

On Languages Piecewise Testable in the Strict Sense

James Rogers* Jeffrey Heinz† Gil Bailey*
Matt Edlefsen* Molly Visscher* David Wellcome*
Sean Wibel*

jrogers@cs.earlham.edu

Abstract

In this paper we explore the class of Strictly Piecewise languages, originally introduced to characterize long-distance phonotactic patterns by Heinz [Hei07] as the Precedence Languages. We provide an abstract characterization, discuss their basic properties, locate them relative to other well-known subregular classes and provide algorithms for translating between the grammars defined here and finite state automata as well as an algorithm for deciding whether a regular language is Strictly Piecewise.

1 Introduction

Although many long-distance dependencies in natural language require expressive formalisms that are at least context-free [Cho56, Jos85, Shi85, Kob06], some non-local patterns in natural language do not. An example from Heinz [Hei07, Hei08] comes from the sibilant harmony process of Sarcee, where [-anterior] sibilants like [ʃ] and [ʒ] regressively require [+anterior] sibilants like [s] and [z] to assimilate in anteriority, but not vice versa [Coo78, Coo84].¹ As a consequence of this phonological rule, there are no words in Sarcee where [-anterior] sibilants may follow [+anterior] sibilants as in (1b), though the reverse is possible (1a) (data from Cook [Coo78]).

1. a. /si-tʃiz-aʔ/ → ʃítʃídʒàʔ ‘my duck’ b. cf. *sítʃídʒàʔ

Heinz [Hei07, Hei08] observes that these kinds of long-distance dependencies can be described according to the well-formedness of subsequences: in Sarcee discontinuous subsequences like [ʒ] and [ʃs] are well-formed but discontinuous subsequences like [zʃ] and [sʃ] are not (since the phonological rule requires the [s] to become [ʃ] when followed by [ʃ]).

The Piecewise Testable (PT) languages [Sim75] are a subclass of the regular languages that can describe this kind of non-local pattern. These languages are

*Earlham College

†University of Delaware. The author acknowledges the support of a 2008-2009 University of Delaware Research Fund Grant.

¹Linguistic descriptions of Sarcee (and many other languages with consonantal harmony [Han01, RW04]) are clear that agreeing consonants can be arbitrarily distant.

similar, in many respects, to the Locally Testable (LT) languages [MP71, RP07] except that the two classes differ in how they determine an expression’s well-formedness: for LT languages, an expression’s well-formedness depends entirely on the set of *contiguous* subsequences (up to some length k , known as k -factors) in the expression, whereas for PT languages, an expression’s well-formedness depends entirely on the set of subsequences (not necessarily contiguous and up to some length k) found within the expression.

In fact, Sarcee-like non-local patterns are describable by a proper subclass of the PT languages, which we call Strictly Piecewise (SP). This name reflects the fact that the relationship between the SP languages and the PT languages is precisely analogous to the relationship between the Strictly Local (SL) languages and the LT languages [MP71, RP07]. The SP class completes a *dual hierarchy* of subregular language classes with the Local branch being characterized by immediate adjacency (successor) and the Piecewise branch by precedence (less-than).

The structure of the paper is as follows. Section 2 defines basic notation. Section 3 reviews the Piecewise Testable languages. Most of the results in this section are well-known (see [Sim75], [SS83], [Lot05], [KCM06], [Tra01]); the rest are probably best attributed to folklore. The primary contributions of this paper are in Sections 4 and 5. Section 4 defines the Strictly Piecewise Testable languages, explores some of their properties and provides an abstract characterization of the class. Section 5 presents algorithms for extracting a SP_k grammar from a minimal Deterministic Finite-State Automaton (DFA) recognizing a SP_k language and for constructing a minimal DFA recognizing an SP_k language from its grammar. Together these provide an algorithm for deciding if an arbitrary regular language is SP and for determining the least k for which it is SP_k if it is. In section 6 we consider the parallels between the Piecewise Testable and Locally Testable hierarchies from a descriptive perspective.

2 Preliminaries

We start with some mostly standard notation. $\mathcal{P}(S)$ denotes the power set of the set S ; $S_1 - S_2$ set-theoretic difference. Σ denotes a finite set of symbols and a string over Σ is a finite sequence of symbols drawn from that set. Σ^k , $\Sigma^{\leq k}$, Σ^* denote all strings over this alphabet of length k , of length less than or equal to k , and of any finite length, respectively. ϵ denotes the empty string. $|w|$ denotes the length of string w and $|w|_\sigma$ denotes the number of occurrences of $\sigma \in \Sigma$ in w . A language L is a subset of Σ^* . Concatenation of sets of strings is denoted $L_1 \cdot L_2 = \{uv \mid u \in L_1 \text{ and } v \in L_2\}$. When discussing partial functions, we use the notation \uparrow and \downarrow to indicate that the function is undefined, respectively is defined, for some particular arguments.

A *Deterministic Finite-state Automaton* (DFA) is a tuple $\mathcal{M} = \langle Q, \Sigma, q_0, \delta, F \rangle$ where Q is the state set, Σ is the alphabet, q_0 is the start state, δ is a deterministic transition function and F is the set of accepting states. Let $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ be the (partial) path function of \mathcal{M} , i.e., $\hat{\delta}(q, w)$ is the (unique) state reachable from state q via the sequence w , if any, or $\hat{\delta}(q, w)\uparrow$ otherwise. The language recognized by a DFA \mathcal{M} is $L(\mathcal{M}) \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid \hat{\delta}(q_0, w)\downarrow \in F\}$.

Two strings w and v over Σ are *distinguished* by a DFA \mathcal{M} iff $\hat{\delta}(q_0, w) \neq \hat{\delta}(q_0, v)$. They are *Nerode equivalent* with respect to a language L if and only if $wu \in L \iff vu \in L$ for all $u \in \Sigma^*$. All DFAs which recognize L must distinguish strings which are inequivalent in this sense, but no DFA recognizing L necessarily distinguishes any strings which are equivalent. Hence the number of equivalence classes of strings over Σ modulo Nerode equivalence with respect to L gives a (tight) lower bound on the number of states required to recognize L . A DFA is *minimal* if the size of its state set is minimal among DFAs accepting the same language. A minimal DFA is *trimmed* if the (unique) sink state has been removed. The reader is referred to [HMU01] for details.

The relation between strings which is fundamental to Piecewise Testability is the *subsequence* relation, which is a partial order on Σ^* :

$$w \sqsubseteq v \stackrel{\text{def}}{\iff} w = \varepsilon \text{ or } w = \sigma_1 \cdots \sigma_n \text{ and } (\exists w_0, \dots, w_n \in \Sigma^*) [v = w_0 \sigma_1 w_1 \cdots \sigma_n w_n].$$

in which case we say w is a *subsequence* of v . \sqsubseteq is compatible with concatenation: $w_1 \sqsubseteq v_1$ and $w_2 \sqsubseteq v_2$ implies that $w_1 w_2 \sqsubseteq v_1 v_2$.

For $w \in \Sigma^*$, let

$$P_k(w) \stackrel{\text{def}}{=} \{v \in \Sigma^k \mid v \sqsubseteq w\} \text{ and } P_{\leq k}(w) \stackrel{\text{def}}{=} \{v \in \Sigma^{\leq k} \mid v \sqsubseteq w\}.$$

Let $P_k(L)$ and $P_{\leq k}(L)$ be the natural extensions of these to sets of strings. Note that $P_0(w) = \{\varepsilon\}$, for all $w \in \Sigma^*$, that $P_1(w)$ is the set of symbols occurring in w and that $P_{\leq k}(L)$ is finite, for all $L \subseteq \Sigma^*$.

3 Piecewise Testable Languages

The class of Piecewise Testable languages (PT) was introduced by Simon [Sim75] as the Boolean closure of the class of languages of the form $\Sigma^* \sigma_1 \Sigma^* \cdots \Sigma^* \sigma_n \Sigma^*$ where $\sigma_1 \cdots \sigma_n$ is a possibly empty word in Σ^* .

Following Sakarovitch and Simon [SS83], Lothaire [Lot05] and Kontorovich, et al., [KCM08] we call the set of strings that contain w as a subsequence the *shuffle ideal*² of w :

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

Then the class of Piecewise Testable (PT) languages is the smallest class of languages including $\text{SI}(w)$ for all $w \in \Sigma^*$ and closed under Boolean operations. Similarly, the class of k -Piecewise Testable (PT_k) languages is the smallest class of languages including $\text{SI}(w)$ for all $w \in \Sigma^{\leq k}$ and closed under Boolean operations.

From a model-theoretic perspective, PT_k is the class of languages definable by propositional formulae in which the atomic formulae are strings in $\Sigma^{\leq k}$ with a string $w \in \Sigma^*$ satisfying a formula $p \in \Sigma^{\leq k}$ iff $w \in \text{SI}(p)$. PT is the class of languages definable by arbitrary finite formulae of this type.

The class of Piecewise Testable languages has a well-known characterization (sometimes taken to be its definition):

Theorem 1 $L \subseteq \Sigma^*$ is in the class PT iff there exists some k such that, for all $w, v \in \Sigma^*$,

$$P_{\leq k}(w) = P_{\leq k}(v) \Rightarrow (w \in L \Leftrightarrow v \in L).$$

²Properly $\text{SI}(w)$ is the principal ideal generated by $\{w\}$ wrt the inverse of \sqsubseteq .

Since $P_{\leq k}(\Sigma^*)$ is finite for all k and Σ , one consequence of this characterization is that a language is in PT iff it is the union of a finite set of equivalence classes modulo the relation $w \equiv_k v \stackrel{\text{def}}{\iff} P_{\leq k}(w) = P_{\leq k}(v)$. Given this, we can take a PT_k language to be generated by a finite set of subsets of $\Sigma^{\leq k}$.

Definition 1 (PT_k Grammar) A PT_k grammar is a pair $\mathcal{G} = \langle \Sigma, T \rangle$ where $T \subseteq \mathcal{P}(\Sigma^{\leq k})$. The language licensed by a PT_k grammar is

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

Note that $L(\mathcal{G}) = \emptyset$ iff $T = \emptyset$ and $\varepsilon \in L(\mathcal{G})$ iff $\{\varepsilon\} \in T$.

Theorem 2 The classes PT_k form a proper hierarchy in k : $(\forall k)[\text{PT}_k \subsetneq \text{PT}_{k+1}]$.

The inclusion follows from the fact that $P_{k+1}(w) = P_{k+1}(v) \Rightarrow P_k(w) = P_k(v)$ and, hence, \equiv_{k+1} is a refinement of \equiv_k . To see that it is proper, let

$$L_{\leq kb} \stackrel{\text{def}}{=} \{w \in \{a, b\}^* \mid |w|_b \leq k\}.$$

This is not in PT_k since $P_{\leq k}(b^k) = P_{\leq k}(b^{k+1})$. On the other hand, it is in PT_{k+1} since it is the complement of $\text{SI}(b^{k+1})$.

Theorem 3 The class of finite languages is a proper subset of the class of Piecewise Testable languages.

Any singleton set $\{w\}$ is $\text{PT}_{|w|+1}$, being the intersection of $\text{SI}(w)$ and all $\overline{\text{SI}(v)}$ for $v \in \Sigma^{|w|+1}$. Hence every finite set L is in PT_k for every k greater than the longest string in L . On the other hand, there is no k for which the class of finite languages is included in PT_k .

Theorem 4 PT and PT_k , for any $k > 0$, are not closed under concatenation.

The languages $L_{\leq kb} = L_{\leq (k-1)b} \cdot L_{\leq 1b}$ witness that PT_k is not closed under concatenation. For the general case consider the language $L_{awb} = \{a\} \cdot \Sigma^* \cdot \{b\}$. This is the concatenation of three PT_2 languages, but it is not, itself, PT. Suppose, by way of contradiction, that it was PT. Then it would be PT_k for some k . But then the string $(ab)^k \in L_{awb}$ while $(ab)^k a \notin L_{awb}$ despite the fact that $P_{\leq k}((ab)^k) = P_{\leq k}(\{a, b\}^*) = P_{\leq k}((ab)^k a)$, contradicting Theorem 1.

Theorem 5 (Simon 1975) The class of Piecewise Testable languages is a proper subset of the class of Star-Free languages: $\text{PT} \subsetneq \text{SF}$.

$\text{SI}(w)$, where $w = \sigma_1 \sigma_2 \cdots \sigma_{|w|}$, is denoted by the SF expression

$$\bar{\emptyset} \cdot \sigma_1 \cdot \bar{\emptyset} \cdot \dots \cdot \bar{\emptyset} \cdot \sigma_{|w|} \cdot \bar{\emptyset}.$$

and SF is closed under Boolean operations. That the inclusion is proper is witnessed by the fact that $L_{awb} \in \text{SF}$.

Theorem 6 The class of Star-Free languages is the closure of the class of Piecewise Testable languages under concatenation and Boolean operations.

SF is the closure of the class of Finite languages under union, concatenation and complement (hence concatenation and Boolean operations).

4 Strictly Piecewise Languages

Languages that are *Locally Testable in the Strict Sense* (Strictly Local, SL) are defined only in terms of the set of k -factors which are licensed to occur in the string (equivalently the complement of that set with respect to $\Sigma^{\leq k}$, the *forbidden factors*) [MP71]. In this section we introduce the class of languages obtained by the analogous restriction to PT, which we call *Piecewise Testable in the Strict Sense* (Strictly Piecewise, SP).

Definition 2 (SP_k Grammar) A SP_k grammar is a pair $\mathcal{G} = \langle \Sigma, T \rangle$ where $T \subseteq \Sigma^k$. The language licensed by a SP_k grammar is

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

A language is SP_k iff it is $L(\mathcal{G})$ for some SP_k grammar \mathcal{G} . It is SP iff it is SP_k for some k .

Lemma 1 Every SP language is the intersection of the complements of a finite set of principal shuffle ideals.

This follows immediately from the definition. A string w is licensed iff none of the sequences in $\Sigma^{\leq k} - P_{\leq k}(T)$ is a subsequence of w , which is to say if w is in none of the (finitely many) principal shuffle ideals generated by those forbidden sequences.

The SP languages are characterized by the following property.

Theorem 7 A language $L \subseteq \Sigma^*$ is SP iff, there is some $k \in \mathbb{N}$ such that, for all $w \in \Sigma^*$,

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

Suppose that $L \in \text{SP}$. Then $L \in \text{SP}_k$ for some k and there is some Σ and $T \subseteq \Sigma^{\leq k}$ for which $L = \{w \in \Sigma^* \mid P_{\leq k}(w) \subseteq P_{\leq k}(T)\}$. Then $P_{\leq k}(L) = \bigcup_{w \in L} [P_{\leq k}(w)] \subseteq P_{\leq k}(T)$. That L is closed under the property of Theorem 7 follows immediately.

Suppose, on the other hand, that L is closed under the property of Theorem 7. Let $\mathcal{G}_L = \langle P_1(L), P_{\leq k}(L) \rangle$. Then $w \in L(\mathcal{G}_L) \Rightarrow P_{\leq k}(w) \subseteq P_{\leq k}(L) \Rightarrow w \in L$ and $w \in L \Rightarrow P_{\leq k}(w) \subseteq P_{\leq k}(L) \Rightarrow w \in L(\mathcal{G}_L)$.

Corollaries:

If $L \in \text{SP}_k$ then:

1. $wv \in L \Rightarrow w, v \in L$ (Prefix and Suffix closure),
2. $w\sigma v \in L \Rightarrow wv \in L$ (Subsequence closure),
3. $P_1(L) \subseteq L$ (Unit strings) and
4. $L \neq \emptyset \Rightarrow \varepsilon \in L$ (Empty string).

Theorem 8 (Proper Hierarchy) $(\forall k)[\text{SP}_k \subsetneq \text{SP}_{k+1}]$.

Inclusion follows from the fact that $P_{\leq k}(w) = P_{\leq k}(P_{\leq k+1}(w))$. The same sequence of languages that witnesses separation of the PT_k classes witnesses separation of the SP_k classes.

Theorem 9 SP and SP_k , for any $k > 0$, are closed under intersection but not union or complement. SP_k is not closed under concatenation, although SP is.

Closure under intersection follows immediately from Lemma 1. Non-closure under union is witnessed by the language $L = \{b^i, ab^i \mid i \geq 0\} \cup \{a^i, ba^i \mid i \geq 0\}$. This is the union of two SL_2 sets ($L(\langle\{a, b\}, \{ab, bb\}\rangle)$ and $L(\langle\{a, b\}, \{ba, aa\}\rangle)$, respectively), but it is not SL_k for any k since $P_{\leq k}(ab^k a) \subseteq P_{\leq k}(L)$. Non-closure under complement follows by DeMorgan's theorem. For concatenation, note that $L_{\leq (k-1)b}$ and $L_{\leq 1b}$ are both SL_k but $L_{\leq (k-1)b} \cdot L_{\leq 1b} = L_{\leq kb} \notin SL_k$. On the other hand if $L_1 \in SP_k$ and $L_2 \in SP_l$ then $L_1 \cdot L_2 \in SP_{k+l}$.

Theorem 10 PT (respectively, PT_k) is the closure of SP (respectively, SP_k) under complement, equivalently, under Boolean operations.

That every SP_k language is PT_k follows from Theorems 1 and 7. That every PT_k language is a Boolean combination of SP_k languages follows from the fact that $SI(w)$, for any $w \in \Sigma^*$ is the complement of a SP_k language.

It's striking that, while PT is the closure of SP under complement, the latter is closed under concatenation while the former is not.

5 SP and the Regular Languages

Since $SP \subseteq PT \subseteq \text{Star-Free} \subseteq \text{Regular}$ every SP language is recognizable by a Deterministic Finite-State Automaton (DFA). Theorem 7 has a number of consequences for the structure of the trimmed, minimal DFAs which recognize SP languages. In particular, 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:

- All states of \mathcal{M} are accepting states: $F = Q$.
- For all $q_1, q_2 \in Q$ and $\sigma \in \Sigma$, if $\hat{\delta}(q_1, \sigma) \uparrow$ and $\hat{\delta}(q_1, w) = q_2$ for some $w \in \Sigma^*$ then $\hat{\delta}(q_2, \sigma) \uparrow$. (Missing edges propagate down.)
- All cycles are self-edges.
- For all $q_1, q_2 \in Q$ and $u, v, w \in \Sigma^*$, if $\hat{\delta}(q_0, w) = q_1$, $\hat{\delta}(q_1, v) = q_2$ and $q_1 \neq q_2$ then:
 - $(\exists u \in \Sigma^*)[\hat{\delta}(q_0, wu) \downarrow \text{ and } \hat{\delta}(q_0, wvu) \uparrow]$ and
 - $(\forall u \in \Sigma^*)[\hat{\delta}(q_0, wvu) \downarrow \Rightarrow \hat{\delta}(q_0, wu) \downarrow]$

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

This follows from closure under subsequence.

Lemma 2 provides an algorithm which, given a DFA that recognizes an SP_k language, constructs the SP_k grammar for that language. 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. The time complexity of this algorithm is $\Theta(\text{card}(\Sigma)^k)$.

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

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

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

Lemma 3 provides the foundation for an algorithm which, given an SP_k grammar $\langle \Sigma, T \rangle$ for a language L , constructs a minimal, trimmed DFA which recognizes L . One constructs the trimmed, minimal DFA for $\overline{\text{SI}(w)}$ for each $w \in \Sigma^{\leq k} - P_{\leq k}(T)$, and then constructs the trimmed, minimal DFA for their intersection. The complexity of this algorithm is $\Theta(\text{card}(\Sigma)^k)$ (since $\text{card}(T) = \Theta(\text{card}(\Sigma)^k)$).

Lemma 4 Suppose $L \in \text{SP}_k - \text{SP}_{k-1}$. Then every DFA that recognizes L has at least k states.

$L \in \text{SP}_k - \text{SP}_{k-1}$ implies that there is at least one $w \in \Sigma^k$ such that $\text{SI}(w) \cap L = \emptyset$ but for all proper subsequences v of w it is the case that $\text{SI}(v) \cap L \neq \emptyset$. In fact, since SP_k languages are closed under subsequence v itself must be in L .

Suppose v_1 and v_2 are distinct proper prefixes of w , $|v_1| < |v_2|$. Then there is some u such that $v_2u = w \notin L$. On the other hand, v_1u is a proper subsequence of w and thus, by choice of w , $v_1u \in L$. Hence none of the k proper prefixes of w are Nerode equivalent (with respect to L) to each other or to w and any DFA recognizing L will need to distinguish at least $k+1$ classes of strings, hence have at least k states.

Corollary 1 Suppose $L \in \text{SP}$. 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 $\Theta(\text{card}(\Sigma)^{\text{card}(Q)})$.

Theorem 11 There is an algorithm which, given any Regular language 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 .

Assume, wlog, that L is presented as a trimmed, minimal DFA. The algorithm constructs potential SP_k grammars for increasing k using Lemma 2 and constructs trimmed, minimal DFAs for each using Lemma 3. The first grammar constructed in this way that is isomorphic to the DFA for L will be an SP_k grammar for the least k for which L is SP_k . By Lemma 4, if no such DFA is found for $k \leq \text{card}(Q)$ then L is not SP_k for any k .

The time complexity of this algorithm is $\Theta(\text{card}(\Sigma)^{\text{card}(Q)})$, i.e., it is exponential time. By Corollary 1, however, this is optimal for algorithms that actually construct an SP_k grammar for L .

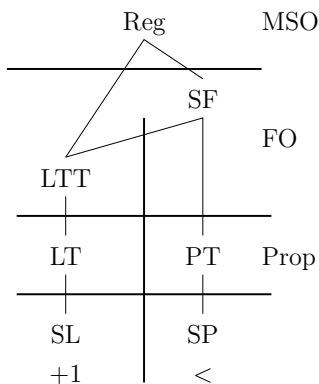


Figure 1: Parallel Sub-regular Hierarchies.

6 Dual Subregular Hierarchies

The hierarchy of Local classes of languages has a very attractive model-theoretic characterization. The class of Strictly Local languages is properly extended by the class of Locally Testable languages, which is the class of languages definable by propositional formulae in which the atomic formulae are blocks of symbols interpreted as factors of the string. This is properly extended by the class of Locally Threshold Testable languages, which is the class of languages definable by First-Order formulae with adjacency (successor) but not precedence (less-than). This is properly extended by the class of Regular languages, which is the class of languages definable by Monadic Second-Order formulae with either adjacency or precedence, equivalently with both (since they are MSO definable from each other).

As we have seen here, the Piecewise classes provide a parallel sequence of classes. The class of SP languages corresponds to the Strictly Local languages, except that they are defined in terms of subsequences rather than factors. This is extended by the class of PT languages, which is the class of languages definable over propositional formulae in which the atomic formulae are blocks of symbols interpreted as subsequences of the string. Hence PT corresponds to LT except that, again, it is defined in terms of subsequences rather than factors. The class of languages definable by First-Order formulae with precedence, corresponding to LTT on the adjacency side, is the class of Star-Free sets. Since adjacency is FO definable from precedence, LTT is actually a (proper) subclass of SF. Finally, the two branches become indistinguishable at the MSO level in the class of Regular languages.³

7 Conclusion

We have characterized the Strictly Piecewise languages and presented their basic properties. Additionally, it was shown that SP languages complete a subregular

³Interestingly, it can also be said that they join at the bottom of the hierarchy as well, since SP_1 and SL_1 both contain just \emptyset and Γ^* for each $\Gamma \subseteq \Sigma$.

hierarchy based on precedence in the same way the Local classes form a hierarchy based on adjacency. We have also provided algorithms for translating between the SP grammars defined here and finite state automata, as well as an algorithm for deciding if some regular language is SP.

The theoretical contributions above provide a better understanding of linguistic, cognitive, and natural language processing models. We have already mentioned the capability of the SP languages to describe certain kinds of phonotactic patterns [Hei07, Hei08]. The introductory Sarcee pattern, for example, is given by a SP_2 grammar which only prohibits subsequences consisting of a [+anterior] sibilant followed by a [-anterior] sibilant. Interestingly, SP languages also appear in models of reading comprehension [GW04, WC08] as well as in text classification [LCSTW02, CGGR03] (see also Shawe-Taylor [STC05, chap. 11]). We hope that this paper continues to spur interest in the utility and beauty of languages described piecewise.

References

- [CGGR03] Nicola Cancedda, Eric Gaussier, Cyril Goutte, and Jean-Michel Renders. Word-sequence kernels. *Journal of Machine Learning Research*, 3:1059–1082, 2003.
- [Cho56] Noam Chomsky. Three models for the description of language. *I.R.E. Transactions on Information Theory*, IT-2:113–123, September 1956. Reprinted in *Readings in Mathematical Psychology*, Volume II, ed. by R. Duncan Luce, Robert R. Bush, and Eugene Galanter, 105–124; New York: John Wiley & Sons, 1965.
- [Coo78] Eung-Do Cook. The synchronic and diachronic status of Sarcee g^y . *International Journal of American Linguistics*, 4:192–196, 1978.
- [Coo84] Eung-Do Cook. *A Sarcee Grammar*. University of British Columbia Press, 1984.
- [GW04] J. Grainger and C. Whitney. Does the huamn mnid raed wrods as a wlohe? *Trends in Cognitive Science*, 8:58–59, 2004.
- [Han01] Gunnar Hansson. *Theoretical and typological issues in consonant harmony*. PhD thesis, University of California, Berkeley, 2001.
- [Hei07] Jeffrey Heinz. *The Inductive Learning of Phonotactic Patterns*. PhD thesis, University of California, Los Angeles, 2007.
- [Hei08] Jeffrey Heinz. Learning long distance phonotactics. Submitted manuscript, 2008.
- [HMU01] John Hopcroft, Rajeev Motwani, and Jeffrey Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 2001.
- [Jos85] A. K. Joshi. Tree-adjoining grammars: How much context sensitivity is required to provide reasonable structural descriptions? In D. Dowty, L. Karttunen, and A. Zwicky, editors, *Natural Language Parsing*, pages 206–250. Cambridge University Press, 1985.

- [KCM06] Leonid Kontorovich, Corinna Cortes, and Mehryar Mohri. Learning linearly separable languages. In *The 17th International Conference on Algorithmic Learning Theory (ALT 2006)*, volume 4264 of Lecture Notes in Computer Science, pages 288–303. Springer, Heidelberg, Germany, October 2006.
- [KCM08] Leonid (Aryeh) Kontorovich, Corinna Cortes, and Mehryar Mohri. Kernel methods for learning languages. *Theoretical Computer Science*, 405(3):223 – 236, 2008. Algorithmic Learning Theory.
- [Kob06] Gregory Kobele. *Generating Copies: An Investigation into Structural Identity in Language and Grammar*. PhD thesis, University of California, Los Angeles, 2006.
- [LCSTW02] H. Lodhi, N. Cristianini, J. Shawe-Taylor, and C. Watkins. Text classification using string kernels. *Journal of Machine Language Research*, 2:419–444, 2002.
- [Lot05] M. Lothaire, editor. *Applied Combinatorics on Words*. Cambridge University Press, Cambridge, UK, New York, 2nd edition, 2005.
- [MP71] R. McNaughton and S. Papert. *Counter-Free Automata*. MIT Press, 1971.
- [RP07] James Rogers and Geoffrey Pullum. Aural pattern recognition experiments and the subregular hierarchy. In Marcus Kracht, editor, *Proceedings of 10th Mathematics of Language Conference*, pages 1–7, 2007. University of California, Los Angeles.
- [RW04] Sharon Rose and Rachel Walker. A typology of consonant agreement as correspondence. *Language*, 80(3):475–531, 2004.
- [Shi85] Stuart Shieber. Evidence against the context-freeness of natural language. *Linguistics and Philosophy*, 8:333–343, 1985.
- [Sim75] Imre Simon. Piecewise testable events. In *Automata Theory and Formal Languages: 2nd Grammatical Inference conference*, pages 214–222, Berlin ; New York, 1975. Springer-Verlag.
- [SS83] 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.
- [STC05] John Shawe-Taylor and Nello Christianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2005.
- [Tra01] A.N. Trahtman. Piecewise and local threshold testability of DFA. In Rūsiņš Freivalds, editor, *Fundamentals of Computation Theory, 13th International Symposium, FCT2001*, pages 347–358, New York, 2001. Springer.
- [WC08] C. Whitney and P. Cornelissen. SERIOL reading. *Language and Cognitive Processes*, 23:143–164, 2008.