

Deciding the Strictly Piecewise Testable Stringsets

Gil Bailey
Dept. of Computer Science
Earlham College
Richmond, IN
bailey.gil@gmail.com

Matt Edlefsen
Dept. of Computer Science
Earlham College
Richmond, IN
edlefma@earlham.edu

Molly Visscher
Dept. of Computer Science
Earlham College
Richmond, IN
visscmo@earlham.edu

David Wellcome
Dept. of Computer Science
Earlham College
Richmond, IN
dwellcome@gmail.com

Sean Wibel
Dept. of Computer Science
Earlham College
Richmond, IN
sawibel07@earlham.edu

ABSTRACT

In this paper we introduce the the class of Strictly Piecewise (SP) stringsets which restricts the class of Piecewise Testable (PT) stringsets in a way that is analogous to the way the class of Strictly Local (SL) stringsets restricts the class of Locally Testable (LT) stringsets. We show SP to be a decidable subclass of the class of Regular stringsets by providing an algorithm that determines, given a Regular stringset in the form of a DFA, whether that DFA licenses an SP stringset and, if it does, constructs an SP_k grammar for the minimal k for which that stringset is SP_k . The time complexity of the algorithm is exponential in the size of the DFA, but we show this to be optimal for algorithms that actually construct the grammar.

The SP class of stringsets is interesting on purely formal grounds. But, it is also useful as a class of patterns that is computationally and cognitively very simple, but that still has the power to capture many phonotactic patterns in natural languages.

1. INTRODUCTION

Phonology is the study of sound patterns that occur in natural languages. For example, in Sarcee (a Native American language) the sound 'sh' never follows the sound 's' [1]. What this means is that if there is an 's' in a word there will be no 'sh' from that point to the end of the word. Phonologists usually describe sound patterns as Regular stringsets (e.g., with DFAs or Regular Ex-

pressions). But for many patterns the Regular stringsets are stronger computationally (and thus cognitively) than is necessary. The Sarcee pattern can be specified as a *Piecewise Testable* (PT) stringset [6, 5]. The class of Piecewise Testable stringsets is the Boolean closure of the class of stringsets of the form

$$\Sigma^* \sigma_1 \Sigma^* \sigma_2 \Sigma^* \dots \Sigma^* \sigma_k \Sigma^*. \quad (1)$$

PT is, however, still stronger than is necessary to describe the stringsets that have rules like Sarcee's exclusion of 's..sh' [2]. In fact, we only need to *rule out* the subsequences that are not allowed.

$$\overline{\Sigma^* s \Sigma^* sh \Sigma^*}$$

(where the vinculum denotes complement with respect to Σ^*) is the set of strings that include no 's..sh'¹.

There is a well-known class of stringsets that is similar to PT except that it works on consecutive sequences (*factors*) rather than subsequences. The *Locally Testable* (LT) stringsets [4] are the Boolean closure of the class of stringsets of the form

$$\Sigma^* w \Sigma^* \quad (2)$$

LT is analogous to PT. If LT is limited to only ruling out forbidden factors the result is the (also well-known) class of *Strictly Local* (SL) stringsets [4]. SL stringsets are finite intersections of stringsets of the form

$$\overline{\Sigma^* w \Sigma^*} \quad (3)$$

Proceedings of the 2009 Midstates Conference on Undergraduate Research in Computer Science and Mathematics

¹ Σ is the set of phonemes (sounds) that occur in Sarcee and 'sh' is a single phoneme.

where w is the forbidden factor. In this paper we define an analogous class of stringsets: a subclass of PT that is restricted to forbidding subsequences. We call this class the *Strictly Piecewise* (SP) stringsets. A stringset is SP iff (if and only if) it is the finite intersection of the class of stringsets of the form

$$\overline{\Sigma^* \sigma_1 \Sigma^* \sigma_2 \Sigma^* \dots \Sigma^* \sigma_k \Sigma^*} \quad (4)$$

where $\sigma_1 \sigma_2 \dots \sigma_k$ is a forbidden subsequence.

The structure of the paper is as follows. Section 2 begins with definitions of some basic notation and introduces SP stringsets and the SP_k grammars. It then gives an abstract characterization of SP stringsets along with some useful corollaries. It finishes by establishing a proper hierarchy of SP_k classes of differing k . Section 3 then shows SP to be a decidable subclass of the class of Regular stringsets by giving an algorithm that determines if a given Regular stringset is SP, and if it is, finding the minimal k for which it is SP_k and constructing an SP_k grammar for the stringset.

2. STRICTLY PIECEWISE STRINGSETS

The fundamental relationship between strings that defines Piecewise stringsets is the relationship of one string being a *subsequence* of another. We denote this relationship with the symbol \sqsubseteq :

$$w \sqsubseteq v \stackrel{\text{def}}{\iff} w = \sigma_1 \dots \sigma_k \text{ and } v \in \Sigma^* \sigma_1 \dots \Sigma^* \sigma_k \Sigma^*$$

Following Sakarovitch and Simon [5] we will refer to the set of all strings that w is a subsequence of as the *Shuffle Ideal* of w .

DEFINITION 1 (SHUFFLE IDEAL).

$$SI(w) \stackrel{\text{def}}{=} \{v \in \Sigma^* \mid w \sqsubseteq v\}$$

So a stringset of Form 1 is $SI(\sigma_1 \dots \sigma_k)$ and a stringset of Form 4 is $\overline{SI(\sigma_1 \dots \sigma_k)}$. A stringset will be SP iff it is the intersection of a finite set of the complements of these Shuffle Ideals.

The class of Piecewise Testable stringsets can be subdivided into a proper hierarchy of subclasses distinguished by the maximum length of the subsequences used in defining a stringset: PT_k stringsets are those that can be specified using subsequences of length k . SP is the union of the corresponding hierarchy of SP_k subclasses.

Let $\Sigma^{\leq k} \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid |w| \leq k\}$.

DEFINITION 2. A stringset is SP_k iff it is the intersection of the complements of a finite set of shuffle ideals generated by strings of length k .

$$L \in SP_k \stackrel{\text{def}}{\iff} L = \bigcap_{w_i \in T} \overline{SI(w_i)}, T \subseteq \Sigma^{\leq k}, \text{ finite.}$$

A stringset is SP iff it is SP_k for some k , i.e.,

$$SP \stackrel{\text{def}}{=} \bigcup_{k>0} [SP_k].$$

EXAMPLE 1. Strings that do not violate Sarcee's 's'...'sh' rule are captured by $\overline{SI(ssh)}$.

We refer to the subsequences of a string which are of length k as the k -subsequences of that string and those that are of length no greater than k as the ($\leq k$)-subsequences. We denote these as P_k and $P_{\leq k}$ respectively:

DEFINITION 3 (k -SUBSEQUENCES).

$$P_k(w) \stackrel{\text{def}}{=} \{v \mid v \sqsubseteq w \wedge |v| = k\}$$

$$P_{\leq k}(w) \stackrel{\text{def}}{=} \{v \mid v \sqsubseteq w \wedge |v| \leq k\}$$

Let $P_k(L)$ and $P_{\leq k}(L)$, for $L \subseteq \Sigma^*$, be the natural extension of these to sets of strings. Note that if $L \subseteq \Sigma^*$ then $P_1(L) \subseteq \Sigma$.

In linguistic applications such as statistical models and learning algorithms it is more convenient to think in terms of what subsequences do occur rather than what subsequences do not occur. We can define SP stringsets in terms of what subsequences are permitted to occur with SP_k grammars. These grammars provide an initial set of sequences with length less than or equal to k and license the set of all the strings in which only those sequences occur as subsequences. They are analogous to the grammars that license SL stringsets, except that they specify subsequences rather than factors.

DEFINITION 4. (SP_k Grammar) A grammar is a pair $\mathcal{G} = \langle \Sigma, T \rangle$ where $T \subseteq \Sigma^{\leq k}$. The stringset

licensed by an SP_k grammar is

$$L(\mathcal{G}) = \{w \in \Sigma^* \mid P_{\leq k}(w) \subseteq P_{\leq k}(T)\}.$$

EXAMPLE 2. For example, let

$$\mathcal{G} = \langle \Sigma, T \rangle$$

where

$$\begin{aligned} \Sigma &= \{a, b\} \text{ and} \\ T &= \{aaa, aab, abb, aba, baa, bab, bba, \\ &\quad ba, bb, ab, aa, a, b\} \end{aligned}$$

then $L(\mathcal{G})$ is $\{w \in \{a, b\}^* \mid |w|_b \leq 2\}$ i.e., the set of all strings over that alphabet that have at most two 'b's.

These grammars capture exactly the same stringsets as Definition 2. To show this we show how to convert a grammar into a set of forbidden subsequences and how to convert a set of forbidden subsequences into a grammar.

THEOREM 1. A stringset L is $L(\mathcal{G})$ for an SP_k grammar \mathcal{G} iff it is SP_k .

PROOF. To show that every SP stringset is $L(\mathcal{G})$ for some SP_k grammar \mathcal{G} , let a stringset L be an arbitrary SP stringset. Then $L = \bigcap_{w_i \in T} [\overline{\text{SI}(w_i)}]$ for some finite $T \subseteq \Sigma^{\leq k}$, where T is the set of forbidden subsequences. Equivalently,

$$L = \{w \in \Sigma^* \mid P_{\leq k}(w) \cap P_{\leq k}(T) = \emptyset\}.$$

Let T_L be the set of subsequences that are allowed to occur in a string in L . That is to say $T_L = \Sigma^{\leq k} - P_{\leq k}(T)$. Because $\Sigma^{\leq k}$ is finite, T_L is finite.

Let $\mathcal{G}_L \stackrel{\text{def}}{=} \langle \Sigma, T_L \rangle$.

We claim that $L(\mathcal{G}_L) = L$. To see this, first suppose that $w \in L(\mathcal{G}_L)$. Then $P_{\leq k}(w) \subseteq P_{\leq k}(T_L)$. Consequently $P_{\leq k}(w) \cap \text{SI}(v) = \emptyset$ for all $v \in T$ and thus $w \in \bigcap_{v_i \in T} [\overline{\text{SI}(v_i)}]$. Hence, $L(\mathcal{G}_L) \subseteq L$.

For the other direction let $w \in L$. Then there is no $v \in T$ for which $v \sqsubseteq w$. Thus, $P_{\leq k}(w) \subseteq \overline{\text{SI}(v)}$ for all $v \in T$. Consequently, $P_{\leq k}(w) \cap P_{\leq k}(T) = \emptyset$ and $P_{\leq k}(w) \subseteq \Sigma^{\leq k} - T = T_L$. Thus $P_{\leq k}(w) \subseteq P_{\leq k}(T_L)$ and $w \in L(\mathcal{G}_L)$. Thus $L(\mathcal{G}_L) \subseteq L$.

A similar construction based on $T_{\mathcal{G}} \stackrel{\text{def}}{=} \Sigma^{\leq k} - P_{\leq k}(T)$ shows that every stringset L that is $L(\mathcal{G})$ for some SP_k grammar \mathcal{G} is, in fact SP_k . \square

To show that a stringset is SP_k it is only necessary to express it in terms of one of the previous characterizations. To show that it is not SP_k we require a more abstract characterization of the SP_k stringsets. These can be characterized, purely in terms of the strings in the set, by the following closure condition.

THEOREM 2. A stringset $L \subseteq \Sigma^*$ is SP iff there is some $k \in \mathbb{N}$ such that for all $w \in \Sigma^*$,

$$P_{\leq k}(w) \subseteq P_{\leq k}(L) \Rightarrow w \in L. \quad (5)$$

PROOF. First assume that $L \subseteq \Sigma^*$ is SP. Because L is SP there must be some minimum k for which L is SP_k and a set $T \subseteq \Sigma^{\leq k}$ for which

$$L = \{w \in \Sigma^* \mid P_{\leq k}(w) \subseteq P_{\leq k}(T)\}$$

Since $P_{\leq k}(L) = \bigcup_{w \in L} [P_{\leq k}(w)]$ and, for all $w \in L$, $P_{\leq k}(w) \subseteq P_{\leq k}(T)$, it follows that $P_{\leq k}(L) \subseteq P_{\leq k}(T)$. Thus,

$$P_{\leq k}(w) \subseteq P_{\leq k}(L) \Rightarrow P_{\leq k}(w) \subseteq P_{\leq k}(T) \Rightarrow w \in L.$$

Now assume that there is some $k \in \mathbb{N}$ such that

$$(\forall w \in \Sigma^*) [P_{\leq k}(w) \subseteq P_{\leq k}(L) \Rightarrow w \in L]$$

Define a SP_k grammar $\mathcal{G}_L \stackrel{\text{def}}{=} \langle P_1(L), P_{\leq k}(L) \rangle$. Then

$$w \in L(\mathcal{G}_L) \Leftrightarrow P_{\leq k}(w) \subseteq P_{\leq k}(L) \Leftrightarrow w \in L.$$

\square

This characterization implies four very interesting properties of the SP stringsets.

COROLLARY 1. If $L \in SP_k$ then:

1. $w\sigma v \in L \Rightarrow wv \in L$ (Subsequence Closure)
2. $wv \in L \Rightarrow w, v \in L$ (Prefix and Suffix Closure)
3. $P_1(L) \subseteq L$ (Unit Strings)
4. $L \neq \emptyset \Rightarrow \epsilon \in L$ (Empty String)

To see that SP stringsets are closed under subsequence (via the contrapositive) assume that $wv \notin L$. It must be the case, then, that $P_{\leq k}(wv) \not\subseteq$

```

SPk( $\mathcal{M}$ )
  Given  $\mathcal{M} = \langle Q, \Sigma, q_0, \delta, F \rangle$ 
  Return an SPk grammar for  $L(\mathcal{M})$  for the least  $k$  such that  $L(\mathcal{M}) \in \text{SP}_k$ 
  or “Not SP” if there is no such  $k$ .
1  Let  $k = 1$ 
2  while  $k \leq \text{card}(Q)$ 
    do
3     Let  $\mathcal{G}_k$  be the SPk grammar for  $L(\mathcal{M})$  (construction of Theorem 5).
4     Let  $\mathcal{M}_k$  be the trimmed minimal DFA for  $L(\mathcal{G}_k)$  (construction of Theorem 6).
5     if  $\mathcal{M}$  is isomorphic to  $\mathcal{M}_k$ 
6       then Return  $k$  and  $\mathcal{G}_{\mathcal{M},k}$ 
7      $k \leftarrow k + 1$ 
8  Return “Not SP”

```

Figure 1: Algorithm

$P_{\leq k}(T)$, i.e., $P_{\leq k}(wv) - P_{\leq k}(T) \neq \emptyset$. Then, since $P_{\leq k}(wv) \subseteq P_{\leq k}(w\sigma v)$, it must also be the case that $P_{\leq k}(w\sigma v) - P_{\leq k}(T) \neq \emptyset$. Hence, $w\sigma v \notin L$.

The remaining corollaries follow from closure under subsequence.

Now to show that a stringset is not SP_k it suffices to give a counter-example to Property 5. That is to say, a string w such that $P_{\leq k}(w) \subseteq P_{\leq k}(L)$ but $w \notin L$. For example, the set of strings in which the subsequence ‘sh’ *must* occur is not SP₂ because $P_{\leq 2}(s) \subseteq P_{\leq 2}(ssh) \subseteq P_{\leq 2}(L)$ but ‘s’ is not a string in the stringset. Hence, this stringset is not closed under Property 5 for $k = 2$ and cannot be SP₂.

To show that a stringset is not SP, i.e., not SP_k for any k , one shows how to construct a string w_k , parameterized by k , such that, for each k , $P_{\leq k}(w_k) \subseteq P_{\leq k}(L)$ but w_k is not a string in L .

With the abstract characterization defined in Theorem 2 we can now show that there is a proper hierarchy among the SP_k stringsets, i.e., that for any k there is a stringset that is not SP_k but is SP_(k+1).

THEOREM 3. [*Proper Hierarchy*]

$$(\forall k)[\text{SP}_k \subsetneq \text{SP}_{k+1}].$$

PROOF. To show that $\text{SP}_{\leq k} \subseteq \text{SP}_{\leq k+1}$, note that $P_{\leq k}(w) = P_{\leq k}(P_{\leq k+1}(w))$. It follows, then, that $P_{\leq k+1}(w) \subseteq P_{\leq k+1}(L) \Rightarrow P_{\leq k}(w) \subseteq P_{\leq k}(L)$ and if L is closed under the property of Theorem 2 for k it must, *a fortiori*, be so closed for $k + 1$.

To show that the inclusion proper we show how to construct a stringset that is not SP_{≤k} but is SP_{≤k+1}.

Let $\Sigma = \{a, b\}$ and define the stringset $L_{\leq kb}$ to be the set of strings in Σ^* in which no more than k ‘b’s occur:

$$L_{\leq kb} \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid |w|_b \leq k\}$$

By Theorem 2, this stringset is not SP_k because $P_{\leq k}(b^{k+1}) \subseteq P_{\leq k}(L_{\leq kb})$ but $b^{k+1} \notin L_{\leq kb}$.

This stringset is, on the other hand, SP_{≤k+1} because it is $\overline{\text{SI}(b^{k+1})}$. \square

3. SP AND THE REGULAR STRINGSETS

In introducing the class of SP stringsets we stated that they are finite intersections of stringsets of Form 4. This form is a regular expression, hence stringsets with that form are Regular. It then follows immediately, by closure of Regular under finite intersection, that SP is a subclass of Regular. We want to show that SP is a decidable subclass of Regular, that is to say that there is an algorithm which determines whether a given Regular stringset is SP or not. Our algorithm is constructive in that, if the stringset is SP, it returns the minimal k for which it is SP_k and the SP_k grammar for the stringset.

THEOREM 4. *There is an algorithm which, given any Regular stringset L , decides if L is SP and, if it is, determines the least k for which L is SP_k and returns an SP_k grammar for L .*

Without loss of generality we can assume that the Regular stringset is given as a trimmed, minimal Deterministic Finite-State Automaton (DFA).

To fix notation, let $\mathcal{M} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a DFA, where Q is the set of states, Σ is the input alphabet, q_0 is the start state, $\delta : Q \times \Sigma \rightarrow Q$ is the *transition function* and $F \subseteq Q$ is the set of accepting states.

Let $\hat{\delta}$ be the natural extension of δ to $Q \times \Sigma^* \rightarrow Q$. This is well defined because, in a DFA, there is exactly one path from a state q labeled w for each $w \in \Sigma^*$.

Note that $L(\mathcal{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$ [3].

A DFA is *minimal* iff its state set Q corresponds to the set of Nerode-equivalence classes of $L(\mathcal{M})$, i.e.,

$$\begin{aligned} & \{(w, v) \mid \hat{\delta}(q_0, w) = \hat{\delta}(q_0, v)\} = \\ & \{(w, v) \mid (\forall u \in \Sigma^*) [wu \in L(\mathcal{M}) \Leftrightarrow vu \in L(\mathcal{M})]\} \end{aligned}$$

A *sink state* is a state q_s such that for all $w \in \Sigma^*$, $\hat{\delta}(q_s, w) \notin F$. If \mathcal{M} is minimal there can be at most one sink state.

A DFA is *trimmed* if its sink states have been removed. $\hat{\delta}(q_0, w)$ is then partial.

$\hat{\delta}(q_0, w)$ is defined ($\hat{\delta}(q_0, w) \downarrow$) if there is a path from q_0 labeled w in the transition graph of the DFA.

$\hat{\delta}(q_0, w)$ is undefined ($\hat{\delta}(q_0, w) \uparrow$) if there is no path from q_0 labeled w in the transition graph. This will be the case when a path labeled w would lead to a (missing) sink state of a trimmed DFA.

Theorem 4 relies on being able to convert DFAs to SP_k grammars and SP_k grammars to DFAs. We start by showing how to convert a DFA to an SP_k grammar with the following theorem.

THEOREM 5. *There is an algorithm which, given a DFA that recognizes an SP_k stringset, constructs an SP_k grammar $\mathcal{G}_{\mathcal{M}}$ such that $L(\mathcal{G}_{\mathcal{M}}) = L(\mathcal{M})$.*

To build the grammar we need to find all of the $(\leq k)$ -subsequences of the strings of $L(\mathcal{M})$ i.e., $P_{\leq k}(L)$. These will be the set T of the SP_k grammar. Conveniently, because SP is closed under subsequence, a DFA recognizing an SP_k stringset will have a particularly simple form that makes it easy to find the $(\leq k)$ -subsequences.

LEMMA 1. *If \mathcal{M} is a trimmed, minimal DFA and $L(\mathcal{M})$ is SP_k , then every state is an accepting state*

$$L(\mathcal{M}) \in SP_k \Rightarrow L(\mathcal{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \downarrow\}$$

PROOF OF LEMMA 1. Since SP_k stringsets are closed under subsequence by Theorem 2, if there is some v for which $\hat{\delta}(\hat{\delta}(q_0, w), v) \in F$ then $\hat{\delta}(q_0, w) \in F$ as well. If, on the other hand there is no v for which $\hat{\delta}(\hat{\delta}(q_0, w), v) \in F$ then $\hat{\delta}(q_0, w)$ would be a sink state. But by our assumption that the DFA is trimmed, this implies $\hat{\delta}(q_0, w) \uparrow$. Thus if $\hat{\delta}(q_0, w) \downarrow$ there is some v for which $\hat{\delta}(\hat{\delta}(q_0, w), v) \in F$ and, hence, $\hat{\delta}(q_0, w) \in F$ as well. This establishes that $\hat{\delta}(q_0, w) \downarrow \Rightarrow \hat{\delta}(q_0, w) \in F$.

That $\hat{\delta}(q_0, w) \uparrow \Rightarrow \hat{\delta}(q_0, w) \notin F$ is a consequence of the definition of $L(\mathcal{M})$. Hence the lemma. \square

A consequence of Lemma 1 is that instead of having to find all of the $(\leq k)$ -subsequences of the strings in $L(\mathcal{M})$, we can find all of the $(\leq k)$ -subsequences of the strings labeling the paths in the DFA. Moreover if $L(\mathcal{M})$ is SP_k for a given k , we can limit our search to the paths of length k from the start state.

LEMMA 2. *Let $\mathcal{M} = \langle Q, \Sigma, q_0, \delta, F \rangle$ be a trimmed, minimal DFA for which $L(\mathcal{M}) \in SP_k$. Then*

$$P_{\leq k}(L(\mathcal{M})) = L(\mathcal{M}) \cap \Sigma^{\leq k}.$$

PROOF OF LEMMA 2. $L(\mathcal{M})$ is closed under subsequence by Corollary 1 to Theorem 2. Consequently, $P_{\leq k}(L(\mathcal{M})) \subseteq L(\mathcal{M})$. By definition of $P_{\leq k}$, $P_{\leq k}(L(\mathcal{M})) \subseteq \Sigma^{\leq k}$. Hence,

$$P_{\leq k}(L(\mathcal{M})) \subseteq L(\mathcal{M}) \cap \Sigma^{\leq k}.$$

Conversely, $L(\mathcal{M}) \cap \Sigma^{\leq k} \subseteq P_{\leq k}(L(\mathcal{M}))$ also by definition of $P_{\leq k}(L(\mathcal{M}))$.

Therefore $P_{\leq k}(L(\mathcal{M})) = L(\mathcal{M}) \cap \Sigma^{\leq k}$ \square

PROOF OF THEOREM 5. By Lemma 2,

$$P_{\leq k}(L(\mathcal{M})) = L(\mathcal{M}) \cap \Sigma^{\leq k}.$$

By Lemma 1,

$$L(\mathcal{M}) \cap \Sigma^{\leq k} = \{w \in \Sigma^{\leq k} \mid \hat{\delta}(q_0, w) \downarrow\}.$$

So one simply does a search of the transition graph of the DFA with the depth limited to k , recording the strings labeling the paths traversed. Those

strings form the set T of an SP_k grammar $\mathcal{G}_{\mathcal{M}} = \langle \Sigma, T \rangle$. If $L(\mathcal{M}) \in SP_k$ then $L(\mathcal{G}_{\mathcal{M}}) = L(\mathcal{M})$. In the worst case there is one path for each $w \in \Sigma^k$. Therefore, this algorithm terminates in time $\Theta(\text{card}(\Sigma)^k)$. \square

Now that we have the method for constructing SP_k grammars from DFAs we need to construct DFAs from SP_k grammars. We do this by finding the forbidden ($\leq k$)-sequences, the ($\leq k$)-sequences not in $P_{\leq k}(T)$, and building a DFA for the intersection of the complements of their Shuffle Ideals.

THEOREM 6. *There is an algorithm that, given any SP stringset L , constructs a DFA that recognizes L .*

Since Regular stringsets are effectively closed under finite intersection, by the definition of SP_k it suffices show how to construct a trimmed, minimal DFA recognizing the set of strings in which a forbidden sequence w does not occur, in other words $\overline{SI(w)}$.

LEMMA 3. *Suppose $w \in \Sigma^k$, $w = \sigma_1 \dots \sigma_k$. Let $\mathcal{M}_{\overline{SI(w)}} = \langle Q, \Sigma, q_0, \delta, F \rangle$, Where $Q = \{i \mid 1 \leq i \leq k\}$, $q_0 = 1$, $F = Q$ and for all $i \in Q$, $\sigma \in \Sigma$:*

$$\delta(i, \sigma) = \begin{cases} i + 1 & \text{if } \sigma = \sigma_i \text{ and } i < k \\ \uparrow & \text{if } \sigma = \sigma_i \text{ and } i = k \\ i & \text{otherwise} \end{cases}$$

Then $\mathcal{M}_{\overline{SI(w)}}$ is a minimal, trimmed DFA that recognizes the complement of $SI(w)$, i.e., $\overline{SI(w)} = L(\mathcal{M}_{\overline{SI(w)}})$.

PROOF OF LEMMA 3. By construction \mathcal{M} is both trimmed and minimal and $L(\mathcal{M}) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \downarrow\}$. But $\hat{\delta}(q_0, v) \uparrow$ if and only if $w \sqsubseteq v$. Hence $L(\mathcal{M}_{\overline{SI(w)}}) = \overline{SI(w)}$. \square

Finally, the constructions of Theorem 5 and Theorem 6 provide the foundation of the algorithm for deciding the question of whether a given Regular stringset is SP. We can assume without loss of generality that the Regular stringset is in the form of a trimmed, minimal DFA, \mathcal{M} . We take \mathcal{M} , and for successively larger k , use the process of Theorem 5 to construct $\mathcal{G}_{\mathcal{M},k} = \langle \Sigma, T_{\mathcal{M},k} \rangle$. $L(\mathcal{G}_{\mathcal{M},k}) = L(\mathcal{M})$ iff $L(\mathcal{M})$ is SP_k .

From Theorem 1 we determine the forbidden sequences, $\Sigma^{\leq k} - T_{\mathcal{M},k}$. Then use the process of Theorem 6 to build a DFA, \mathcal{M}_k , such that $L(\mathcal{M}_k) = L(\mathcal{G}_{\mathcal{M},k})$. Then $L(\mathcal{M}_k) = L(\mathcal{M})$ iff $L(\mathcal{M})$ is SP_k .

Since both \mathcal{M}_k and \mathcal{M} are trimmed, minimal DFAs, $L(\mathcal{M}_k) = L(\mathcal{M})$ iff \mathcal{M}_k is isomorphic to \mathcal{M} . If they are we return k and $\mathcal{G}_{\mathcal{M},k}$. Otherwise we increase k and repeat. If k eventually exceeds $\text{card}(Q)$ we declare that $L(\mathcal{M})$ is not SP for any k .

PROOF (PARTIAL CORRECTNESS). If the algorithm returns $\mathcal{G}_{\mathcal{M},k}$ then \mathcal{M}_k is isomorphic to \mathcal{M} . This is only true iff $L(\mathcal{M}_k) = L(\mathcal{M})$, which is true iff $L(\mathcal{M})$ is SP_k . Therefore the algorithm returns $\mathcal{G}_{\mathcal{M},k}$ iff $L(\mathcal{M})$ is SP_k and since $L(\mathcal{G}_{\mathcal{M},k}) = L(\mathcal{M}_k) = L(\mathcal{M})$, $\mathcal{G}_{\mathcal{M},k}$ is an SP_k grammar for the stringset.

Since the algorithm did not return $\mathcal{G}_{\mathcal{M},k-1}$, There is no SP_{k-1} grammar for $L(\mathcal{M})$ and, hence, k is the minimal value for which the stringset is SP_k . This gives Partial Correctness of the Algorithm. \square

Termination depends on the the ability to limit k based only on the structure of the DFA. By the following lemma, k can be limited to $\text{card}(Q)$.

LEMMA 4. *Suppose $L \in SP_k - SP_{k-1}$. Then every DFA that recognizes L has at least k states.*

PROOF LEMMA 4. Let $L \in SP_k - SP_{k-1}$. Since $L \in SP_k$ but is not in SP_{k-1} by Theorem 3 there must be some string $w \in \Sigma^k$ for which $SI(w) \cap L = \emptyset$ but, for all strict subsequences v of w , $SI(v) \cap L \neq \emptyset$. Take v_1, v_2 to be distinct proper prefixes of w such that $|v_1| < |v_2|$. Then there is some u such that $v_2u = w$ while v_1u is a proper subsequence of w . Thus $v_2u \notin L$ but $v_1u \in L$.

Therefore none of the proper prefixes of w are Nerode equivalent (with respect to L) to each other or to w and any DFA that recognizes L will need to distinguish at least $k+1$ classes of strings. Consequently every trimmed DFA that recognizes L will need at least k states. \square

THEOREM 7. *Suppose $L \in SP$. Let $\text{card}(Q)$ be the size of the state set of a trimmed minimal DFA recognizing L . Then the worst case size of the grammar for L is $\Theta(\text{card}(\Sigma)^{\text{card}(Q)})$.*

PROOF. Since $T \subseteq \Sigma^k$, no SP_k grammar is larger than $\text{card}(\Sigma)^k$. By Lemma 4, no DFA for an SP_k

(but not SP_{k-1}) stringset has fewer than k states. Hence no grammar for an SP stringset can be larger than $\Theta(\text{card}(\Sigma)\text{card}(Q))$, where Q is the state set of a trimmed minimal DFA recognizing the language.

To see that grammars of this size do exist, note that $\overline{SI(w)} = L(\mathcal{G})$ for $\mathcal{G} = \langle \Sigma, T_w \rangle$ where $T_w = \Sigma^k - \{w\}$. Then $\text{card}(T_w) = \text{card}(\Sigma^k) - 1$. By the construction of Lemma 3 the size of the state set of a minimal DFA for $\overline{SI(w)}$ is $\text{card}(Q) = k$. Therefore $\text{card}(T_w) = \Theta(\text{card}(\Sigma)\text{card}(Q))$. \square

Consequently, although the algorithm is exponential in the size of the DFA, it is optimal among algorithms that actually construct SP_k grammars for $L(\mathcal{M})$.

4. CONCLUSION

We have introduced the the class of Strictly Piecewise (SP) stringsets which restricts the class of Piecewise Testable (PT) stringsets in a way that is analogous to the way the class of Strictly Local (SL) stringsets restrict the class of Locally Testable (LT) stringsets. We have shown SP to be a subclass of the class of Regular stringsets by providing an algorithm that determines, given a Regular stringset in the form of a DFA, whether that DFA licenses an SP stringset and, if it does, constructs an SP_k grammar for the minimal k for which that stringset is SP_k . The time complexity of the algorithm is exponential in the size of the DFA, but we have shown this to be optimal for algorithms that actually construct the grammar.

The SP class of stringsets is interesting on purely formal grounds. But, it is also useful as a class of patterns that is computationally and cognitively very simple, but that still has the power to capture many phonotactic patterns in natural languages.

5. REFERENCES

- [1] Eung-Do Cook. *A Sarcee Grammar*. University of British Columbia Press, 1984.
- [2] Jeffrey Heinz. Learning long distance phonotactics. Submitted manuscript, 2008.
- [3] John Hopcroft, Rajeev Motwani, and Jeffrey Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2001.
- [4] R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.
- [5] Jacques Sakarovitch and Imre Simon. Subwords. In M. Lothaire, editor, *Combinatorics on Words*, volume 17 of *Encyclopedia of Mathematics and Its Applications*, chapter 6, pages 105–134. Addison-Wesley, Reading, Massachusetts, 1983.
- [6] Imre Simon. Piecewise testable events. In *Automata Theory and Formal Languages: 2nd Grammatical Inference conference*, pages 214–222, Berlin ; New York, 1975. Springer-Verlag.