CrossMark

*Phonology* **35** (2018) 171–205. © Cambridge University Press 2018
doi:10.1017/S0952675718000027

# *Input Strictly Local opaque maps**

**Jane Chandlee**
Haverford College

**Jeffrey Heinz**
Stony Brook University

**Adam Jardine**
Rutgers University

This paper gives a computational characterisation of opaque interactions in phonology. Specifically, a range of opaque interactions are shown to be Input Strictly Local (ISL) maps (Chandlee 2014), which are string-to-string functions that determine output based only on contiguous sequences of input symbols. Examples from Baković's (2007) extended typology of counterfeeding, counterbleeding, self-destructive feeding, non-gratuitous feeding and cross-derivational feeding, as well as a case of fed counterfeeding from Kavitskaya & Staroverov (2010), are all given ISL analyses, which show that these interactions can be computed based on sequences of bounded length in the input. It is discussed how ISL maps are restrictive in their typological predictions, have guaranteed learning results and known methods for generation and recognition, and thus compare favourably to rule-based and constraint-based approaches to these interactions.

## 1 Introduction

Opacity has long been a topic of interest in phonology (Kiparsky 1971, 1973), but has taken on new importance with the shift from rule-based to constraint-based theories of the phonological grammar. The inability of classic Optimality Theory (Prince & Smolensky 1993, Kager 1999) to straightforwardly account for cases traditionally analysed with ordered rules (Idsardi 1998, 2000) has led to a range of proposals for how to model seemingly non-optimal cases within the general OT framework. Such proposals include correspondence theory (McCarthy & Prince 1999), output–output faithfulness (Benua 1997), sympathy (McCarthy

* E-mail: JCHANDLEE@HAVERFORD.EDU, JEFFREY.HEINZ@STONYBROOK.EDU, ADAM.JARDINE@RUTGERS.EDU.

The authors would like to thank the following people for valuable feedback and discussion related to this work: Bill Idsardi, the three anonymous reviewers and audiences at GALANA 2015, GLOW 2015, UPenn, UC Berkeley, UCSD and the University of Chicago.



https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

1999) and candidate chains (McCarthy 2007). An alternative approach has been to call into question the psychological reality of opaque generalisations (Sanders 2003, Kawahara 2015), under the assumption that if they are not synchronically productive then the theory need not account for them anyway. In other words, the lack of a mechanism for accounting for opaque interactions would be an advantage, not a shortcoming, of a theory of the synchronic phonological grammar. Whether or not opacity is 'real' is an empirical question, one we do not aim to settle here. Readers interested in this question may consult McCarthy (2007) and references therein for a review of the evidence against denying that opacity is a distinct and psychologically real phenomenon.

Opacity is of interest from a computational perspective because of the complications it has posed for modern theories of generative phonology and because of claims that it is more 'complex' or 'difficult to learn' (as suggested in Kiparsky 1971, 1973). In this paper we investigate the computational nature of opacity, and show that opaque interactions are in fact computationally just as simple as single processes.

Specifically, we study the typology of opaque generalisations provided by Baković (2007) in the light of Chandlee's (2014) theory of local, subregular functions. We show that the variety of opaque generalisations discussed by Baković, and an additional case from Kavitskaya & Staroverov (2010), have the property of being Input Strictly Local (ISL), the definition of which is given in §2. Baković's typology includes a variety of opaque cases, including ones analysed as counterfeeding (both on environment and on focus), counterbleeding, self-destructive feeding, non-gratuitous feeding and cross-derivational feeding. Kavitskaya & Staroverov (2010) call their case fed counterfeeding. The fact that all such generalisations are ISL functions shows they are not necessarily more complex than single processes, and also comes with strong learnability results – the ISL class is known to be efficiently learnable from positive data (Chandlee *et al*. 2014, Jardine *et al*. 2014), in the sense of de la Higuera (1997) (a paradigm which can be thought of as adding efficiency guarantees to the paradigm introduced in Gold 1967).

It should be noted that in the spirit of inclusiveness, we do not attempt to define opacity formally, as any such definition will likely exclude one or more of the above generalisations that others have called 'opaque'.[1] In addition, it is not clear how to construct a formal definition in a theory-neutral way – in fact, Baković (2007) notes that cross-derivational feeding in Lithuanian may or may not be opaque, depending on how one views it (for more on this see §3.2). Without such a definition, we cannot establish a formal, provable relationship between opaque maps and ISL ones, but that is not the point of the paper. Rather, our goal is to demonstrate that the various types of interactions which researchers have identified as opaque all share the computational property of being ISL. In other words, opacity does not necessarily disrupt the ISL property.

[1] One potential candidate would draw on Tesar's notion of output-drivenness, in that opaque maps are those that are *not* output-driven. But Tesar himself rejects this, saying that 'the two properties are fundamentally distinct' (2014: 14).

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

In this paper, the only assumption we make is that phonological grammars relate underlying representations to surface representations. Following Tesar (2014), we use the term 'map' to describe this phonological transformation.[2] We then establish that, in terms of a well-defined computational property, a variety of specific reported instances of opacity are no different from single processes (i.e. maps without interactions).

Because our analyses focus on the computational nature of the maps themselves, the analyses are *independent* of traditional phonological formalisms. This approach has precedent in the literature (see Baković 2013, Tesar 2014). This focus on the maps themselves means the conclusions we draw are not about the nature of phonological rules or phonological constraint-based grammars, but the nature of the input–output map that all such formalisms aim to describe. This focus notwithstanding, there are significant ramifications for theories of phonology, which we discuss in §4. There we argue that these results make the case that a theory of phonological grammar which embraces subregular properties like ISL has more support from generation, recognition, typology and learnability than a theory which embraces optimisation.

This paper is organised as follows. §2 presents the background needed for the analyses of opaque interactions presented in §3. In §4 we compare our approach to other types of phonological theories (i.e. rule- and constraint-based) before concluding in §5.

## 2 Background

The foundational premise to the computational analyses presented in this paper is that phonological grammars posit transformations: they ultimately describe a function or map that takes URs as inputs and computes corresponding SRs as output.

### 2.1 Intensional and extensional descriptions

It will be helpful to make clear a distinction between grammars, on the one hand, and the objects grammars describe, on the other. Grammars are INTENSIONAL descriptions of objects, and the objects themselves are EXTENSIONAL. The objects may be infinite in size, but the intensional descriptions must be finite. These distinctions are common in mathematics. Lines on a Cartesian plane are sets of infinitely many points, and have many intensional descriptions. For example, they can be described with an equation of finite size of the form $y = ax + b$; they can also be described with any two distinct points on the line, such as $(x, y)$ and $(x', y')$. However the equations and points are not the lines themselves, but descriptions of them. They are intensional descriptions (grammars), while the infinite set of

[2] Throughout the paper we will use the more general terms 'input' and 'output' instead of UR and SR, since individual phonological maps based on rule representations are embedded in a larger grammar such that the input/output of a given map may not correspond to the UR/SR.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

points is the extension of these descriptions. While these are familiar concepts in mathematics, and perhaps hardly worth mentioning, it is important to keep this distinction in the forefront of one's mind when thinking about grammars, since the distinction is equally valid. What is the extension that a grammar describes?

To make the point concrete in phonology, consider a language where word-final vowels delete. There are different grammars which can describe this fact. One could formulate the *SPE*-style rewrite rule shown in (1a), while in an OT grammar one might (in part) employ the ranking shown in (1b).

(1) a. $V \rightarrow \emptyset \,/\, __ \#$ b. $\textsc{Final}(C) \gg \textsc{Max}(V)$

What are the extensions of these grammars? We argue that, like lines on Cartesian planes, the extension can be understood as an infinite set of points, with the left coordinate being the input form and the right coordinate being the output form. This infinite set of points includes pairs like (kapa, kap), (tiki, tik) and (plabo, plab). We assume this extension also includes points like (rek, rek), (malarakalark, malarakalark) and (b, b). This is consonant with the principle of the rich base (Prince & Smolensky 1993) and with the assumption that a rule's output is identical to its input when it fails to apply. We can now ask what *kind* of set of points this is. More specifically, what computational properties explain the nature of the sets of points that represent phonological maps? In the next section we define one such property: Input Strict Locality.

## 2.2 Input Strictly Local functions

We will first define ISL functions through a metaphor, and then provide a more precise, formal characterisation. First, the metaphor. You are a computer asked to return the surface representation $y$ of some underlying representation $x$. You begin this process by reading from the beginning of $x$ and moving through the string left to right. As you read $x$, and perhaps before you finish reading $x$, you begin writing the output $y$ also from left to right. The key idea behind Input Strictly Local functions is this: there is some positive integer $k$, such that what you write at any given point in this process only depends on the last $k$ symbols you read in $x$. Figure 1 illustrates this point, with $k = 2$.

The metaphor of reading and writing from left to right and paying attention to only the most recent $k$ symbols in the input is useful, but inaccurate in one important way. As it turns out, it does not matter if the string is read left to right or right to left. All that matters is that if the input $x$ is $n$ symbols long (so $x = x_1 x_2 \ldots x_n$, with each $x_i$ a letter), we can think of the output $y$ as the concatenation of $n$ strings (so $y = y_1 y_2 \ldots y_n$, with each $y_i$ a string), where each $y_i$ only depends on the string $x_{i-k+1} \ldots x_i$ (which is of length $k$). The order in which each $y_i$ is computed does not really matter.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

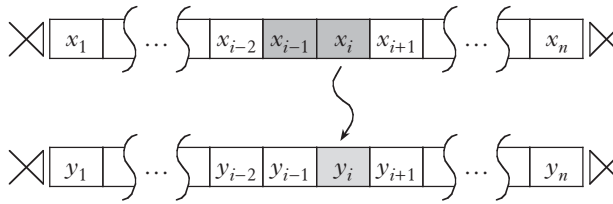



*Figure 1*

A schematic illustrating the Markovian nature of Input Strictly $k$-Local functions. For every Input Strictly 2-Local function, the output string $y_i$ of each input element $x_i$ depends only on $x_i$ and $x_{i-1}$. In other words, the light grey cell depends only on the dark grey cells.

One might think that ISL functions can only describe surface representations which are at least as long as the underlying representations they are computed from, because there are $n$-many $y_i$ strings. However, we should not forget that $y_i$ can also be the EMPTY string, the unique string of length zero (which we denote $\lambda$). Not only does this make ISL functions able to model deletion processes, it also allows them to describe phonological processes with both left and right contexts. Examples like this are provided in the following section and throughout §3.

## 2.3 Finite-state characterisation of ISL functions

ISL functions have been characterised in terms of automata theory, formal language theory (Chandlee 2014, Chandlee *et al*. 2014) and first order logic (Chandlee & Lindell to appear). Here we provide a finite-state transducer (FST) characterisation. This characterisation is derived from the original definition, which is provided in Appendix A.[3]

What is a FST? Figure 2 shows a FST for the example of word-final deletion discussed earlier. A FST includes a finite set of states (shown as circles), with one state being designated the start state (the one with the unlabelled incoming arrow) and some set of states designated as final states (circles with double perimeters). It also includes a set of transitions between states that determine (i) how the state changes when reading some input letter, and (ii) what output is written; transitions are shown as arrows between states with the label a:w, meaning letter a is read and string w is written.

Figure 2 is a FST whose extension is a map for word-final vowel deletion. An input $x$ is mapped to an output $y$ only if there is a path from state 1 (the start state) to state 2 (in this FST the only final state) which reads a sequence corresponding to $x$ and writes a sequence corresponding to $y$. In this FST, C and V serve as variables such that any input segment in the class of consonants or vowels can follow a transition marked C or V.

[3] The appendices are available as online supplementary materials at https://doi.org/10.1017/S0952675718000027.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

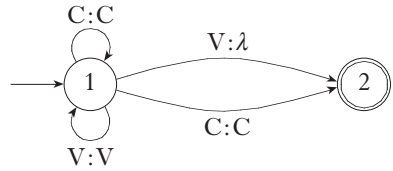



*Figure 2*
FST for word-final vowel deletion.

For example, the input string /CVC/ is mapped to [CVC] because the path in (2a) is a valid path in the FST (i.e. starts in the start state and ends in a final state). (2b) is a path which computes the output of /CVCV/.

(2) a.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| *input* | | C | | V | | C | | |
| *state* | 1 | → | 1 | → | 1 | → | 2 | |
| *output* | | C | | V | | C | | |

b.

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| *input* | | C | | V | | C | | V | |
| *state* | 1 | → | 1 | → | 1 | → | 1 | → | 2 |
| *output* | | C | | V | | C | | λ | |

Note that any path which ends in state 1 is not valid. Also, note that an input like /CVV/ will map to [CV], so only one final vowel is deleted. Thus this map is identical to the extension of the *SPE*-style rule in (1a) applying simultaneously, and it is not identical to the extension of the OT grammar in (1b), which deletes all final vowels. To make a FST which deletes all final vowels one would have to add a transition from state 2 to itself with the label V:$\lambda$. This FST would now match the extension of the OT grammar in (1b), as well as the extension of the *SPE*-style rule in (1a) applying right to left or persistently.

One kind of FST is necessary and sufficient to describe ISL functions. Figure 3 shows an ISL FST with the same extension as the FST in Fig. 2. Recall that a map is ISL if there is a positive integer $k$ such that the $i$th part of the output string depends only on the last $k$ letters up to and including the $i$th symbol read. In a FST, this is encoded in the states: basically, each state corresponds to the last $k-1$ letters read in the input, which means there must be a state for each $k-1$ sequence of input that *could* be read.[4] This in turn means the ISL FST will have more states than the non-ISL version, so for readability the FST in Fig. 3 assumes a reduced alphabet with one consonant and one vowel: {t, a}.[5]

[4] If the input read so far contains fewer than $k$ letters, then the state just represents the string read so far (there is an exception involving the final state, discussed below).

[5] A reviewer questions whether an ISL FST with an alphabet that includes all consonants and vowels is in fact equivalent to the representations in (1a) and (1b) and to

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

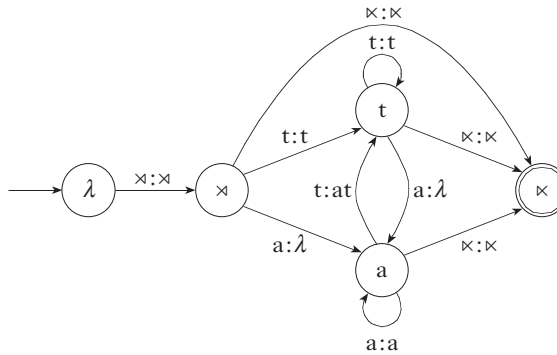



*Figure 3*
The ISL FST for word-final vowel deletion.

Instead of indexing states with numbers as we did in Fig. 2, it makes sense to label states with the sequence they correspond to. Another difference in Fig. 3 is that both the input and the output strings explicitly encode beginning and end markers, using the symbols ⋊ and ⋉ respectively.[6] In Fig. 3, $k = 2$, so five states are needed to keep track of possible input sequences up to length $k - 1$ or 1. These states are $\lambda$, t, a, ⋊ and ⋉.

The transitions in an ISL FST are organised in a very simple, specific way. Basically, if the FST is in a state corresponding to the sequence t and the letter a is read, then the FST transitions to the state which corresponds to the last $k - 1$ letters of ta. When $k = 2$, as in Fig. 3, this means whenever a is read, the FST transitions to state a, and whenever t is read it transitions to state t. There is only one final state (labelled ⋉), and every transition which reads the end marker ⋉ goes to this final state.

To see how this FST works, the valid paths in (3a) and (b) show the mapping of *tat* to *tat* and *tata* to *tat* respectively.

(3) a.

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *input* | | ⋊ | | t | | a | | t | | ⋉ | | |
| *state* | $\lambda$ | → | ⋊ | → | t | → | a | → | t | → | ⋉ | |
| *output* | | ⋊ | | t | | $\lambda$ | | at | | ⋉ | | |

b.

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *input* | | ⋊ | | t | | a | | t | | a | | ⋉ |
| *state* | $\lambda$ | → | ⋊ | → | t | → | a | → | t | → | a | → | ⋉ |
| *output* | | ⋊ | | t | | $\lambda$ | | at | | $\lambda$ | | ⋉ |

These examples illustrate how the states and the ability to write the empty string $\lambda$ allow the ISL FST to deterministically decide whether to write a

the FST in Fig. 2, which all use standard natural class variables. It would be equivalent, because it would have the same extension.

[6] The technical motivation for explicitly encoding the beginnings and ends of strings is explained in Jardine *et al.* (2014).

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

given vowel or not. When an a is read, it is not written out right away. This is because, at that point, as the string is processed left to right, it is unknown whether that a is word-final or not. If it is word-final, nothing further is written, and the FST moves to the final state upon reaching the end marker. If it is not, the first symbol of the next part of the output is always a.

## 2.4 Properties of ISL functions

ISL functions have several important properties that are important to the study of phonology. First, Chandlee (2014) and Chandlee & Heinz (2018) discuss how all phonological processes which can be described with a rule $A \rightarrow B \,/\, C __ D$ applying simultaneously are ISL functions, provided only finitely many strings match the structural description (or equivalently, provided there is a longest string in CAD). This excludes long-distance phenomena (such as consonant harmony) that involve segments interacting over an arbitrary number of intervening segments.

Second, Chandlee & Heinz also explain that ISL functions do not necessarily correspond to individual processes. Maps that include multiple phenomena (such as multiple ordered rules in rule-based grammars or many interacting constraints in OT grammars) can still be ISL functions. A precise account of which such maps are ISL and which are not is an outstanding goal. This paper makes a contribution to this goal, by showing that a variety of opaque maps are ISL.

Third, ISL functions are a proper subclass not only of the regular relations, but also of the class of SUBSEQUENTIAL functions. Regular relations are those definable with FSTs of the sort illustrated in Fig. 2. Subsequential functions are definable by DETERMINISTIC FSTs (see Chandlee & Heinz 2016 for further discussion). Thus the ISL functions are an extremely restrictive class in terms of the kind of computations they can represent. The importance of this restriction to phonological theory will be discussed further in §2.5.

Fourth, ISL functions have a clear cognitive interpretation. Rogers *et al.* (2013) explain how different subregular classes of formal languages can be understood in terms of the different kinds of memory they require (and hence distinguish). The kind of memory needed to compute an ISL function is fleeting: it is only necessary to scan the input word $w$ and remember the most recent substring of length $k$. Everything else can be thrown out. It is not necessary to compare a substring in one part of $w$ with a substring in another part of $w$. Nor is it necessary to remember the number of times particular substrings occurred in $w$. This is how ISL functions are like the Strictly Local (SL) formal languages discussed by Rogers & Pullum (2011) and Rogers *et al.* (2013). Cognitively, this is one of the simplest forms of memory.

Fifth, $k$-ISL functions are provably learnable by algorithms that are very efficient in both time and data (Chandlee 2014, Jardine *et al.* 2014, Chandlee, Jardine & Heinz 2015). This means that, given any $k$-ISL function $f$ and a sufficient finite set of points exemplifying $f$, these algorithms

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

invariably return a FST whose extension is $f$. Furthermore, the time the algorithm takes is reasonable with respect to the size of its input data. Additionally, the amount of data needed is also reasonable with respect to the size of the ISL FST corresponding to $f$. Here, 'reasonable' has a clear meaning from the theory of computational complexity; the above-mentioned papers provide the details. These results are significant, because comparable results do not exist for the subsequential class nor for the class of regular relations. While a learning algorithm exists for the subsequential class (Oncina *et al*. 1993), it is less efficient than the algorithms for $k$-ISL. And, provably, no algorithm exists which exactly identifies the class of regular relations from positive data.

### 2.5 Relevance to phonological theory

Above it was stated that the ISL functions are a restrictive class of maps – they are subclasses of both the subsequential and regular classes of maps. To better situate the results presented in the next section, we first comment here on why this is important for phonological theory, and where other phonological theories lie in terms of the maps that they can describe.

Those unfamiliar with formal language theory might wonder why restrictiveness matters, but an analogy with OT will hopefully help make the relevance clear. Suppose there were three competing sets of constraints under typological investigation (call them $\textsc{Con}_1$, $\textsc{Con}_2$ and $\textsc{Con}_3$). If it became known that the typology of $\textsc{Con}_3$ was properly included in $\textsc{Con}_2$, which was in turn properly included in $\textsc{Con}_1$, this would be a fact worth knowing. It would tell us that any map describable with $\textsc{Con}_3$ is describable with $\textsc{Con}_2$ and $\textsc{Con}_1$, but not *vice versa*.

Likewise with these classes of maps. Every ISL function is subsequential and regular, but not *vice versa*. This means ISL functions are restrictive in a computationally important way, and so an ISL theory of phonology draws a small circle around a large class of phonological maps. Such maps are indeed also regular (as shown by Johnson 1972 and Kaplan & Kay 1994), but the 'ISL circle' is a tighter, and therefore stronger, characterisation. Importantly, this is a better characterisation – in computational terms – than characterisations in OT and *SPE*. *SPE* rules describe the entirety of the regular maps (Johnson 1972, Kaplan & Kay 1994), and OT can describe maps outside of the regular class (Frank & Satta 1998, Riggle 2004, Gerdemann & Hulden 2012).

It is often pointed out that *SPE* rules or OT grammars more easily capture the distinction between 'natural' and 'unnatural' processes, and therefore provide better typological predictions than computational restrictions. However, 'naturalness', in common practice, is implemented as a metatheoretic restriction based on phonetic principles on possible rules or constraints, and their interactions or rankings (as in the papers in Hayes *et al*. 2004). We argue that these same types of restrictions could also be used to further restrict the class of ISL functions to a class that is both computationally appealing and phonetically natural.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

However, since the focus of the current paper is to characterise the *computational* nature of opaque maps, we do not attempt to do so here.

To summarise, subregular function classes like ISL present a learnable, restrictive, computational theory of phonological maps. The next section explores the kinds of opaque maps ISL functions are capable of describing, using Baković (2007) and Kavitskaya & Staroverov (2010) as a catalogue of the known typology of opaque maps. These works were chosen because they provide a more comprehensive collection of opaque phenomena that go beyond the classic examples of counterbleeding and counterfeeding (Kiparsky 1971, 1973). Despite the nuanced differences among the types of opaque maps we will analyse, the fact is that all have the ISL property.

## 3 Opaque ISL maps

In this section we demonstrate that four categories of opaque maps can be modelled with ISL functions. These cases are listed in (4a). The first three are discussed by Baković (2007); the case of fed counterfeeding in Tundra Nenets comes from Kavitskaya & Staroverov (2010). (4b) contains three additional categories of opaque maps, detailed in Appendix B.

(4) a. Cross-derivational feeding in Lithuanian
Counterbleeding in Yowlumne
Non-gratuitous feeding in Classical Arabic
Fed counterfeeding in Tundra Nenets

b. Counterfeeding on environment in Bedouin Arabic
Counterfeeding on focus in Bedouin Arabic
Self-destructive feeding in Turkish

We will present each example using the original author's analysis, though this does not amount to a claim that their analysis is correct. Our aim is simply to demonstrate that interactions that have been called opaque are ISL maps. It is also important to note that while the examples are presented in a rule-based framework (as is common for opaque generalisations), and therefore include two distinct generalisations that interact opaquely, our analyses study the opaque generalisation *directly* as an input–output map. That is, we did not proceed by first modelling each generalisation as an ISL function and then taking the composition of the two. Informally, the composition of two functions is a single function that directly computes the result of running the two component functions one after the other.[7] This is important, because it means our results do not depend on any particular decomposition of a given opaque generalisation.

[7] Formally, if $f$ and $g$ are functions then their composition $f \circ g$ is equivalent to using the output of $g$ as the input to $f$. For example, if $g(x) = y$ and $f(y) = z$, then $f \circ g(x) = f(g(x)) = f(y) = z$.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

To clarify the exposition, we shall discuss the examples in order of their $k$-value. Recall that $k$ is the size of the input substring that the function is mindful of at any given time. Cross-derivational feeding in Lithuanian is 2-ISL (i.e. $k = 2$); the next three cases are 3-ISL. (Two of the cases in Appendix B are 3-ISL, while self-destructive feeding in Turkish is 5-ISL.)

All it takes to accomplish our goal is to present an ISL FST for each opaque map (i.e. a FST whose extension is equivalent to the reported generalisation). However, the size of these FSTs can make it inconvenient to study them in the graphical form of Fig. 3. Therefore, in this section we introduce an alternative, tabular notation for ISL FSTs, which makes them much easier to examine. In particular, we will use what we call an ABBREVIATED INPUT/OUTPUT TABLE, which makes it easier to verify that a map is ISL and to determine the output string for a given input. Essentially, these tables provide all the information one needs to construct the ISL FST. In the online supplementary materials we also provide the complete FSTs for all of the example maps in §3 and Appendix B, in both table and graphical format.[8]

### 3.1 Input/output tables

The example ISL FST in §2 (Fig. 3) has a simple alphabet of two letters (t and a). To describe the phonology of a language at the level of its phonemic and phonetic inventory, it will be necessary to have symbols for each element of these inventories. Consequently, the number of states and transitions will increase dramatically, making the FST harder to read in graphical form. Generally, if there are $n$ letters in the alphabet, the number of states in a $k$-ISL FST is $n^k + 1$, because there are $n^k - 1$ states corresponding to the sequences up to length $k - 1$ plus the $\rtimes$ and $\ltimes$ states.

While the size of ISL FSTs may seem daunting, keep in mind that these are not the smallest FSTs with these extensions. There are efficient procedures which can make them much more compact (Mohri 1997). The size of ISL FSTs follows from their particular structure, which is designed to make clear that the functions they compute are ISL.

What we are calling an input/output table is basically a list of the transitions of the ISL FST. A full input/output table for the ISL FST in Fig. 3 which models word-final vowel deletion is shown in Table I. The rows in an input/output table serve the same purpose as the transitions of a FST: the current output is determined by the current $(k - 1)$-suffix (the sequence of $k - 1$ previous symbols) and the current input symbol. Recall that for the function depicted in Fig. 3, $k = 2$, so Table I lists 1-suffixes. For example, row (g) indicates that an a following a t should be output as $\lambda$. Because of the particular structure of ISL FSTs, the destination state of this transition can be inferred: it is the last $k - 1$ symbols of the 1-suffix concatenated to the new input. In this example, that would be the last $k - 1$, or 1, symbols of ta, so the FST would proceed to state a.

[8] A Python script is also provided that readers can use to generate the output string for any input using the FSTs for all of the languages analysed in §3.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

| | 1-suffix | input | output |
|---|---|---|---|
| a. | $\lambda$ | ⋊ | ⋊ |
| b. | ⋊ | ⋉ | ⋉ |
| c. | ⋊ | t | t |
| d. | ⋊ | a | $\lambda$ |

| | 1-suffix | input | output |
|---|---|---|---|
| e. | t | ⋉ | ⋉ |
| f. | t | t | t |
| g. | t | a | $\lambda$ |

| | 1-suffix | input | output |
|---|---|---|---|
| h. | a | ⋉ | ⋉ |
| i. | a | t | at |
| j. | a | a | a |

*Table I*
Full input/output table for the ISL FST in Fig. 3.

The process of mapping an input string to its output string can be thought of as sliding a window of length $k$ over the input from left to right. This window is moved one input symbol at a time, not unlike a FST. This process is demonstrated in (5) with the derivation /tat/ → [tat]. The left portion of the window reveals the current $(k-1)$-suffix (again in this example, it is a 1-suffix, since $k=2$). The right portion of the window includes the current input symbol. For each new input symbol, we look up its corresponding output given the $(k-1)$-suffix in the input/output table. To clarify the roles of each symbol, the shading in the derivation matches the shading in the table. Additionally, each step is labelled with its corresponding row letter from Table I. The first step, which corresponds to row (a), has been omitted.

(5) (c) ⋊ t a t ⋉ / ⋊ t  (g) ⋊ t a t ⋉ / ⋊ t $\lambda$  (i) ⋊ t a t ⋉ / ⋊ t $\lambda$ at  (e) ⋊ t a t ⋉ / ⋊ t $\lambda$ at ⋉

We will make these input/output tables even more succinct, by not showing those rows where the inputs and outputs are identical. An abbreviated input/output table is one which only shows cases where the inputs do not match the outputs. Table II shows an abbreviated input/output table for the ISL FST in Fig. 3. In contrast to Table I, Table II has only three rows. In addition to improving readability, the advantage of an abbreviated input/output table is that we can isolate the 'interesting' part of the FST, or the part that represents the structural change of the phonological map. For any input symbol and $(k-1)$-suffix not found in the abbreviated input/output table, it is understood that the output written is identical to the input (as in steps (c) and (e) in (5)).

| | 1-suffix | input | output |
|---|---|---|---|
| d. | ⋊ | a | $\lambda$ |
| g. | t | a | $\lambda$ |
| i. | a | t | at |

*Table II*
Abbreviated input/output table for the ISL FST in Fig. 3.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

To illustrate the kind of analysis we will conduct in this paper, consider a classic case of counterfeeding opacity: vowel lowering in Danish (Lundskaer-Nielsen & Holmes 2011). In Danish, front vowels lower in proximity to /r/. This occurs in a chain shift, in which each vowel lowers to the vowel one 'step' in height below it. To simplify the discussion, we focus on front vowels following /r/. A simple rule-based analysis is presented in (6), with the rules applying in the order presented.

(6) a. ɛ → æ / r__ b. e → ɛ / r__ c. i → e / r__

This lowering process is a 2-ISL function: whether or not an input /i/ is output faithfully depends on whether or not the preceding segment in the input is an /r/. We can describe this function exactly with the abbreviated input/output table in Table III.

| | 1-suffix | input | output |
|---|---|---|---|
| a. | r | i | e |
| b. | r | e | ɛ |
| c. | r | ɛ | æ |

*Table III*
Abbreviated input/output table for lowering in Danish.

An example derivation is given in (7).

(7) (c) ⋊ i r i ⋉ → ⋊ i r i ⋉ (a) ⋊ i r i ⋉ → ⋊ i r i ⋉
⋊ i ⋊ i r ⋊ i r e ⋊ i r e ⋉

Table III succinctly represents the map which lowers front vowels after /r/. We use such tables throughout this paper to show that opaque interactions of ISL phonological maps are themselves ISL – that is, for these interactions, there exists some $k$ such that the relevant changes from input to output are determined by $(k-1)$-suffixes in the input.[9] To make the tables more compact and readable, we will use symbols like C, N and V as variables in the conventional manner, though in the actual FSTs each segment would again be represented separately.

### 3.2 Cross-derivational feeding in Lithuanian

We begin with an interaction between epenthesis and voicing assimilation in Lithuanian that is 2-ISL. Baković (2007) identifies this as an example of what he calls cross-derivational feeding, a type of opacity in which one process applies in order to avoid a derivation in which another process would create a marked structure.

[9] It follows that a map that is not ISL could not be represented with such a table.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

In Lithuanian, [i]-epenthesis occurs between identical obstruents and voiceless obstruents assimilate to following voiced obstruents. In (8), D represents a voiced obstruent and K any obstruent, regardless of voicing.[10]

(8) a. *Epenthesis:* $\emptyset \rightarrow$ i / $K_1$ __ $K_2$, where $K_1 = K_2$
/at-taiki:ti/ ↦ [atitaiki:ti] 'to make fit well'
b. *Voicing assimilation:* K → [+voice] / __ D
/ap-gauti/ ↦ [abgauti] 'to deceive'

In Odden (2005)'s formulation, epenthesis occurs only between homorganic obstruents, but Baković (2007: 235) argues against this analysis, because it misses the generalisation that assimilation 'is bled in precisely those contexts where it would otherwise create pairs of completely adjacent consonants'. He instead suggests the version shown in (8a), though with either version of the rule the interaction of epenthesis followed by assimilation is not actually opaque, but a standard case of bleeding.

Opacity comes into play when we address the fact that these two rules cannot account for the example in (9), under either ordering.

(9) *Cross-derivational feeding*
/ap-berti/ ↦ [apiberti], *[abberti], *[abiberti] 'to strew all over'

The application of epenthesis and not assimilation to /ap-berti/ is a case of overapplication. Baković calls this cross-derivational feeding because the epenthesis appears to take place in order to *avoid a derivation* in which voicing assimilation would create the marked structure of identical obstruents. Though this is missed by the rule-based analysis, Baković shows that it can be handled rather straightforwardly in OT. As mentioned above, our analyses do not depend on the rule-based descriptions or on the ability to decompose the generalisation into multiple individual ones. Therefore, while not considered opaque in the rule-based analysis, we include this case in our survey because its *extension* (which is identical in both Odden's and Baković's analyses) reflects the type of opacity called cross-derivational feeding.

We now show that this interaction of epenthesis and assimilation can be modelled with a single 2-ISL map. The essence of this map's computation is the following. When a voiceless obstruent appears in the input, its output is temporarily delayed, until it can be determined whether or not it will be subject to voicing assimilation in (8b). This depends on the following segment. If the following segment is a voiced obstruent of a *different* place, the voiceless obstruent should be output as voiced, because epenthesis will not be triggered. If the following segment is an obstruent of the *same* place, then both the voiceless obstruent and [i] are output.

[10] For simplicity, we focus on voicing, and omit palatalisation from the transcriptions and discussion.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

| | 1-suffix | input | output |
|---|---|---|---|
| a. | ⋊ | t | $\lambda$ |
| b. | ⋊ | p | $\lambda$ |
| c. | ⋊ | d | d |
| d. | ⋊ | b | b |
| e. | V | t | $\lambda$ |
| f. | V | p | $\lambda$ |
| g. | V | d | d |

| | 1-suffix | input | output |
|---|---|---|---|
| h. | V | b | b |
| i. | t | V | tV |
| j. | t | ⋉ | t⋉ |
| k. | t | p | t |
| l. | t | b | db |
| m. | p | V | pV |
| n. | p | ⋉ | p⋉ |

| | 1-suffix | input | output |
|---|---|---|---|
| o. | p | t | p |
| p. | p | d | bd |
| q. | t | t | ti |
| r. | t | d | tid |
| s. | p | p | pi |
| t. | p | b | pib |

*Table IV*
Abbreviated input/output table for cross-derivational feeding in Lithuanian.

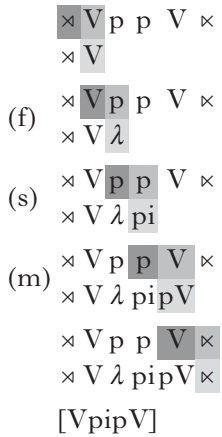
This interaction is 2-ISL, because a scanning window of size 2 is sufficient to distinguish the above conditions. Table IV illustrates these various conditions. First, voiceless obstruents, which may or may not be subject to assimilation, are initially output as $\lambda$ (rows (a, b, e, f)). Voiced obstruents, on the other hand, are always output unchanged (rows (c, d, g, h)).

When a voiceless obstruent is followed by a vowel (rows (i, m)), the end of the word (rows (j, n)) or a heterorganic voiceless obstruent (rows (k, o)), the voiceless obstruent is output faithfully. (In the latter case, the following heterorganic voiceless obstruent is not included in the output, as it may itself be subject to assimilation, depending on what comes next, per the discussion above.) If the following symbol is a heterorganic voiced obstruent, the preceding voiceless obstruent is output as voiced (rows (l, p)). The remaining cases of interest are when a voiceless obstruent precedes a homorganic obstruent. These are the cases in which epenthesis applies. When a voiceless obstruent is followed by an identical voiceless obstruent (rows (q, s)), the first obstruent is output followed by an [i]; thus for example, in (q) a /tt/ sequence is output as [ti]. In this case, the output of the second voiceless obstruent is withheld, as it itself may be followed by a trigger for voicing assimilation. When a voiceless obstruent is followed by a homorganic voiced obstruent, the voiceless obstruent is output followed by an epenthesised [i] and the voiced obstruent; thus in (r) an input /td/ sequence is output as [tid].

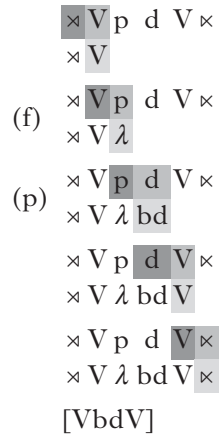
This final case is exactly the case of overapplication of epenthesis due to cross-derivational feeding. The preceding tables have shown this interaction to be definable by the 1-suffix of each input; thus it is 2-ISL. (10) illustrates how the correct outputs for the three crucial cases – epenthesis of identical segments, voicing in heterorganic segments and cross-derivationally fed epenthesis between homorganic segments differing in voicing – are obtained by a scanning window of size 2 implementing the input/output relations listed in the tables above. The row letters are from Table IV.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

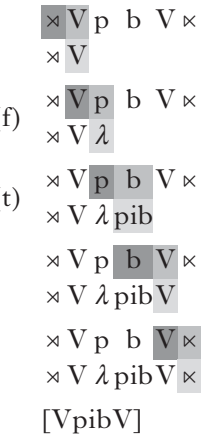
(10) a. /VppV/




b. /VpdV/




c. /VpbV/




### 3.3 Counterbleeding in Yowlumne

We now turn to a case of counterbleeding, from Yowlumne, and show that it is 3-ISL.[11] In Yowlumne, all long vowels become [−high], and vowels in closed syllables shorten, as shown in (11).

(11) a. [+long] → [−high]
b. V → [−long] / __ C{C, ⋉}

With long vowels in closed syllables, the application of shortening obscures the environment for lowering. The example in (12), from McCarthy (1999), shows apparent overapplication of (11a), because the subsequent application of (11b) removes the triggering long vowel.

(12) /mi:k-hin/ ↦ [mekhin] 'swallowed'

These processes can be viewed as a single map, whose computation can be summarised as follows: when a high long vowel is read, the output is temporarily delayed, until it can be determined whether (i) to both lower and shorten the vowel or (ii) just lower it. Condition (i) holds if the two segments after the vowel are /C⋉/ or /CC/, where /C/ ranges over all consonants. Condition (ii) holds if the next two segments are anything except /C⋉/ or /CC/.

This map is 3-ISL, because these conditions can be checked by scanning a three-segment window in the input.[12] (Note that /i:/ counts as one

[11] This language is referred to as Yawelmani Yokuts in McCarthy (1999), but the currently accepted name is Yowlumne (see Weigel 2005). We thank an anonymous reviewer for pointing this out, in addition to identifying an error in our original description and analysis of the shortening process.

[12] An anonymous reviewer adds that Yowlumne also has a rounding harmony process that both counterfeeds and counterbleeds lowering. The interaction of all three of

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

symbol.) The following abbreviated input/output tables, focusing on the lowering and/or shortening of an input /i:/, illustrate this. We assume the alphabet {i:, V, C}, where /i:/ represents long high vowels, and /C/ and /V/ are variables ranging over the set of consonants and the set of short vowels plus long non-high vowels respectively.

First, any input /i:/ will be initially output as $\lambda$, temporarily delaying its corresponding output – whether it is eventually output as [e:] or [e] depends on whether the following input is /C⋉/, /CC/ or something else. Thus its corresponding output is $\lambda$, regardless of the 2-suffix, as Table V illustrates with example rows for the input /i:/.

| | 2-suffix | input | output |
|---|---|---|---|
| a. | ⋉C | i: | $\lambda$ |
| b. | CC | i: | $\lambda$ |
| c. | VC | i: | $\lambda$ |
| d. | i:C | i: | $\lambda$ |
| e. | ⋉V | i: | $\lambda$ |
| f. | CV | i: | $\lambda$ |
| g. | VV | i: | $\lambda$ |
| h. | i:V | i: | $\lambda$ |
| i. | ⋉i: | V | e:V |

| | 2-suffix | input | output |
|---|---|---|---|
| j. | Ci: | V | e:V |
| k. | Vi: | V | e:V |
| l. | ⋉i: | C | $\lambda$ |
| m. | Ci: | C | $\lambda$ |
| n. | Vi: | C | $\lambda$ |
| o. | i:C | V | e:CV |
| p. | i:C | C | eCC |
| q. | i:C | ⋉ | eC⋉ |

| | 2-suffix | input | output |
|---|---|---|---|
| r. | CC | V: | $\lambda$ |
| s. | CV | V: | $\lambda$ |
| t. | VC | V: | $\lambda$ |
| u. | VC | V: | $\lambda$ |
| v. | CV: | C | $\lambda$ |
| w. | VV: | C | $\lambda$ |
| x. | V:C | C | VCC |
| y. | V:C | ⋉ | VC |

*Table V*

Abbreviated input/output table for counterbleeding in Yowlumne.

The change, then, hinges on inputs following an /i:/. If an /i:/ is followed by anything other than a /C/, it is not in the shortening environment. Thus, it can be output as [e:], with only lowering applying. Rows (i–k) in Table V illustrate this: a /V/ immediately following a /i:/ will be output as [e:V]. However, if /i:/ is followed by a /C/ (rows (l–n)), the output still cannot be determined: if it is a word-final /i:C/ sequence or if another /C/ follows, then it should be output as [eC], with both lowering and shortening; otherwise, it should be output as [e:C], with only lowering. Thus, the output must be again delayed, and so a /C/ following /i:/ is always output as $\lambda$.

Thus an input /i:C/ sequence is output as $\lambda$, as whether or not it is realised as [e:C] or [eC] is contingent on what comes next. An input /V/ following an /i:C/ sequence is output as [e:CV] (row (o)). This output comprises the output [e:C] of the /i:C/ 2-suffix (which was previously withheld), concatenated to the output [V] for the new input /V/. This can be

these processes is still ISL, but for $k = 6$. Under the assumption that the harmony takes place over at most two intervening consonants (e.g. /uC(C)i:/), a window of 6 would be required to check whether an input long high vowel is in both the contexts for rounding and shortening: /uC(C)i:C#/. We thank the reviewer for bringing out this interesting extension of the Yowlumne example.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

interpreted as follows: now that the /i:C/ has been seen to precede a /V/, it is clear that the /i:/ is not in the shortening environment, and so the output reflects only lowering. In contrast, as shown in rows (p) and (q), an input /C/ or /⋉/ following an input /i:C/ is output as [eCC] or [eC⋉] respectively, reflecting both lowering and shortening.

Lastly, rows (r–y) demonstrate the application of shortening without lowering. The output for /V:/ (which abbreviates all non-high long vowels) is [V] in the same contexts that /i:/ is shortened (i.e. closed syllables).

(13) presents the derivations /Ci:CV/ → [Ce:CV] and /Ci:C/ → [CeC], to show how the input–output correspondences in these tables correctly capture the opaque interaction. Row letters are from Table V.

(13) a. /Ci:CV/

| | | |
|---|---|---|
| | ⋉ C i: C V ⋉<br>⋉ C | |
| (a) | ⋉ C i: C V ⋉<br>⋉ C λ | |
| (m) | ⋉ C i: C V ⋉<br>⋉ C λ λ | |
| (p) | ⋉ C i: C V ⋉<br>⋉ C λ λ e:CV | |
| | ⋉ C i: C V ⋉<br>⋉ C λ λ e:CV ⋉ | |
| | [Ce:CV] | |

b. /Ci:C/

| | |
|---|---|
| | ⋉ C i: C ⋉<br>⋉ C |
| (a) | ⋉ C i: C ⋉<br>⋉ C λ |
| (m) | ⋉ C i: C ⋉<br>⋉ C λ λ |
| (q) | ⋉ C i: C ⋉<br>⋉ C λ λ eC⋉ |
| | [CeC] |

These derivations highlight the fact that a scanning window of size 3 is sufficient to model the combined shortening and lowering map – the longest $(k-1)$-suffix needed to produce the correct output for any input symbol is /i:C/, which is of length 2. Thus this opaque interaction is 3-ISL.

### 3.4 Non-gratuitous feeding in Classical Arabic

Another type of opacity discussed by Baković (2007) is non-gratuitous feeding, in which the structural description of one process is obscured by a second process that is fed by the first. For example, in Classical Arabic a vowel is epenthesised before an initial consonant cluster (rule (14a)).[13] This then feeds glottal stop epenthesis in (14b), as initial vowels must be preceded by a glottal stop in Classical Arabic.

[13] The epenthesised V is in fact identical to the V following the consonant cluster (see McCarthy 2007), but for notational simplicity we ignore the featural makeup of V. Ensuring that the vowels are identical would add 1 to the $k$-value for this process, in order to look ahead to see what vowel follows the consonant cluster. We also follow Baković (2007) in considering the linear version of these two processes (that is, with no reference to syllable structure). See that work for discussion.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

(14) a. /ktub/ ↦ [ʔuktub] 'write.MASC SG' b. i. $\emptyset \rightarrow V / \#__CC$
ii. $\emptyset \rightarrow ʔ / \#__V$

This interaction can be considered opaque because both the triggering environment for vowel epenthesis and its output are altered by the application of glottal stop epenthesis – that is, the resulting word-initial #VCC sequence is not word-initial in the output.

The combined map is 3-ISL. This can be shown fairly easily, as the crucial input–output correspondences are defined in the same word-initial environment, as summarised in Table VI.

| | 2-suffix | input | output |
|---|---|---|---|
| a. | ⋊ | V | ʔV |
| b. | ⋊ | C | $\lambda$ |

| | 2-suffix | input | output |
|---|---|---|---|
| c. | ⋊C | V | CV |
| d. | ⋊C | C | ʔVCC |

*Table VI*
Abbreviated input/output table for non-gratuitous feeding in Classical Arabic.

When the suffix is /⋊/, an input /V/ is output as [ʔV]. For the same suffix /⋊/, an input /C/ is instead output as $\lambda$. This is to withhold a word-initial /C/, as it may be the first member of a word-initial CC sequence, and thus subject to vowel epenthesis. The application of vowel epenthesis is then defined for inputs when the 2-suffix is /⋊C/. If a /V/ follows (row (c)), then vowel epenthesis does not apply, and [CV] is output. If a /C/ follows (row (d)), then both vowel epenthesis and glottal epenthesis apply, and so [ʔVCC] is output. Contrasting derivations for /CVC/ → [CVC] (when neither process is triggered) and /CCVC/ → [ʔVCCVC] (when both processes are triggered, corresponding to (12)) are shown in (15). As illustrated in (15), the interaction between these two processes is 3-ISL, as it can be captured with a scanning window of size 3.

(15) a. /CVC/

(b) ⋊ C V C ⋊
⋊ $\lambda$

(c) ⋊ C V C ⋊
⋊ $\lambda$ C V
⋊ C V C ⋊
⋊ $\lambda$ C V C
⋊ C V C ⋊
⋊ $\lambda$ C V C ⋊

[CVC]

b. /CCVC/

(b) ⋊ C C V C ⋊
⋊ $\lambda$

(d) ⋊ C C V C ⋊
⋊ $\lambda$ ʔ V C C
⋊ C C V C ⋊
⋊ $\lambda$ ʔ V C C V
⋊ C C V C ⋊
⋊ $\lambda$ ʔ V C C V C
⋊ C C V C ⋊
⋊ $\lambda$ ʔ V C C V C ⋊

[ʔVCCVC]

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

### 3.5 Fed counterfeeding in Tundra Nenets

Kavitskaya & Staroverov (2010) describe another type of opacity, fed counterfeeding. (16a) gives an example from Tundra Nenets, with the debuccalisation and vowel-deletion rules shown in (16b).

(16) a. i. /tasʌ/ ↦ [tas] 'whole'
ii. /tʲimjʌs/ ↦ [tʲimjʔ] 'it rotted'

b. i. {t, d, s, n, ŋ} → ʔ / __ #
ii. ʌ → Ø / __ (ʔ) #

In a mapping like (16a.i), we observe counterfeeding: if vowel deletion applied first then debuccalisation would also apply to derive *[taʔ]. But (16a.ii), which in a rule-based analysis proceeds as /tʲimjʌs/ ↦ tʲimjʌʔ ↦ [tʲimjʔ], instead demonstrates feeding. Thus the same two rules can exhibit both feeding and counterfeeding, depending on the input form.

Using the alphabet {ʌ, ʔ, T, C, V} (where T = {t, d, s, n, ŋ}, C is any consonant aside from /ʔ/ and those in T, and V is any vowel except /ʌ/), we can show that this combined fed counterfeeding map is 3-ISL. Since /ʌ/-deletion doesn't have a left context, the output of /ʌ/ is $\lambda$, no matter what 2-suffix precedes it. This is indicated in rows (a–e) of Table VII, where X denotes any segment.[14]

| | 2-suffix | input | output |
|---|---|---|---|
| a. | XC | ʌ | $\lambda$ |
| b. | XV | ʌ | $\lambda$ |
| c. | Xʔ | ʌ | $\lambda$ |
| d. | Xʌ | ʌ | ʌ |
| e. | XT | ʌ | T |
| f. | XC | T | $\lambda$ |
| g. | XV | T | $\lambda$ |
| h. | Xʔ | T | $\lambda$ |

| | 2-suffix | input | output |
|---|---|---|---|
| i. | Xʌ | T | $\lambda$ |
| j. | XT | T | T |
| k. | Xʌ | ʔ | $\lambda$ |
| l. | Xʌ | C | ʌC |
| m. | Xʌ | V | ʌV |
| n. | ʌT | ⋉ | ʔ⋉ |
| o. | CT | ⋉ | ʔ⋉ |
| p. | VT | ⋉ | ʔ⋉ |

| | 2-suffix | input | output |
|---|---|---|---|
| q. | TT | ⋉ | ʔ⋉ |
| r. | ʔT | ⋉ | ʔ⋉ |
| s. | ʌʔ | ⋉ | ʔ⋉ |
| t. | Cʌ | ⋉ | ⋉ |
| u. | Vʌ | ⋉ | ⋉ |
| v. | Tʌ | ⋉ | ⋉ |
| w. | ʔʌ | ⋉ | ⋉ |

*Table VII*
Abbreviated input/output table for fed counterfeeding in Tundra Nenets.

The situation is similar when the input is T. As the debuccalisation rule has no left context, whenever a T is input it must be withheld until it can be determined whether it is word-final (rows (f–j)). Again, the output of T when the 2-suffix ends in T (row (j)) is T, representing the return of the T in the suffix, which is now known not to be word-final. This is also why the output is T in row (e): the additional input of /ʌ/ verifies that the suffix T is not word-final, and therefore should be output unchanged. Note this is the

[14] The output in row (d) is [ʌ] because the /ʌ/ of the suffix was already output as $\lambda$. Since this suffix /ʌ/ is now known to not be word-final (because another /ʌ/ follows it), it can now be safely output.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

key to the fed counterfeeding interaction originally shown in (16a.i), where an input /s/ preceding a deleted /ʌ/ escapes debuccalisation.

When an input /ʔ/ follows a 2-suffix that ends in /ʌ/ (row (k)), the output is also *λ*, because if (and only if) it is word-final, this will also trigger deletion of /ʌ/. In contrast, if the input following a 2-suffix that ends in /ʌ/ is /C/ or /V/, then the output returns the withheld [ʌ], which we now know will not be deleted (rows (l, m)).

Since these are both word-final processes, the crucial rows of the input/output table (n–w) are those for which the input is /⋉/. When /⋉/ follows the 2-suffix /ʌT/ (row (n)), the context for both deletion and debuccalisation has been found, so the output is [ʔ⋉]. Debuccalisation also applies when any of the 2-suffixes /CT, VT, TT, ʔT/ (rows (o–r)) are followed by /⋉/. Deletion alone applies when /⋉/ follows the 2-suffix /ʌʔ/ (row (s)). When any 2-suffix that ends in /ʌ/ is followed by /⋉/, the /ʌ/ is deleted, as shown in rows (t–w). (17) illustrates this interaction with example derivations.

(17) a.

| | /CVTʌ/ |
|---|---|
| | ⋊ C V T ʌ ⋉ |
| | ⋊ C |
| | ⋊ C V T ʌ ⋉ |
| | ⋊ C V |
| (g) | ⋊ C V T ʌ ⋉ |
| | ⋊ C V *λ* |
| (e) | ⋊ C V T ʌ ⋉ |
| | ⋊ C V *λ* T |
| (v) | ⋊ C V T ʌ ⋉ |
| | ⋊ C V *λ* T ⋉ |
| | [CVT] |

b.

| | /CVCʌT/ |
|---|---|
| | ⋊ C V C ʌ T ⋉ |
| | ⋊ C |
| | ⋊ C V C ʌ T ⋉ |
| | ⋊ C V |
| | ⋊ C V C ʌ T ⋉ |
| | ⋊ C V C |
| (a) | ⋊ C V C ʌ T ⋉ |
| | ⋊ C V C *λ* |
| (i) | ⋊ C V C ʌ T ⋉ |
| | ⋊ C V C *λ* *λ* |
| (n) | ⋊ C V C ʌ T ⋉ |
| | ⋊ C V C *λ* *λ* ʔ ⋉ |
| | [CVCʔ] |

## 3.6 Discussion

This concludes our analysis of four distinct types of opaque phonological maps, all of which have been shown to be ISL functions for some value of *k*. Similar analyses of the other three cases in this typology are presented in Appendix B. Together, these analyses show that a seemingly diverse set of interactions can be given a unified computational analysis based on the common property of ISL. This does not, however, amount to a claim that opacity *itself* is ISL. Long-distance processes, which are not ISL, can also be involved in opaque interactions. Nonetheless, it is still a non-trivial result that every example in this typology of opaque maps is ISL.

While the object of each analysis presented above was the extension of the opaque interaction as a whole, each case can be (and has been)

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

thought of as an interaction between two individual processes. This raises two important questions. First, given two ISL maps with possibly different $k$-values, what will the $k$-value of the map that describes their interaction be? The second question is closely related: is the class of ISL functions closed under composition – that is, is the composition (see note 7) of any two ISL functions guaranteed to also be ISL? We address these two questions in turn.

Table VIII reviews the results of this section and Appendix B, presenting the $k$-values for each opaque map discussed, along with the $k$-values for the individual processes of each interaction.[15] For ISL maps corresponding to single phonological rules, determining the $k$-value is straightforward: $k$ is the length of the longest string in the structural description of the process. The longest $k$-value for one of the interactions, listed in the rightmost column, is 5, for Turkish, whereas the others are either 2 or 3. Note that there is no clear correlation between the size of the $k$-values for the basic processes and the $k$-value of the resulting interaction. For example, both epenthesis and assimilation in Lithuanian (§3.2) are 2-ISL, as is their interaction. In contrast, epenthesis in Turkish is 3-ISL and deletion is 4-ISL, whereas their interaction is 5-ISL. Thus the composition of two ISL processes is not simply the larger of their respective $k$-values, nor their sum. Predicting the $k$-value of the composition of two ISL processes appears to be a more subtle matter, involving how the output of the second process depends on the output of the first.

| opacity type | language | process A | process B | inter-action |
|---|---|---|---|---|
| cross-derivational feeding | Lithuanian | $k=2$ | $k=2$ | $k=2$ |
| counterbleeding | Yowlumne | $k=1$ | $k=3$ | $k=3$ |
| non-gratuitous feeding | Classical Arabic | $k=3$ | $k=2$ | $k=3$ |
| fed counterfeeding | Tundra Nenets | $k=2$ | $k=3$ | $k=3$ |
| counterfeeding on environment | Bedouin Arabic | $k=3$ | $k=3$ | $k=3$ |
| counterfeeding on focus | Bedouin Arabic | $k=3$ | $k=3$ | $k=3$ |
| self-destructive feeding | Turkish | $k=3$ | $k=4$ | $k=5$ |

*Table VIII*

The $k$-values for the ISL maps analysed in §3 and in Appendix B.

Nonetheless, in all of the cases analysed here, there exists some $k$ for which the map that combines the two ISL processes is $k$-ISL. This is not, however, true generally, as the ISL functions as defined in Chandlee (2014) are not closed under composition. A proof of this fact

[15] As stated in note 13, the $k$-value for process A in Classical Arabic would increase by 1, to reflect the fact that the inserted vowel is actually a copy of the first vowel in the word. The interaction $k$-value would then likewise increase to 4.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

can be found in Appendix B. Thus the interaction of two ISL processes is not itself *guaranteed* to be ISL. However, subsets of the ISL class may be closed under composition. Chandlee & Lindell (to appear) study ISL maps from a logical perspective, and identify a subclass which excludes the counterexample given in Appendix B, but includes all of the opaque interactions studied in this paper.

In sum, the preceding sections demonstrated that many cases of opacity are ISL for some $k$, as summarised in Table VIII. In this subsection, we have addressed two important theoretical questions about ISL and map interaction: (i) can we predict the $k$-value of the interaction of two ISL processes?, and (ii) is the class of ISL functions closed under composition? The answer to both of these questions is currently 'no', but ongoing research aims to identify the conditions under which interactions preserve the ISL property.

## 4 Comparison to other theories of phonology

It is natural to wonder how a theory of phonology which asserts that phonological transformations from underlying to surface forms are characterised by ISL functions compares to other theories of phonology. In this section we compare this theory to (i) a theory that asserts that phonological maps are generated by a list of rewrite rules in the style of Chomsky & Halle (1968) and (ii) classic Optimality Theory (OT; Kager 1999, Prince & Smolensky 1993), which asserts that phonological maps are generated by optimising over a set of ranked, violable constraints.

We acknowledge that theories are not monolithic entities, but contain many different variants and parts. OT is a good example. It is not possible to address every set of constraints proposed in the OT framework, as well as its variants, such as Harmonic Grammar (HG; Legendre *et al*. 1990) and Harmonic Serialism (HS; McCarthy 2000, Pater 2012). However, these theories all understand maps as optimisations of violable constraints, and there are consequences that necessarily follow from this core assumption. For this reason, we focus on their computational nature in the following discussion, and put aside other aspects.

The most important aspect we leave aside here is how phonetically grounded the theories are (Hayes *et al*. 2004). This is not a commitment to the belief that phonetic factors play no role in phonological theory – indeed we believe the most explanatory theory will identify how computational *and* phonetic factors interact to shape phonological typology. The focus of this paper is the contribution of the *computational* factors.

### 4.1 Points of comparison

We consider the following points of comparison: typological predictions, learnability, generation and recognition. Typological predictions refer to a theory's predictions as to the kinds of maps that should and should not

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

occur. Learnability refers to the results that have been obtained with respect to how the kinds of grammars the theories posit can be learned. Generation refers to whether algorithms exist which reliably compute a surface form from a given underlying form using the grammar. Recognition refers to the converse of the generation process: given a surface form and grammar, is there a procedure that can recover the possible underlying forms?

All four points of comparison are important. Typologically, it is preferable for theories to neither overgenerate nor undergenerate, though we acknowledge there is room for debate on the nature of this over- and undergeneration. For instance, it can be difficult to decide between two theories that overgenerate in different ways. However, computational complexity is a clear measure for overgeneration. We thus adopt the principle that theories which overgenerate simple patterns (computationally speaking) are to be preferred to theories which overgenerate complex patterns. This gives us a strong starting point from which to further restrict the theory, based on substantive constraints such as phonetic naturalness.

In addition, as human children somehow acquire their native language's phonology, a theory of phonology should provide some guarantee that the grammars it predicts to exist can be learned by provably correct learning algorithms. There are different ways to characterise what it means 'to learn' (Kearns & Vazirani 1994, Jain *et al*. 1999, de la Higuera 2010), and different types of learning problems in phonology can be identified. None of the results mentioned in this paper are complete solutions to the problem of phonological acquisition. Nonetheless, it ought not be controversial that algorithms which are well understood analytically and computationally efficient provide important landmarks in understanding how a child could generalise from linguistic stimuli. Indeed, theoretical learnability results have played an important role in the literature (Tesar & Smolensky 1996, 1998; see also Heinz & Riggle 2011 for a review of the relationship between learnability and phonology).

Finally, it is desirable for there to exist specific algorithms that solve the generation and recognition problems. Generation allows us to reliably compute predictions within a language. Without a generation algorithm, we cannot make reliable predictions about the output form of a given input form. And recognition is a form of parsing, allowing us to reliably interpret surface forms in terms of their potential underlying structure. While recognition may not receive as much attention as generation, recognition algorithms may play a role in some learning problems.

In the remainder of this section, we compare the ISL function theory with rule-based theories like *SPE* and with OT. We first compare the theories with respect to the generation and recognition problems (§4.2). Then we consider the learnability criterion (§4.3). Finally, we address typology (§4.4). Our conclusion is that, despite some known deficiencies, the ISL theory of phonology compares well, and has sufficient merit to be worthy of further study.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

### 4.2 Generation and recognition

The fact that ISL functions can be characterised by FSTs is significant. The generation and recognition problems are solved for FSTs. Algorithms like the ones mentioned in §2.4 guarantee that the correct output is generated from a given input. Likewise, it is well-known that FSTs can be inverted to do recognition (Beesley & Karttunen 2003). In other words, if the solution to the generation problem is in terms of FSTs, then one automatically obtains a solution to the recognition problem.

Rule-based grammars also have solutions to the recognition and generation problems. This is because they can be faithfully compiled into FSTs, provided it is clear whether they apply simultaneously, left to right, right to left and/or optionally (Johnson 1972, Kaplan & Kay 1994). The fact that all maps for phonology can be expressed as a list of ordered *SPE*-style rewrite rules is strong evidence for the claim that phonological maps are regular (i.e. describable with a FST). (See Karttunen 1993 and Heinz 2011a, b for additional discussion on this point.)

In OT, there are solutions to the generation and recognitions problems under certain assumptions. Frank & Satta (1998) show that OT grammars can be faithfully compiled into FSTs provided the three conditions in (18) hold. (See Frank & Satta 1998, Karttunen 1998, Gerdemann & van Noord 2000, Jäger 2002, Riggle 2004 and Gerdemann & Hulden 2012 for more details.)

(18) a. GEN can be represented with a FST (and so is a regular relation).
b. Each constraint must also be describable with a FST.
c. There must be a maximum number of violations that each constraint can assign.[16]

However, many theoretical phonologists do not accept these assumptions. Correspondence theory is the most widely adopted theory of GEN, but it is not known whether such a GEN defines a regular relation. Also, not all constraints widely used in the literature can be described with FSTs, such as the constraints of the ALIGN family (Eisner 1997). Finally, while some theoretical phonologists have argued for assumption (18c) (e.g. McCarthy 2003), most do not adopt it. Consequently, OT theories of phonology which fail to adopt these three assumptions lack a comprehensive solution to the generation problem.

There are some other approaches to generation and recognition in OT. For instance, Riggle's (2004) solution does not require bounding the number of violations constraints assign ((18c)), but it does depend on (18a) and (b). His solution is guaranteed to be correct, provided the map that the OT grammar describes is in fact a regular relation (but not all of

[16] Jäger (2002) shows that this assumption can be relaxed provided the grammar has 'global optimality', meaning an output that is optimal for some input must be optimal for all inputs for which it is a candidate.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

them are; one example is discussed below). Another solution is found in Albro (2005), which provides a comprehensive OT analysis of the phonology of Malagasy; it assumes GEN is represented by a mildly context-sensitive grammar.

The above discussion indicates that solutions to the generation and recognition problem under typical assumptions in OT, even if currently absent, may be forthcoming. Nonetheless, their absence contrasts with existing algorithms that reliably solve the generation and recognition problems for *SPE* and the ISL function theories.

### 4.3 Learnability

With respect to learnability, there are algorithms which provably learn any ISL function from finitely many examples of (input, output) pairs (Chandlee *et al*. 2014, Jardine *et al*. 2014). These algorithms are provably efficient in the size of the input to the learner. Additionally, it is the case that the amount of data required to learn the pattern exactly is polynomial in the size of the target grammar. In fact, Chandlee *et al*.'s results are quadratic, and Jardine *et al*.'s linear.

With respect to rule-based grammars, there are some results, but they are not as strong. In each case, the input data is assumed to be (input, output) pairs. Johnson (1984) shows how to generate all rules consistent with the input data, and Oncina *et al*. (1993) prove that the Onward Subsequential Transducer Inference Algorithm (OSTIA) can learn total subsequential functions in time cubic in the size of the input data. Since many segmental phonological maps appear to be subsequential (Chandlee *et al*. 2012, Chandlee & Heinz 2012, Gainor *et al*. 2012, Heinz & Lai 2013, Jardine 2016), and since Kaplan & Kay (1994) show how to convert a FST into a rule, this suggests a method for learning rules. However, Gildea & Jurafsky (1996) have argued that OSTIA, while well understood analytically, needs more phonology-specific information to be successful on corpus data.

There are numerous results for both the core OT theory and its variants. Many of these results are impressive and analytical. The first is Recursive Constraint Demotion, introduced by Tesar & Smolensky (1998), which was shown to provably learn OT grammars with a polynomial mistake bound. It has subsequently been shown that established machine-learning techniques can be applied to Harmonic Grammar (Pater 2008, Bane *et al*. 2010, Potts *et al*. 2010), and also to categorical OT (Magri 2013). Finally, Tesar (2014) proves that, under certain conditions, an OT-based learner can learn both a grammar and underlying forms for a map, given that the map is output-driven.

However, to our knowledge there are no analytical results for optimisation-based learners that can learn an entire class of grammars that includes the diversity of opaque maps that we have discussed here. In the absence of analytical results, there are interesting modelling studies which bear on this question. For example, Jarosz (2016) provides results from

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

simulations demonstrating an algorithm that can successfully learn Harmonic Serialism grammars capable of describing some opaque maps. Also, Nazarov & Pater (2017) provide simulations from a learner capable of learning some opaque maps represented in Stratal Maximum Entropy Grammar. While results based on specific models tested with specific input may in the future lead to more general results, they are at present less conclusive than results based on proven theorems.

### 4.4 Typology

In this section, we compare the ISL approach to phonological maps with OT and *SPE* in terms of its generative power (i.e. does it over- or undergenerate the observed typology?). As noted in §2.5, in this paper we focus on generation with respect to computational properties, putting aside any treatment of the important questions of phonetic naturalness, such as the idea that phonological grammars are organised around natural classes and not individual speech sounds. For instance, in the same way that *SPE*-style rules and OT constraints can be required to refer to natural classes, this could also be required for ISL functions, which would be a further restriction.

Earlier we explained that many phonological maps, including opaque ones, are ISL for some $k$. However, as Chandlee (2014), Chandlee, Eyraud & Heinz (2015) and Chandlee & Heinz (2018) explain, neither progressive and regressive iterative spreading nor unbounded consonant and vowel harmony are ISL. Consequently, this theory undergenerates.

This undergeneration problem requires adjusting the theory. However, as noted by Chandlee & Heinz (2018), the change is not expected to be significant. Chandlee, Eyraud & Heinz (2015) show that iterative spreading can be described with a comparable class of functions called Output Strictly Local functions, which differ from ISL in that the output is determined based on a bounded window of the previous *output* rather than the *input*.

For long-distance phenomena, an approach is readily available which builds on previous work on phonotactics and the Subregular Hierarchy, a hierarchy of formal languages that are all properly contained by (i.e. are less complex than) the regular languages (Rogers & Pullum 2011, Rogers *et al*. 2013). Heinz (2010) argues that long-distance phonotactic constraints can be characterised by Strictly Piecewise (SP) languages. Another, related, view holds that long-distance phonotactics can be characterised as Tier-based Strictly Local (TSL) languages (Heinz *et al*. 2011, McMullin 2016). In the same way that the ISL functions are based on the notion of locality that defines the SL languages, there are likely to be functional counterparts to the SP and TSL classes. These classes are expected to characterise long-distance maps, though the precise nature of these functions is an area of current research. Taken together, this means that the current theory can be modified as follows to address the undergeneration problem: phonological grammars are not just ISL functions, but

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

rather some combination of ISL, OSL, SP and TSL functions. What these functions have in common is they are all SUBREGULAR.

As for overgeneration, one of the ways in which grammars can overgenerate is to permit maps like majority rules (MR) vowel harmony and sour grapes (SG) vowel harmony (Baković 2000, Finley 2008, Heinz & Lai 2013).[17] ISL maps do not overgenerate in this way. This is because (i) every ISL function by definition is also both a regular relation and a subsequential function, and (ii) it has been proven that MR maps are not regular relations and SG maps are not subsequential (Heinz & Lai 2013).

Turning to the typological predictions of rule-based grammars, it is generally understood that ordered rules do not undergenerate. They also can describe opaque maps, despite some criticisms (e.g. Baković 2007, 2011). With respect to overgeneration, the SG pattern is regular, and therefore describable with an ordered list of rules according to Kaplan & Kay's analysis. On the other hand, rule-based grammars cannot generate non-regular maps like MR.

Turning to optimisation-based theories, it is well-known that classic OT cannot generate many opaque maps (Idsardi 1998, 2000, McCarthy 2007, Buccola 2013) – as discussed by Baković (2007), OT analyses of non-gratuitous feeding and cross-derivational feeding are possible and straightforward. In response, many adjustments to classic OT have been proposed, including constraint conjunction (Smolensky 2006), sympathy theory (McCarthy 1999), turbidity theory (Goldrick 2000), output-to-output correspondence (Benua 1997), stratal OT (Kiparsky 1998, 2000, Bermúdez-Otero 2018), candidate chains (McCarthy 2007), Harmonic Serialism (McCarthy 2000), targeted constraints (Wilson 2000), contrast preservation (Łubowicz 2003), comparative markedness (McCarthy 2003) and serial markedness reduction (Jarosz 2014). See McCarthy (2007) for a review, meta-analysis and more references to these proposals.

These approaches all invoke different representational schemes, constraint types and/or architectural changes to classic OT. In comparison, *no special modifications are needed* to establish the ISL nature of the opaque maps we have studied in this paper. Furthermore, the typological and learnability ramifications of these changes to classic OT are not yet well understood in many cases, though they are the subject of current research (e.g. Nazarov & Pater 2017). For instance, it is not obvious whether the results of Recursive Constraint Demotion carry over from classic OT to Stratal OT.

It is also the case that classic OT overgenerates. This is not controversial. It has been shown that non-regular maps are obtainable via the interaction of very simple constraints (Frank & Satta 1998, Riggle 2004, Gerdemann & Hulden 2012, Heinz & Lai 2013). MR vowel harmony is one

[17] In MR harmony, the feature with the greatest number of occurrences is the one that harmonises (e.g. $+++--\mapsto+++++$, but $++---\mapsto-----$). In SG harmony, harmony only takes place if the harmonising feature can spread through the entire string. If any blockers are present, harmony doesn't occur at all (e.g. $+++--\mapsto+++++$ but $+++-B\mapsto+++-B$, where B is a harmony blocker).

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

example (Baković 2000, Finley 2008). Hansson (2007) and McMullin & Hansson (2015) draw similar conclusions with respect to Agreement by Correspondence. Gerdemann & Hulden's (2012) example is given in (19).

(19) IDENT, DEP ≫ *ab ≫ MAX

Under this ranking, an input string $a^n b^m$ maps to $a^n$ if $m < n$, but to $b^m$ if $n < m$. No FST can compute this function (i.e. it is non-regular).

Maps like MR and the one above make computations based on global access to information in the string. In this respect, optimisation misses an important generalisation. When computing the output of phonological maps, the necessary information is contained within substructures of bounded size, which, as this paper shows, are often present in the input representation. This is neither expected nor predicted under global optimisation. On the other hand, it is one of the defining characteristics of ISL functions.

To summarise, both rule-based and optimisation-based grammars overgenerate in a way that ISL maps do not. Additionally, optimisation-based grammars and ISL maps undergenerate when compared to rule-based grammars, but in different ways. ISL maps cannot account for iteration and long-distance phenomena. Optimisation-based grammars struggle with opaque maps, as witnessed by the past two decades of research in this area. We believe that the prospects for extending subregular functions to include long-distance phenomena are bright, and this line of research ought to be pursued.

Figure 4 summarises the typological results. A successful theory of phonology ought to include patterns like Bedouin raising,[18] Bedouin

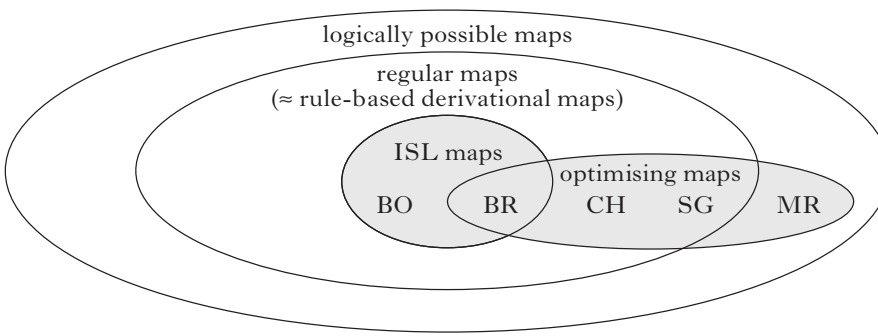



*Figure 4*

Alignment of typological predictions of rule-based theories, classic OT and subregular theory (BO = Bedouin opacity, BR = Bedouin raising, CH = consonant harmony, SG = sour grapes harmony, MR = majority rules harmony).

[18] Bedouin raising here refers to just the vowel-raising rule that participates in both counterfeeding on environment and counterfeeding on focus in Bedouin Arabic, discussed in Appendix B.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

opacity and consonant harmony, while excluding sour grapes harmony and majority rules harmony. While no such theory exists to our knowledge, we hope to have convinced readers that extending ISL maps in the ways mentioned above is a route worthy of further research.

### 4.5 Summary of theory comparison

A theory of phonology which characterises UR–SR maps as ISL functions provides theoretically strong results for learnability, recognition and generation. With respect to typology, certain types of overgeneration are avoided, such as MR and SG harmony. On the other hand, the theory cannot generate long-distance maps, though it could be modified to include them by generalising additional subregular language classes to functions. Rule-based theories, on the other hand, do not undergenerate at all. They do overgenerate SG harmony, but not MR harmony, as the former, but not the latter, is regular. While some learnability results exist, they were not thoroughly pursued, in part due to the advent of constraint-based theories. Lastly, due to their known correspondence with FSTs, rule-based grammars can also be successfully used for both generation and recognition.

Optimisation-based theories have theoretical results for generation and recognition under certain assumptions, even if they are not widely adopted in practice. There are also learnability results for a variety of optimisation frameworks. With respect to typology, we find both overgeneration and undergeneration. It is interesting to observe that the signature of a successful OT analysis involves complex phenomena being understood as the interaction of simple constraints. But the overgeneration problem is precisely this: complex but strange phenomena emerging from the interaction of simple constraints. One approach is to put the burden of explanation on Con: the right set of constraints will generate the attested patterns, but not ones like MR or SG. This is a reasonable approach, but it misses the forest for the trees. Under this approach, the choice of Con blunts the power of optimisation, to avoid generating non-regular maps. There is an important typological generalisation – that phonological maps are regular – whose explanation is obtusely stated on this line of thinking.

## 5 Conclusion

This paper has shown that a range of phonological generalisations labelled as opaque all have the computational property of being Input Strictly Local. This means that, despite the challenges that some types of opacity have presented for constraint-based theories like classic OT, in computational terms this set of opaque maps is very limited in its complexity.

The result that many opaque maps are ISL is also significant, because ISL functions as a class have several advantages for a theory of phonology:

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

they are restrictive, they are learnable and they have a clear cognitive interpretation. While they cannot account for long-distance phenomena, there is precedent from the study of other subregular formal languages which indicates that ISL maps can be extended in the right sorts of ways. The results reported here increase the viability of subregular properties such as Input Strict Locality speaking directly to the computational nature of phonology. As such, whether these subregular properties will ultimately supplant optimisation as the core computational component of phonological grammars merits further study.

## REFERENCES


Albright, Adam & Michelle A. Fullwood (eds.) (2015). *Proceedings of the 2014 Meeting on Phonology*. http://dx.doi.org/10.3765/amp.

Albro, Daniel M. (2005). *Studies in computational Optimality Theory, with special reference to the phonological system of Malagasy*. PhD dissertation, University of California, Los Angeles.

Baković, Eric (2000). *Harmony, dominance and control*. PhD dissertation, Rutgers University.

Baković, Eric (2005). Antigemination, assimilation and the determination of identity. *Phonology* **22**. 279–315.

Baković, Eric (2007). A revised typology of opaque generalisations. *Phonology* **24**. 217–259.

Baković, Eric (2011). Opacity and ordering. In John Goldsmith, Jason Riggle & Alan Yu (eds.) *The handbook of phonological theory*. 2nd edn. Malden, Mass. & Oxford: Wiley-Blackwell. 40–67.

Baković, Eric (2013). *Blocking and complementarity in phonological theory*. Sheffield & Bristol, Conn.: Equinox.

Bane, Max, Jason Riggle & Morgan Sonderegger (2010). The VC dimension of constraint-based grammars. *Lingua* **120**. 1194–1208.

Beesley, Kenneth R. & Lauri Karttunen (2003). *Finite state morphology*. Stanford: CSLI.

Benua, Laura (1997). *Transderivational identity: phonological relations between words*. PhD dissertation, University of Massachusetts, Amherst.

Bermúdez-Otero, Ricardo (2018). Stratal Phonology. In S. J. Hannahs & Anna R. K. Bosch (eds.) *The Routledge handbook of phonological theory*. Abingdon & New York: Routledge. 100–134.

Buccola, Brian (2013). On the expressivity of Optimality Theory versus ordered rewrite rules. In Glyn Morrill & Mark-Jan Nederhof (eds.) *Formal grammar: Proceedings of the 17th and 18th International Conferences, FG 2012/2013*. Berlin & Heidelberg : Springer. 142–158.

Chandlee, Jane (2014). *Strictly local phonological processes*. PhD dissertation, University of Delaware.

Chandlee, Jane, Angeliki Athanasopoulou & Jeffrey Heinz (2012). Evidence for classifying metathesis patterns as subsequential. *WCCFL* **29**. 303–309.

Chandlee, Jane, Rémi Eyraud & Jeffrey Heinz (2014). Learning Strictly Local subsequential functions. *Transactions of the Association for Computational Linguistics* **2**. 491–503.

Chandlee, Jane, Rémi Eyraud & Jeffrey Heinz (2015). Output strictly local functions. In Marco Kuhlmann, Makoto Kanazawa & Gregory M. Kobele (eds.) *Proceedings of*


https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

*the 14th Meeting on the Mathematics of Language*. Association for Computational Linguistics. 112–125.
Chandlee, Jane & Jeffrey Heinz (2012). Bounded copying is subsequential: implications for metathesis and reduplication. In *Proceedings of the 12th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*, pages. Stroudsburg, PA: Association for Computational Linguistics. 42–51.
Chandlee, Jane & Jeffrey Heinz (2016). Computational phonology. In Mark Aronoff (ed.) *Oxford research encyclopedia of linguistics*. Oxford: Oxford University Press. http://dx.doi.org/10.1093/acrefore/9780199384655.013.116.
Chandlee, Jane & Jeffrey Heinz (2018). Strict locality and phonological maps. *LI* **49**. 23–60.
Chandlee, Jane, Adam Jardine & Jeffrey Heinz (2015). Learning repairs for marked structures. In Albright & Fullwood (2015). http://dx.doi.org/10.3765/amp.v2i0.3760.
Chandlee, Jane & Steven Lindell (to appear). A logical characterization of strictly local functions. In Jeffrey Heinz (ed.) *Doing computational phonology*. Oxford: Oxford University Press.
Chomsky, Noam & Morris Halle (1968). *The sound pattern of English*. New York: Harper & Row.
Eisner, Jason (1997). Efficient generation in primitive Optimality Theory. In *Proceedings of the 35th Annual Meeting of the ACL and 8th Conference of the European Chapter of the Association for Computational Linguistics*. Morristown, NJ: Association for Computational Linguistics. 313–320.
Finley, Sara (2008). *Formal and cognitive restrictions on vowel harmony*. PhD dissertation, Johns Hopkins University.
Frank, Robert & Giorgio Satta (1998). Optimality Theory and the generative complexity of constraint violability. *Computational Linguistics* **24**. 307–315.
Gainor, Brian, Regine Lai & Jeffrey Heinz (2012). Computational characterizations of vowel harmony patterns and pathologies. *WCCFL* **29**. 63–71.
Gerdemann, Dale & Mans Hulden (2012). Practical finite state Optimality Theory. In *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*. Association for Computational Linguistics. 10–19.
Gerdemann, Dale & Gertjan van Noord (2000). Approximation and exactness in finite state optimality theory. In Jason Eisner, Lauri Karttunen & Alain Thériault (eds.) *Finite-state phonology: proceedings of the 5th Workshop of the ACL Special Interest Group in Computational Phonology* (*SIGPHON*). Luxemburg. 34–45.
Gildea, Daniel & Daniel Jurafsky (1996). Learning bias and phonological-rule induction. *Computational Linguistics* **22**. 497–530.
Gold, E. Mark (1967). Language identification in the limit. *Information and Control* **10**. 447–474.
Goldrick, Matthew (2000). Turbid output representations and the unity of opacity. *NELS* **30**. 231–245.
Hansson, Gunnar Ólafur (2007). Blocking effects in agreement by correspondence. *LI* **38**. 395–409.
Hayes, Bruce, Robert Kirchner & Donca Steriade (eds.) (2004). *Phonetically based phonology*. Cambridge: Cambridge University Press.
Heinz, Jeffrey (2010). Learning long-distance phonotactics. *LI* **41**. 623–661.
Heinz, Jeffrey (2011a). Computational phonology. Part I: Foundations. *Language and Linguistics Compass* **5**. 140–152.
Heinz, Jeffrey (2011b). Computational phonology. Part II: Grammars, learning, and the future. *Language and Linguistics Compass* **5**. 153–168.
Heinz, Jeffrey & Regine Lai (2013). Vowel harmony and subsequentiality. In András Kornai & Marco Kuhlmann (eds.) *Proceedings of the 13th Meeting on the Mathematics of Language*. Sofia: Association for Computational Linguistics. 52–63.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

Heinz, Jeffrey, Chetan Rawal & Herbert G. Tanner (2011). Tier-based strictly local constraints in phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*. Vol. 2. Association for Computational Linguistics. 58–64.
Heinz, Jeffrey & Jason Riggle (2011). Learnability. In Marc van Oostendorp, Colin J. Ewen, Elizabeth Hume & Keren Rice (eds.) *The Blackwell companion to phonology*. Malden, Mass.: Wiley-Blackwell. 54–78.
Higuera, Colin de la (1997). Characteristic sets for polynomial grammatical inference. *Machine Learning* **27**. 125–138.
Higuera, Colin de la (2010). *Grammatical inference: learning automata and grammars*. Cambridge: Cambridge University Press.
Idsardi, William J. (1998). Tiberian Hebrew spirantization and phonological derivations. *LI* **29**. 37–73.
Idsardi, William J. (2000). Clarifying opacity. *The Linguistic Review* **17**. 337–350.
Jäger, Gerhard (2002). Gradient constraints in finite state OT: the unidirectional and the bidirectional case. In Ingrid Kaufmann & Barbara Stiebels (eds.) *More than words: a Festschrift for Dieter Wunderlich*. Berlin: Akademie Verlag. 299–325.
Jain, Sanjay, Daniel Osherson, James S. Royer & Arun Sharma (1999). *Systems that learn: an introduction to learning theory*. 2nd edn. Cambridge, Mass.: MIT Press.
Jardine, Adam (2016). Computationally, tone is different. *Phonology* **33**. 247–283.
Jardine, Adam, Jane Chandlee, René Eyraud & Jeffrey Heinz (2014). Very efficient learning of structured classes of subsequential functions from positive data. In Alexander Clark, Makoto Kanazawa & Ryo Yoshinaka (eds.) *Proceedings of the 12th International Conference on Grammatical Inference*. 94–108.
Jarosz, Gaja (2014). Serial markedness reduction. In John Kingston, Claire Moore-Cantwell, Joe Pater & Robert Staubs (eds.) *Proceedings of the 2013 Meeting on Phonology*. http://dx.doi.org/10.3765/amp.v1i1.40.
Jarosz, Gaja (2016). Learning opaque and transparent interactions in Harmonic Serialism. In Gunnar Ólafur Hansson, Ashley Farris-Trimble, Kevin McMullin & Douglas Pulleyblank (eds.) *Proceedings of the 2015 Annual Meeting on Phonology*. http://dx.doi.org/10.3765/amp.v3i0.3671.
Johnson, C. Douglas (1972). *Formal aspects of phonological description*. The Hague & Paris: Mouton.
Johnson, Mark (1984). A discovery procedure for certain phonological rules. In *Proceedings of the 10th International Conference on Computational Linguistics and the 22nd Annual Meeting of the ACL*. 344–347.
Kager, René (1999). *Optimality Theory*. Cambridge: Cambridge University Press.
Kaplan, Ronald & Martin Kay (1994). Regular models of phonological rule systems. *Computational Linguistics* **20**. 331–378.
Karttunen, Lauri (1993). Finite-state constraints. In John A. Goldsmith (ed.) *The last phonological rule: reflections on constraints and derivations*. Chicago: University of Chicago Press. 173–194.
Karttunen, Lauri (1998). The proper treatment of optimality in computational phonology. In Lauri Karttunen & Kemal Oflazer (eds.) *Proceedings of the International Workshop on Finite State Methods in Natural Language Processing*. Ankara: Bilkent University. 1–12.
Kavitskaya, Darya & Peter Staroverov (2010). When an interaction is both opaque and transparent: the paradox of fed counterfeeding. *Phonology* **27**. 255–288.
Kawahara, Shigeto (2015). A catalogue of phonological opacity in Japanese. *Reports of the Keio Institute of Cultural and Linguistic Studies* **46**. 145–174.
Kearns, Michael & Umesh Vazirani (1994). *An introduction to computational learning theory*. Cambridge, Mass.: MIT Press.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

Kiparsky, Paul (1971). Historical linguistics. In William Orr Dingwall (ed.) *A survey of linguistic science*. College Park: University of Maryland Linguistics Program. 576–642.
Kiparsky, Paul (1973). Abstractness, opacity, and global rules. In Osamu Fujimura (ed.) *Three dimensions in linguistic theory*. Tokyo: TEC. 57–86.
Kiparsky, Paul (1998). *Paradigm effects and opacity*. Ms, Stanford University.
Kiparsky, Paul (2000). Opacity and cyclicity. *The Linguistic Review* **17**. 351–365.
Legendre, Géraldine, Yoshiro Miyata & Paul Smolensky (1990). Harmonic Grammar: a formal multi-level connectionist theory of linguistic well-formedness: theoretical foundations. In *Proceedings of the 12th Annual Conference of the Cognitive Science Society*. Hillsdale: Erlbaum. 388–395.
Łubowicz, Anna (2003). *Contrast preservation in phonological mappings*. PhD dissertation, University of Massachusetts, Amherst.
Lundskaer-Nielsen, Tom & Philip Holmes (2011). *Danish: an essential grammar*. 2nd edn. London & New York: Routledge.
McCarthy, John J. (1999). Sympathy and phonological opacity. *Phonology* **16**. 331–399.
McCarthy, John J. (2000). Harmonic serialism and parallelism. *NELS* **30**. 501–524.
McCarthy, John J. (2003). Comparative markedness. *Theoretical Linguistics* **29**. 1–51.
McCarthy, John J. (2007). *Hidden generalizations: phonological opacity in Optimality Theory*. Sheffield & Bristol, Conn.: Equinox.
McCarthy, John J. & Alan Prince (1999). Faithfulness and identity in Prosodic Morphology. In René Kager, Harry van der Hulst & Wim Zonneveld (eds.) *The prosody–morphology interface*. Cambridge: Cambridge University Press. 218–309.
McMullin, Kevin (2016). *Tier-based locality in long-distance phonotactics: learnability and typology*. PhD dissertation, University of British Columbia.
McMullin, Kevin & Gunnar Ólafur Hansson (2015). Long-distance phonotactics as Tier-Based Strictly 2-Local languages. In Albright & Fullwood (2015). http://dx.doi.org/10.3765/amp.v2i0.3750.
Magri, Giorgio (2013). HG has no computational advantages over OT: toward a new toolkit for computational OT. *LI* **44**. 569–609.
Mohri, Mehryar (1997). Finite-state transducers in language and speech processing. *Computational Linguistics* **23**. 269–311.
Nazarov, Aleksei & Joe Pater (2017). Learning opacity in Stratal Maximum Entropy Grammar. *Phonology* **34**. 299–324.
Odden, David (2005). *Introducing phonology*. Cambridge: Cambridge University Press.
Oncina, José, Pedro García & Enrique Vidal (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **15**. 448–458.
Pater, Joe (2008). Gradual learning and convergence. *LI* **39**. 334–345.
Pater, Joe (2012). Serial Harmonic Grammar and Berber syllabification. In Toni Borowsky, Shigeto Kawahara, Takahito Shinya & Mariko Sugahara (eds.) *Prosody matters: essays in honor of Elisabeth Selkirk*. London: Equinox. 43–72.
Potts, Christopher, Joe Pater, Karen Jesney, Rajesh Bhatt & Michael Becker (2010). Harmonic Grammar with linear programming: from linear systems to linguistic typology. *Phonology* **27**. 77–117.
Prince, Alan & Paul Smolensky (2004). *Optimality Theory: constraint interaction in generative grammar*. Malden, Mass. & Oxford: Blackwell.
Riggle, Jason (2004). *Generation, recognition, and learning in finite-state Optimality Theory*. PhD dissertation, University of California, Los Angeles.
Rogers, James, Jeffrey Heinz, Margaret Fero, Jeremy Hurst, Dakotah Lambert & Sean Wibel (2013). Cognitive and sub-regular complexity. In Glyn Morrill & Mark-Jan Nederhof (eds.) *Formal grammar*. Berlin & Heidelberg: Springer. 90–108.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press

Rogers, James & Geoffrey K. Pullum (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* **20**. 329–342.
Sanders, Nathan (2003). *Opacity and sound change in the Polish lexicon*. PhD dissertation, University of California, Santa Cruz.
Smolensky, Paul (2006). Optimality in phonology II: harmonic completeness, local constraint conjunction, and feature domain markedness. In Paul Smolensky & Géraldine Legendre (eds.) *The harmonic mind: from neural computation to optimality-theoretic grammar*. Vol. 2: *Linguistic and philosophical implications*. Cambridge, Mass.: MIT Press. 27–160.
Sprouse, Ronald L. (1997). A case for enriched inputs. Handout of paper presented at TREND (Trilateral Phonology Weekend) 3. Available as ROA-193 from the Rutgers Optimality Archive.
Tesar, Bruce (2014). *Output-driven phonology: theory and learning*. Cambridge: Cambridge University Press.
Tesar, Bruce & Paul Smolensky (1996). *Learnability in Optimality Theory* (*long version*). Technical Report 96:3, Department of Cognitive Science, Johns Hopkins University.
Tesar, Bruce & Paul Smolensky (1998). Learnability in Optimality Theory. *LI* **29**. 229–268.
Weigel, William F. (2005). *Yowlumne in the twentieth century*. PhD thesis, University of California, Berkeley.
Wilson, Colin (2000). *Targeted constraints: an approach to contextual neutralization in Optimality Theory*. PhD dissertation, Johns Hopkins University.

https://doi.org/10.1017/S0952675718000027 Published online by Cambridge University Press