Learning Repairs for Marked Structures

Jane Chandlee, Adam Jardine, Jeffrey Heinz
University of Delaware

1 Introduction

This paper presents an algorithm which learns phonological grammars. These phonological grammars—like grammars in Optimality Theory (OT, Prince & Smolensky, 2004) or rule-based grammars (Chomsky & Halle, 1968)—describe phonological maps, or the transformations from input (underlying) forms to output (surface) forms (Tesar, 2014). Like OT grammars, the grammars which underlie the algorithm distinguish marked structures from their possible repairs. The advantages of this algorithm include 1) it is provably efficient and correct, and 2) it has significant empirical coverage in that the grammars it acquires can model many phonological maps, including opaque ones which are hard to represent in classical OT.

To illustrate the algorithm and these advantages, we focus on *repair typologies*, by which we mean 'international conspiracies' in which different processes in different languages all target a single marked structure. In a classic example from Pater (2004), languages that avoid nasal-voiceless obstruent sequences (NC̥) employ various processes such as fusion, denasalization, and voicing of the obstruent. Examples from Indonesian, Toba Batak, and Quechua, respectively, are presented below in (1), (2), and (3).

(1) Fusion in Indonesian
  a. /məN+pilih/↦[məmilih], 'to choose'

(2) Denasalization in Toba Batak (Hayes, 1986)
  a. /maŋinum tuak/↦[maŋinup tuak], 'drink palm wine'

(3) Obstruent Voicing in Quechua
  a. /kam+pa/↦[kamba], 'yours'

Pater (2004) accounts for this repair typology by permuting the rankings of a handful of FAITHFULNESS constraints and a single MARKEDNESS constraint, *NC̥.

Our proposed learner exploits the fact that the maps that make up a repair typology all belong to a well-defined class of functions and therefore share a common structure. These functions are the *Input Strictly Local (ISL) functions*, defined by Chandlee (2014) (see also Chandlee et al. (2014)), who shows that a significant range of phonological maps belong to this class and that they are learnable under certain conditions. These functions, which are defined below in §3, are defined in terms of a Markovian-like property that restricts the information needed to compute the output for a given input. This restrictive nature of the functions corresponds to a highly structured characterization in terms of a finite state transducer (FST). This structure allows for efficient, provably correct learning from positive data. We argue that this structure of the ISL class of functions is universal and therefore serves as *a priori* knowledge that guides learning.

The learning algorithm we illustrate in this paper, which is called the Structured Onward Subsequential Function Inference Algorithm (SOSFIA) is presented in detail in Jardine et al. (2014).[1] Like Recursive Constraint Demotion (RCD, Tesar & Smolensky (1998)), the input to SOSFIA is a finite set of input-output pairs and some *a priori* knowledge of the grammar. In RCD, this *a priori* knowledge is essentially the contents of CON. Here the *a priori* knowledge is that phonological maps are ISL functions. As we explain, this knowledge has two components: the allowable marked structures are of a bounded length k, and the

[1] SOSFIA is similar to the Input Strictly Local Function Learning Algorithm (ISLFLA) presented in Chandlee (2014) and Chandlee et al. (2014). We choose to present SOSFIA here for two reasons. First, it is easier to explain. Second, Jardine et al. (2014) establish lower complexity bounds on the runtime of SOSFIA as compared to the ISLFLA (so it is more efficient, as far as we know).

Proceedings of *AMP 2014*
Completed February 27, 2015

repairs to those marked structures are also determined by information which is local to the locus of violation. This knowledge can be represented with a single finite state transducer (FST) in which the state set and the input side of the transitions are known and fixed in advance. The algorithm then fills in the initially empty output side of the transitions by parsing the input data set based on the notion of *longest common prefix*. Jardine et al. (2014) show that any class of *transductions* (i.e., input-output maps) defined in terms of the structure of a particular FST is learnable in efficient time and data (de la Higuera, 2010). In this paper, we show that the structure of the FST establishes a range of possible markedness constraints, and that the maps that SOSFIA can learn correspond to any of the possible repair strategies for these marked structures.

The paper outline is as follows. In §2 we present the computational perspective that underlies our analyses and learning approach. In §3 we define Input Strictly Local functions and show how the repair typology in Pater (2004) is describable by functions in this class. In §4 we present and provide a demonstration of how SOSFIA learns phonological maps, even opaque ones. In §5 we discuss the advantages of this model for phonology and phonological learning. In §6 we discuss remaining issues and future directions of this work, and in §7 we conclude.

2 Phonological maps

Both the computational analyses of the phonologies of languages and the approach to learning them presented in this paper are based on the view that a phonological grammar is a function that maps an input (underlying representation, UR) to an output (surface representation, SR). Such maps are fundamental to all generative theories of the phonological grammar. Despite the many differences between rule-based theories like SPE (Chomsky & Halle, 1968) and constraint-based theories like Optimality Theory (Prince & Smolensky, 2004), Harmonic Grammar (Legendre et al., 1990), and Harmonic Serialism (McCarthy, 2000; Pater, 2012), each of these formalisms concur on the existence of a grammar which maps underlying representations to surface representations.

Our approach then is to model these maps with functions with the goal of identifying computational properties of the maps that are independent of grammatical formalisms like rules and constraints. For example, post-nasal obstruent voicing, exemplified in (3) for Quechua, can be represented by the function f shown in (4).

(4) a. Post-nasal obstruents become voiced.
b. For example: f(kampa) = [kamba]
c. Thus: $f = \{$(kampa,kamba),(kama,kama),(mpmpa,mbmba),...$\}$

The function f is represented in three ways in (4). In (4-a) it is expressed in English. In (4-b), it is illustrated ostensively by showing a particular input and its corresponding output. In (4-c) it is shown as a infinite set of input-output pairs. In this light, phonological grammars of post-nasal obstruent voicing are *intensional* descriptions of f, which is the (infinitely-sized) extension these grammars describe.

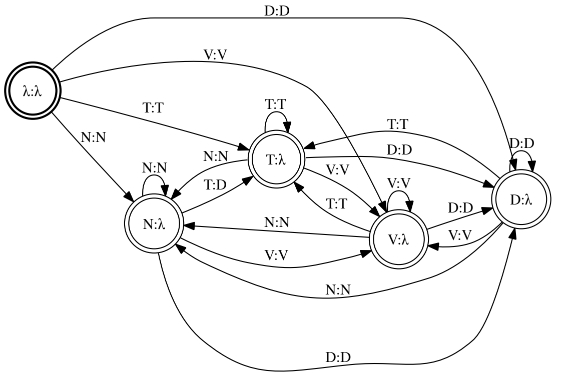
Two important assumptions of our analyses are highlighted by (4-c). One is that the function for a given process will map all inputs to an output. For example, there is no obstruent following the nasal in the input $kama$, but that does not mean that f is undefined for $kama$. Rather, it maps this form to itself. Secondly, the function is also defined for inputs that do not necessarily correspond to lexical items in the language in question. Hence f applies post-nasal obstruent voicing to the form $mpmpa$, as shown in (4-c), even though this is likely not a well-formed word in Quechua.

In other words, these functions are *total*, or defined for all possible inputs. This means that, like OT, our analysis assumes there are no morpheme structure constraints (MSCs) that restrict the set of input forms. This assumption is therefore compatible with the principle of Richness of the Base (Prince & Smolensky, 2004). This set of possible inputs for a function (or its *domain*) is the set of all possible strings that can be formed using a given set of input symbols, or *alphabet*. We use Σ to represent this alphabet, and Σ^* to represent all possible strings formed by concatenating symbols from Σ. In general, the alphabet for a given function can be thought of as the set of symbols needed to represent URs, including phonemes and boundary symbols.

In addition to the representations shown in (4), certain types of functions can also be described with FSTs. FSTs are a finite means of representing an infinite function like (4). They provide a computationally well-understood, intensional description of maps. When presented visually as diagrams, FSTs allow us to view the entire function at once. An FST for (4) is shown in Figure 1 on the following page.

An FST is composed of a set of *states*, represented in the figure with circles, and a set of *transitions* between states, represented in the figure with labeled arrows. The *initial* or *start* state is represented in bold, and states with double circles are *final* or *accepting* states. States are labeled with a *name*, a colon, and then an *output string*. Generally, the names are arbitrary, but the particular names chosen here will become clear in §3. The output strings can also be ignored for now and will be discussed in §3.

For a given input like /kampa/, the FST begins in its initial state and reads the input one symbol at a time, proceeding to other states by following the transition for which the portion of the label to the left of the colon matches the current input symbol. As it does so, it also produces an output string in accordance with the portion of the transition label to the right of the colon. In other words, the transition labels can be interpreted as $input : output$.



$$\begin{array}{ccccccccccccc}
 & \text{k} & & \text{a} & & \text{m} & & \text{p} & & \text{a} & & \# \\
\lambda & \Rightarrow & \text{T} & \Rightarrow & \text{V} & \Rightarrow & \text{N} & \Rightarrow & \text{T} & \Rightarrow & \text{V} & \Rightarrow \\
 & \text{k} & & \text{a} & & \text{m} & & \text{b} & & \text{a} & & \lambda
\end{array}$$

Figure 1: FST for post-nasal obstruent voicing

As do functions, a given FST has a specified alphabet. The alphabet of the FST in Figure 1 is $\Sigma = \{$N, T, D, V$\}$, where N=nasals, T=voiceless obstruents, D=voiced obstruents, and V=vowels. In practice, we often use such symbols that correspond to natural classes for illustration purposes. This helps make the FSTs more readable, since it is not uncommon for the number of states and transitions to increase with the size of the alphabet. As we discuss later in more detail, it is important to note that the properties of the functions and the learner's ability to learn them do not depend on the use of this reduced symbol set.

A sample derivation for /kampa/↦[kamba] is also shown in the figure. Starting in the initial state λ, the FST reads /k/ and proceeds to state T, outputting [k].[2] It then reads /a/, proceeds to state V and outputs [a], and then reads /m/, proceeds to state N and outputs [m]. At this point, then, it is merely reproducing the input faithfully. When it next reads /p/, however, it outputs [b], as indicated by the T:D transition. Lastly, it reads /a/, returns to state V, and outputs [a]. Because the state it has ended in is a final state, /kampa/↦[kamba] is a valid mapping.

This example demonstrates how a phonological process can be modeled as a function and represented with a FST. The larger question of interest for phonology is what *class* of functions does f belong to? It is a well-known result in computational phonology that maps describable with SPE-style rules which do not re-apply to the locus of their structural change belong to the class of *regular relations* (Johnson, 1972; Kaplan & Kay, 1994; Koskenniemi, 1983).[3] The regular relations have multiple characterizations, but one is that they are the class of maps that can be modeled with FSTs. Though this result was originally expressed in terms of rules, it does not depend on a rule-based formalism since, as mentioned above, constraint-based formalisms may describe the *same* UR-SR maps. For instance, Baković (2013) shows that a given rule A →

[2] The λ after the colon on the state label is the output string and will be explained in the next section.

[3] Both the terms *function* and *relation* refer to an input-output map. The difference is whether a given input can be mapped to at most one output (as in a function) or multiple outputs (as in a relation). Since the maps we assume in our analysis are deterministic, we use the word *function*.

B / C __ D can be translated into a core constraint ranking *CAD $\gg$ FAITH(A$\mapsto$B), in which the markedness constraint *CAD outranks whatever faithfulness constraints are violated when A surfaces as B. This core ranking is assumed to be embedded in the complete ranking of the OT grammar. Crucially, the same post-nasal obstruent voicing function in (4) is described by rule-based analyses, OT analyses, and analyses with FSTs alike. These are all different intensional descriptions of the same extensional object. So what kind of extensional object—what kind of function—is it?

It has been suggested more recently that the class of regular relations is too large for phonology, meaning the kinds of maps that correspond to the range of attested processes are more restrictive than just regular. In particular, there is evidence that phonological maps in fact belong to a proper subclass of the regular relations called the *subsequential* functions (Mohri, 1997; Chandlee et al., 2012; Gainor et al., 2012; Chandlee & Heinz, 2012; Heinz & Lai, 2013; Jardine, 2013; Luo, 2013; Payne, 2014). Taking it a step further, Chandlee (2014) has argued that local phonological processes are best characterized as a proper subclass of the subsequential functions called the *Input Strictly Local functions*, which we define in the next section.

Thus the goal of this line of research is to identify the *most restrictive* set of functions needed for phonological maps in order to better characterize the components of phonological grammars, be they rules or constraints. Such a characterization has typological implications, in that it makes a strong prediction for what can be a possible phonological process. And, as the current work demonstrates, the restrictive nature of the characterization leads to an efficient and provably correct algorithm for learning any map in the class, even ones which are opaque.

3 Input Strictly Local Functions

3.1 *Definitions and properties* As mentioned in the previous section, one definition of the regular relations is the class of maps describable with finite state transducers. But not all FSTs are equal, and there are proper subclasses of the regular relations that correspond to particular types of FSTs. These classes are *proper subclasses* because there are restrictions on the FSTs that limit the kinds of maps they can represent.

For example, the subsequential functions mentioned above are one such proper subclass of regular relations (Mohri, 1997), and the FSTs that describe them (called subsequential FSTs or SFSTs) have a few special properties. These properties include being deterministic on the input, which means that from any given state there is at most one outgoing transition per possible input symbol. Another property is that the definition of an SFST includes a *final output function* that maps the states to strings. The string mapped to a given state is appended to the output of any input that ends in that state. Figure 1 is in fact an SFST, and its final output function is represented in the labels on its states. An input string that ends in state V will have the empty string λ appended to its output. So, for example, the input string $kampa$ is actually mapped to the output $kamba \cdot \lambda$, which is just $kamba$. This final output function provides a way to increase the expressivity of these transducers. This increased expressivity is not illustrated with this particular example since all states are mapped to the empty string, but examples with non-empty output strings are provided later.

Certain maps cannot be described with FSTs that have these added properties; such maps therefore fall outside of the subsequential function class. This is a benefit when the goal is to model some maps while excluding others. When it comes to phonology, our goal is to model the range of attested maps while excluding those that would correspond to implausible phonological ones. As mentioned above, the restriction of phonological maps to the subsequential class has been argued to exclude phonologically bizarre maps (Heinz & Lai, 2013). It is also a useful restriction from the perspective of learning. The regular relations are not identifiable in the limit from positive data (in the sense of Gold (1967)), but the subsequential functions are (Oncina et al., 1993).

However, there are still many logically possible subsequential maps that we want to rule out in order to better approximate the class of maps found in natural language phonology. Chandlee (2014) proposes a further restriction on phonological maps to the Input Strictly Local (ISL) functions, which are a proper subclass of the subsequential functions. Informally, a function is Input Strictly k-Local if the output of every input string $a_0a_1 \cdots a_n$ is $u_0u_1 \cdots u_n$ where $a_i \in \Sigma$ and u_i is a string which only depends on a_i and the $k-1$ symbols before a_i in the input string (so $a_{i-k+1}a_{i-k+2} \cdots a_{i-1}$). Thus an ISL function is in fact ISL for a particular value of k (to abbreviate, we write that the function is k-ISL). In a phonological map expressed by a rewrite rule, k is the length of the target of the rule plus the length of its triggering context. For example, the k value of post-nasal obstruent voicing is 2, since the target is T and the triggering context is N (i.e., the length of NT is 2). If there is no bound on the material that intervenes between the target and the triggering context then the map will not be ISL for any k, a point we return to in the discussion section.

The notion of Strict Locality that characterizes the ISL functions is based on the well-studied class of subregular formal languages called the Strictly Local (SL) languages. Strictly k-Local languages model markedness constraints that ban substrings of length k in words. For example, *NC̥ is a Strictly 2-Local constraint. See Rogers & Pullum (2011) and Rogers et al. (2010, 2013) for details of these and other subregular languages. A formal definition of ISL functions similar to the informal one above is provided in Chandlee et al. (2014), where mathematical relationships to the Strictly Local languages are explained and studied in more detail than we can provide here. Two points are worth mentioning. First, k-ISL functions are sensitive to (and only sensitive to) Strictly k-Local markedness constraints. Second, the formal definition, like the informal one given above, is *independent* of any particular grammatical formalism. It is a property of a set of input-output pairs like the one in (4-c).

The fact that ISL functions are a proper subclass of subsequential ones means that the FSTs that can describe ISL functions have certain properties that limit the kinds of maps they can represent. As established in Chandlee et al. (2014), these properties restrict the state set and their transitions. Each state in an ISL FST corresponds to a possible string of symbols from Σ of length less than or equal to $k-1$. The transitions must be arranged as follows: a transition with input symbol a departing from a state corresponding to a string w must lead to a state which corresponds to the longest suffix of wa up to length $k-1$. For post-nasal obstruent voicing, $k-1 = 2-1 = 1$, which is why we have labeled the states in Figure 1 with all possible strings up to length 1 ($\{\lambda, N, T, D, V\}$). These states represent the last symbol that was read. This can be easily verified in the figure: all V transitions go to state V, all N transitions go to state N, and so on.

These restrictions on the state set and transitions can be interpreted as the function having a Markovian quality as it computes the output for a given input: it can only make use of the most recent input, with 'most recent' being quantified by the specified value of k. In other words, the function can only pay attention to a single piece of information at a time, and that information is the previous $k-1$ input symbols. The memory of the previous $k-1$ symbols is encoded in the states of the FST.

Any map whose computation requires more information than this is not ISL. For example, imagine a version of post-nasal obstruent voicing in which voicing only occurs following an even number of nasals. Such a logically possible—but phonologically impossible—map cannot be modeled with an ISL function, because determining whether a string of Ns is of even or odd length requires more information than just the previous $k-1$ symbols.[4] This phonologically bizarre map is subsequential, but it is provably not ISL.

The particular properties of ISL FSTs have the result that the class of functions as a whole is highly structured, in that we know a lot about the map just by knowing that it is an ISL function. This structure plays an important role in learning, as explained by Chandlee (2014), Chandlee et al. (2014), and Chandlee & Heinz (to appear). That research also demonstrates that a significant range of phonological maps can be modeled with ISL functions, including processes like local substitution, deletion, epenthesis, and synchronic metathesis, as well as morphological processes like local partial reduplication, prefixation, suffixation, and infixation. Examples of attested phonological maps that are not ISL are iterative spreading processes (i.e., NVVV ↦ NṼṼṼ) and long-distance processes like vowel harmony (with transparent vowels) (Nevins, 2010) and long-distance consonant agreement (Hansson, 2001; Rose & Walker, 2004) and dissimilation (Suzuki, 1998; Bennett, 2013). We comment on possible extensions of the ISL analysis to these cases in §6.

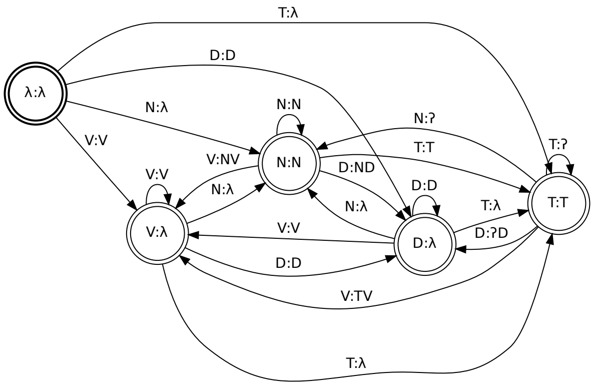
3.2 *The Repair Typology of *NC̥ is ISL.*

Here we establish that all of Pater (2004)'s *NC̥ typology can be modeled with ISL functions with $k = 2$. It will thus fall out that the entire typology can be learned using SOSFIA (explained in the next section).

The processes in Pater (2004)'s typology all fall into a particular ISL class because they all implement *local* repairs for a *local* marked structure. This marked structure, which in OT is represented by the *NC̥ constraint, corresponds in terms of a function (using our alphabet) to a NT sequence in the input. As such, they are all 2-ISL. This means that all of the maps in the typology can be described with FSTs of the same structure as the one in Figure 1, with each state representing the last symbol seen in the input. As we will now show, when viewed this way the maps in the typology vary only in the outputs of the transitions in and out of

[4] This is true regardless of how high k is. If $k = 5$, for example, then the function could at most remember that it has seen a string of 4 Ns. It cannot recall whether or not the symbol that preceded those 4 Ns was also an N, and therefore it cannot determine whether it has seen an even or odd number of Ns.

the state labeled N, in particular the T transition out of the N state (representing an input NT sequence). We will show in §4 how SOSFIA takes advantage of this shared structure by representing it directly.



/kampa/→[kappa]

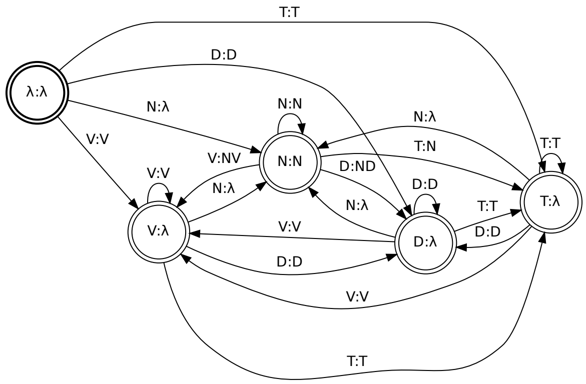
	k		a		m		p		a		#
λ	⇒	T	⇒	V	⇒	N	⇒	T	⇒	V	⇒
	k		a		λ		p		pa		λ

/kama/→[kama]

	k		a		m		a		#
λ	⇒	T	⇒	V	⇒	N	⇒	V	⇒
	k		a		λ		ma		λ

Figure 2: FST for Toba-Batak denasalization and debuccalization

Take, for example, denasalization in Toba Batak, originally given in (2). In Toba Batak, an NC̥ sequence is resolved by denasalizing the N to an obstruent of the same place. In the terms we have used here, this means that input NT sequences are transformed into output TT sequences. Complicating Toba Batak somewhat is the additional fact of debuccalization, whereby underlying /TC/ sequences surface as [ʔC], as in /ganup taɔn/→[ganuʔ taɔn] 'every year' (Hayes, 1986). However, [TC] sequences are present on the surface provided they derive from underlying /NT/ sequences which denasalize; e.g., from (2) /maŋinum tuak/↦[maŋinup tuak], 'drink palm wine' and not *[maŋinuʔtuak]. The fact that debuccalization underapplies in these cases makes the map for Toba Batak opaque. The FST for this map is shown in Figure 2. Again, for readability of the machines we have abstracted away from place features; this has no bearing on whether or not the map is 2-ISL.



	m		a		N		p		a		#
λ	⇒	T	⇒	V	⇒	N	⇒	T	⇒	V	⇒
	m		a		λ		m		a		λ

Figure 3: FST for Indonesian fusion

This FST shares the same structure as the one in Figure 1 for Quechua post-nasal voicing. However, here, note that the output side of all incoming transitions to the N state is the empty string, λ. This means that the machine has seen an N in the input but cannot yet decide what to do with it. If the next symbol is a T, it needs to change the N to a T; otherwise, the N should stay an N. This is achieved by 'waiting' to output the symbol corresponding to the input N until the output transition corresponding to the next input symbol. This can be seen in both of the examples given with Figure 2, /kampa/→[kappa] and /kama/→[kama]. In both cases, the FST writes nothing when seeing the input /m/ following the vowel, as it is 'waiting' to see the next input symbol before making a decision as to what to write for the /m/. In /kampa/, the next symbol is a /p/, and so the FST takes the T:T transition from the N state to the T state, writing out a [p] corresponding to the input /m/. (Due to debuccalization, the machine here 'waits' again to write the output corresponding to the input /p/, which may need to be written out as [ʔ] if another consonant follows. However, the vowel /a/, follows, and it takes the V:TV transition to the V state, outputting [pa].) In this way, the FST denasalizes the /m/ in the offending input NT substring. In contrast, there is no such substring in /kama/, and so after the /m/ it next sees a V, and thus it outputs [ma], leaving the /m/ unchanged.

Thus, while the details of Toba Batak denasalization differ from the Quechua devoicing process, it is still ISL for $k = 2$. As such, both functions are describable with FSTs sharing the *same* state and transition structure. The only differences between the FSTs in Figure 2 and Figure 1 is in the *outputs* on the transitions.

Likewise, the Indonesian fusion case is no different; its FST is given in Figure 3. Like the FSTs presented earlier, the FST representing Indonesian fusion has the same state set and the same transitional structure. In this way, these machines are sensitive to marked strings up to length 2 (so Strictly 2-Local constraints). The difference is in the outputs of the transitions, which correspond to the type of repair.

We have thus established that processes which repair a NC̥ structure can be modeled by a 2-ISL FST. More generally, any typology resulting from differing repairs for markedness constraints that are Strictly k-Local can be modeled with a k-ISL FST. The structure of the k-ISL FST represents every Strictly k-Local constraint, and the outputs on the transitions represent different repairs for each of these constraints.

4 Structured Onward Subsequential Inference Algorithm (SOSFIA)

As mentioned in the previous section, membership in the ISL class guarantees that a function will have a particular structure. The insight behind the Structured Onward Subsequential Inference Algorithm (SOSFIA, Jardine et al., 2014) is to represent this structure directly and manipulate it for learning. As discussed above, each state in an FST representing a function which is k-ISL represents the previous $k - 1$ symbols seen in the input. This means that *all* FSTs representing ISL functions for some k can be seen as sharing the *same* set of states and input transitions—they only vary in their output. Thus, we can represent k-ISL functions with what Jardine et al. call an 'output-empty FST', or an FST whose outputs are blank (represented here with □). The output-empty FST for the class of ISL functions with input alphabet $\Sigma = \{$N, T, D, V$\}$ and $k = 2$ is given below in Figure 4.

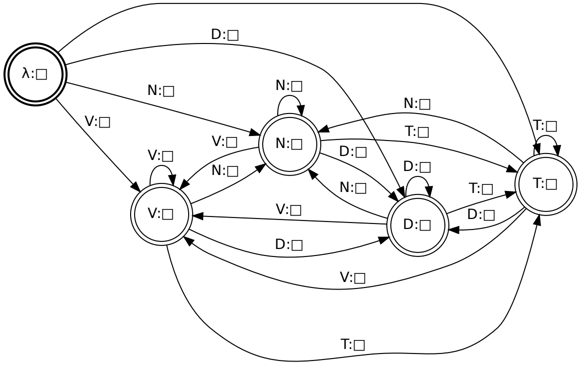


Figure 4: Output-empty FST for ISL functions where $\Sigma = \{$N, T, D, V$\}$ and $k = 2$

The reader can confirm that the FSTs presented earlier share exactly the same set of states and transitions as the output-empty FST in Figure 4. The only difference is that in those FSTs, the outputs on the transitions have been filled in. Consider for instance the voicing repair in Quechua. The output is always identical to the input (e.g., T : T) except in the case of the T transition from the N state, which represents a NT sequence in the input. In this case, the transition is T : D, which changes a T to a D exactly in that context (i.e., having just seen an N). By altering the output in this context-dependent way, we can model language-specific phonological behavior. How we can use positive data to learn an FST like Figure 1 from an output-empty FST like Figure 4 will be discussed next.

Given an output-empty transducer, SOSFIA learns any function in the class from positive data in the form of a finite set of input-output pairs that sample the target map. In simple terms, the goal is to fill in the blanks in the output-empty FST. This entails, for each transition in the FST, determining what its input symbol 'contributes' to the output. This contribution is calculated using the *longest common prefix* (LCP) of a set of strings, which is simply the longest initial sequence of symbols that set shares. For example, the LCP of the strings $\{abb, abbcd, abbbbbbba\}$ is abb. For the set of strings $\{aab, aabcd, bab\}$, the LCP is the empty string λ, because bab shares no initial symbols with aab and $aabcd$. Matching initial sequences of symbols in input strings in the data with the LCP of their corresponding output strings allows SOSFIA to align pieces of the input with pieces of the output in a principled way.[5]

To illustrate, we walk through how SOSFIA would calculate the outputs of the transitions in the output-empty FST in Figure 4 using the data in Table 1, which exhibits post-nasal obstruent voicing as in Quechua.

a) Data	b) LCP of	c) LCP of outputs	d) LCP of outputs	e) LCP of outputs
	outputs for **T**: **T**	for **TV**: **TV**	for **TVN**: **TVN**	for **TVNT**: **TVND**
(klap, klap),	(**k**lap, **k**lap),	(klap, klap),	(klap, klap),	(klap, klap),
(kap, kap),	(**k**ap, **k**ap),	(**ka**p, **ka**p),	(kap, kap),	(kap, kap),
(kam, kam),	(**k**am, **k**am),	(**ka**m, **ka**m),	(**kam**, **kam**),	(kam, kam),
(kampa, kamba),	(**k**ampa, **k**amba),	(**ka**mpa, **ka**mba),	(**kam**pa, **kam**ba),	(**kamp**a, **kamb**a),
(kampla, kambla)	(**k**ampla,**k**ambla)	(**ka**mpla,**ka**mbla)	(**kam**pla,**kam**bla)	(**kamp**la,**kamb**la)

Table 1: Sample post-nasal obstruent voicing data

Take the transition in Figure 4 labeled T from the state labeled λ to the state labeled T (the transition that appears at the very top of the diagram). This transition represents seeing a voiceless obstruent at the beginning of the word. To calculate the output for this transition, SOSFIA looks at the pairs in the data whose input strings start with the initial sequence T; that is, those which begin with a voiceless obstruent. As can be seen in column (b) in Table 1, this includes all of the pairs in the data. SOSFIA then matches this input T sequence with the LCP of the set of output strings for these pairs, {klap, kap, kam, kamba, kambla}. The LCP of this set is T, as the longest initial sequence they all share is the single initial voiceless obstruent. SOSFIA thus infers that, given an initial T in the input, the function outputs a T, and thus the output for this transition is set to T. In other words, based on this dataset, the algorithm can be confident that no matter what comes next in the input, an input string that begins with T will be mapped to an output that begins with T.

Similarly, to calculate the output for the transition labeled V from the T state to the V state, SOSFIA considers pairs in the data whose input string begins with a TV sequence. As seen in column (c) in Table 1, this is all of the pairs except (klap, klap). The LCP for the output strings of these pairs is TV. To determine the contribution of the V in the input TV sequence, SOSFIA removes from this LCP the contribution of the T, which it determined in the previous step to be T. This leaves V as the contribution of V to the output, and so SOSFIA assigns V to the output of this transition. The same process assigns N as the output to the N transition from state V to N, by using the pairs highlighted in column (d).

We can now examine how SOSFIA assigns the output to the transition crucial for the voicing process, the transition with the input label T from state N to state T. Crucially, in step (e), SOSFIA finds that the LCP of the output strings corresponding to a TVNT sequence is TVND. As it has determined in previous steps that the contributions of T, V, and N in the input are T, V, and N, respectively, the remaining D in the output must correspond to the final T in the input. In this way, SOSFIA learns that the T in an NT sequence

[5] This principle is called *onwardness*, and while it is not necessary for representing functions with FSTs, it is crucial for the known algorithms that learn them. For formal details on the utility of onwardness, see Oncina et al. (1993), Chandlee et al. (2014), and Jardine et al. (2014).

should be output as a D. The partially filled-in machine after these 5 steps is given in Figure 5, with the filled-in transitions highlighted in bold. Jardine et al. (2014) prove that given any sufficient sample of input-output pairs, SOSFIA will correctly determine the output for all transitions and therefore return the FST that generated the data set S. They further prove that a sample is sufficient if it contains a particular finite set of pairs whose size is linear in the size of the FST and that SOSFIA runs in linear time in the size of S. In other words, the learning is not just successful, it is *efficient*.

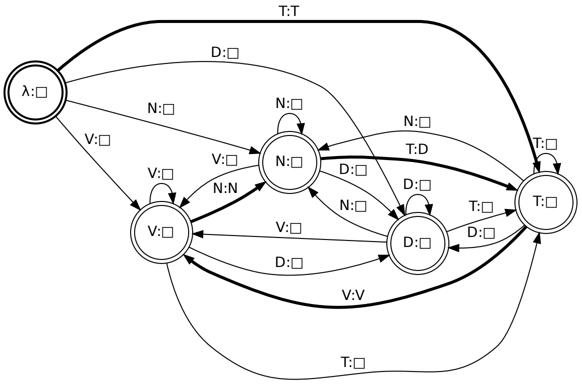


Figure 5: Result of running SOSFIA on data from Table 1

It is important to note that although it calculates outputs for transitions based on initial sequences in the input, the structure of the machine forces SOSFIA to *generalize* beyond its input. Thus, while the machine in Figure 5 was created through learning on the data points in Table 1, it will correctly devoice multiple NT sequences in an input. For example, the reader can confirm that it transforms an input /kampampa/, which was not seen in Table 1, to [kambamba].

5 Advantages

Here we argue that a theory of phonological maps based on the computational properties like those that characterize ISL functions has some advantages over theories based on optimization. The next section discusses current limitations of this theory and suggests avenues of future research that may address them.

The proposed theory has much in common with OT. Like OT, ISL functions divorce markedness from their repairs. In OT markedness is given in terms of the markedness constraints and the different possible repairs are given according to different faithfulness constraints. The actual repair in a language is determined according to some language-specific prioritization of the constraints. For k-ISL functions, the markedness constraints are exactly the Strictly k-Local ones, a fact realized in the FSTs through the state set and the organization of the transitions. Faithfulness is not determined according to faithfulness constraints, however, but is instead constrained by the previous $k - 1$ input elements. The repair for a given constraint is realized as an output label of a transition that differs from its input label. So OT maps and ISL maps both distinguish marked structures from repairs, though these concepts are expressed differently in the two formalisms. Also like OT, a theory based on ISL maps does not make use of intermediate representations, as was common with rule-based analyses.

Another important similarity to OT is the existence of learnability results with theoretical guarantees. RCD takes as input a constraint set CON and a set S of UR-SR pairs, and there is an important theoretical guarantee that RCD will return a ranking of the constraints consistent with S (if one exists). Much subsequent work on learning in OT is predicated on this result. Similarly, SOSFIA takes as input an output-empty transducer (for example one for k-ISL functions) and a set S of UR-SR pairs, and there are likewise important theoretical guarantees about the conditions under which the target grammar can be acquired.

Major differences between the two theories also exist, however. The theory presented here can represent opaque maps, as long as they are ISL. The map for Toba Batak in Figure 2 is a case in point. It is well known that opaque maps are problematic for classic OT (Idsardi, 1998, 2000). In OT, debuccalization in Toba Batak would be motivated by a highly ranked *TC constraint. However, denasalizing underlying /NT/ sequence creates a surface [TC] structure in violation of this constraint that would be less optimal than a debuccalizing candidate. Classic OT grammars cannot describe counterfeeding chains of this type without additional devices. However, the FST in Figure 2 represents both denasalization and glottalization (see the transitions in and out of the T state), which shows that the map is 2-ISL. Thus, there is no inherent difficulty to representing (and learning) opaque maps in the theory offered here. 2-ISL functions can be opaque, and SOSFIA can learn these maps, due to the fact that it is focusing on the computational nature of the map directly.

In addition to undergenerating the typology of phonology, OT also has been shown to overgenerate. It is uncontroversial that OT generates non-regular maps from simple constraints (Frank & Satta, 1998; Riggle, 2004; Gerdemann & Hulden, 2012). This is in contrast to the fact that non-regular maps are considered to be phonologically bizarre. Majority Rules vowel harmony is one example of such a non-phonological, non-regular map (Baković, 2000; Wilson, 2003; Finley, 2008; Heinz & Lai, 2013). Strong evidence has yet to be presented that the right analysis of the phonology of any language requires positing a non-regular map.

It is true that some variants of classic OT have made headway on the undergeneration problem. For example Stratal OT (Kiparsky, 2000) can represent opaque maps. Unfortunately, Stratal OT does not yet come with learning results, nor does it eliminate the overgeneration problem, since nothing prevents the last stratum from including the ranking which generates Majority Rules. Harmonic Serialism (HS) has also been argued to eliminate some of the unwanted overgeneration in the factorial typology of classic OT (McCarthy, 2008). However, there are no theoretical results for learning HS grammars, and there is no guarantee that non-regular maps like Majority Rules are impossible for them to generate.[6]

Finally, there is a sense in which OT misses an important generalization about the nature of locality in phonological maps, which ISL functions capture directly. As explained here and in more detail by Chandlee (2014) and Chandlee & Heinz (to appear), maps that are 'local' have the ISL property. The fact that these maps are ISL reveals an important aspect of their true nature: *the output strings in the map fundamentally depend on sequences of length k in the input*. OT, with its markedness constraints, squarely puts a focus on *output* forms. This division of constraints into markedness and faithfulness—combined with optimization over some ranking of them—misses the generalization that phonological grammars describing local maps only need to pay attention to pieces of length k of the input string.

6 Current Limitations and Future Directions

We have demonstrated the contribution to phonological learning of the computational property of Input Strict Locality, but as a complete theory of phonology, this approach of course has limitations. One is that SOSFIA's input data is a set of input-output string pairs corresponding to UR-SR pairs, but in a more naturalistic language learning environment only the surface forms are provided. How are the URs identified? This is an ongoing area of research for many models of phonological learning, with recent results in classic OT under the assumption that maps are output-driven (Tesar, 2014).

A second question relates to the types of UR-SR pairs that SOSFIA needs to correctly infer the target mapping. Certain pairs may not be made available due to certain phonotactic restrictions in the language in question. The learner will then need to deduce the needed missing data from the data that is present, in some manner along the lines of the 'faithfulness' and 'community' biases employed in the work of Gildea & Jurafsky (1996). The best way to incorporate these kinds of biases in SOSFIA remains open.

A third issue is the 'too many solutions' problem. Not all logically possible repairs appear to be available to marked structures (Lombardi, 2001; Blumenfeld, 2005, 2006). Like OT, ISL maps on their own do not account for such facts.

Lastly, as mentioned above, not all phonological maps are ISL. In other work, we have shown how iterative spreading processes, like the nasal spreading shown in (5), can only be modeled with a comparable notion of *Output Strict Locality* (Chandlee, 2014; Chandlee & Heinz, 2015).

(5) /maap/ $\mapsto$ [mããp], 'pardon', Johore Malay (Onn, 1980; McCarthy, 2011)

The reason such a process is not ISL is because the trigger for nasalization of the second vowel is not present in the input (i.e., underlyingly). Instead, it is present on the output side of the transition after the first vowel is

[6] It is clear that there are harmonically improving paths from underlying forms like $/- - + + +/$ and $/- - - + +/$ to surface forms like $[+ + + + +]$ and $[- - - - -]$, respectively.

read and output as ã. Such processes therefore belong not to the ISL class but to the class of Output Strictly Local functions (Chandlee, 2014; Chandlee et al., 2014), which have much in common with the ISL class but for which the details of their FST characterization and other properties are still being worked out.

In addition, long-distance processes in which the target and trigger do not form a contiguous substring of bounded length (such as vowel harmony with transparent vowels or long-distance consonant agreement and dissimilation) are also not ISL. However, we believe, following the work of Heinz (2009, 2010), that additional subregular classes of functions can be defined that will model these long-distance maps. We further expect such functions are also likely to be learnable in a manner similar to the one presented here.

7 Conclusion

We have presented an approach to modeling the phonological grammar that, like OT and its variants, assumes a separation between marked structures and repairs. Using the restrictive subclass of regular relations called the Input Strictly Local functions, we have demonstrated how a single FST can represent a repair typology. The particular output of the FST at the locus of violation of the marked structure corresponds to the chosen repair strategy.

There are several attractive aspects of this proposal. It is restrictive since ISL functions are the strongest computational characterization to date of local phonological processes. It is expressive; even opaque maps, contra classic OT, can be represented with ISL functions. It is learnable; both the ISLFLA and SOSFIA algorithms provide theoretical learning guarantees. These algorithms exploit the *a priori* knowledge of the structure of the ISL class to efficiently generalize the target phonological map from a finite amount of positive data. While further issues remain, we believe the shape of the theory being offered holds much promise.

References

Baković, Eric (2000). *Harmony, Dominance and Control*. Ph.D. thesis, Rutgers University.

Baković, Eric (2013). *Blocking and Complementarity in Phonological Theory*. Equinox, Bristol, CT.

Bennett, William (2013). *Dissimilation, Consonant Harmony, and Surface Correspondence*. Ph.D. thesis, Rutgers.

Blumenfeld, Lev (2005). Too many solutions: prosody and segmental effect. *Proceedings from the Annual Meeting of the Chicago Linguistics Society*, vol. 41, 17–31.

Blumenfeld, Lev (2006). *Constraints on Phonological Interactions*. Ph.D. thesis, Stanford University.

Chandlee, Jane (2014). *Strictly Local Phonological Processes*. Ph.D. thesis, University of Delaware.

Chandlee, Jane & Jeffrey Heinz (2012). Bounded copying is subsequential: implications for metathesis and reduplication. *Proceedings of SIGMORPHON 12*.

Chandlee, Jane & Jeffrey Heinz (2015). Using locality to learn long-distance phonological processes. Talk given at the Annual Meeting of the Linguistics Society of America, Portland, OR, January 8-11, 2015.

Chandlee, Jane & Jeffrey Heinz (to appear). Strictly local phonological processes. *Linguistic Inquiry,* Under revision.

Chandlee, Jane, Angeliki Athanasopoulou & Jeffrey Heinz (2012). Evidence for classifying metathesis patterns as subsequential. Choi, Jaehoon, E. Alan Hogue, Jeffrey Punske, Deniz Tat, Jessamyn Schertz & Alex Trueman (eds.), *WCCFL 29: Proceedings of the 29th West Coast Conference on Formal Linguistics*, Cascadilla, Somerville, MA, 303–309.

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

Chomsky, Noam & Morris Halle (1968). *The Sound Pattern of English*. Harper & Row, New York.

Finley, Sara (2008). *The formal and cognitive restrictions on vowel harmony*. Ph.D. thesis, Johns Hopkins University, Baltimore, MD.

Frank, Robert & Giorgio Satta (1998). Optimality Theory and the generative complexity of constraint violability. *Computational Linguistics* 24:2, 307–315.

Gainor, Brian, Regine Lai & Jeffrey Heinz (2012). Computational characterizations of vowel harmony patterns and pathologies. Choi, Jaehoon, E. Alan Hogue, Jeffrey Punske, Deniz Tat, Jessamyn Schertz & Alex Trueman (eds.), *WCCFL 29: Proceedings of the 29th West Coast Conference on Formal Linguistics*, Cascadilla, Somerville, MA, 63–71.

Gerdemann, Dale & Mans Hulden (2012). Practical finite state Optimality Theory. *Proceedings of the 10th International Workshop on Finite State Methods and Natural Language Processing*, Association for Computational Linguistics, Donostia–San Sebastin, 10–19.

Gildea, Daniel & Daniel Jurafsky (1996). Learning bias and phonological rule induction. *Computational Linguistics* 24:4.

Gold, Mark E. (1967). Language identification in the limit. *Information and Control* 10, 447–474.

Hansson, Gunnar (2001). *Theoretical and Typological Issues in Consonant Harmony*. Ph.D. thesis, University of California, Berkeley.

Hayes, Bruce (1986). Assimilation as spreading in Toba Batak. *Linguistic Inquiry* 17:3, 467–499.

Heinz, Jeffrey (2009). On the role of locality in learning stress patterns. *Phonology* 26, 303–351.

Heinz, Jeffrey (2010). Learning long-distance phonotactics. *Linguistic Inquiry* 41, 623–661.

Heinz, Jeffrey & Regine Lai (2013). Vowel harmony and subsequentiality. Kornai, Andras & Marco Kuhlmann (eds.), *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, 52–63.

de la Higuera, Colin (2010). *Grammatical Inference: Learning Automata Grammars*. Cambridge University Press.

Idsardi, William (1998). Tiberian Hebrew spirantization and phonological derivations. *Linguistic Inquiry* 29, 37–73.

Idsardi, William J. (2000). Clarifying opacity. *The Linguistic Review* 17, 337–350.

Jardine, Adam (2013). Tone is (computationally) different. Unpublished manuscript, University of Delaware.

Jardine, Adam, Jane Chandlee, Rémi Eyraud & Jeffrey Heinz (2014). Very efficient learning of structured classes of subsequential functions from positive data. Paper submitted to the 12th International Conference on Grammatical Inference (ICGI 2014).

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

Kaplan, Ronald & Martin Kay (1994). Regular models of phonological rule systems. *Computational Linguistics* 20, 331–78.

Kiparsky, Paul (2000). Opacity and cyclicity. *The Linguistic Review* 17, 351–366.

Koskenniemi, Kimmo (1983). *Two-Level Morphology: A general computational model for word-form recognition and production*. University of Helsinki, Department of General Linguistics.

Legendre, Geraldine, Yoshiro Miyata & Paul Smolensky (1990). Harmonic grammar: A formal multi level connectionist theory of linguistic well formedness: Theoretical foundations. *Proceedings of the Twelfth Annual Conference of the Cognitive Science Society*, Cambridge, MA., 388–395.

Lombardi, Linda (2001). Why place and voice are different. Lombardi, Linda (ed.), *Segmental Phonology in Optimality Theory*, Cambridge University Press.

Luo, Huan (2013). Long-distance consonant harmony and subsequentiality. Unpublished manuscript, University of Delaware.

McCarthy, John J. (2000). Harmonic serialism and parallelism. Hirotani, Masako, Andries Coetzee, Nancy Hall & Jiyung Kim (eds.), *NELS 30: Proceedings of the 30th Annual Meeting of the North East Linguistic Society*, GLSA, University of Massachusetts Amherst, 501–524.

McCarthy, John J. (2008). The gradual path to cluster simplification. *Phonology* 25:2, 271–319.

McCarthy, John J. (2011). Autosegmental spreading in Optimality Theory. Goldsmith, John, Elizabeth Hume & Leo Wetzels (eds.), *Tones and Features (Clements Memorial Volume)*, Berlin: Mouton de Gruyter.

Mohri, Mehryar (1997). Finite-state transducers in language and speech processing. *Computational Linguistics* 23:2, 269–311.

Nevins, A. (2010). *Locality in Vowel Harmony*. MIT Press.

Oncina, José, Pedro García & Enrique Vidal (1993). Learning subsequential transducers for pattern recognition tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15, 448–458.

Onn, Farid M. (1980). *Aspects of Malay Phonology and Morphology: A Generative Approach*. Kuala Lumpur: Universiti Kebangsaan Malaysia.

Pater, Joe (2004). Austronesian nasal substitution and other NC effects. McCarthy, John (ed.), *Optimality Theory in Phonology: A Reader*, Oxford and Malden, MA: Blackwell, 271–289.

Pater, Joe (2012). Serial Harmonic Grammar and Berber syllabification. Borowsky, Toni, Shigeto Kawahara, Takahito Shinya & Mariko Sugahara (eds.), *Prosody Matters: Essays in Honor of Elisabeth O. Selkirk*, London, Equinox Press, 43–72.

Payne, Amanda (2014). Dissimilation as a subsequential process. *Proceedings of the 44th Meeting of the North East Linguistic Society (NELS)*.

Prince, Alan & Paul Smolensky (2004). *Optimality Theory: Constraint Interaction in Generative Grammar*. Blackwell.

Riggle, Jason (2004). *Generation, Recognition, and Learning in Finite State Optimality Theory*. Ph.D. thesis, UCLA.

Rogers, James & Geoffrey Pullum (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20, 329–342.

Rogers, James, Jeffrey Heinz, Gil Bailey, Matt Edlefsen, Molly Visscher, David Wellcome & Sean Wibel (2010). On languages piecewise testable in the strict sense. Ebert, Christian, Gerhard Jäger & Jens Michaelis (eds.), *The Mathematics of Language*, Springer, vol. 6149 of *Lecture Notes in Artifical Intelligence*, 255–265.

Rogers, James, Jeffrey Heinz, Margaret Fero, Jeremy Hurst, Dakotah Lambert & Sean Wibel (2013). Cognitive and sub-regular complexity. *Formal Grammar*, Springer, vol. 8036 of *Lecture Notes in Computer Science*, 90–108.

Rose, Sharon & Rachel Walker (2004). A typology of consonant agreement as correspondence. *Language* 80, 475–531.

Suzuki, Keiichiro (1998). *A Typological Investigation of Dissimilation*. Ph.D. thesis, University of Arizona.

Tesar, Bruce (2014). *Output-Driven Phonology: Theory and Learning*. Cambridge, Cambridge University Press.

Tesar, Bruce & Paul Smolensky (1998). Learnability in Optimality Theory. *Linguistic Inquiry* 29, 229–268.

Wilson, Colin (2003). Analyzing unbounded spreading with constraints: marks, targets, and derivations. Unpublished manuscript, UCLA.