Multi-Tiered Strictly Local Functions

Phillip Burness
University of Ottawa
pburn036@uottawa.ca

Kevin McMullin
University of Ottawa
kevin.mcmullin@uottawa.ca

Abstract

Tier-based Strictly Local functions, as they have so far been defined, are equipped with just a single tier. In light of this fact, they are currently incapable of modelling simultaneous phonological processes that would require different tiers. In this paper we consider whether and how we can allow a single function to operate over more than one tier. We conclude that multiple tiers can and should be permitted, but that the relationships between them must be restricted in some way to avoid overgeneration. The particular restriction that we propose comes in two parts. First, each input element is associated with a set of tiers that on their own can fully determine what the element is mapped to. Second, the set of tiers associated to a given input element must form a strict superset-subset hierarchy. In this way, we can track multiple, related sources of information when deciding how to process a particular input element. We demonstrate that doing so enables simple and intuitive analyses to otherwise challenging phonological phenomena.

1 Introduction

Many theoretical analyses of long-distance phonological patterns share the core intuition that a *tier* or *projection* of segments can allow for 'local' relationships to be established between non-adjacent elements by excluding material that is irrelevant from the implicated level of representation. For example, Samala (also known as Ineseño Chumash) has a long-distance process of sibilant harmony in which an underlying /s/ surfaces as [ʃ] if another [ʃ] appears anywhere later in the word, as in /ha-s-xintila-waʃ/ → [haʃxintilawaʃ] 'his former gentile name' (Applegate, 1972). The harmony target can be arbitrarily far from the harmony trigger in the full string of segments, but the trigger and target are rendered adjacent on a level of representation containing all and only the sibilant consonants.

Within the subregular hierarchy of formal languages, the class of Strictly Local languages, which model local phonotactic restrictions (McNaughton and Papert, 1971; Rogers and Pullum, 2011; Rogers et al., 2013) have been extended to incorporate the notion of a tier, resulting in the class of *Tier-based Strictly Local* (TSL) languages (Heinz et al., 2011). Likewise, the Input Strictly Local (ISL) and Output Strictly Local (OSL) functions—which characterize locally-bounded phonological processes as subregular maps (Chandlee, 2014; Chandlee et al., 2014, 2015)—have been generalized to the classes of Input Tier-based Strictly Local (ITSL) and Output Tier-based Strictly Local (OTSL) functions in order to account for non-local phonological processes (Burness and McMullin, 2019; Hao and Andersson, 2019; Hao and Bowers, 2019).

These TSL formal languages and functions successfully model a wide range of long-distance phonological patterns (McMullin, 2016; McMullin and Hansson, 2016; Burness and McMullin, 2019; Hao and Bowers, 2019), and moreover have desirable properties for learnability (Jardine and Heinz, 2016; Jardine and McMullin, 2017; Burness and McMullin, 2019). However, they suffer from a major drawback in that they are restricted to a single tier, and as a consequence are ill-equipped to deal with multiple, simultaneous long-distance dependencies. For example, the Tamashek dialect of Tuareg exhibits regressive long-distance sibilant harmony and regressive long-distance labial dissimilation (Heath, 2005; McMullin, 2016). Each of the two processes can be modelled in isolation as a TSL function, but there is no single TSL function that can apply both rules simultaneously. The solution we pursue in this paper is to give functions access to more than

Proceedings of the Seventeenth SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology, pages 245–255

https://doi.org/10.18653/v1/P17

one tier.

Incorporating multiple tiers into a single function is relatively straightforward; the difficulty lies in understanding the computational properties of such functions and establishing appropriate restrictions on the number of tiers and the relationships between tiers. Progress has been made with regards to properly restricting multi-tiered languages (Aksënova and Deskmukh, 2018; McMullin et al., 2019), and this paper considers the restrictions that need to be imposed onto the multiple tiers of a multi-tiered function. Our proposal is to associate each input element with a set of tiers that must fall into a strict subset-superset hierarchy (i.e., each tier in the set is a strict subset of the next largest tier in the set, if it exists). The output corresponding to an input element then depends on its dedicated set of tiers, and no others.

The rest of this paper is structured as follows. Section 2 introduces the notation that will be used throughout the paper. Section 3 presents the TSL functions as they are currently defined. Section 4 discusses some limitations of TSL functions that this paper aims to address. Section 5 formally defines the Multi-Tiered Strictly Local (MTSL) functions, and formalizes the restrictions that we propose must hold over the relationships among multiple tiers. Section 6 demonstrates how appropriately restricted MTSL functions overcome the limitations highlighted in Section 4. Finally, Section 7 concludes and provides directions for future research.

2 Preliminaries

To start, let Σ be an alphabet of symbols, which in the context of phonotactics represents a language's inventory of surface phones. A string w is a finite contiguous sequence of symbols from Σ, and $|w|$ denotes the length of w. We write λ for the unique string of length 0 (the empty string). We use Σ^* to denote the set of all strings of any length that can be made from elements in Σ. Note that Σ^* includes the empty string. Given two strings u and v, we write $u \cdot v$ to denote their concatenation, though we will often simply write uv when context permits. A k-factor of a string w is any contiguous substring of w with length k, though in the special case that $|w| \leq k$, w is its own and only k-factor. In what follows, $\texttt{fac}_k(w)$ denotes all the k-factors contained in a string w.

A prefix of some string $w \in \Sigma^*$ is any string $u \in \Sigma^*$ such that $w = u \cdot x$ and $x \in \Sigma^*$. A suffix of some string $w \in \Sigma^*$ is any string $u \in \Sigma^*$ such that $w = x \cdot u$ and $x \in \Sigma^*$. Note that any string is a suffix of itself, and that λ is a suffix of every string. When $|w| \geq n$, $\texttt{suff}^n(w)$ denotes the unique suffix of w with a length of n; when $|w| < n$, it simply denotes w itself. Given a string w and one of its prefixes u we write $u^{-1} \cdot w$ to denote w with u removed from its front. For example, $ab^{-1} \cdot abcde = cde$. Finally, given a set of strings S, we write $\texttt{lcp}(S)$ to denote the *longest common prefix* of S, which is the string u such that u is a prefix of every $w \in S$, and there exists no other string v such that $|v| > |u|$ and v is also a prefix of every $w \in S$.

A string-to-string function pairs every $w \in \Sigma^*$ with one $y \in \Delta^*$, where Σ and Δ are the input alphabet and output alphabet respectively. Given a set of input strings $I \subseteq \Sigma^*$, $f(I) = \bigcup_{i \in I}\{f(i)\}$ is the set of all outputs associated to at least one of the inputs. An important concept is that of the *tails* of an input string w with respect to a function f.

Definition 1. Tails (Oncina and Garcia, 1991) *Given a function f and an input $w \in \Sigma^*$, $\texttt{tails}_f(w) = \{(y, v) \mid f(wy) = uv \wedge u = \texttt{lcp}(f(w\Sigma^*))\}$.*

In words, $\texttt{tails}_f(w)$ pairs every possible string $y \in \Sigma^*$ with the portion of $f(wy)$ that is *directly attributable* to y. Put another way, the tails of w are the effect that w has on the output of any subsequent string of input symbols. Consider a function f computing post-nasal voicing. The tails of w_1 = /tan/ relative to f will include pairs such as ⟨t, d⟩ since f(tant) = [tand] and ⟨opu, opu⟩ since f(tanopu) = [tanopu]. When $\texttt{tails}_f(w_1) = \texttt{tails}_f(w_2)$, we say that w_1 and w_2 are *tail-equivalent* with respect to f. The string w_2 = /ken/ is tail-equivalent to w_1 with respect to f because they both end in a nasal consonant, and therefore both trigger post-nasal voicing. An example of a string that is not tail-equivalent to w_1 or w_2 would be w_3 = /sini/; its tail for /t/ is ⟨t, t⟩ since it does not end in a nasal consonant and so does not trigger post-nasal voicing.

Throughout the rest of this paper, we will need to be able to pick out the portion of the output that corresponds to actual input material. Viewed from another perspective, we need to be able to ignore the portion of the output that would correspond to a word-end symbol. To make this distinction,

Chandlee et al. (2015) defined the prefix function f^p associated with a function f such that $f^p(w)$ is equal to the longest common prefix of all strings $f(wx)$, where x is some member of Σ^*.

Definition 2. Prefix function
Given a function f, its associated prefix function f^p is such that $f^p(w) = \texttt{lcp}(f(w\Sigma^*))$.

An example where $f(w)$ and $f^p(w)$ differ would be a function that appends a to the end of every input string. In this case, f^p is simply the identity map, so $f^p(abc) = abc$ whereas $f(abc) = abca$.

3 Background

Like their name suggests, the Input Strictly Local (ISL) and Output Strictly Local (OSL) functions take the Strictly Local (SL) languages as their base. Theorem 1 states the property of the SL languages that allowed for the jump to ISL and OSL functions. This property is known as Suffix Substitution Closure (SSC; Rogers and Pullum, 2011; Rogers et al., 2013). Informally, if two grammatical strings share a middle portion of at least length $k-1$ (i.e., if $w_1 = axb$, $w_2 = cxd$, and $|x| \geq k-1$), then we can substitute the suffixes that come after the overlap without causing ungrammaticality. A corollary of SSC is that any two grammatical strings from an SL_k language that end in the same $k-1$ (or more) symbols can be legally continued by the exact same set of strings (Chandlee et al., 2015).

Theorem 1. Suffix Substitution Closure (Rogers et al., 2013)
A stringset L is Strictly k-Local if and only if whenever there is a string x of length $k-1$ and strings u_1, u_2, v_1, v_2, it is the case that:

$$[u_1xv_1 \in L \wedge u_2xv_2 \in L] \Rightarrow u_1xv_2 \in L$$

Corollary 1. Suffix-defined Residuals (Chandlee et al., 2015)
A stringset L is Strictly k-Local if and only if for all pairs $w_1, w_2 \in \Sigma^*$:

$$\texttt{suff}^{k-1}(w_1) = \texttt{suff}^{k-1}(w_2) \Rightarrow$$
$$\{v \mid w_1 \cdot v \in L\} = \{v \mid w_2 \cdot v \in L\}$$

The definitions of the ISL and OSL functions take the property in Corollary 1 and adapt it so that it applies to function tails. Informally, a function f is ISL if the tail-equivalence classes of f correspond to input suffixes, and a function f is OSL if the tail-equivalence classes of f correspond to output suffixes (or more accurately, suffixes of f^p).

Definition 3. Input Strictly k-Local Functions
A function $f : \Sigma^* \to \Delta^*$ is ISL_k if for all w_1, w_2 in Σ^*:

$$\texttt{suff}^{k-1}(w_1) = \texttt{suff}^{k-1}(w_2) \Rightarrow$$
$$\texttt{tails}_f(w_1) = \texttt{tails}_f(w_2)$$

Definition 4. Output Strictly k-Local Functions
A function $f : \Sigma^* \to \Delta^*$ is OSL_k if for all w_1, w_2 in Σ^*:

$$\texttt{suff}^{k-1}(f^p(w_1)) = \texttt{suff}^{k-1}(f^p(w_2)) \Rightarrow$$
$$\texttt{tails}_f(w_1) = \texttt{tails}_f(w_2)$$

Chandlee (2014) and Chandlee et al. (2014, 2015) show that most iterative phonological processes can be modelled with an OSL function, with an important exception being long-distance iterative processes like consonant harmony. This is parallel to the fact that long-distance phonotactics cannot be represented with an SL stringset, which motivated Heinz et al. (2011) to define the Tier-based Strictly Local (TSL) languages—stringsets that are SL after an erasure function has applied, masking all symbols that are irrelevant to the restrictions that the language places on its strings. The erasure function takes a tier τ and a string w, returning w with all non-tier elements removed.

Definition 5. Erasure function
Given an alphabet Σ, a tier $\tau \subseteq \Sigma$, and a string $w = a_1 \ldots a_n$, $\texttt{erase}_\tau(w) = b_1 \ldots b_n$ where for all $i \leq n$, $b_i = a_i$ if $a_i \in \tau$, else $b_i = \lambda$.

Definition 6. Tier-based Strictly k-Local languages
A language L is TSL_k if there is a tier $\tau \subseteq \Sigma$ and a subset $S \subseteq \texttt{fac}_k(\rtimes\tau^*\ltimes)$ such that:

$$L = \{w \in \Sigma^* \mid \texttt{fac}_k(\rtimes\texttt{erase}_\tau(w)\ltimes) \subseteq S\}$$

As it turns out, the TSL languages also exhibit a form of Suffix Substitution Closure (Lambert and Rogers, 2020). In light of this fact, the legal continuations of any string w in a TSL_k language can be inferred simply by looking at the $k-1$ suffix of $\texttt{erase}_\tau(w)$. Just as we did for the ISL and OSL functions, then, we can define the ITSL and OTSL functions according to how they partition Σ^* into tail-equivalence classes. For convenience, we will write $\texttt{suff}^n_\tau(w)$ to mean $\texttt{suff}^n(\texttt{erase}_\tau(w))$ in what follows.

Definition 7. Input Tier-based Strictly k-Local Functions
A function $f : \Sigma^* \to \Delta^*$ is ITSL$_k$ if there is a tier $\tau \subseteq \Sigma$ such that for all w_1, w_2 in Σ^*:

$$\mathtt{suff}_\tau^{k-1}(w_1) = \mathtt{suff}_\tau^{k-1}(w_2) \Rightarrow \mathtt{tails}_f(w_1) = \mathtt{tails}_f(w_2)$$

Definition 8. Output Tier-based Strictly k-Local Functions
A function $f : \Sigma^* \to \Delta^*$ is OTSL$_k$ if there is a tier $\tau \subseteq \Delta$ such that for all w_1, w_2 in Σ^*:

$$\mathtt{suff}_\tau^{k-1}(f^p(w_1)) = \mathtt{suff}_\tau^{k-1}(f^p(w_2)) \Rightarrow \mathtt{tails}_f(w_1) = \mathtt{tails}_f(w_2)$$

Informally, a function f is ITSL if the tail-equivalence classes of f correspond to input tier suffixes (where the tier is a subset of the input alphabet Σ), and a function f is OTSL if the tail-equivalence classes of f correspond to output tier suffixes (where the tier is a subset of the output alphabet Δ). Important to note is that the ITSL functions properly contain the ISL functions and that the OTSL functions properly contain the OSL functions. This is because anything SL is also TSL when the tier is simply the entire alphabet.

The above definitions of the ITSL and OTSL functions abstract away from reading direction, which divides each class into two overlapping but distinct classes. This is similar to how the subsequential functions can be divided into the left-subsequential functions which read the input from left to right and the right-subsequential functions which read the input from right to left (Heinz and Lai, 2013). We will also abstract away from reading direction when defining our multi-tiered functions, but will specify the directionality of individual functions.

4 Limitations of TSL functions

As discussed in Burness and McMullin (2019), the TSL functions are quite versatile, being able to model long-distance harmony and long-distance dissimilation, both with and without blocking effects. This is, however, only the case when we model each phonological process of a language in isolation.

Consider the Tamashek dialect of Tuareg, which contains a process of long-distance regressive sibilant harmony and a process of long-distance regressive labial dissimilation (Heath, 2005; McMullin, 2016). The sibilant harmony can be seen in words where the causative prefix /s-/ is followed non-locally by another sibilant, whereupon it takes that other sibilant's values for anteriority, voicing, and pharyngealization as shown by the data in (1) from Heath (2005, p. 442). Note that the language has considerable vowel allophony, and we write 'V' where Heath (2005) does not provide the surface vowel quality. The labial dissimilation can be seen in words where a prefix /m/ (such as in the mediopassive) is followed non-locally by a labial consonant other than /w/, whereupon the prefix /m/ will dissimilate to [n]. Data for this process is shown in (2) from Heath (2005, p. 472). That both processes can occur simultaneously is demonstrated by the word in (3) from Heath (2005, p. 462), which contains the causative and mediopassive prefixes together.

(1) Sibilant harmony: causative /s-/
-s-VŋŋV- 'cook'
-s-VsVfVr- 'treat (patient)'
-sˤ-VsˤuhV- 'strengthen'
-ʃ-VluʃV- 'clean sand from'
-z-VjVzzV 'scrutinize'

(2) Labial dissimilation: mediopassive /m-/
-m-VrtVj- 'become mixed'
-n-VkmVm- 'be squeezed'

(3) Both prefixes/processes
ɑ-zˤ-ənː-ət-əlməzˤ 'spitting saliva'

The combination of sibilant harmony and labial dissimilation cannot, however, be computed by a single TSL function. To see why, consider what happens when we try with an OTSL$_2$ function whose tier consists of all sibilants and labial consonants. Producing a sibilant will push the most recent labial consonant (if any) out of the $k-1$ window, and producing a labial consonant will push the most recent sibilant (if any) out of the $k-1$ window. At any given point, then, we can only know how to correctly map an input /s/ or only know how to correctly map an input /m/. Increasing k does not eliminate the issue, since any number of sibilants can in principle occur between two labial consonants, and any number of labial consonants can in principle occur between two sibilants.

A further difficulty for TSL functions posed by Tamashek Tuareg is that long-distance regressive labial dissimilation interacts with a local process of regressive nasal place assimilation. The complication arises from the fact that the local pro-

cess overrides the long-distance process: /m/ fails to dissimilate specifically when it is immediately followed by an oral labial stop, in order to avoid a heterorganic cluster. This can be seen in a word like (4) from Heath (2005, p. 476) where the reciprocal prefix /Vm-/ comes immediately before a /b/. We are faced with a paradox if we attempt to model this interaction as a single TSL function. In order to know when an input labial needs to dissimilate we need to ignore everything that is not a labial consonant, but in order to know when an input labial needs to obey place assimilation we cannot ignore anything.

(4) Local blocking of dissimilation
-æm-bæbbɑ- 'carried each other'

Whether these interactions are problematic depends on whether we conceive of a language's phonological system as a series of input-output maps or as a single "master" input-output map. The latter position is explored by Chandlee and Heinz (2018) and Chandlee et al. (2018) in the context of formal language theory and automata theory. They remark that the ISL functions are not closed under composition, but that certain opaque rule interactions (counterfeeding, counterbleeding, etc.) can nevertheless be modelled using a single ISL function when the component rules are also ISL functions.

5 Multi-Tiered Strictly Local functions

The main method that we will use to tackle the limitations presented in the previous Section is by allowing a single function to operate over multiple tiers. We call these the Multi-Tiered Strictly k-Local (MTSL$_k$) functions, for lack of a better name. Note that we use a large subscripted conjunction symbol to collapse a series of formulae that are identical aside from using different tiers.

Definition 9. Input Multi-Tiered Strictly k-Local functions
A function $f : \Sigma^* \to \Delta^*$ is IMTSL$_k$ if there is a finite set T of tiers $\tau \subseteq \Sigma$ such that for all $w_1, w_2 \in \Sigma^*$:

$$[\bigwedge_{\tau \in T} [\mathtt{suff}_\tau^{k-1}(w_1) = \mathtt{suff}_\tau^{k-1}(w_2)]] \Rightarrow [\mathtt{tails}_f(w_1) = \mathtt{tails}_f(w_2)]$$

Definition 10. Output Multi-Tiered Strictly k-Local functions
A function $f : \Sigma^* \to \Delta^*$ is OMTSL$_k$ if there is a finite set T of tiers $\tau \subseteq \Delta$ such that for all $w_1, w_2 \in \Sigma^*$:

$$[\bigwedge_{\tau \in T} [\mathtt{suff}_\tau^{k-1}(f^p(w_1)) = \mathtt{suff}_\tau^{k-1}(f^p(w_2))]] \Rightarrow [\mathtt{tails}_f(w_1) = \mathtt{tails}_f(w_2)]$$

In words, there is a finite set of tiers such that if two input strings share a $k-1$ suffix *on all tiers*, then they will have the same tails. Notice how this is a direct extension of the TSL$_k$ functions, since the singleton tier of a TSL$_k$ function will satisfy the above definition.

Of course, allowing any conceivable number of tiers and allowing any conceivable relationship between their contents is too powerful. One pathological behaviour that can arise from excessively free tier sets would be 'gang-up' effects. For example, given a collection of disjoint tiers that each contain a single element, we can describe processes where the output of some input element depends on the exact set of preceding elements regardless of their order. A similar behaviour can arise from overlapping but disjoint tiers. Given $\tau_1 = \{\text{s}, \text{ʃ}, \text{t}\}$ and $\tau_2 = \{\text{s}, \text{ʃ}, \text{n}\}$ we could describe a process of sibilant harmony that is blocked only when *both* 't' and 'n' intervene, regardless of their order. To the best of our knowledge, such processes are unattested and should be excluded.

In order to limit the "tier multiverse", we propose to reference the contribution of each $\sigma \in \Sigma$ separately, rather than referencing the entirety of $\mathtt{tails}_f$. Informally, the contribution of a relative to w is the portion of $f(wa)$ uniquely and directly attributable to a (e.g., it is what we would append to the output upon reading a in a transducer after having read w). A more formal definition is provided below.

Definition 11. Contribution
Given a function f, some $a \in \Sigma$, and some $w \in \Sigma^*$, $\mathtt{cont}_f(a, w) = \mathtt{lcp}(f(w\Sigma^*))^{-1} \cdot \mathtt{lcp}(f(wa\Sigma^*)) = f^p(w)^{-1} \cdot f^p(wa)$.

Using this notion of a contribution relative to w, which singles out a single element of $\mathtt{tails}_f(w)$, we place the following condition on MTSL$_k$ functions, which we call *target specification*.

Definition 12. Target specification
An IMTSL$_k$ function $f : \Sigma^* \to \Delta^*$ is target specified *if for each $\sigma \in \Sigma$ there is a finite set T_σ of tiers $\tau \subseteq \Sigma$ that form a strict superset-subset hierarchy,*

and the following holds for all $w_1, w_2 \in \Sigma^*$

$$[\bigwedge_{\tau \in T_\sigma} [\mathtt{suff}_\tau^{k-1}(w_1) = \mathtt{suff}_\tau^{k-1}(w_2)]] \Rightarrow [\mathtt{cont}_f(\sigma, w_1) = \mathtt{cont}_f(\sigma, w_2)]$$

An OMTSL$_k$ function $f : \Sigma^* \to \Delta^*$ *is* target specified *if for each* $\sigma \in \Sigma$ *there is a finite set* T_σ *of tiers* $\tau \subseteq \Delta$ *that form a strict superset-subset hierarchy, and the following holds for all* $w_1, w_2 \in \Sigma^*$*:*

$$[\bigwedge_{\tau \in T_\sigma} [\mathtt{suff}_\tau^{k-1}(f^p(w_1)) = \mathtt{suff}_\tau^{k-1}(f^p(w_2))]] \Rightarrow [\mathtt{cont}_f(\sigma, w_1) = \mathtt{cont}_f(\sigma, w_2)]$$

In words, when an MTSL function is target specified, each $\sigma \in \Sigma$ (i.e., each potential target of one or more processes) can be associated with a group of tiers that together determine the contribution of σ. Viewed from another perspective, each input element specifies the tiers that it wants to track, and the contribution of an input element can always be determined by tracking only its specified set of tiers; tracking other tiers provides information that is either redundant or irrelevant. Importantly, each member of the tier group is a proper subset of the next largest member.

Requiring an input segment's multiple associated tiers to fall into a strict superset-subset hierarchy has two apparent positive consequences. First, this restriction will be beneficial for learning, since it drastically reduces the number of possible tier combinations that can influence a given input element. Aksënova and Deskmukh (2018) show that even with a small inventory of 10 elements, there are 1022 ways to create a superset-subset pair, 511 ways to create a pair of fully disjoint sets, and 27990 ways to create a pair of partially overlapping sets. This last number is already 95% larger than the other two combined, and the difference only increases as the size of the alphabet grows. Second, research into multi-tiered stringsets suggests that we never see a single restriction enforced on two partially overlapping tiers (Aksënova and Deskmukh, 2018; McMullin et al., 2019).

6 Multiple Tiers in Action

This section now considers a range of individual patterns and pattern interactions that standard TSL functions are incapable of capturing. We show that each of the considered patterns and interactions are simply and intuitively captured by the target-specified MTSL functions defined in the previous Section. In order, we consider interactions between independent processes (6.1), interactions between conflicting processes (6.2), cases where a single target is subject to multiple harmonies (6.3), and cases where some segments act as last-resort harmony triggers in the absence of canonical triggers (6.4).

6.1 Independent processes

Recall from Section 4 that Tamashek Tuareg contains simultaneous long-distance sibilant harmony and long-distance labial dissimilation. Postponing discussion of the complication that arises from local nasal place assimilation, we can describe the two processes with a single target-specified OMTSL$_2$ function as follows.

First, we associate input /s/ with the tier $\tau_s = \{\text{s}, \text{s}^\curlywedge, \text{z}, \text{z}^\curlywedge, \text{ʃ}, \text{ʒ}\}$ containing all and only the sibilant consonants. Doing so, we need only specify that if $\mathtt{suff}^1_{\tau_s}(f^p(w)) \neq \lambda$ when reading from right to left, then $\mathtt{cont}_f(\text{s}, w) = \mathtt{suff}^1_{\tau_s}(f^p(w))$, else $\mathtt{cont}_f(\text{s}, w) =$[s]. In other words, /s/ harmonizes with the closest sibilant to its right, if there is one, else it surfaces faithfully.

Second, we associate input /m/ with the tier $\tau_m = \{\text{m}, \text{b}, \text{f}\}$ containing all and only the labial obstruents. Doing so, we need only specify that if $\mathtt{suff}^1_{\tau_m}(f^p(w)) \neq \lambda$ when reading from right to left, then $\mathtt{cont}_f(\text{m}, w) =$ [n], else $\mathtt{cont}_f(\text{m}, w) =$ [m]. In other words, /m/ dissimilates to [n] if there is a labial obstruent somewhere to its right, else it surfaces faithfully.

Each of the input elements we are considering here is associated to a single tier, and so the definition of target specification is satisfied. By virtue of tracking separate tiers, the two processes do not interfere with each other and so can be computed in tandem, exactly as desired. The simultaneous computation of the two rules is shown pictorially in Figure 1 for the word [ɑ-z$^\curlywedge$-ənː-ət-əlməz$^\curlywedge$] 'act of spitting up saliva'. The string in the center of the figure is the output string. Solid lines represent projection to an output tier and dashed lines represent an output tier element's influence on an input element.

6.2 Conflicting processes

The previous Section abstracted away from the interaction between long-distance labial dissimilation and local nasal place assimilation, which we

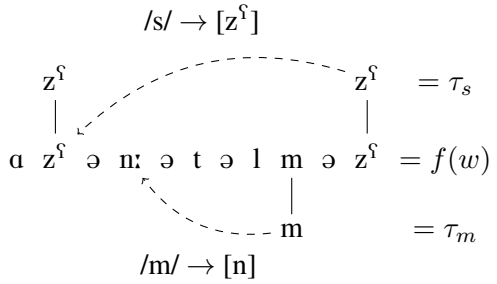

Figure 1: Simultaneous sibilant harmony and labial dissimilation in Tamashek Tuareg.

will now consider here. The interaction can be modelled with a target-specified OMTSL$_2$ functions as follows. First, we associate /m/ with a first tier $\tau_{m1} = \Delta$ containing all elements in the output alphabet. We furthermore associate /m/ with a second tier $\tau_{m2} = \{\text{m, b, f}\}$ containing all and only the labial obstruents. Assuming that the input is read from right to left, the full behaviour of /m/ can be described as follows. If $\texttt{suff}^1_{\tau_{m1}}(f^p(w))$ is a labial obstruent, then $\texttt{cont}_f(\text{m}, w)$ = [m] since /m/ is required to assimilate in place with the adjacent obstruent. If $\texttt{suff}^1_{\tau_{m1}}(f^p(w)) \neq$ [b] and $\texttt{suff}^1_{\tau_{m2}}(f^p(w)) \neq \lambda$, then $\texttt{cont}_f(\text{m}, w)$ = [n] since /m/ is not immediately adjacent to a labial consonant, but nonetheless is preceded non-locally by a labial and must dissimilate. In all other cases, /m/ surfaces faithfully as [m]. Note especially how $\tau_{m2} \subset \tau_{m1}$, and so the function meets the definition of target specification.

This is not to say that local processes always override non-local processes. Indeed, the opposite is witnessed in Samala. The Samala language is well-known for its process of regressive sibilant harmony, whereby the anteriority of the rightmost sibilant overrides the anteriority of all sibilants to the left. As it turns out, the language contains an additional rule affecting the sibilant /s/. Namely, /s/ regressively palatalizes to [ʃ] when immediately followed by [t], [n], or [l] (Applegate, 1972; Poser, 1982, 1993; McCarthy, 2007; Hansson, 2010; McMullin, 2016). The long-distance process is given priority here, such that the local sequences [sn], [st] and [sl] are permitted precisely when palatalization would create a disharmonic sequence of non-local sibilants.[1] Data exemplifying the two processes and their interaction are provided in (5) through (7) taken from Applegate (1972, p. 117-120).

(5) Unbounded sibilant harmony
/s-xalam-ʃ/ → [ʃ-xalamʃ] ‘it is wrapped’

(6) Local palatalization
/s-niʔ/ → [ʃ-niʔ] ‘his neck’

(7) Harmony overrides palatalization
/s-net-us/ → [s-net-us] ‘he does it to him’

We can also describe this with a target-specified OMTSL$_2$ function. In this case, the contribution of /s/ is dependent on two tiers: a tier $\tau_{s1} = \Delta$ relative to which local palatalization is considered, and a tier $\tau_{s2} = \{\text{s, ʃ}\}$ relative to which sibilant harmony is considered. Assuming that the input is read from right to left, the behaviour of /s/ can be described as follows. If $\texttt{suff}^1_{\tau_{s2}}(f^p(w)) = \lambda$ and $\texttt{suff}^1_{\tau_{s1}}(f^p(w))$ is one of [t], [n], or [l], then $\texttt{cont}_f(\text{s}, w)$ = [ʃ] since it is adjacent to a palatalization trigger but not preceded non-locally by a [-anterior] sibilant. If $\texttt{suff}^1_{\tau_{s2}}(f^p(w)) =$ [ʃ], then $\texttt{cont}_f(\text{s}, w)$ = [ʃ] since /s/ is preceded non-locally by a [-anterior] sibilant and must harmonize. In all other contexts, /s/ surfaces faithfully as [s]. Once again, $\tau_{s2} \subset \tau_{s1}$ and so the function meets the definition of target specification.

6.3 Single target, multiple harmonies

The target-specified MTSL functions are not limited to describing interactions between a local and non-local process, they can equally describe cases where a single element is subject to two long-distance dependencies operating over different featural dimensions. An example of this comes from then Imdlawn dialect of Tashlhiyt. The language’s causative prefix /s-/ agrees in both anteriority and voicing with a following sibilant, as shown in (8), although the voicing dimension of the harmony is blocked by an intervening voiceless obstruent (Elmedlaoui, 1995; Hansson, 2010; McMullin, 2016), as shown in (9).

(8) Anteriority and voicing harmony
/s-gruʒːm/ → [ʒ-gruʒːm]
‘CAUS-be.extinguished’

(9) Voicing harmony blocked
/s-mħaraʒ/ → [ʃ-mħaraʒ]
‘CAUS-get.angry.with.each.other’

We can describe this as a target-specified OMTSL$_2$ function wherein the contribution of /s/

[1] Some accounts of the data claim that the local process takes priority. See Heinz and Idsardi (2010) for a discussion and resolution of this inconsistency.

depends on one tier τ_{s1} for its voicing and another tier τ_{s2} for its anteriority. The voicing tier τ_{s1} contains all sibilants and all voiceless obstruents, while the anteriority tier τ_{s2} contains only the sibilants. Assuming that the input is read from right to left, the behaviour of /s/ can be described as follows. If $\texttt{suff}^1_{\tau_{s1}}(f^p(w)) = \texttt{suff}^1_{\tau_{s2}}(f^p(w))$, then an input /s/ harmonizes in both anteriority and voicing with that sibilant. Note that both tiers contain all sibilants and so the two tier suffixes can never be different sibilants. If $\texttt{suff}^1_{\tau_{s2}}(f^p(w))$ is a sibilant and $\texttt{suff}^1_{\tau_{s1}}(f^p(w))$ is a voiceless obstruent, then /s/ harmonizes in anteriority but not voicing with $\texttt{suff}^1_{\tau_{s2}}(f^p(w))$. The voicing dimension of the harmony is blocked because $\tau_{s2} \subset \tau_{s1}$, and so the voiceless obstruent necessarily intervenes between the triggering sibilant and the target sibilant. In all other cases, /s/ surfaces faithfully as [s].

In this Imdlawn Tashlhiyt case, the harmony target agrees with the harmony source along two dimensions, but there are cases where a single target can agree with multiple sources in specific configurations. Suffixal high vowels in Turkish typically agree in backness and rounding with the next vowel to the left, as shown in (10) from Nevins (2010, p. 27). An interesting exception, however, concerns the consonants /k/, /g/, and /l/ which are contrastively [+back] and have the [-back] counterparts /kʲ/, /gʲ/, and /lʲ/ (Clements and Sezer, 1982). If one of these consonants intervenes between the suffixal high vowel and the next vowel to the left, the suffixal high vowel takes its backness value from the consonant and its rounding feature from the vowel, as shown in (11) from Nevins (2010, p. 54).

(10) Two harmonies, same source
son-un ‘end-GEN
sap-ɯn ‘stalk-GEN’

(11) Two harmonies, two sources
usulʲ-y ‘system-Acc.SG’
sualʲ-i ‘question-ACC.SG’

Let us assume that the harmonizing suffixal high vowels are underlyingly /U/, which lacks values for [back] and [round]. The potentially split harmony can be analyzed with the following target-specified OMTSL$_2$ function. We have a backness tier τ_{U1} containing all the vowels and the abovementioned consonants. We also also have a rounding tier τ_{U2} containing all and only the vowels. Assuming that the input is read from left to right, the behaviour of /U/ can be described as follows. If $\texttt{suff}^1_{\tau_{U1}}(f^p(w)) = \texttt{suff}^1_{\tau_{U2}}(f^p(w))$, then the nearest source of backness and the nearest source of rounding are both the nearest leftward vowel. If, however, $\texttt{suff}^1_{\tau_{U1}}(f^p(w)) \neq \texttt{suff}^1_{\tau_{U2}}(f^p(w))$, then one of the relevant consonants must intervene between the suffixal vowel and the nearest leftward vowel. In this case, the suffixal vowel will take on the backness value of the consonant; the source of rounding will, however, always be the nearest leftward vowel.

6.4 Last-resort triggers

The final type of behaviour we will consider comes from a case of harmony that is triggered by a special class of segments in the absence of a more “preferred” trigger. Uyghur vowels are subject to a progressive backness harmony for which the non-low front vowels [i] and [e] are transparent (Lindblad, 1990; Vaux, 2000; Mayer and Major, 2018). The language’s locative suffix shows that the harmony is an active process; its vowel alternates between front [æ] and back [a] as shown in (12) taken from Mayer and Major (2018).

(12) Uyghur vowel harmony
aʁinæ-dæ ‘friend-LOC’
qoichi-da ‘shepherd-LOC’

Interestingly, dorsal consonants can trigger harmony in the absence of a harmonic vowel, with velar consonants causing front allomorphs of suffixes and uvulars causing back allomorphs (Lindblad, 1990; Vaux, 2000; Mayer and Major, 2018), as shown in (13). The harmony “prefers” to be triggered by vowels, however, which is witnessed by the fact that a back vowel can trigger back allormorphs across a velar consonant (Lindblad, 1990; Vaux, 2000; Mayer and Major, 2018), as shown in (14). Dorsal consonants only decide the front/back status of a suffix when the base lacks non-transparent vowels altogether, in a sense acting as a “last-resort trigger” (Lindblad, 1990; Vaux, 2000; Mayer and Major, 2018).

(13) Harmony with a dorsal
gezit-tæ ‘newspaper-LOC’
qiʁiz-da ‘Kyrgyz-LOC’

(14) Harmony across a dorsal
rak-ta ‘exercise-LOC’

This pattern too can be generated by a target-specified OMTSL$_2$ function. The suffixal low

vowel /A/ is associated to one tier τ_{A1} containing all non-transparent vowels and dorsal consonants, and is also associated to a second tier τ_{A2} containing only the non-transparent vowels. Assuming that the input is read from left to right, the harmony can be described as follows. If $\texttt{suff}^1_{\tau_{A2}}(f^p(w)) \neq \lambda$, then there is a non-transparent vowel in the stem that triggers harmony. If, however, $\texttt{suff}^1_{\tau_{A2}}(f^p(w)) = \lambda$ and $\texttt{suff}^1_{\tau_{A1}}(f^p(w)) \neq \lambda$, then all stem vowels are transparent, but there is a dorsal consonant available to trigger harmony as a last resort.

7 Conclusion

This paper defined an extension of the Tier-based Strictly Local functions, allowing them to track multiple tiers so long as the inter-tier relationships obey certain restrictions. Rather than having the entirety of the functional tails depending on just one tier, we allow the contribution of each input element to separately depend on its own set of tiers, provided that the set falls into a strict superset-subset hierarchy.

Our strategy of dividing the work amongst several independent sets of tiers greatly resembles the search procedure of Nevins (2010), see also Andersson et al. (2020). In this search procedure, specific input elements initiate a search for the nearest item relevant to them in some specified direction, and their output fate depends on the identity of the first relevant element found. Our approach is in many ways the mirror of the search procedure: by tracking the tiers specified by σ, we preemptively remember the most recent element(s) that would be relevant to the output fate of σ, and we therefore always know what to do when σ is encountered. We demonstrated that a wide variety of otherwise challenging phonological processes receive simple and intuitive analyses from this perspective.

A reviewer asks about the closure of the TSL functions under composition, pointing out that the MTSL languages are the closure of the TSL languages under intersection. While we have not yet formally proven so, it is likely that the left-reading MTSL functions are indeed the closure of the left-reading TSL functions under composition (*mutatis mutandis* for right-reading functions). At the very least, all of the language patterns analyzed in Section 6 can be described as the composition of two same-direction TSL functions. Similar to how Aksënova and Deskmukh (2018) and McMullin et al. (2019) argue that the full class of MTSL languages is too powerful, though, we believe that our proposal of target specification is necessary since free composition of TSL functions can lead to processes with bizarre properties (see Section 5).

One noteworthy behaviour not covered by the proposed class is bidirectional application. Research into bidirectional patterns has established that they can be modelled by *weakly deterministic functions* (Heinz and Lai, 2013). These are those functions from Σ^* to Δ^* that can be modelled as a pair of subsequential functions that apply in sequence and meet the following criteria: (i) the two functions read in opposite directions, (ii) the first function is from Σ^* to Σ^*, and (iii) the first function is not permitted to increase the length of the string. Since the function class considered in this paper is a proper subclass of the subsequential functions, it would be interesting to see whether we can lower the complexity bound of these bidirectional processes. Namely, can we instead model them as a pair of MTSL functions running in opposite directions?

Finally, future work will consider the learnability of MTSL functions. Burness and McMullin (2019) showed that any OTSL function is efficiently learnable from positive data if the tier is known in advance. This result can likely be carried over to MTSL functions once they receive a suitable automata characterization. When the tier is not known in advance, however, Burness and McMullin (2019) show that only total OTSL_2 functions are learnable from positive data. The induction of multiple tiers, necessary for the learning of MTSL functions, will thus likely be a significant challenge. That being said, McMullin et al. (2019) developed a method for learning the Multi-Tiered Strictly 2-Local languages, and their methods may perhaps be fruitfully applied to the function case.

Acknowledgements

We thank reviewers of this paper for their helpful comments. This research was supported by the Social Sciences and Humanities Research Council of Canada.

References

Aksënova, A. and Deskmukh, S. (2018). Formal restrictions on multiple tiers. In *Proceedings of the*

Society for Computation in Linguistics (SCiL) 2018, pages 64–73.

Andersson, S., Dolatian, H., and Hao, Y. (2020). Computing vowel harmony: The generative capacity of search and copy. In Baek, H., Takahashi, C., and Hong-Lun Yeung, A., editors, *Proceedings of the 2019 Annual Meeting on Phonology*.

Applegate, R. B. (1972). *Ineseño Chumash Grammar*. Doctoral Dissertation, University of California, Berkeley.

Burness, P. and McMullin, K. (2019). Efficient learning of Output Tier-based Strictly 2-Local functions. In *Proceedings of the 16th Meeting on the Mathematics of Language*, pages 78–90. Association for Computational Linguistics.

Chandlee, J. (2014). *Strictly Local Phonological Processes*. Doctoral Dissertation, University of Delaware.

Chandlee, J., Eyraud, R., and Heinz, J. (2014). Learning strictly local subsequential functions. *Transactions of the Association for Computational Linguistics*, 2:491–503.

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

Chandlee, J. and Heinz, J. (2018). Strict Locality and Phonological Maps. *Linguistic Inquiry*, 49(1):23–59.

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

Clements, G. N. and Sezer, E. (1982). Vowel and consonant disharmony in Turkish. In van der Hulst, H. and Smith, N., editors, *The Structure of Phonological Representations (Part II)*, pages 213–255. Foris, Dordrecht.

Elmedlaoui, M. (1995). *Aspects de Représentations Phonolgiques Dans Certains Langues Chamito-Sémitiques [Aspects of Phonological Representations in Certain Chamito-Semitic Languages]*. Doctoral Dissertation, Université Mohammed V, Rabat, Morocco.

Hansson, G. Ó. (2010). *Consonant Harmony: Long-Distance Interaction in Phonology*. Number 145 in University of California Publications in Linguistics. University of California Press, Berkeley, CA.

Hao, Y. and Andersson, S. (2019). Action-Sensitive Phonological Dependencies. In *Proceedings of the 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology and Morphology*, pages 135–143, Florence, Italy. Association for Computational Linguistics.

Hao, Y. and Bowers, D. (2019). Action-Sensitive Phonological Dependencies. In *Proceedings of the 16th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology and Morphology*, pages 218–228, Florence, Italy. Association for Computational Linguistics.

Heath, J. (2005). *A Grammar of Tamashek (Tuareg of Mali)*. Mouton de Gruyter, Berlin.

Heinz, J. and Idsardi, W. J. (2010). Learning opaque generalizations: the case of Samala. Unpublished Manuscript.

Heinz, J. and Lai, R. (2013). Vowel harmony and subsequentiality. In Kornai, A. and Kuhlmann, M., editors, *Proceedings of the 13th Meeting on the Mathematics of Language (MoL13)*, pages 52–63, Sofia, Bulgaria. Association for Computational Linguistics.

Heinz, J., Rawal, C., and Tanner, H. G. (2011). Tier-based strictly local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 58–64, Portland, OR. Association for Computational Linguistics.

Jardine, A. and Heinz, J. (2016). Learning Tier-based Strictly 2-Local languages. *Transactions of the Association for Computational Linguistics*, 4:87–98.

Jardine, A. and McMullin, K. (2017). Efficient Learning of Tier-Based Strictly k-Local Languages. In *International Conference on Language and Automata Theory and Applications (LATA 2017)*, pages 64–76.

Lambert, D. and Rogers, J. (2020). Tier-Based Strictly Local Stringsets: Perspectives from Model and Automata Theory. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2020*, pages 330–337, New Orleans, Louisianna.

Lindblad, V. M. (1990). *Neutralization in Uyghur*. Master's Thesis, University of Washington.

Mayer, C. and Major, T. (2018). A challenge for tier-based strict locality from Uyghur backness harmony. In *Formal Grammar 2018*, number 10950 in Lecture Notes in Computer Science, pages 62–83. Springer, Berlin.

McCarthy, J. (2007). Consonant harmony via correspondence: Evidence from Chumash. In Bateman, L., O'Keefe, M., Reilly, E., and Werle, A., editors, *Papers in Optimality Theory III*, pages 223–237. University of Massachusetts Occasional Papers in Linguistics.

McMullin, K. (2016). *Tier-Based Locality in Long-Distance Phonotactics: Learnability and Typology*. Doctoral Dissertation, University of British Columbia, Vancouver, BC.

McMullin, K., Aksënova, A., and De Santo, A. (2019). Learning phonotactic restrictions on multiple tiers. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, volume 2, pages 377–378.

McMullin, K. and Hansson, G. Ó. (2016). Long-distance phonotactics as Tier-based Strictly 2-Local Languages. In Albright, A. and Fullwood, M. A., editors, *Proceedings of the 2014 Annual Meeting on Phonology*, Washington, DC. Linguistic Society of America.

McNaughton, R. and Papert, S. A. (1971). *Counter-Free Automata*. MIT Press, Cambridge, MA.

Nevins, A. (2010). *Locality in Vowel Harmony*. Number 55 in Linguistic Inquiry Monographs. MIT Press.

Oncina, J. and Garcia, P. (1991). Inductive learning of subsequential functions. Technical Report DSIC II-34, University Politecnia de Valencia.

Poser, W. (1982). Phonological representations and action-at-a-distance. In van der Hulst, H. and Smith, N., editors, *The Structure of Phonological Representations*, volume 2, pages 121–158. Foris, Dordrecht.

Poser, W. (1993). Are strict cycle effects derivable? In Hargus, S. and Kaisse, E., editors, *Studies in Lexical Phonology*, pages 315–321. Academic Press, New York.

Rogers, J., Heinz, J., Fero, M., Hurst, J., Lambert, D., and Wibel, S. (2013). Cognitive and sub-regular complexity. In *Formal Grammar*, number 8036 in Lecture Notes in Artificial Intelligence, pages 90–108. Springer.

Rogers, J. and Pullum, G. K. (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information*, 20:329–342.

Vaux, B. (2000). Disharmony and derived transparency in Uyghur vowel harmony. In *Proceedings of NELS 30*, pages 671–698.