associated $\top$, and a *blocking structure* if it is associated with a $\bot$. For example, in the BMRS definition in (16) for the non-iterative rule $a \rightarrow b \;/\; \_\_\; b$, $[\mathrm{B}]_i(s(x))$ is a licensing structure for $[\mathrm{B}]_o(x)$. This naming is intended to communicate that whenever $[\mathrm{B}]_i(s(x))$ is true for x, it allows $[\mathrm{B}]_o(x)$ to return true as well.

If $S_i(x)$ evaluates to $\bot$, evaluation goes along to the structure in $S_{i+1}(x)$. In this way, the `if ... then ... else ...` syntax of a BMRS definition of an output property outlines a hierarchy of licensing and blocking structures, where structure $S_i(x)$ takes priority over all structures $S_j(x)$ for $j > i$ that appear later in the hierarchy, as it is evaluated first.

The final structure S_n in the hierarchy is the *default* condition; that is, the condition that is evaluated when all other structures in the hierarchy return $\bot$. In (16) the default condition is simply $[\mathrm{B}]_i(x)$—the input value of x for $[\pm\mathrm{B}]$. By setting this as the default, (16) states that, as a default, x remains faithful to its input value for $[\pm\mathrm{B}]$. Faithfulness is violated only if preempted by a structure higher up in the hierarchy.

Given a set P_1, P_2, P_3, ..., P_k, that fully describes output representations, a full BMRS is a list of definitions $P_1 =$ ..., $P_2 =$ ..., $P_3 =$ ..., ..., $P_k =$ ... that follow the general format of (20). These definitions thus can determine an output structure for any input structure. Thus, as the feature $[\pm\mathrm{B}]$ is sufficient to describe any string of *a*s and *b*s, the single definition in (16) is thus a full BMRS that can transform any string of *a*s and *b*s into another. The analysis below will provide a more sophisticated example involving the definition of more than one output property.

Importantly, a BMRS definition of a mapping can be recursive, in that these output properties may appear in their own definitions. To illustrate, let us return to the iterative $a \rightarrow b \;/\; \_\_\; b$ rule in (21), repeated from (4).

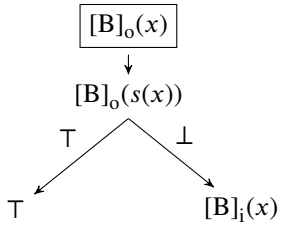
(21) a. $a \rightarrow b \;/\; \_\_\; b$ (iterative)

b. $aaa \mapsto aaa$
$aba \mapsto bba$
$aab \mapsto bbb$
$aaab \mapsto bbbb$
$\vdots \mapsto \vdots$

To write a BMRS expression for this map, we cannot simply use the input property $[\mathrm{B}]_i(x)$. As already explained in Section 2.2, this map is properly OSL, and as such cannot be captured by referring solely to properties of the input string. Instead, we can use a *recursive* BMRS definition to refer to the properties of the output string. Such a definition is given below in (22).

(22) $[\mathrm{B}]_o(x) =$ `if` $[\mathrm{B}]_o(s(x))$ `then` $\top$ `else` $[\mathrm{B}]_i(x)$

(23) Flow of computation of (22)



This definition is identical to that in (16) except that the licensing structure $[\mathrm{B}]_i(s(x))$ in (16) has been replaced with $[\mathrm{B}]_o(s(x))$. Thus, instead of checking the input value for $[\pm\mathrm{B}]$ in the following segment, (22) checks its output value for $[\pm\mathrm{B}]$. As is perhaps clear, this is recursive because $[\mathrm{B}]_o(s(x))$ is being used in the definition for $[\mathrm{B}]_o(x)$ itself. Thus, in order to evaluate $[\mathrm{B}]_o(x)$ for any x, it is necessary to first evaluate $[\mathrm{B}]_o(s(x))$.

To see how this works, Fig. 2 gives an example valuation for segment 1 in the string *aab* which, following the same indexing scheme as above, is *a*, segment 2 is *a* (the full indexing is also given below in Fig. 3a).

Following (22), to evaluate $[\mathrm{B}]_o(1)$, the licensing structure $[\mathrm{B}]_o(s(1))$ must first be evaluated. This begins the recursive evaluation: as $s(1) = 2$, this requires computing $[\mathrm{B}]_o(2)$. This can be seen in the flow diagram in Fig. 2 as the arrow from $[\mathrm{B}]_o(s(1))$ into the evaluation of $[\mathrm{B}]_o(2)$. The first step in evaluating $[\mathrm{B}]_o(2)$ is, of course, $[\mathrm{B}]_o(s(2))$. In turn, then, the evaluation moves to compute $[\mathrm{B}]_o(3)$. Here, the recursion stops at the `if` statement in $[\mathrm{B}]_o(3)$, namely $[\mathrm{B}]_o(s(3))$: as $s(3)$ does not exist, $[\mathrm{B}]_o(s(3))$ immediately returns $\bot$. The computation thus moves to the `else` statement, $[\mathrm{B}]_i(3)$, which returns $\top$ (as 3 is a *b*). This $\top$ thus becomes the output of $[\mathrm{B}]_o(3)$.

As shown in the diagram, this $\top$ becomes the output of $[\mathrm{B}]_o(s(2))$ in the evaluation of $[\mathrm{B}]_o(2)$. The `if` statement having evaluated to $\top$, the computation goes to the `then` statement in (22), which is simply $\top$ (see the flow diagram in

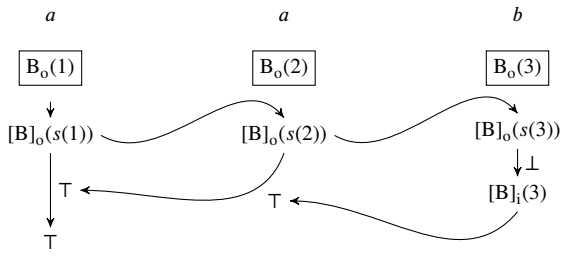


Figure 2 Flow of computation for (22) for segment 1 (*a*) in the string *aab*.

(23)). This ⊤ thus becomes the output of $[B]_o(2)$, which in turn becomes the output of $[B]_o(s(1))$ in the evaluation of $[B]_o(1)$, ⊤. This in turns results in the output of $[B]_o(1)$, ⊤.

a.

index	1	2	3
UR	*a*	*a*	*b*
$[B]_i(x)$	⊥	⊥	⊤
$[B]_o(x)$	⊤	⊤	⊤
SR	*b*	*b*	*b*

b.

index	1	2	3	4	5	6
UR	*a*	*a*	*a*	*b*	*a*	*a*
$[B]_i(x)$	⊥	⊥	⊥	⊤	⊥	⊥
$[B]_o(x)$	⊤	⊤	⊤	⊤	⊥	⊥
SR	*b*	*b*	*b*	*b*	*a*	*a*

Figure 3 Truth values and mappings based on (22) in the example strings (a) *abb* and (b) *baaaba*.

Thus, the output for $[B]_o(x)$ of one segment can factor into the computation of $[B]_o(x)$ for multiple other segments. Because the output condition $[B]_o(s(x))$ is a licensing structure for $[B]_o(x)$, any *a* preceding a segment that returns ⊤ for $[B]_o(x)$—that is, is a *b* in the output—will also itself return ⊤ for $[B]_o(x)$. As Fig. 3a shows, when interpreted as a description of a map, it changes any input *a* preceding an output *b* to a *b*. In other words, $[B]_o(x)$ in (22) describes the iterative $a \rightarrow b \,/\, \_\_\, b$ rule in (21).

We have now seen how a BMRS definition can compute the properly ISL map described by the non-iterative version of $a \rightarrow b \,/\, \_\_\, b$ as well as the properly OSL map described by the iterative version of the rule. In the former case, this was accomplished through the reference in (16) to the input property $[B]_i(s(x))$, and in the latter case, through the reference in (22) to the output property $[B]_o(s(x))$. We now show how this ability to directly refer to either the input or the output to capture the Iny pattern, which is the composition of properly ISL and properly OSL generalizations.

4.2 Analysis of Iny harmony in BMRS

Using a combination of input and output conditions, the BMRS formalism can directly capture the distinct spreading behavior of mid and high vowels in Iny. Recall that in Iny, input high vowels spread [+ATR], whereas output mid vowels spread [+ATR]. Furthermore, in order to spread [+ATR] the mid vowels must also be oral. We thus first define what it means to be a [+ATR, +high] vowel in the input and a [+ATR, −high, −low, −nasal] vowel in the output (24) and in (25), respectively, below.

(24) $[+\text{ATR}, +\text{hi}]_i(x) = \texttt{if } [\text{ATR}]_i(x) \texttt{ then } [\text{high}]_i(x) \texttt{ else } \bot$

(25) $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(x) =$
```
if [high]_o(x) then ⊥ else
if [low]_o(x) then ⊥ else
if [nasal]_o(x) then ⊥ else
[ATR]_o(x)
```

(26) $[\text{high}]_o(x) = [\text{high}]_i(x)$

(27) $[\text{low}]_o(x) = [\text{low}]_i(x)$

(28) $[\text{nasal}]_o(x) = [\text{nasal}]_i(x)$

The expression on the right-hand side of (24) first checks if x is $[\text{ATR}]_{\text{i}}(x)$; if so, then it returns the value of $[\text{high}]_{\text{i}}(x)$. In this way, the expression returns $\top$ if and only if x is both [+ATR] and [+high] in the input representation. (Thus, equivalently, one could write it as the boolean formula "$[\text{ATR}]_{\text{i}}(x) \wedge [\text{high}]_{\text{i}}(x)$").

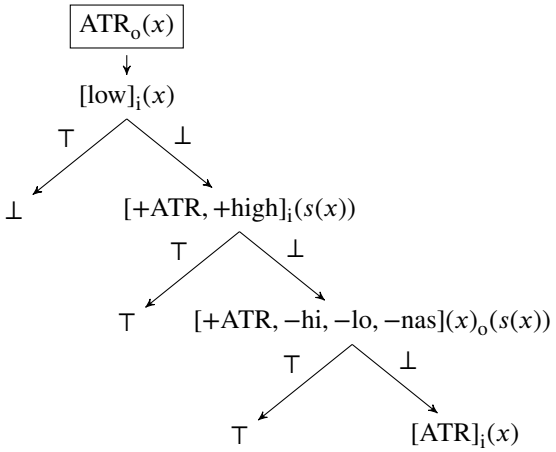
Similarly, the expression on the right-hand side of (25) first checks $[\text{high}]_{\text{o}}(x)$. If this returns $\top$, the entire statement returns $\bot$: we want $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}]_{\text{o}}(x)$ to only return true for [−high] vowels. If this returns $\bot$, evaluation moves on to the next if condition, which is $[\text{low}]_{\text{o}}(x)$. Similarly, as we only want [−low] vowels, if this returns $\top$ then the entire statement returns $\bot$. The next condition does the same for [−nasal] vowels. Finally, if the evaluation moves past each of these `if` conditions, the statement returns the value of $[\text{ATR}]_{\text{o}}(x)$. It thus returns true if and only if x is [+ATR], [−high], [−low], and [−nasal] *in the output*. We must, of course, define $[\text{ATR}]_{\text{o}}(x)$, $[\text{high}]_{\text{o}}(x)$, [−low], and [−nasal]. The latter three are defined in (26), which equates the output feature with its input value. In other words, a segment is always faithful in height and in nasality.[2]

The definition for $[+\text{ATR}]_{\text{o}}(x)$, as it is determined itself by harmony, is the focus of the BMRS analysis. This definition is given below in (29).

(29) $[\text{ATR}]_{\text{o}}(x) =$ `if` $[\text{low}]_{\text{i}}(x)$ `then` $\bot$ `else`
`if` $[+\text{ATR}, +\text{hi}]_{\text{i}}(s(x))$ `then` $\top$ `else`
`if` $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_{\text{o}}(s(x))$ `then` $\top$ `else`
$[\text{ATR}]_{\text{i}}(x)$

This definition reads as follows. First, if x is [+low], then return $\bot$; the only [+low] vowels are [−ATR], [a ã]. If not, then continue to the next line. If then the *successor* of x satisfies $[+\text{ATR}, +\text{high}]_{\text{i}}(x)$—that is, the expression defined in (24)—then return $\top$. In other words, if the following vowel is [+ATR, +high] in the input, then x surfaces as [+ATR]. (Following the discussion in Sec. 2.2, we will assume here that $s(x)$ refers to the next vowel—that is, it operates over some vowel tier.) Likewise, if the successor of x satisfies $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}]_{\text{o}}(x)$ (i.e., (25)), then return $\top$. Thus, both $[+\text{ATR}, +\text{high}]_{\text{i}}(s(x))$ and $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}](x)_{\text{o}}(s(x))$ are licensing structures for [ATR]. If the segment x satisfies neither of these conditions, then the last line states specifies the default return value—namely, x's value for [±ATR] in the input. The logic of the computation is visualized in (30).

(30) Flow of computation of (29)



Importantly, the third condition, $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}](x)_{\text{o}}(s(x))$, contains recursion: the definition for $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}]_{\text{o}}(x)$ in (25) makes reference to $[\text{ATR}]_{\text{o}}(x)$, but here $[+\text{ATR}, -\text{high}, -\text{low}, -\text{nas}]_{\text{o}}(x)$ is being used itself in the definition for $[\text{ATR}]_{\text{o}}(x)$. This allows the value of one segment to depend on the output value for [±ATR] of another.

Table 1a and b a show how the value of $[\text{ATR}]_{\text{o}}(x)$ is calculated for each vowel in the URs /kɔɗʊ-dĩ/ and /r-ɛ-hʊkɔɗɛ=r-e/, respectively.

[2]This claim may seem at odds with the apparently shifted nasality in /dĩ/ [ni] reported in (1), (10), and (13). However, (Ribeiro 2012, 91) indicates that the nasality of /ĩ/ is "much less pronounced after [n]." In the absence of any dedicated description of the apparent transfer of nasality from /ĩ/ to /d/ – which does not contrast for nasality with [n] – we conclude that Ribeiro's surface transcriptions record a phonetic shift in the onset and offset of velum lowering; there is no phonological relationship between the nasality and orality of the high front vowels and harmony.This, however, is not crucial to our analysis.

a. /kɔɗʊ-dĩ/

vowel	1	2	3
UR	ɔ	ʊ	ĩ
$[\text{low}]_i(x)$	⊥	⊥	⊥
$[+\text{ATR}, +\text{hi}]_i(s(x))$	⊥	⊤	⊥
$[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(s(x))$	⊥		⊥
$[\text{ATR}]_i(x)$	⊥		⊤
$[\text{ATR}]_o(x)$	⊥	⊤	⊤
SR	ɔ	u	i

[kɔɗu-ni]

b. /r-ɛ-hʊkɔɗɛ=r-e/

vowel	1	2	3	4	5
UR	ɛ	ʊ	ɔ	ɛ	e
$[\text{low}]_i(x)$	⊥	⊥	⊥	⊥	⊥
$[+\text{ATR}, +\text{hi}]_i(s(x))$	⊥	⊥	⊥	⊥	⊥
$[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(s(x))$	⊥	⊤	⊤	⊤	⊥
$[\text{ATR}]_i(x)$	⊥				⊤
$[\text{ATR}]_o(x)$	⊥	⊤	⊤	⊤	⊤
SR	ɛ	u	o	e	e

[rɛhukoɗere]

Table 1 Derivation of (13a) /kɔɗʊ-dĩ/ [kɔɗu-ni] 'turtle-similar.to' and (13d) /r-ɛ-hʊkɔɗɛ=r-e/ [rɛhukoɗere] 'CTFG-1+TRANS-lend-CTFG-IMPERF'.

Let us begin with vowel 1 in Table 1; i.e., the /ɔ/, of /kɔɗʊ-dĩ/. To compute its output value for [ATR], the computation evaluates $[\text{ATR}]_o(1)$—that is, $[\text{ATR}]_o(x)$ where x is interpreted as vowel 1. To do this, we substitute 1 for x in the right-hand side of definition (29) for $[\text{ATR}]_o(x)$, and then evaluate it.

Following (29), the first condition evaluates $[\text{low}]_i(1)$—that is, it checks to see if 1 is low. As is true of all of the vowels in these examples, 1 /ɔ/ is [−low], so this condition fails and the computation moves onto the next licensing structure. The next condition in the definition is $[+\text{ATR}, +\text{hi}]_i(s(x))$, so the computation checks $[+\text{ATR}, +\text{hi}]_i(s(1))$. As $s(1) = 2$ (i.e, the following vowel /ʊ/), $[+\text{ATR}, +\text{hi}]_i(2)$ is evaluated. Clearly, this returns ⊥, as /ʊ/ is not [+ATR]. (More precisely: definition (24) for $[+\text{ATR}, +\text{hi}]_i(x)$ states "`if` $[\text{ATR}]_i(x)$ `then` $[\text{high}]_i(x)$ `else` ⊥"; as $[\text{ATR}]_i(2)$ is ⊥, it returns the value of the `else` clause, which is ⊥.) Thus, the computation moves to the next licensing structure in the definition. This is $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(s(1))$, so the computation must evaluate $[+\text{ATR}, -\text{hi}]_o(2)$. By definition (25) of $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(x)$, this requires checking $[\text{ATR}]_o(2)$.

We have encountered recursion: the computation now must go back into the definition of $[\text{ATR}]_o(x)$, this time evaluating x as 2. Again, the first licensing structure in (29) is $[\text{low}]_i(x)$, which is false for 2 (/ʊ/). The computation then moves on to $[+\text{ATR}, +\text{hi}]_i(s(x))$. As $s(2) = 3$, the computation next evaluates $[+\text{ATR}, +\text{hi}]_i(3)$. This returns ⊤: 3, which is the vowel /ĩ/, is both [+ATR] and [+high]. As the licensing structure has evaluated to ⊤, the evaluation returns ⊤. As shown in the column for vowel 2 in Table 1, the computation has thus returned a value for $[\text{ATR}]_o(2)$: ⊤.

Returning to the evaluation for $[\text{ATR}]_o(1)$, the evaluation for $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(s(1))$—i.e. $[+\text{ATR}, -\text{hi}]_o(2)$—can now be completed. As the `if` condition $[\text{ATR}]_o(2)$ returns ⊤, according to (25) the computation returns the value of its `else` statement $\neg[\text{high}]_o(2)$. Since 2 is indeed [+high], this returns ⊥. Thus, the second licensing structure for $[\text{ATR}]_o(2)$ has returned ⊥, and so the evaluation turns to the last expression in definition (29), which is simply to return $[\text{ATR}]_i(1)$. As 1 is [−ATR], this returns ⊥, and so the final value for $[\text{ATR}]_o(1)$ is ⊥.

Finally, the computation evaluates 3, the final vowel /ĩ/. As this is the last vowel in the word, it has no successor—meaning that $s(3)$ is undefined. This means that the semantic convention that $\boldsymbol{B}(s(x))$ is ⊥ whenever $s(x)$ is undefined, mentioned in the previous section, plays a large role in the evaluation of $[\text{ATR}]_o(3)$. Because $s(3)$ does not exist, $[+\text{ATR}, +\text{hi}]_i(s(3))$ and $[+\text{ATR}, -\text{hi}, -\text{lo}, -\text{nas}]_o(s(3))$ immediately evaluate to ⊥, as shown in the final column of Table 1a. The value of $[\text{ATR}]_o(3)$ thus falls to the final condition of (29), $[\text{ATR}]_i(x)$. As /ĩ/ is [+ATR] in the input, $[\text{ATR}]_i(3)$ returns ⊤, and so does $[\text{ATR}]_o(3)$, as also shown in the final column of Table 1a.

Thus, as shown in the last two rows of Table 1a, vowels 1, 2, and 3 return values of ⊥, ⊤, and ⊤, for $[\text{ATR}]_o(x)$, respectively. That is, /ɔ/is correctly output as [−ATR], while /ʊ/, and /ĩ/ are correctly output as [+ATR].

In the evaluation for Table 1b for an input /r-ɛ-hʊkɔɗɛ=r-e/, recursion results in the propagation of a ⊤ value leftward. For brevity, we begin with vowel 5, /e/. (Because it is a logical expression, $[\text{ATR}]_o(x)$ can be evaluated for each segment in any order and the result will be the same.)

As with the last vowel in Table 1a, because /e/ is word-final it will return ⊥ for the licensing conditions for $[\text{ATR}]_o(x)$. Evaluation will then go to the default, $[\text{ATR}]_i(x)$. As this evaluates to ⊤ for /e/, the value for the entire expression $[\text{ATR}]_o(5)$ evaluates to ⊤.

The computation for $[\text{ATR}]_o(4)$ then shows how this ⊤ value propagates leftward. To evaluate $[\text{ATR}]_o(x)$ for vowel 4 (i.e., the second /ɛ/), after first checking $[\text{low}]_i(x)$ (and failing), the computation then checks the licensing structure $[+\text{ATR}, +\text{hi}]_i(s(x))$. This returns ⊥, as $s(4) = 5$ is [−hi] (as 5 is /e/). The computation thus checks the next licensing

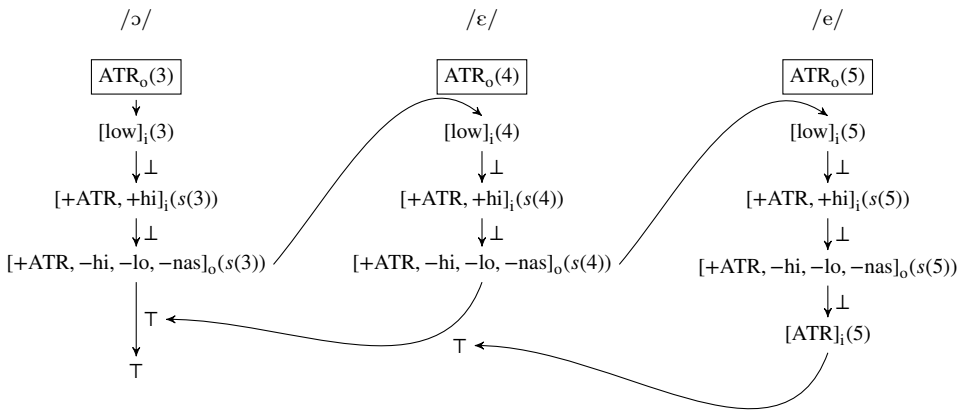


Figure 4 Flow of computation for segment 3 /ɔ/ in Table 1b.

structure, [+ATR, −hi, −lo, −nas]$_o$($s(x)$). This returns ⊤: for $s(4) = 5$, [ATR]$_o$(x) evaluates to ⊤, as just established. As /e/ is also [−hi], [+ATR, −hi, −lo, −nas]$_o$(s(4)) evaluates to ⊤. Thus, the licensing structure is satisfied and so [ATR]$_o$(4) returns ⊤, as shown in the column for 4 in Table 1b. Thus, the computation correctly computes this second /ɛ/ to be output as [e].

Vowels 2 /ʊ/ and 3 /ɔ/ follow the exact same computation as 4. For each vowel, its successor is a mid vowel that evaluates to ⊤ for [ATR]$_o$(x). Thus, for both 2 and 3 the licensing structure [+ATR, −hi, −lo, −nas]$_o$($s(x)$) will evaluate to ⊤, thus causing [ATR]$_o$(x) to return ⊤ for that vowel. In this way, recursive evaluation of [ATR]$_o$(x) allows [+ATR] to propogate leftward. This is shown explicitly in Figure 4, which depicts how the logic of the computation given in (30) is recursively evaluated in order to obtain the output value for vowel 3 /ɛ/.

The recursion stops at vowel 1, also /ɛ/, as it returns ⊥ for both licensing [+ATR, +hi]$_i$($s(x)$) and [+ATR, −hi, −lo, −nas]$_o$($s(x)$), and it is itself [−ATR] in the input so it returns ⊥ for the default condition [ATR]$_i$(x). Thus, as shown in Table 1b, the computation produces the correct output values for [±ATR] for each vowel in [rɛ-hukoɗere] 'CTFG-1+TRANS-lend-CTFG-IMPERF'.

4.3 Notes on the analysis

We have now shown how the definition in (29) correctly describes ATR spreading in Iny. Two related points are of note.

First, while the two licensing structures in (29) are ordered such that [+ATR, +hi]$_i$($s(x)$) precedes [+ATR, −hi, −lo, −nas]$_o$($s(x)$), this is not necessary. Because [+ATR, +hi]$_i$($s(x)$) could only ever be true when the successor of x is a high vowel, and [+ATR, −hi, −lo, −nas]$_o$($s(x)$) could only ever be true when x's successor is a nonhigh vowel, the two structures denote complementary environments. Thus, the first licensing structure could equally be ordered after the second.

Second, there is no need to encode ordering in the formalism because BMRS predicates explicitly mark when a structure holds in the input and when it holds in the output. This directly captures what is happening in Iny: mid vowels spread their output [+ATR] value, but high vowels only spread their input [−ATR] value. In the next section we demonstrate that the ability to refer to input values enables a more observationally adequate and theoretically parsimonious account than is available in other formalisms.

5 Theory comparison

In §1 we indicated three key concerns for the generative tradition: (1) explanation, (2) typology, and (3) computation. In the following subsections we focus on the observational and descriptive (in)adequacy of various analyses of Iny, arguing that our BMRS analysis generates the correct outputs better than alternative OT analyses, and encodes the central descriptive generalizations without the unneccessary machinery required by a serial, rule-based analysis. As such, we contend that the BMRS analysis of Iny encodes the sort of analytical generalizations that linguists desire. This result complements existing work demonstrating the typological and computational aspects of the formalism. Notably,

Chandlee and Jardine (2021a) illustrates the typological potential for BMRS with the well-known typology of *NC̥ repairs while Bhaskar et al. (2020) outlines the computational properties of the formalism.

5.1 OT alternatives

5.1.1 Metrical analysis

McCollum (2022) argues that a metrical analysis of Iny is necessary in OT, contending that analyses employing string-based representations are unable to provide a descriptively accurate or formally tractable account of the pattern. The issue is that underlying high vowels initiate and undergo [+ATR] spreading but derived [+hi,+ATR] vowels fail to further propagate [+ATR] leftward. He contends that this is a form of non-iterativity, which is generally problematic for OT (Kaplan 2008). Following Jurgec (2011) and McCollum and Kavitskaya (2022), which present evidence that non-linear representations can encode non-iterative spreading in OT, McCollum (2022). develops a metrical analysis of the pattern.

In McCollum's analysis, leftward harmony in Iny is motivated by ALIGN-L, which interacts with a set of metrical constraints on harmony domains.[3] Crucially, in his analysis, harmony domains involve heads and non-heads, as in McCarthy (2004), Smolensky (2006), and Jurgec (2011). Harmony domains may overlap (Hyde 2002), and are regulated by two constraints; *HEAD, which delimits which segments may head a domain. For Iny, *HEAD[+HI,+ATR] militates against [+hi] vowel heads of [+ATR] domains. Extending the metrical analysis in Jurgec (2011), McCollum (2022) encodes a preference for binary domains via SPANBIN, which like FOOTBIN prohibits non-binary feet (McCarthy and Prince 1986, Prince and Smolensky 2004, Key and Bickmore 2014). In the tableaux below, metrical domains are notated with parentheses, and heads are indicated with underlining. In (31-32) we see how this OT analysis generates iterative regressive harmony among mid vowels, as well as the non-iterative assimilation of high vowels (icy targets). In (31), since there is no constraint banning mid vowels as heads of [+ATR] domains, iterative leftward spreading is generated. The key ranking in (31) is ALIGN-L » ID[ATR], though SPANBIN also compels some harmony in order to prevent a monosyllabic harmony span in (31a). Additionally, SPANBIN prevents unbounded autosegmental association in (31d) since it would result in a ternary domain.

(31) Mid vowel triggers and targets in Iny

	/r-ɛ-rɔ=r-e/	SPANBIN[+ATR]	ALIGN-L[+ATR]	ID[ATR]
a.	rɛrɔ(re)	*!	**	
b.	rɛ(rore)		*!	*
☞ c.	(rɛ(rɔ)re)			**
d.	(rɛrɔre)	*!		**

In (31) no high vowels were present, but in (32) two high vowels are present, one triggering and one undergoing harmony. Since SPANBIN is highly ranked, candidate (32a), the faithful candidate, and candidate (32d), the unbounded autosegmental spreading candidate, are both ruled out since they each contain non-binary harmony spans. Candidate (32b) is preferred over candidate (32c) because only one high vowel heads a harmony span. In this manner, the icy target behavior of high vowels is analyzed as the byproduct of two rankings, SPANBIN » *HEAD and *HEAD » ALIGN-L. Highly ranked SPANBIN forces spreading from an underlying high vowel, even though that results in a violation of *HEAD. In turn, the ranking *HEAD » ALIGN-L limits the extent of spreading from derived [+hi, +ATR] vowels, producing high vowel icy targets.

(32) High vowel triggers and targets in Iny

	/kɔɗʊ-d̃i/	SPANBIN[+ATR]	*HEAD[+HI,+ATR]	ALIGN-L[+ATR]	ID[ATR]
a.	kɔɗʊ(ni)	*!		**	
☞ b.	kɔ(ɗuni)		*	*	
c.	(ko(ɗu)ni)		**!		**
d.	(koɗuni)	*!	*		**

While McCollum's (2022) OT analysis can generate the correct outputs for these forms, there are a number of drawbacks to his analysis. First, his analysis requires metrical representations, which are known to be more expressive than the simpler, string-based representations employed in our analysis. Arguments for the superiority of non-linear

[3] McCollum (2022) and Ozburn and Leduc (2022) do not address the relationship between nasality and harmony in their analyses, and we follow suit since it does not bear on the larger comparisons to be made.

representations rest upon two bases: the inability of simpler representations to generate the pattern, and in cases where both can generate the pattern, the insight offered by the non-linear representation system that is either missed entirely or obscured by string-based representations. In the present case, neither of these are issues. First, string-based representations are capable of generating the pattern, as shown in Section 4. Second, our string-based analysis actually accounts for the data better than McCollum's analysis, which incorrectly predicts rightward spreading in some cases. Consider a word like [ritʃɔrɛ] "offspring" from (11i). In such a word, [+ATR] is underlyingly aligned with the left edge of the word, and so we expect no spreading. However, McCollum's analysis predicts non-iterative rightward spreading to satisfy SPANBIN (33).[4]

(33) Rightward spreading predicted by McCollum's analysis

	/ritʃɔrɛ/	SPANBIN[+ATR]	*HEAD[+HI,+ATR]	ID[ATR]
a.	(ri)tʃɔrɛ	*!		
✖ b.	(ritʃo)rɛ		*	*
c.	(ri(tʃo)re)		*	**!
d.	(ritʃore)	*!	*	**

Despite the fact that leftward directionality is directly encoded in the analysis, rightward spreading is possible. In addition to this directionality reversal, McCollum's analysis predicts the possibility of satisfying SPANBIN by epenthesis, metathesis, and a range of other possible repairs, an instance of the well-known "too many repairs" problem in OT (Steriade 2001).

5.2 Contrast preservation

Ozburn and Leduc (2022) contends that the Iny pattern is derivable with string-based representations in OT. Their analysis draws on work arguing for a direct implementation of contrast preservation in OT (Lubowicz 2003, Tessier 2004). Specifically, such work requires a family of constraints to evaluate the potential neutralization of a phonological contrast across different inputs. Thus, outputs of an input /x/, [x_1] ... [x_n] must be compared with outputs of related inputs, /y/ ... /z/, [y_1] ... [y_n] ... [z_1] ... [z_n]. If some output from underlying /x/, [x_n] is identical to some output of some comparison input, [y_n] from /y/, then a member of the family of PRESERVE CONTRAST constraints assigns a violation. In addition to comparing input-output mappings across related forms for PRESERVE CONTRAST constraints, their analysis requires each constraint in Con, both faithfulness and markedness constraints, to evaluate each input-output mapping in the tableau. Commas are used in the tableaux below to differentiate violations incurred from different input-output mappings.

This departure from canonical OT is noteworthy. Given that contrast in OT is entirely epiphenomenal, building theoretical devices into the grammar that crucially reference contrast across related input-output mappings is truly significant, and it is not at all clear what the empirical coverage or typological consequences of such an analysis are. In many respects, Tessier's (2004) amended version of Contrast Preservation Theory (Lubowicz 2003) bear similarities to output-output correspondence (Benua 2000) and Sympathy Theory (McCarthy 1999). In these various frameworks, the correspondence relation is extended beyond input-output mappings. In output-output correspondence, different outputs from morphologically related inputs enter into correspondence, while in Sympathy Theory outputs enter into a correspondence relation with another potential output (the Sympathy candidate). In each of these theories, faithfulness between these pairs of outputs is used to derive a range of empirical facts. In contrast, in Tessier (2004) and Ozburn and Leduc (2022), output candidates must *differ* from their correspondence pairs. The requirement that output candidates differ from related outputs is a tenant of anti-faithfulness (Alderete 2001). Therefore, Contrast Preservation relies on mechanisms similar to output-output correspondence and Sympathy to extend the correspondence relation to words derived from distinct inputs. Further, the correspondence relation is then leveraged to enforce anti-faithfulness between corresponding outputs. As Potts and Pullum (2002) note, the extended correspondence relation required by these theories departs from the syntax of canonical OT constraints and the larger logic of Optimality Theory.

Empirically, Ozburn and Leduc's (2022) analysis makes one problematic prediction, that the realization of an underlying [+hi, -ATR] vowel depends not only on the height and ATR features of the following vowel, but also the height of the preceding vowel. In cases where a high vowel target is preceded by a mid vowel, their analysis outputs a tie between icy target behavior – the high vowel undergoes but does not further spread the harmonic feature, and feature propagation, as seen in (34). Note, Ozburn and Leduc (2022) claim that this ranking predicts icy target behavior only

[4]Though it is also possible to treat medial [o] as the head of the initial domain in (33) to avoid violating *HEAD, this does not change the fundamental issue.

in this context. They do not however consider an input scenario with full harmony on the base and faithfulness to the neighboring input that differs only from the base in it's input value for the [+hi] vowel.

(34) Icy targets in Iny as contrast preservation

	/ɛ...ʊ...e/	/ɛ...u...e/	/e...u...e/	*[ʊ][+ATR]	PRESERVE [+HI, +ATR]	*[-ATR][+ATR]	Id[-ATR]
a.	ɛ...ʊ...e	ɛ...u...e	e...u...e	*!		*,*	
b.	ɛ...ʊ...e	e...u...e	e...u...e	*!		*	*
c.	ɛ...u...e	ɛ...u...e	e...u...e		*!	*,*	*
☞ d.	ɛ...u...e	e...u...e	e...u...e			*	*,*
☞ e.	e...u...e	ɛ...u...e	e...u...e			*	**
f.	e...u...e	e...u...e	e...u...e		*!		**,*

However, given the ranking *[ʊ][+ATR] » PRESERVE [+HI, +ATR], their analysis predicts that [+hi, -ATR] vowels will both alternate and spread the harmonic feature when preceded by another [+hi, -ATR] vowel, in contrast to their behavior after mid vowels. This prediction is seen in (35). In (34) a tie is generated, but in (35) the optimal candidate exhibits full harmony.

(35) Icy targets in Iny as contrast preservation

	/ʊ...ʊ...e/	/ʊ...u...e/	/u...u...e/	*[ʊ][+ATR]	PRESERVE [+HI, +ATR]	*[-ATR][+ATR]	Id[-ATR]
a.	ʊ...ʊ...e	ʊ...u...e	u...u...e	*!,*		*!,*	
b.	ʊ...ʊ...e	u...u...e	u...u...e	*!		*	*
c.	ʊ...u...e	ʊ...u...e	u...u...e	*!,*	*	*,*	*
d.	ʊ...u...e	u...u...e	u...u...e	*!		*	*,*
e.	u...u...e	ʊ...u...e	u...u...e	*!		*	**
☞ f.	u...u...e	u...u...e	u...u...e		*,*		**,*

Ozburn and Leduc (2022) defend the prediction that the behavior of the high vowels depends on both the [ATR] value of the following vowel and the height of the preceding vowel based on transcriptions from the description of verbal morphology in Ribeiro (2012). In (36a,b), we see that Ribeiro transcribes iterative harmony among the high vowels in these verbs, which they cite as evidence that high vowels propagate [+ATR] when preceded by highvowels. Problematically for Ozburn and Leduc (2022), in his discussion of verbal morphology Ribeiro also reports words like (36c-e). In these words, high vowels both undergo and propagate [+ATR] even when preceded by mid vowels. In (36c) an underlying mid vowel precedes the high vowel, and both surface as [+ATR]. In (36d,e) the surface mid vowel that undergoes harmony is derived to resolve [aɪ] in hiatus. In (36f) we see that [aɪ] in hiatus is generally fused to [ɛ]. Therefore, the [e] that surfaces in (36d,e) must be due to harmony.

(36) Verbal morphology in Iny

a.	/ɗa-d-ɪ-Ø-wɨ=ɗ-e/	[ɗadiwɨ̰ɗe]	'2-CTPT-TRANS-3-carry=2-IMP'
b.	/Ø-d-ɪ-Ø-wɨ=d-e/	[diwɨ̰de]	'3-CTPT-TRANS-3-carry=3-IMP'
c.	/d-a-d-Ø-ɛbʊrɛ=d-e/	[nadeburede]	'CTPT-1-CTPT-INTR-see=2-CTPT-IMP'
d.	/ɗa-Ø-ɪ-Ø-wɨ=ɗ-e/	[ɗediwɨ̰ɗe]	'2-CTFG-TRANS-3-carry=2-IMP'
e.	/r-a-Ø-Ø-ɛbʊrɛ=r-e/	[reburere]	'CTFG-1-CTFG-INTR-get.angry=CTFG-IMP'
f.	/r-a-Ø-ɪ-rɔ=bə̃/	[rɛrɔbə̃]	'CTFG-1-CTFG-TRANS-eat=CONV'

How can we resolve the apparent inconsistency between the data marshalled in (13) and the data just discussed in (36)? Let us first consider Ribeiro's description, quoted below:

> Therefore, high and mid [-ATR] vowels differ in the way they undergo vowel harmony. As illustrated by the examples (11) – (15) above, the open-mid vowels /ɛ/ and /ɔ/ undergo vowel harmony iteratively, while with the high lax vowels /ɪ/, /ɨ/, and /ʊ/, harmony may optionally occur only locally. There is actually a great deal of variation in the extent to which vowel harmony can take place with high [-ATR] vowels, not only from speaker to speaker, but also within the speech of a single individual. Such variations need to be further investigated (108–109).

> The examples below, involving the suffix -dĩ 'similar to', further illustrate the differences between mid and high [-ATR] vowels. While harmonization of [-ATR] high vowels may optionally take place iteratively when triggered by aspectual clitics such as =r-e 'imperfective', it seems to be strictly local [they are icy targets] when triggered by the derivational suffix -dĩ (109).

Ribeiro's description makes it clear that there is a difference between the mid and high vowels, and that lexical and/or morphological differences are at play. Ribeiro's transcriptions rarely indicate optionality, so it is difficult to ascertain from his grammar more than his prose suggests. We conclude that the most reasonable description of the pattern is as follows: mid vowels iteratively undergo and spread [+ATR]; high vowels undergo [+ATR] harmony, but further spreading is optional, and additionally depends on lexical and morphological factors. Analytically, we have two choices, to treat the high and mid vowels the same, which is inconsistent with Ribeiro's general description, or to differentiate them in the way that we have. We do not have sufficient data to formulate gradient categories between categorical participation and non-participation. With this in mind, it is clear that Ozburn and Leduc's (2022) analysis is not empirically adequate. If their analysis can account for (36a,b), it cannot account for (36c-e). More generally, their analysis depends on a bidirectional dependency that simply is not there. The behavior of high vowels does not depend on the height of the preceding vowel, but rather the [ATR] value of the following vowel along with poorly understood morphological distinctions.

Thus, the Contrast Preservation analysis of Iny suffers from numerous shortcomings. On the empirical side, their analysis cannot adequately account for the data as presented in Ribeiro (2002, 2012). On the formal side, it critically depends on theoretical devices that radically transform the architecture of the grammar and greatly increase the power of the formalism, i.e., output-output correspondence and anti-faithfulness.

5.3 Rule-based alternative

In contrast to the two constraint-based analyses just discussed, a rule-based analysis in the formalism developed in Chomsky and Halle (1968) is actually fairly straightforward. This rule-based analysis requires two interacting rules, shown in (37). Non-linear representations and output-output correspondence are unnecessary, a simple set of ordered rewrite rules is sufficient to generate the pattern.

(37) *SPE*-style rules
High trigger rule: V[-lo] → [+ATR] / ___ C_0 [+hi, +ATR]; [-iterative]
Mid trigger rule: V[-lo] → [+ATR] / ___ C_0 [-hi, -nasal, +ATR]; [+iterative]

These two rules are able to account for all the data in (9-13). We demonstrate this in the derivations in (38–39). In the leftmost column, the two rules do not interact, and the correct output obtains via the non-iterativity of the high trigger rule, HTR, which assimilates only a single preceding [-ATR vowel]. In both the middle and rightmost columns, the two rules interact. In the center column, the mid trigger rule, MTR, motivates the assimilation of /ɪ/ to [i], with the icy target behavior of the high vowel owing to the ordering of HTR before MTR. In the rightmost column, HTR feeds MTR, resulting in the full assimilation of all preceding mid vowels in the word.

(38) Sample derivations

UR	/kɔɗʊ-dĩ/	/r-ɛ-hɪ=r-e/	/brɔrɛ-dĩ
HTR	|kɔɗuni|		|brɔreni|
MTR		|rɛhire|	|brorenɪ|
SR	[kɔɗuni]	[rɛhire]	[broreni]

(39) Sample derivations with rule ordering reversed

UR	/kɔɗʊ-dĩ/	/r-ɛ-hɪ=r-e/	/brɔrɛ-dĩ
MTR		[rɛhire]	
HTR	|kɔɗuni|	|rehire|	|brɔrenɪ|
SR	[kɔɗuni]	*[rehire]	*[brɔreni]

The *SPE*-style analysis depends on two crucial components – direct reference to (non)iterativity, and rule ordering. Concerning iterativity, the rules must specify whether or not the output of each rule can feed itself, for HTR must not feed itself while MTR must be self-feeding to generate the data at hand. Our BMRS analysis is superficially similar, referring to the input qualities of following high vowels, but the output quality of following mid vowels. Despite the similarity between referencing inputs and outputs in BMRS and the rule feature [±iterative], there are substantial

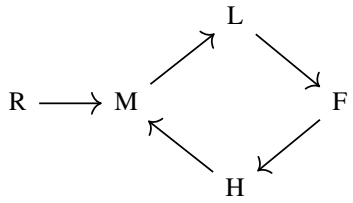
differences between the two. Non-iterative patterns are input-oriented (input strictly local), but input-oriented patterns are not all non-iterative, as discussed in (Chandlee 2014, Heinz 2018, Chandlee and Jardine 2021b). Input-orientedness thus subsumes non-iterativity as a construct. BMRS can directly reference input or output feature values, but an *SPE*-style analysis can only do so via rule ordering. The first rule may reference the input, but subsequent rules may only reference the output of earlier rules. Moreover, the need for extrinsic rule ordering is evident above – if HTR and MTR are reordered, two of the three words are output incorrectly (39). However, our BMRS analysis does not require any necessary ordering. Despite the fact that the triggers for each rule in (37) are non-intersecting, the rewrite rule analysis requires ordering. Yet, the BMRS analysis is successful without extrinsic ordering precisely because the two generalizations involve non-intersecting triggers - the set of [+hi,+ATR] inputs does not intersect with the set of [-hi, +ATR] outputs.

The issue here is the fact that rewrite rules cannot reference features values from earlier stages in the derivation; each rule has access only to the output of the previous rule. In many cases, this dependence on the output of earlier rules has no obvious effect on the empirical coverage of the theory. However, in some cases the differences between access to inputs and outputs and access only to the output of earlier rules emerges. One particularly problematic case involves circular chain shifts, which present a rule ordering paradox. In general, a counterfeeding order is necessary for chain shifts, but in circular shifts one rule inevitably feeds another, generating the wrong output. As Oakden (2021) demonstrates, this crucial dependence on ordering renders formalisms predicated on ordered rewrite rules largely incapable of accounting for circular chain shifts. Consider the data from Xiamen (Cheng 1968, Chen 1987, 2000) below in (40), diagrammed in (41). Each tone undergoes sandhi in non-final position. In non-final positions, rising tones surface as mid (40a), mid tones surface as low (40b), low tones surface as falling (40c), falling tones surface as high (40d), and high tones surface as mid (40e).

(40) Xiamen tone circle (Chen 1987)

a.	H /p'ang	F tsui/	⟶	M [p'ang	F tsui]	"perfume" (literally, fragrant + water)
b.	R /we	L tua/	⟶	M [we	L tua]	"shoe laces"
c.	M /pĩ	R lang/	⟶	L [pĩ	R lang]	"patient" (literally, sick + person)
d.	L /ts'u	F ting/	⟶	F [ts'u	F ting]	"roof" (literally, house + top)
e.	F /hai	R kĩ/	⟶	H [hai	R kĩ]	"ocean front"

(41) Diagram of Xiamen tone circle



If we attempt an *SPE*-style analysis of these data, as shown in (42), there is no way to generate the correct output for all input tones. Each pair of rule demands a counterfeeding order, but there is no way to generate this; inevitably, one rule will feed another. In this derivation, the informal rule mapping falling tones to high feeds the rule mapping highs to mids, incorrectly predicting surface *MT from /FT/.

(42) *SPE*-style analysis of Xiamen tone sandhi

UR	/RT/	/MT/	/LT/	/FT/	/HT/
F-to-H	–	–	–	HT	–
L-to-F	–	–	FT	–	–
M-to-L	–	LT	–	–	–
R-to-M	MT	–	–	–	–
H-to-M	–	–	–	MT	MT
SR	[MT]	[LT]	[FT]	*[MT]	MT

The challenge of the Xiamen tone circle isn't its computational complexity, as Oakden (2021) shows that the pattern is input strictly local (Chandlee 2014, Heinz 2018), one of the simplest classes of subregular functions. More generally, opaque maps like counterfeeding are input-oriented (Chandlee et al. 2018), further illustrating the fact that input-orientedness is not the same as non-iterativity.

Moreover, the challenge of the tone circle isn't the seeming arbitrariness of the mappings. The problem for an *SPE*-style analysis, and an OT analysis for that matter, is the circularity of the mapping (Moreton 2004, Barrie 2006, Tesar 2014). Representational solutions have been posited (Yip 1980), but these are have been subject to critique.[5] Without some ad hoc representational patch or way to stipulate disjunctive rule application, circular counterfeeding is not tenable for a formalism built upon ordered rewrite rules, even though circular tone shifts are attested in number of language families. Despite the fact that similar patterns are both computationally simple and attested in human language this formalism cannot adequately model them, leading some authors to – in our estimation – incorrectly dismiss them as non-phonological, historical accidents (Anderson 1978, Schuh 1978, Moreton 2004).

However, Oakden (2021) develops a simple BMRS analysis of the Xiamen tone circle. The following briefly summarizes his analysis. First, the following define what it means to be a non-final R tone, a non-final M tone, etc. In what follows, $\mathrm{R_i}(x)$ is true iff x is rising, $\mathrm{M_i}(x)$ if it is mid, etc., in the input, and $\mathrm{T_i}(x)$ refers to x being a syllable of any tone.

(43) a. $\mathrm{RT_i}(x) = \texttt{if } \mathrm{R_i}(x) \texttt{ then } \mathrm{T_i}(s(x)) \texttt{ else } \bot$

b. $\mathrm{HT_i}(x) = \texttt{if } \mathrm{H_i}(x) \texttt{ then } \mathrm{T_i}(s(x)) \texttt{ else } \bot$

c. $\mathrm{MT_i}(x) = \texttt{if } \mathrm{M_i}(x) \texttt{ then } \mathrm{T_i}(s(x)) \texttt{ else } \bot$

d. $\mathrm{FT_i}(x) = \texttt{if } \mathrm{F_i}(x) \texttt{ then } \mathrm{T_i}(s(x)) \texttt{ else } \bot$

e. $\mathrm{LT_i}(x) = \texttt{if } \mathrm{L_i}(x) \texttt{ then } \mathrm{T_i}(s(x)) \texttt{ else } \bot$

A BMRS analysis of the Xiamen then defines how a syllable becomes rising, mid, low, falling, or high in the output. More formally, it is a series of definitions of $\mathrm{R_o}(x)$, $\mathrm{M_o}(x)$, $\mathrm{L_o}(x)$, $\mathrm{F_o}(x)$, and $\mathrm{H_o}(x)$, as given in (44).

(44)
$$\begin{aligned}
\mathrm{M_o}(x) &= \texttt{if } \mathrm{RT_i}(x) \texttt{ then } \top \texttt{ else} \\
&\quad\ \texttt{if } \mathrm{HT_i}(x) \texttt{ then } \top \texttt{ else} \\
&\quad\ \texttt{if } \mathrm{MT_i}(x) \texttt{ then } \bot \texttt{ else } \mathrm{M_i}(x) \\
\mathrm{L_o}(x) &= \texttt{if } \mathrm{MT_i}(x) \texttt{ then } \top \texttt{ else} \\
&\quad\ \texttt{if } \mathrm{LT_i}(x) \texttt{ then } \bot \texttt{ else } \mathrm{L_i}(x) \\
\mathrm{H_o}(x) &= \texttt{if } \mathrm{MT_i}(x) \texttt{ then } \top \texttt{ else} \\
&\quad\ \texttt{if } \mathrm{HT_i}(x) \texttt{ then } \bot \texttt{ else } \mathrm{H_i}(x) \\
\mathrm{F_o}(x) &= \texttt{if } \mathrm{LT_i}(x) \texttt{ then } \top \texttt{ else} \\
&\quad\ \texttt{if } \mathrm{FT_i}(x) \texttt{ then } \bot \texttt{ else } \mathrm{F_i}(x) \\
\mathrm{R_o}(x) &= \texttt{if } \mathrm{RT_i}(x) \texttt{ then } \bot \texttt{ else } \mathrm{R_i}(x)
\end{aligned}$$

The definitions in (44) specify the conditions under which each tone surfaces. For example, the definition of $\mathrm{M_o}(x)$ first checks if x is either R followed by some tone ($\mathrm{RT_i}(x)$) or H followed by some tone ($\mathrm{HT_i}(x)$). If either of these is true, then it returns $\top$—that is, that x is an M in the output. Next, it checks to see if $\mathrm{MT_i}(x)$ holds, to see if x is an M in the input. If so, the expression returns $\bot$—that is, x cannot be output as M if it is a non-final M in the input. If none of the above conditions hold, then the expression returns $\mathrm{M_i}(x)$, the which indicates whether x was an M in the input.

The reader can confirm that these definitions correctly calculate the outputs in (40). Crucially, all of these definitions are computed in parallel—there is no ordering in how they are evaluated. Like the Iny analysis in (29), this analysis

[5] See Barrie (2006) and Oakden (2021) for further discussion of autosegmental approaches to circular tone shifts.

avoids any ordering by directly accessing the input. This reinforces the difference between the BMRS analyses above and the alternative rewrite rule-based analysis. The rule-based analysis posits a potentially large number of intermediate levels between inputs and outputs while the BMRS analysis, like OT and various other theories, posits only inputs and outputs with no intermediate representations. The *SPE*-style analysis mimics the input-orientedness of high vowel behavior by rule ordering – the high vowel rule must apply before the mid vowel rule, but this tactic fails for circular chain shifts. As our brief discussion of Xiamen illustrates, rule ordering is not equivalent to the direct reference to inputs and ouputs available to BMRS. Serial rule-based theories like *SPE* cannot account for some input-oriented patterns, like the chain shift in Xiamen, and they also rely on a potentially large number of independently unmotivated intermediate levels of representation.

5.4 Discussion

In the previous subsections we argued that a BMRS analysis of harmony in Iny is superior to those available in rule- and constraint-based formalisms. For OT, the challenge is a familiar one – when a phonological generalization is input-oriented, OT requires some amendment or additional theoretical device to capture it. For instance, linear chain shifts have been leveraged to introduce elaborate cross-derivational correspondence (Lubowicz 2003), constraint conjunction (Kirchner 1996), Sympathy (McCarthy 1999), and the reintroduction of derivational levels (Rubach 2000), among others. Yet, in the analysis of, to use serialist terms, fed counterfeeding in Iny many of those same devices fail.

For a derivational theory, rule ordering offers a single device with which to derive input-oriented generalizations. As discussed in McCarthy (1999), Rubach (2000), and Burzio (2001), accounting for a range of patterns with a single theoretical tool is appealing. However, rule ordering cannot account for all input-oriented generalizations because each rule only has access to the output of the preceding rule, and not direct access to the input. As a result, circular chain shifts like that in Xiamen (Chen 1987, Oakden 2021) fail under rule ordering. Relatedly, rule ordering cannot, in and of itself, account for (non)iterativity, as seen by work in the late 60s and early 70s on rule application (Chomsky and Halle 1968, Johnson 1972, Jensen and Stong-Jensen 1973, Anderson 1974). While bounded assimilation of a single target was unproblematic, inducing unbounded assimilation across a string of targets was more challenging for *SPE*. Parenthesis-star notation was found to be too powerful, predicting a range of decidedly non-local effects. Other devices, including directional rule application or tagging rules with an iterativity feature all require the introduction of additional machinery. In BMRS, though, both (non)iterativity and pairwise rule interactions are accounted for by direct reference to input and outputs. Direct access to inputs obviates the need to mark rules for iterativity, and in the case of Iny (and as far as we know, all cases of fed counterfeeding) to rely on anything resembling rule ordering. Although licensing and blocking structures in BMRS are capable of capturing a wide range of phenomena (Chandlee and Jardine 2021a), the conditions for harmony are entirely disjoint in Iny, and do not interact in our analysis.

At this point, one might wonder if our argumentation is really predicated on a particularly niche character of BMRS. More directly, perhaps BMRS is uniquely suited to handling input-oriented generalizations, including both counterfeeding and non-iterativity, while less successful at accounting for other types of patterns. Though a number of theories have exhibited this shortcoming (see Kenstowicz and Kisseberth 1979 for discussion), BMRS is just as adept at accounting for output-oriented generalizations. Recall from above that vowels in Iny are output as [+ATR] if they are followed by an output [-hi, +ATR] vowel. This dependence on outputs is canonical of iterative as well as feeding interactions. BMRS accounts for these generalizations, which is further evident in Oakden's (2021) discussion of various Asian tonal patterns.

The total Iny pattern is composed of input- and output-oriented generalizations. OT struggles with input-oriented papers because markedness constraints in OT have no access to inputs, though these are crucial to an analysis of ATR harmony in Iny. In contrast, derivational rule-based theories struggle on two fronts, limited access to inputs renders them largely unable to model circular chain shifts, while no access to outputs prevents the kind of analytical and typological insights offered by OT. Thus, it would seem that an adequate theory of phonology needs access to both inputs and outputs, a core feauture of BMRS.

6 Conclusion

In this paper we've presented a BMRS analysis of regressive ATR harmony in Iny. Direct reference to inputs and outputs allows BMRS to capture the height-based generalizations in Iny without necessary reference to rule ordering or iterativity. Further, direct access to inputs and outputs enables BMRS to capture a wider range of phonological patterns than ordered rewrite rule formalisms, e.g., circular chain shifts. BMRS offers intensional characterizations of phonological patterns in a manner similar to OT, which is how the formalism is capable of explicit typological predictions. The ways these generalizations are encoded permits the type of falsifiable typologies that phonologists

have used over the last 30 years to assess model fit. Additionally, BMRS does so by pitting competing generalizations against one another within the `if ... then ... else ...` syntax of the formalism. These conflicting pressures, crucially, are resolved locally, whereas optimization in OT is global. The result of this is that BMRS is computationally reasonable while OT critically fails to provide a sufficiently expressive or maximally restrictive model of human grammar. In these ways, we argue that BMRS better meets the general criteria laid out at the outset of the paper. It encodes the sort of generalizations that phonologists expect of a formal model. BMRS does this while restricting the space of possible language to the regular languages, and is thus computationally more akin to rule-based formalisms. As such, BMRS offers not only a superior analysis of Iny, but a larger grammatical architecture that combines the best properties of its various formal predecessors.

References

Alderete, J. D. (2001). Dominance effects as transderivational anti-faithfulness. *Phonology*, 18(2):201–253.

Anderson, S. R. (1974). *The organization of phonology*. Academic Press.

Anderson, S. R. (1978). Tone features. In Fromkin, V., editor, *Tone: A linguistic survey*, pages 133–175.

Barrie, M. (2006). Tone circles and contrast preservation. *Linguistic Inquiry*, 37:131–141.

Benua, L. (2000). *Phonological relations between words*. Psychology Press.

Bhaskar, S., Chandlee, J., Jardine, A., and Oakden, C. (2020). Boolean monadic recursive schemes as a logical characterization of the subsequential functions. In *International Conference on Language and Automata Theory and Applications*, pages 157–169. Springer.

Burness, P. A., McMullin, K. J., Chandlee, J., Burness, P., and McMullin, K. (2021). Long-distance phonological processes as tier-based strictly local functions. *Glossa: a journal of general linguistics*, 6(1).

Burzio, L. (2001). Zero derivations. *Linguistic Inquiry*, 32(4):658–677.

Chandlee, J. (2014). *Strictly local phonological processes*. PhD thesis, University of Delaware.

Chandlee, J., Eyraud, R., and Heinz, J. (2015). Output strictly local functions. In *14th Meeting on the Mathematics of Language*, pages 112–125.

Chandlee, J. and Heinz, J. (2018). Strictly locality and phonological maps. *LI*, 49:23–60.

Chandlee, J., Heinz, J., and Jardine, A. (2018). Input strictly local opaque maps. *Phonology*, 35(2):171–205.

Chandlee, J. and Jardine, A. (2021a). Computational universals in linguistic theory: Using recursive programs for phonological analysis. *Language*, 97(3):485–519.

Chandlee, J. and Jardine, A. (2021b). Input and output locality and representation. *Glossa: a journal of general linguistics*, 6(1).

Chen, M. Y. (1987). The syntax of xiamen tone sandhi. *Phonology*, 4:109–149.

Chen, M. Y. (2000). *Tone sandhi: Patterns across Chinese dialects*, volume 92. Cambridge University Press.

Cheng, R. L. (1968). Tone sandhi in taiwanese. *Linguistics*, 6:19–42.

Chomsky, N. (1951/2013). *Morphophonemics of Modern Hebrew (Routledge Revivals)*. Routledge.

Chomsky, N. and Halle, M. (1968). *The sound pattern of English.* Harper & Row.

Frank, R. and Satta, G. (1998). Optimality theory and the generative complexity of constraint violability. *Computational linguistics*, 24(2):307–315.

Fulop, S. A. and Warren, R. (2014). An acoustic analysis of advanced tongue root harmony in Karaja. In *Proceedings of Meetings on Acoustics 167ASA*, volume 21. Acoustical Society of America.

Gafos, A. (1996). *The articulatory basis of locality in phonology*. PhD thesis, Johns Hopkins University.

Gerdemann, D. and Hulden, M. (2012). Practical finite state optimality theory. In *Proceedings of the 10th international workshop on finite state methods and natural language processing*, pages 10–19.

Halle, M. (1959). The sound pattern of russian. In *The Sound Pattern of Russian*. Mouton & Company.

Heinz, J. (2018). The computational nature of phonological generalizations. In Hyman, L. M. and Plank, F., editors, *Phonological Typology, Phonetics and Phonology*, pages 126–195. De Gruyter.

Hyde, B. (2002). A restrictive theory of metrical stress. *Phonology*, 19(3):313–359.

Jardine, A. (2016). Computationally, tone is different. *Phonology*, 33(2):247–283.

Jardine, A. (2019). Computation also matters: a response to pater (2018). *Phonology*, 36(2):341–350.

Jensen, J. T. and Stong-Jensen, M. (1973). Ordering and directionality of iterative rules. *Research on Language & Social Interaction*, 6(1-4):66–90.

Johnson, C. D. (1972). *Formal aspects of phonological description*. Mouton.

Jurgec, P. (2011). *Feature spreading 2.0: A unified theory of assimilation*. PhD thesis.

Kager, R. (1999). *Optimality theory*. Cambridge university press.

Kaplan, A. (2008). *Noniterativity is an emergent property of grammar*. PhD thesis, University of California, Santa Cruz.

Kenstowicz, M. and Kisseberth, C. (1979). *Generative phonology: Description and theory*. Academic Press.

Key, M. and Bickmore, L. (2014). Headed tone spans: Binarity and minimal overlap. *Southern African Linguistics and Applied Language Studies*, 32(1):35–53.

Kirchner, R. (1996). Synchronic chain shifts in optimality theory. *Linguistic Inquiry*, 27(2):341–350.

Lubowicz, A. (2003). *Contrast preservation in phonological mappings*. PhD thesis, University of Massachusetts Amherst.

McCarthy, J. J. (1999). Sympathy and phonological opacity. *Phonology*, 16(3):331–399.

McCarthy, J. J. (2004). Headed spans and autosegmental spreading.

McCarthy, J. J. and Prince, A. (1986). Prosodic morphology.

McCollum, A. G. (2022). Non-iterativity, icy targets, and the need for non-linear representations in feature spreading. In *Proceedings of the 2021 Annual Meeting on Phonology*, volume 9. Linguistic Society of America.

McCollum, A. G. and Kavitskaya, D. (2022). On the status of non-iterativity in feature spreading. *Glossa: a journal of general linguistics*, 7(1).

Mielke, J. (2004). P-Base 1.95. http://137.122.133.199/jeff/pbase.

Moreton, E. (2004). Non-computable functions in optimality theory. In McCarthy, J. J., editor, *Optimality Theory in phonology: A reader*, pages 141–164. Blackwell Publishing.

Moschovakis, Y. N. (2019). *Abstract recursion and intrinsic complexity*, volume 48 of *Lecture Notes in Logic*. Cambridge University Press.

Oakden, C. D. (2021). *Modeling phonological interactions using recursive schemes*. PhD thesis, Rutgers, The State University of New Jersey.

Ozburn, A. and Leduc, M. (2022). Icy targets in karajá atr harmony as contrast preservation. In *Supplemental Proceedings of the 2021 Annual Meeting on Phonology*, volume 9. Linguistic Society of America.

Pater, J. (2019). Phonological typology in optimality theory and formal language theory: goals and future directions. *Phonology*, 36(2):351–353.

Potts, C. and Pullum, G. K. (2002). Model theory and the content of ot constraints. *Phonology*, 19(3):361–393.

Prince, A. and Smolensky, P. (2004). *Optimality Theory: Constraint interaction in generative grammar*. John Wiley & Sons.

Ribeiro, E. R. (2000). []ATR] vowel harmony and palatalization in karajá. *Santa Barbara Papers in Linguistics*, 10:80–92.

Ribeiro, E. R. (2002). Directionality in vowel harmony: The case of Karajá (Macro-Jê). In *Proceedings of the 28thAnnual Meeting of the Berkeley Linguistics Society*, pages 475–485.

Ribeiro, E. R. (2012). *A grammar of Karajá*. PhD thesis, The University of Chicago.

Rodrigues, A. D. (1999). Macro-Jê. In Dixon, R. M. and Aikhenvald, A. Y., editors, *The Amazonian languages*, pages 165–206. Cambridge University Press.

Rubach, J. (2000). Glide and glottal stop insertion in slavic languages: A dot analysis. *Linguistic inquiry*, 31(2):271–317.

Schuh, R. G. (1978). Tone rules. In Fromkin, V., editor, *Tone: A linguistic survey*, pages 221–256.

Smolensky, P. (2006). Optimality in phonology ii: Harmonic completeness, local constraint conjunction, and feature domain markedness. In Smolensky, P. and Legendre, G., editors, *The harmonic mind: From neural computation to optimality-theoretic grammar*, volume 2, pages 27–160. MIT Press.

Steriade, D. (2001). The phonology of perceptibility effects: The p-map and its consequences for constraint organization.

Tesar, B. (2014). *Output-driven phonology: Theory and learning*. Number 139. Cambridge University Press.

Tessier, A.-M. (2004). Input “clusters” and contrast preservation in OT. In *Proceedings of WCCFL*, volume 23, pages 101–114.

Vaysse, O. (1986). Addition molle et fonctions *p*-locales. *Semigroup Forum*, 34:157–175.

Yip, M. J. (1980). *The tonal phonology of Chinese*. PhD thesis, Massachusetts Institute of Technology.