Relativized Adjacency

Dakotah Lambert

Journal of Logic, Language and Information manuscript No.
JLLI-D-20-00037R1

Relativized adjacency

Dakotah Lambert

Received: 13 Sep 2020 / Accepted: 05 Apr 2023

Abstract For each class in the piecewise-local subregular hierarchy, a relativized (tier-based) variant is defined. Algebraic as well as automata-, language-, and model-theoretic characterizations are provided for each of these relativized classes, except in cases where this is provably impossible. These various characterizations are necessarily intertwined due to the well-studied logic-automaton connection and the relationship between finite-state automata and (syntactic) semigroups. Closure properties of each class are demonstrated by using automata-theoretic methods to provide constructive proofs for the closures that do hold and giving language-theoretic counterexamples for those that do not. The net result of all of this is that, rather than merely existing as an operationally-defined parallel set of classes, these relativized variants integrate cleanly with the other members of the piecewise-local subregular hierarchy from every perspective. Relativization may even prove useful in the characterization of star-free, as every star-free stringset is the preprojection of another (also star-free) stringset whose syntactic semigroup is not a monoid.

Keywords characterization · finite-state automata · formal language theory · model theory · subregular hierarchy · syntactic semigroups

Mathematics Subject Classification (2010) 68Q19 · 68Q45 · 68Q70 · 20M35

1 Introduction

The piecewise-local subregular hierarchy, henceforth referred to as simply *the subregular hierarchy*, has been extensively studied for decades, with the local branch

D. Lambert
Université Jean Monnet Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, 42023 Saint-Etienne, France ORCiD: 0000-0002-7056-5950
E-mail: dakotahlambert@acm.org

introduced by McNaughton and Papert (1971) and the piecewise branch stemming from Simon (1975). Local constraints are, as the name implies, good at capturing dependencies based on adjacent events, and can do so with even the simplest logics. Even some long-distance dependencies can be captured, such as "A and B do not occur in the same word", but there is no notion of directionality here. Piecewise constraints fall on the other extreme, easily representing certain types of long-distance dependencies but requiring at least first-order logic to be able to refer to adjacent events at all.

In order to more simply state some types of long-distance dependencies, and to account for some that piecewise constraints cannot, a third branch came into existence with the tier-based strictly local (TSL) class imposing adjacency on distant parts of a string (Heinz et al. 2011). A TSL description works by relativizing the concept of adjacency over some subset of the alphabet, referred to as the tier alphabet. Symbols outside this subset are ignored entirely.

Linguistic interest in the TSL class stems from its usefulness in describing long-distance dependencies, especially those that strictly piecewise constraints cannot handle such as blocked harmony patterns (Heinz et al. 2011; McMullin 2016). For example, the liquid dissimilation pattern of Latin (Cser 2010) is a sort of blocked harmony pattern, shown to be 2-TSL by McMullin (2016). This pattern can be shown to be strictly star-free when restricted to the local or piecewise branches of the subregular hierarchy. However, another important consideration for linguistics is learnability, and the star-free class is not learnable (Gold 1967). On the other hand, 2-TSL has been shown to be effectively learnable both by humans (McMullin 2016) and by machines (Jardine and Heinz 2016), and k-TSL for arbitrary k has been shown to be learnable as well (Jardine and McMullin 2017).

The main formal result of Heinz et al. (2011) was a language-theoretic proof that TSL is a subclass of star-free. Originally TSL was defined by applying an erasing homomorphism to a string(set), projecting it to strings formed from the tier alphabet, then applying a strictly local filter to the result. This operational perspective is useful in describing the solution, but it can mask some insights. To provide more clarity, Lambert and Rogers (2020) provide an equivalent alternative definition based on model theory and a new class of ordering relations, and from this they develop language- and automata-theoretic *characterizations* of the class. A characterization of a class is a property such that all and only those sets that have this property are in the class.

The present work extends this further, characterizing not only the TSL class but also relativized variants (introduced here) of the other classes in the subregular hierarchy. These characterizations also go further, including algebraic as well as automata-, language-, and model-theoretic results.

We begin in section 2 with an overview of the model-theoretic concepts that will be used, then section 3 provides definitions as well as model- and language-theoretic characterizations for all of the relativized classes. Section 4 provides automata-theoretic characterizations for some of the classes, deferring the rest to section 6 in which automata are converted to an algebraic structure. In this latter section, algebraic characterizations are given for each new class, primarily synthesizing classical results and applying them to the new structures. Closure properties are proved in section 5, using

properties of automata to prove that some closures do hold, while using the language-theoretic characterizations provided earlier to demonstrate that some other potential properties do not.

2 Model theory

Concepts from finite model theory provide a uniform way to describe relational structures and their parts in logical terms (see Libkin 2004 for a thorough introduction). Applying these concepts to linguistic structure is not a new idea, with applications to syntax by Rogers (1996, 1998) beginning to popularize the approach. In this section we discuss a model-theoretic treatment of relativized adjacency. The relation defined here is then used to define variants of each class of the subregular hierarchy, where the relativized variant of the strictly local class is identical to the tier-based strictly local class of Heinz et al. (2011).

A relational word model consists of a *domain*, $\mathscr{D}$, which is isomorphic to an initial segment $\{1,\ldots,n\}$ of the nonzero natural numbers and represents positions in the word, as well as a collection of relations, $R_i \subseteq \mathscr{D}^{a_i}$, each of which has its own arity a_i.

$$\mathscr{M}(w) = \langle \mathscr{D}; R_i \rangle.$$

Generally we assume that a model consists of at least one *ordering* relation, as well as one or more unary *labeling* relations that partition the domain. Additional relations of any arity are of course permitted. The assumption of a partition is nonrestrictive; one can convert a model whose labeling relations do not form a partition of the domain into a partitioned normal form by using the powerset of these relations instead. One simple example of an ordering relation is that of general precedence ($<$), where $a < b$ if and only if (iff) the domain element a occurs anywhere before b.

The immediate successor relation that defines the local branch of the subregular hierarchy can be derived in first order logic from general precedence.

$$x \lhd y \stackrel{\text{def}}{=} (x < y) \wedge \neg(\exists z)[x < z < y].$$

This is simply the transitive reduction of this general precedence relation. Instead of reduction, we might consider restricting the domain to all and only those elements that satisfy a certain predicate φ.

$$x <^{\varphi} y \stackrel{\text{def}}{=} \varphi(x) \wedge \varphi(y) \wedge (x < y).$$

This is essentially the general precedence relation on the model's projection to those elements that satisfy φ. We can of course combine these to obtain the reduction of this restriction,

$$x \lhd^{\varphi} y \stackrel{\text{def}}{=} (x <^{\varphi} y) \wedge \neg(\exists z)[x <^{\varphi} z <^{\varphi} y],$$

which defines a relativized successor relation. Given some alphabet Σ and some set of salient symbols $T \subseteq \Sigma$, the predicate $\varphi(x) = T(x) = \bigvee_{\tau \in T} \tau(x)$ results in a relation

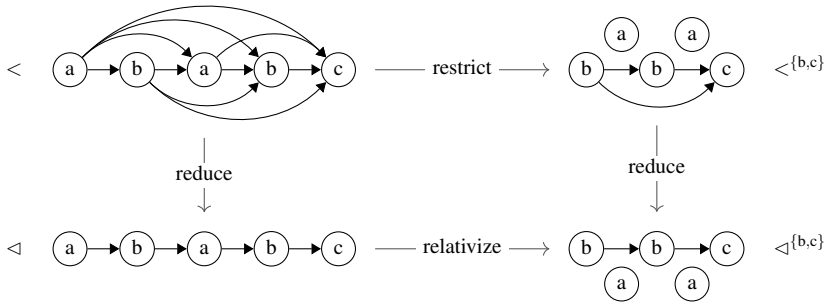


Fig. 1 Word models for "ababc" using general precedence, immediate successor, and relativized variants of each, showing the relationships among these relations. Domain elements that are not ordered are pulled aside from the structure. In these examples, the alphabet is $\Sigma = \{\mathrm{a},\mathrm{b},\mathrm{c}\}$, and the salient symbols for the relativized relations are $T = \{\mathrm{b},\mathrm{c}\}$

that acts as if it were the successor relation on the model's projection to T, reminiscent of the original definition of the TSL^T class. We refer to this specific kind of relativization as *projective relativization*.

Figure 1 shows how the general precedence and immediate successor relations, as well as their relativized variants, relate to one another. These relationships do not only hold for these relations. In fact any binary relation whose transitive closure is antisymmetric may be relativized by taking the transitive reduction of a restriction of its transitive closure. The antisymmetry requirement ensures uniqueness of the result (Aho et al. 1972). Throughout this text we will consider only projective relativization, though with appropriate choice of φ the more general treatment can be shown to capture the structure-sensitive tier-based strictly local class of De Santo and Graf (2019) or the domain- and interval-based strictly piecewise classes of Graf (2017). One important property specific to projective relativization is that the unordered elements truly have no effect on the ordered ones. Whether a domain element is included in the restriction is decided entirely by the unary relation that labels that point, and so the unordered elements can be freely removed, shuffled, or inserted at any point.

Now that we have a notion of a structure, we turn to discussion of contained structures.[1] Following Lambert and Rogers (2020), we distinguish two concepts: a *factor*, which is a connected structure contained within a model, and a *window*, which is a structured collection of domain elements from which a factor may be derived. We begin by defining a window.

Given a (homogeneous) relation R of arity $a \geq 2$, i.e. $R\colon \mathscr{D}^a$, its a-windows are defined by the set

$$\mathscr{W}_a^R \stackrel{\text{def}}{=} \left\{ \left\{ \langle \overset{i}{x_i}, \overset{i+1}{x_{i+1}} \rangle : 1 \leq i < a \right\} : \langle x_1, \ldots, x_a \rangle \in R \right\}.$$

This effectively turns each tuple in the relation into a sequence of overlapping pairs that represent the edges of a (linear) directed graph version of that tuple. Each node

[1] Because our notion of structural containment is distinct from the standard model-theoretic notion of substructures, we are careful to avoid that term.

in this graph is labeled by not only the domain element itself, but also an index so that cycles in the structure do not translate into cycles in the window. For instance, if 1 is a domain element and $\langle 1,1\rangle$ appears in R, the only 2-window in the set of 2-windows that corresponds to this relation is

$$\{\langle \overset{1}{1}, \overset{2}{1}\rangle\}.$$

Discussing smaller windows is simple. Any connected subgraph of an a-window is also a window, of size equal to the number of nodes it contains.

For windows of size larger than the arity of the relation from which they are defined, we can use an inductive definition:

$$\begin{aligned}\mathscr{W}^R_{k+1} \overset{\text{def}}{=} \Big\{ A \cup \langle x_{a-1}^{j_{a-1}}, x_a^{k+1}\rangle : {} & A \in \mathscr{W}^R_k \text{ and } \langle x_1,\ldots,x_a\rangle \in R \\ & \text{and } \{j_1,\ldots,j_{a-1}\} \subseteq \{1,\ldots,k\} \\ & \text{and } \{\langle x_i^{j_i}, x_{i+1}^{j_{i+1}}\rangle : 1 \le i < a-1\} \subseteq A \\ & \text{and } (\exists y,\ell)[\langle x_{a-1}^{j_{a-1}}, y^{\ell}\rangle \in A \text{ or } \langle y^{\ell}, x_{a-1}^{j_{a-1}}\rangle \in A] \\ & \text{and } \big(\forall j_a \in \{1,\ldots,k\}\big)\big[\langle x_{a-1}^{j_{a-1}}, x_a^{j_a}\rangle \notin A\big] \Big\}.\end{aligned}$$

The conditions on the first line select a k-window and an element of the relation. The second line selects $a-1$ indices. The third line, which is never relevant for a binary relation, ensures that these indices form a path from the first to the last, and that this path is labeled by the appropriate domain elements. The fourth line accounts for binary relations, simply asserting that the selected index corresponds to an appropriate domain element. And finally the fifth ensures that edges in the model may only be repeated in the case of cycles.

In short, for each k-window, we find a linear subgraph (a path) that maps to the first $a-1$ elements of a tuple in R, then add an edge from the final node of this path to a newly constructed node representing the final domain element from that tuple.

The conditions assert that adding this new node does not simply repeat the construction of an already-existing path, while still allowing cycles to be iterated without bound. For example, given the 2-window described previously, the only valid 3-window formed from the same $\langle 1,1\rangle$ tuple is

$$\{\langle \overset{1}{1}, \overset{2}{1}\rangle, \langle \overset{2}{1}, \overset{3}{1}\rangle\}.$$

Both index 1 and index 2 provide valid attachment points for a $\langle 1,1\rangle$ edge, but this edge has already been followed from index 1, so that attachment is ruled out by the condition on the fifth line. The result of this induction is that a window is a rooted, connected, acyclic graph of indexed domain elements, where the root is the unique node of in-degree zero.

Although only binary relations will be discussed in this article, this definition applies more generally. Consider $R_3 = \langle 1,2,3\rangle, \langle 1,2,4\rangle, \langle 2,4,5\rangle, \langle 3,4,5\rangle$. Two pairs

can be chained: $\langle 2,4,5\rangle$ can overlay the right of $\langle 1,2,4\rangle$, or $\langle 1,2,4\rangle$ and $\langle 1,2,3\rangle$ can be overlaid at their left portions. So the only 4-windows of R_3 are

$$\left\{\overset{1}{\langle}\overset{2}{1,2}\rangle, \overset{2}{\langle}\overset{3}{2,4}\rangle, \overset{3}{\langle}\overset{4}{4,5}\rangle\right\} \text{ and } \left\{\overset{1}{\langle}\overset{2}{1,2}\rangle, \overset{2}{\langle}\overset{3}{2,4}\rangle, \overset{2}{\langle}\overset{4}{2,3}\rangle\right\} \text{ and } \left\{\overset{1}{\langle}\overset{2}{1,2}\rangle, \overset{2}{\langle}\overset{3}{2,3}\rangle, \overset{2}{\langle}\overset{4}{2,4}\rangle\right\}.$$

It is possible that an alternative definition requiring less overlap might be preferred. Notice though that the latter two windows are identical graphs when the indices are ignored.

In general, there can be several windows that correspond to the same contained structure. Looking only at the domain elements represented unifies these multiple representations: the *factor at* a window x in the word model m (written $[\![x]\!]_m$) is the restriction of m to the domain elements in x. For instance, consider the signature

$$\mathscr{M}^{\lhd^{\{b,c\}}} = \langle \mathscr{D}; \lhd^{\{b,c\}}, \mathrm{a}, \mathrm{b}, \mathrm{c}\rangle$$

and a word model over this signature

$$m = \Big\langle \{1,2,3,4,5\}; \{\langle 2,4\rangle, \langle 4,5\rangle\}, \{1,3\}, \{2,4\}, \{5\}\Big\rangle,$$

which corresponds to the "ababc" example on the $\{b,c\}$ tier shown as the relativized successor model in Figure 1. If we have a window x such that the domain elements included in x are all and only 2, 4, and 5, then the factor at x is the corresponding restriction

$$[\![x]\!]_m = m \restriction x = \Big\langle \{2,4,5\}; \{\langle 2,4\rangle, \langle 4,5\rangle\}, \emptyset, \{2,4\}, \{5\}\Big\rangle.$$

This is the widest possible factor in that example, as the other elements are disconnected from this structure.

The set of all k-factors of a model m is the set

$$\mathscr{F}_k(m) \overset{\text{def}}{=} \{[\![x]\!]_m : x \in \mathscr{W}_k\}.$$

where the windows are built over the ordering relations of m.

The word models shown to this point have been without explicit indication of domain boundaries. Often, however, we wish to consider models in which these boundaries are explicit, which we call *anchored models*. These can be formed by augmenting a model with new positions, self-related under all ordering relations, that are labeled "$\rtimes$" and "$\ltimes$" for head and tail boundaries, respectively. This self-relation allows words shorter than k to be captured by k-windows without special treatment. For any projectively relativized relation, we assume for notational convenience that these boundary symbols are considered salient if they are present. So rather than writing $\lhd^{\{\rtimes,\mathrm{b},\mathrm{c},\ltimes\}}$ we simply write $\lhd^{\{\mathrm{b},\mathrm{c}\}}$ instead. Figure 2 shows an anchored word model for "ababc" under the $\lhd^{\{\mathrm{b},\mathrm{c}\}}$ relation along with all of its nonempty 3-factors. Because the domain boundaries are self-related, assuming the domain elements of "$\rtimes$ababc$\ltimes$" are 1 through 7 in order, the windows

$$\left\{\overset{1}{\langle}\overset{2}{1,1}\rangle, \overset{2}{\langle}\overset{3}{1,3}\rangle\right\} \quad \text{and} \quad \left\{\overset{1}{\langle}\overset{2}{1,1}\rangle, \overset{1}{\langle}\overset{3}{1,3}\rangle\right\}$$

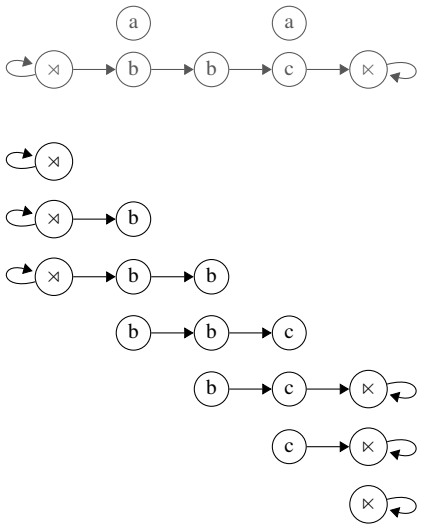


Fig. 2 All 3-factors of "⋊ababc⋉" under the $\lhd^{\{b,c\}}$ relation. Factors that appear shorter than this are formed from windows that repeat the boundary symbols.

both refer to the relativized prefix "⋊b" of this string. (Again, either instance of domain element 1 is a valid attachment point for a $\langle 1,3\rangle$ edge.) The "a" elements are unordered and do not occur in any factor.

One might notice that a model that has a smaller number of domain elements than the arity of its ordering relation might have no factors at all by these definitions. While this is never a problem for anchored models, one might consider alternative constructions when using nonanchored models. The simplest is to construct the factors of the anchored models and then strip away the domain boundaries from the result. In any case, the use of anchored word models will be assumed throughout this text, and the particular definitions used here guarantee that the factors of a model under a relativized relation are exactly those under the corresponding nonrelativized one of the model's projection.

3 Language-theoretic characterizations

A grammar is some representation of a mechanism by which the membership of a string in a stringset may be decided. A class of grammars is denoted by $\mathbb{G}$. The characteristic function $\mathbb{1} : \mathbb{G} \times \mathscr{M} \to \mathbb{B}$ is

$$\mathbb{1}_G(m) \stackrel{\text{def}}{=} \begin{cases} \top & \text{if } m \text{ satisfies } G, \\ \bot & \text{otherwise.} \end{cases}$$

Here, $\mathbb{B}$ represents the binary Boolean ring, $\top$ is true, $\bot$ is false. Functions of more than one argument are sometimes written with their first argument as a subscript; $\mathbb{1}_G$

can be thought of as the partial application of the curried form of $\mathbb{1}$. The stringset represented by G is the set of all and only those strings whose models satisfy it:

$$\mathscr{L}(G) \stackrel{\text{def}}{=} \{w : \mathbb{1}_G(\mathscr{M}(w))\}.$$

Two grammars G_1 and G_2 are equivalent iff they are extensionally equal, that is, $\mathscr{L}(G_1) = \mathscr{L}(G_2)$.

3.1 Strict locality

The original definition of the TSL class from Heinz et al. (2011) was operational. A stringset L is k-TSL^T iff there exists a k-SL grammar such that L contains all and only those strings whose projection to T satisfy this grammar.

A grammar for a k-SL stringset is simply a subset of $\mathscr{F}_k(\Sigma^*)$, that is, a set of k-factors, where an anchored model m satisfies G iff each of its k-factors occurs in G (McNaughton and Papert 1971):

$$\mathbb{1}_G(m) \;=\; \mathscr{F}_k^{\lhd}(m) \subseteq G.$$

Because the symbols not in T never appear in any factors under $<^T$ or $\lhd^T$, any two strings with the same projection to T will have the same set of factors under these relations. Specifically, if $\pi_T(m)$ represents the projection of m to T, it holds that m and $\pi_T(m)$ have exactly the same set of factors under these relations. Moreover, if $T = \Sigma$ it follows by definition that these are equivalent to their nonrelativized analogues. Therefore the only difference between a k-SL grammar and a corresponding k-TSL one is interpretation:

$$\mathbb{1}_G(m) \;=\; \mathscr{F}_k^{\lhd^T}(m) \subseteq G.$$

We will see that this is indeed always the case, so from this point on characteristic functions will be given without the relation specified. Using $\lhd$ yields the local class, $\lhd^T$ its relativization.

With this in mind, we turn to the language-theoretic characterization of the SL class: closure under substitution of suffixes (Rogers and Pullum 2011, see also De Luca and Restivo 1980). A stringset satisfies suffix substitution closure iff there is some k such that for any two strings $w_1 = u_1xv_1$ and $w_2 = u_2xv_2$ where $|x| \geq k-1$, if both w_1 and w_2 are in the set, then so is $w_3 = u_1xv_2$. Lambert and Rogers (2020) provide a similar characterization for the TSL class.

Definition 1 (Preprojective suffix substitution closure: PSSC) A stringset L over the alphabet Σ is closed under T-preprojective suffix substitution (T-PSSC) iff there is some natural number k such that for any two strings $w_1 = u_1x_1v_1$ and $w_2 = u_2x_2v_2$ where $\pi_T(x_1) = \pi_T(x_2)$ and $|\pi_T(x_1)| \geq k-1$, if both w_1 and w_2 are in L, then so is $w_3 = u_1x_1v_2$.

Notice that Σ-PSSC is equivalent to standard suffix substitution closure, as the strings are necessarily equal to their projections. On its own though, T-PSSC is not sufficient to characterize TSL. The other necessary condition is that symbols not in T be freely insertable and deletable, as previously discussed for the relativized relations.

Theorem 1 *A stringset L over an alphabet Σ is* TSL *iff there is some subset $T \subseteq \Sigma$ such that symbols not in T are freely insertable and deletable and L is closed under T-*PSSC.

Proof To prove that such a stringset is TSL, suppose there exists such a T. Then by T-PSSC, the projection of L to T is SL. Further, by insertion and deletion closure of non-T symbols we guarantee that w is in L iff its projection to T is. Together, these facts show that L is TSL.

To prove the reverse implication, suppose that L is TSL. Then it is k-TSL^T for some k and T. By definition of TSL, w is in L iff its projection to T is, and so symbols not in T are freely insertable and deletable. Because L is TSL^T, its projection to T is SL and thus satisfies suffix substitution closure. Further since L is closed under insertion of symbols not in T, it is then also closed under T-PSSC. □

In order to prove that a stringset is TSL, it suffices to provide a grammar. To prove that a stringset cannot be TSL, one can find some set of symbols that are not freely both insertable and deletable, then form strings from those symbols alone that violate PSSC. For instance, a constraint that forbids sequential (not necessarily adjacent) occurrences of "a..b..a" is not TSL because neither "a" nor "b" is freely insertable (so they are necessarily in T) and PSSC is violated by the following words:

$$\begin{array}{ll} \mathrm{a}\overbrace{\mathrm{b}..\mathrm{b}}^{k-1}\mathrm{b} & (\in) \\ \mathrm{b}\,\mathrm{b}..\mathrm{b}\,\mathrm{a} & (\in) \\ \hline \mathrm{a}\,\mathrm{b}..\mathrm{b}\,\mathrm{a} & (\notin). \end{array}$$

3.2 Complements

If an SL stringset contains all strings that satisfy a grammar G, the complement of this set is all strings that do not satisfy G. Since a model satisfies G iff all of its factors are in G, it follows that the model does not satisfy G iff it has at least one factor not in G. We can use the grammar of the complemented SL stringset as the grammar for a coSL stringset. The result is a collection of factors, at least one of which is required to appear in every word. The resulting characteristic function then is

$$\mathbb{1}_G(m) \;=\; G \cap \mathscr{F}_k(m) \neq \emptyset.$$

Definition 2 (Ideal containment: IC) A stringset L is k-coSL iff L contains all and only those strings w that themselves contain at least one factor $f \in \mathscr{F}_k^{\triangleleft}(w)$ such that every string x that contains that factor is also in L:

$$\left\{x : f \in \mathscr{F}_k^{\triangleleft}(x)\right\} \subseteq L.$$

If there exists some k for which L is k-coSL, then L is coSL.

In IC, the factor f is the (not necessarily unique) factor that caused dissatisfaction of the corresponding SL grammar. Then in order to show that a stringset is not

k-coSL, it suffices to find a (preferably small) word $w \in L$ and a set of words S such that $\mathscr{F}_k(w) \subseteq \mathscr{F}_k(S)$, yet no member of S is in L. For instance we can show that a constraint that bans occurrence of the substring "ab" is not coSL because IC is violated by the following:

$$\begin{aligned} w &= \mathrm{a..a} && (\in) \\ S &= \{\underbrace{\mathrm{a..a}}_{k-1}\mathrm{b}\underbrace{\mathrm{a..a}}_{k-1}\} && (\notin). \end{aligned}$$

The factors of w are $\{\rtimes \mathrm{a}^i, \mathrm{a}^i \ltimes : 0 \leq i < k\}$. Each of these factors occurs in the single word in S, and so the presence of any given factor is not sufficient to guarantee acceptance. This extends trivially to *preprojective ideal containment*.

Definition 3 (Preprojective ideal containment: PIC) A stringset L is k-coTSL iff there exists some $T \subseteq \Sigma$ such that L contains all and only those strings w that themselves contain at least one factor $f \in \mathscr{F}_k^{\lhd^T}(w)$ such that every string x that contains that factor is also in L:

$$\left\{x : f \in \mathscr{F}_k^{\lhd^T}(x)\right\}.$$

And L is coTSL iff it is k-coTSL for some k.

Since the factors of w under $\lhd^T$ are the same as those of its T-projection under $\lhd$, this is equivalent in every way to stating that L is k-coTSL^T iff its T-projection is k-coSL and it is closed under insertion and deletion of symbols not in T. Further the order of complementation and relativization is immaterial, as will become clear in section 4.

3.3 Local testability

Much as the SL stringsets consist of words containing only permitted factors, the locally testable (LT) ones consist of words whose set of factors is permitted (McNaughton and Papert 1971). This allows a mechanism to reject "ab" even in the case of accepting "abab", whose 2-factors are $\{\rtimes \mathrm{a}, \mathrm{ab}, \mathrm{b} \ltimes\}$ and $\{\rtimes \mathrm{a}, \mathrm{ab}, \mathrm{ba}, \mathrm{b} \ltimes\}$, respectively. Due to their subset relationship, a mechanism capable only of 2-SL distinctions could not do this, though each of the three other possible combinations of acceptance and rejection of these two strings is possible under 2-SL. For testable stringsets, a grammar is a set of permitted sets of factors, and its characteristic function is

$$\mathbb{1}_G(m) \;=\; \mathscr{F}_k(m) \in G.$$

Rogers and Pullum (2011) state that a stringset is LT iff it is closed under *local test invariance*, where given two strings w_1 and w_2 such that $\mathscr{F}_k^{\lhd}(w_1) = \mathscr{F}_k^{\lhd}(w_2)$, the first is in the set iff the second is as well. This extends trivially to *preprojective local test invariance*.

Definition 4 (Preprojective local test invariance: PLTI) A stringset L is TLT iff there exists some $T \subseteq \Sigma$ and some k such that given two strings w_1 and w_2 such that $\mathscr{F}_k^{\lhd^T}(w_1) = \mathscr{F}_k^{\lhd^T}(w_2)$, the first is in L iff the second is as well.

By the same reasoning employed in discussion of coTSL, this is equivalent in every way to stating that L is k-TLT^T iff its T-projection is k-LT and it is closed under insertion and deletion of symbols not in T.

3.4 Threshold testability

The LT class is characterized by sets of factors. But a set is merely a structure that describes each possible element by a Boolean value, whether or not that element is included. One might consider a natural extension of this structure which saturates its count of occurrences not at 1 but at some arbitrary value t. This is exactly what Beauquier and Pin did when defining the *locally threshold testable* (LTT) stringsets in 1989. We denote this generalized structure by

$$\mathscr{F}_{k,t}(m) \stackrel{\text{def}}{=} \wr [\![x]\!]_m : x \in \mathscr{W}_k \wr_t.$$

For threshold testable stringsets, a grammar is a set of permitted multisets whose characteristic function is

$$\mathbb{1}_G(m) \;=\; \mathscr{F}_{k,t}(m) \in G.$$

The characterization of LTT of course is *local threshold test invariance*, where given two strings w_1 and w_2 such that $\mathscr{F}^{\triangleleft}_{k,t}(w_1) = \mathscr{F}^{\triangleleft}_{k,t}(w_2)$, the first is in the set iff the second is as well. This extends trivially to *preprojective local threshold test invariance*.

Definition 5 (Preprojective local threshold test invariance: PLTTI) A stringset L is TLTT iff there exists some $T \subseteq \Sigma$ and some k and t such that given two strings w_1 and w_2 such that $\mathscr{F}^{\triangleleft T}_{k,t}(w_1) = \mathscr{F}^{\triangleleft T}_{k,t}(w_2)$, the first is in L iff the second is as well.

By the same reasoning employed in discussion of coTSL and TLT, this is equivalent in every way to stating that L is k,t-TLTT^T iff its T-projection is k,t-LTT and it is closed under insertion and deletion of symbols not in T.

Under the definitions of windows and factors used in this text, this actually counts prefixes and suffixes of length less than k more than once, since several windows might correspond to the same factor. Importantly, for fixed k the count is consistent for prefixes (suffixes) of a given length, and since there is at most one length-n prefix (suffix) in any valid word model, this overcounting does not affect the possible distinctions.

3.5 Piecewise relativizations

Using general precedence instead of successor in the definition of the SL class yields the strictly piecewise (SP) class. Characterized by Rogers et al. (2010) as those stringsets closed under deletion (see also Haines 1969 for an earlier treatment of stringsets closed under deletion), we can show that a stringset is TSP iff it is SP. In fact, unlike reduction, relativization provides neither more nor less expressive power in precedence-based models.

By definition $<^{\Sigma}$ is equivalent to $<$, thus all nonrelativized stringsets are also trivially relativized ones. More interesting is the reverse. Recall from Figure 1 that the relativization of the $<$ relation is in fact merely a restriction, and so $\mathscr{F}_k^{<^T}(m) \subseteq \mathscr{F}_k^{<}(m)$. If a factor occurs on the restriction, then it also occurs in the nonrestricted model. Therefore a full piecewise model must be at least as powerful as its relativization, but since T can be equal to Σ they are in fact equivalent.

For this same reason, when the factor width is fixed at $k = 1$ all of the projectively relativized classes are equivalent to their nonrelativized analogues.

4 Automata

In this section we discuss characterizations of the relativized classes and constructions of their constituent stringsets in terms of deterministic finite-state automata (DFAs).

A DFA is a directed graph that represents a machine that computes the well-formedness of a string with respect to some regular stringset. It is represented by a triple $\langle \delta, q_0, F \rangle$, where Q is a set of states, Σ an alphabet, $\delta \colon \Sigma \times Q \to Q$ a transition function which represents edges in the graph, $q_0 \in Q$ an initial state, and $F \subseteq Q$ a set of accepting states.[2] A DFA is *complete* iff δ is total. A word w is accepted by the DFA iff there is some path $q_0 \to \cdots \to q_f$ for some $q_f \in F$ (an *accepting path* from q_0) whose labels spell out w.

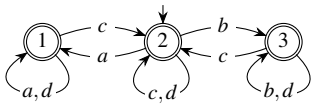
Let $\sim$ represent the equivalence relation over states in Q under which $q_1 \sim q_2$ iff for all $u \in \Sigma^*$ there is an accepting path from q_1 labeled u whenever such a path exists from q_2 and vice-versa. This is *Nerode equivalence*. By the Myhill-Nerode theorem (Nerode 1958), one can construct a *minimal* DFA from a given one by replacing each state in Q by its Nerode-equivalence class, the element of $Q/\!\sim$ that contains it. A minimal DFA might have a unique *nonaccepting sink*, a state q from which there are no accepting paths for any string. A *canonical* DFA is one that is minimal and has had its nonaccepting sink (if any) removed.

4.1 Characterizations

Every relativized class discussed in this text is closed under insertion and deletion of symbols not in T. In other words, such symbols provide no information regarding the well-formedness of a word. It follows then that from a given state q, the state reached by following an edge labeled by such a symbol must be in the same Nerode-equivalence class as q itself. Thus in a canonical DFA, the nonsalient symbols are exactly those that form self-loops on all states simultaneously.

$$\complement T = \bigcap_{q \in Q} \{\sigma \in \Sigma : \delta_\sigma(q) = q\}.$$

[2] The values of Q and Σ are implied by the signature of δ. Some prefer to specify them explicitly, which would result in automata being 5-tuples.



$$\complement T = \bigcap \big\{ \{a,d\}, \{c,d\}, \{b,d\} \big\} = \{d\}$$

Fig. 3 A canonical DFA for which d is a nonsalient symbol.

In other words, these are exactly the symbols σ such that the set of fixed points of δ_σ is the entirety of Q. Figure 3 shows a canonical DFA that represents a TSL stringset in which d is not a salient symbol. The fixed points for a, b, c, and d are $\{1\}$, $\{3\}$, $\{2\}$, and $\{1,2,3\}$, respectively.

Given a canonical automaton for a language L, the automaton for its T-projection is formed by restricting δ to $\Sigma \setminus \complement T$. As discussed previously, with this choice of T a stringset is in the relativized variant of a given class iff its T-projection is in that class. The T found this way is in fact the smallest set for which this holds, though some stringsets might permit several possible values for T. For example, the stringset Σ^* is TSL^P for every $P \subseteq \Sigma$.

Once the projection has been found, any of the numerous existing methods for determining class membership can be used. Most of these tests are based on the algebraic interpretation of the syntactic semigroup corresponding to the automaton, which will be discussed further in section 6. However, for the SL class we can use a result of Edlefsen et al. (2008, see also Caron 1998). Given a canonical DFA $A = \langle \delta, q_0, F \rangle$, construct its *powerset graph* by defining $\delta^{\mathscr{P}} \colon \Sigma \times \mathscr{P}(Q) \to \mathscr{P}(Q)$:

$$\delta^{\mathscr{P}}_\sigma(S) \stackrel{\text{def}}{=} \{ \delta_\sigma(s) : s \in S \}.$$

The stringset represented by A is SL iff the graph formed by $\delta^{\mathscr{P}}$ contains no cycles that iterate a node whose label contains two or more elements. A powerset graph of a non-SL stringset is shown in Figure 4 with the offending path marked.

Edlefsen et al. (2008) also provide a more efficient algorithm in terms of pairs of states. However, the powerset graph construction also allows extraction of a grammar for the target stringset from the automaton itself (Rogers and Lambert 2019a). Since a stringset is coSL iff its complement is SL, this serves as an automata-theoretic method to decide membership in that class as well. Thus by projecting an automaton to the appropriate tier alphabet, not only can we show that the target stringset is TSL or coTSL, but we can also obtain a canonical grammar for this stringset if it is.

4.2 Constructions

The projection mechanism of section 4.1 is invertible. Given a DFA representing a stringset L whose alphabet is some $T \subseteq \Sigma$, the preprojection of L to Σ is given by adding self-edges on each symbol $\sigma \in \Sigma \setminus T$ to every state. If the subregular class of L is known, then the result lies in the corresponding relativized class.

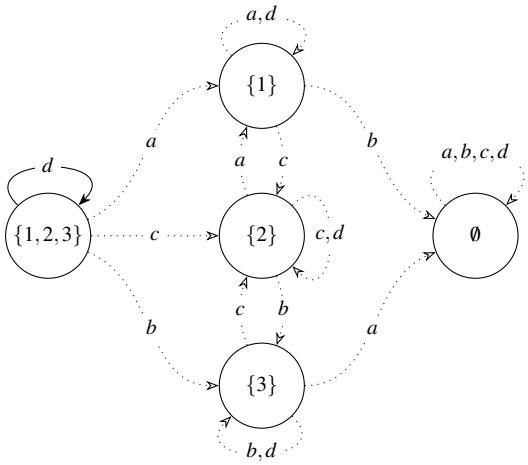


Fig. 4 The powerset graph that corresponds to the DFA of Figure 3, with the cycle marked that proves this stringset is not SL. Notice that if d is removed, there is no such cycle.

However, the purpose of these projectively relativized classes is to represent those stringsets in which only certain symbols are salient. It would be meaningful then to employ *alphabet-agnostic automata* (AAA). Such automata test for the occurrence of a factor within a target string of unknown or unspecified alphabet by augmenting the set of symbols relevant to the factor with a *wildcard* symbol ⍰, much like Beesley and Karttunen (2003). For our purposes, we consider a wildcard that matches all and only those symbols not already listed in the alphabet, like the @ of Hulden (2009). The empty language is represented by a single state which is non-accepting, bearing a self-loop labeled ⍰, which is the only member of the alphabet. The universal acceptor is identical, except its single state is accepting.

Factors under $<$ (piecewise factors) are the simplest to construct. Given a factor $f = \sigma_1 \dots \sigma_n$ under this relation, the AAA is defined in essentially the same way that Rogers et al. (2010) form a DFA:

$$\begin{aligned} Q &= \{0, \dots, n\} \\ \Sigma &= \{\sigma_1, \dots, \sigma_n, ⍰\} \\ q_0 &= 0 \\ F &= \{n\} \\ \delta(\sigma, q) &= \begin{cases} q+1 & \text{if } q < n \text{ and } \sigma = \sigma_{q+1}, \\ q & \text{otherwise.} \end{cases} \end{aligned}$$

An example is shown in Figure 5. Note that since ⍰ is by definition never included in the factor, edges on this symbol are always self-loops. In other words, this construction provides a clear picture as to why SP (and extensions thereof) and TSP (and extensions) should be identical.

For factors under $\lhd$ (local factors), any of the common approaches to substring-matching suffice, including that of Knuth et al. (1977). However, a naïve method

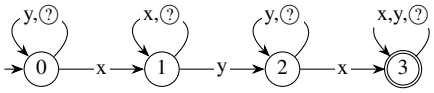


Fig. 5 Canonical AAA for the factor "xyx" under $<$.

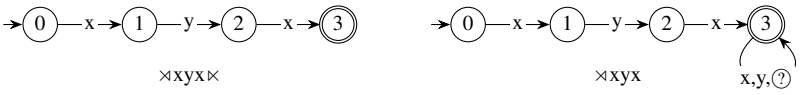


Fig. 6 Canonical automata for head-anchored factors.

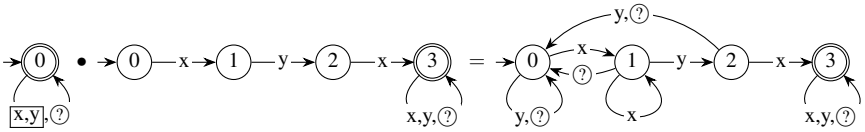


Fig. 7 Canonical AAA for the free factor "xyx" under $\lhd$ constructed via concatenation. The boxed x,y in the first operand are symbols that needed to be added for compatibility.

shown here demonstrates some properties of AAA. Fully-anchored factors of the form $f = \rtimes\sigma_1 \dots \sigma_n \ltimes$ look like piecewise factors, except edges to a non-accepting sink $\perp$ replace the self-loops:

$$Q = \{\perp, 0, \dots, n\}$$
$$\delta(\sigma, q) = \begin{cases} q+1 & \text{if } q < n \text{ and } \sigma = \sigma_{q+1} \\ \perp & \text{otherwise.} \end{cases}$$

For head-anchored but not tail-anchored factors, the difference is that $\delta(\sigma, n) = n$ rather than $\perp$. Figure 6 shows a canonical (trimmed) AAA constructed for each of "$\rtimes$xyx$\ltimes$" and "$\rtimes$xyx".

These automata can then be concatenated after a universal acceptor to represent tail-anchored and free factors, as in Figure 7, but this concatenation requires an extra step compared to standard DFA operations: the two inputs to any binary operation must be made *compatible*. Two AAA are compatible iff they have the same alphabet. Since ? represents any symbol not already in the alphabet, a symbol σ is added by placing new edges labeled by σ in parallel with any existing ? edges. Thus to make two AAA compatible, their alphabets should be extended in this way to the union of their individual alphabets. Two compatible automata can be combined using standard DFA operations, treating ? as just another symbol.

To fix the alphabet of an AAA to a specific set Σ, simply extend it as necessary and then remove any ? edges. Relativizing an automaton over some tier alphabet T is a process of fixing its alphabet to T, then adding ? edges back in as self-loops on every state. For example, Figure 8 shows how this process modifies the factor of Figure 7 to consider only 'x', 'y', and 'z' salient.

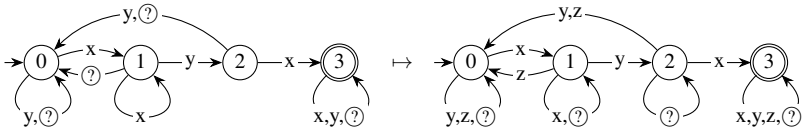


Fig. 8 Relativizing "xyx" to $T = \{\mathrm{x},\mathrm{y},\mathrm{z}\}$: Add 'z' in parallel to any existing ? edges, remove those ? edges, then add new ? edges as self-loops everywhere.

5 Closure properties

In this section we constructively prove some closure properties of relativized classes via their automata-theoretic characterizations, and use the language-theoretic ones to provide counter-examples to other closure properties.

5.1 Products

The intersections and unions of automata are both formed from *the product construction*. Given two automata $A = \langle \delta, q_0, F \rangle$ and $A' = \langle \delta', q_0', F' \rangle$, one constructs the product

$$A \otimes_{op} A' \stackrel{\text{def}}{=} \left\langle \delta^{\otimes}, \langle q_0, q_0' \rangle, F^{op} \right\rangle,$$

where the new transition function is defined pointwise:

$$\delta_\sigma^{\otimes}\left(\langle q, q' \rangle\right) \stackrel{\text{def}}{=} \left\langle \delta_\sigma(q), \delta_\sigma'(q') \right\rangle.$$

The set of accepting states is

$$F^{op} \stackrel{\text{def}}{=} \left\{ \langle q, q' \rangle : q \in F \ op \ q' \in F' \right\},$$

where *op* is "and" for intersection or "or" for union. If σ labels a self-loop on every state of both operands, then this product construction guarantees that the same will hold in the result. By construction then, if $\mathtt{C}T$ is the set of nonsalient symbols for A and $\mathtt{C}T'$ for A', then $\mathtt{C}T^{\otimes} \subseteq \mathtt{C}T \cap \mathtt{C}T'$ is a set for this product.[3] Notably, if A and A' represent stringsets in a relativized class where $T = T'$ and the underlying class is closed under union (intersection), the result of the union (intersection) of A and A' remains in the same relativized class with the same set of salient symbols.

Theorem 2 *The* TLT^T *and* TLTT^T *classes for fixed* T *are closed under both union and intersection.*

Proof Because LT and LTT are each closed under both union and intersection, the product construction guarantees that TLT^T and TLTT^T for fixed T are as well.

[3] Since the result is not necessarily canonical, a larger $\mathtt{C}T^{\otimes}$ (thus a smaller T) may also exist.

For the same reasons, the same holds for TSL^T under intersection and coTSL^T under union.

Note that these closures rely on equality of the sets of salient symbols. Consider the TSL stringset whose projection to $\{\text{a},\text{b}\}$ contains no "ab" factor and the TSL stringset whose projection to $\{\text{a},\text{c}\}$ contains no "aca" factor. In the intersection, none of "a", "b", or "c" is freely insertable, so each must be salient no matter what Σ is. Then even though the two strings

$$\begin{array}{ll} \overbrace{\text{c}..\text{c}}^{k}\,\text{b}\,\overbrace{\text{c}..\text{c}}^{k}\,\text{a}\,\overbrace{\text{c}..\text{c}}^{k} & (\in) \\ \text{c}..\text{c}\,\text{a}\,\text{c}..\text{c}\,\text{b}\,\text{c}..\text{c} & (\notin) \end{array}$$

have exactly the same k-factors and exactly the same counts for each, the first is in the intersection while the second is not. This is a violation of PLTTI (see page 11). Thus the intersection of these two stringsets is not even TLTT, and so by containment it cannot be TLT or TSL. For a more direct proof that this intersection is not TSL, consider the following violation of PSSC:

$$\begin{array}{ll} \text{ac}\overbrace{\text{c}..\text{c}}^{k-1}\text{ca} & (\in) \\ \text{b}\,\text{c}..\text{c}\,\text{b} & (\in) \\ \hline \text{ac}\,\text{c}..\text{c}\,\text{b} & (\notin). \end{array}$$

In this example, the result of PSSC is a string whose projection to $\{\text{a},\text{b}\}$ contains an *ab* factor, which should be forbidden.

5.2 Complements of automata

To find the complement of a complete minimal DFA, simply invert the notion of acceptance. That is, map $\langle \delta, q_0, F\rangle$ to $\langle \delta, q_0, Q \setminus F\rangle$.

Theorem 3 *A regular stringset L and its complement can be defined by expressions over the same tier of salient symbols as one another.*

Proof Because L is regular, it can be represented as a complete minimal DFA, and this DFA will be associated with some set of salient symbols. The complement operation does not affect the transition function δ, so the set of self-loops in the result is exactly the same as that in the input. It follows then that the complement of L has the same set of salient symbols as L itself.

Then if the underlying class is closed under complementation (as is the case for LT and LTT) the corresponding relativized variant is so closed as well. Moreover, since a relativization is formed by merely adding self-loops everywhere, the order of relativization and complementation is immaterial. The two operations cannot interfere with one another.

5.3 Some non-closures

Having shown that, for fixed T, TLT^T and TLTT^T are closed under all Boolean operations and TSL^T and coTSL^T are closed under intersection and union, respectively, we now show that TSL^T is not closed under union or complement, and that coTSL^T is not closed under intersection or complement.

Consider two $\text{TSL}^{\{a,b\}}$ stringsets: one which bans the occurrence of "ab" on the projection to $\{a,b\}$, and another which bans the occurrence of "ba" on this projection. The union of these two stringsets allows the occurrence of either "ab" or "ba", but not both. Then the following is a violation of PSSC:

$$\begin{array}{ll} \text{ab}\overbrace{\text{b}..\text{b}}^{k-1} & (\in) \\ \text{b}..\text{b}\,\text{ba} & (\in) \\ \hline \text{ab}\,\text{b}..\text{b}\,\text{ba} & (\notin). \end{array}$$

The complement of the first of these, that "ab" must occur on the projection to $\{a,b\}$, is also not TSL. The following is a violation of PSSC:

$$\begin{array}{ll} \overbrace{\text{a}..\text{a}}^{k-1}\text{ab} & (\in) \\ \text{ab}\,\text{a}..\text{a} & (\in) \\ \hline \text{a}..\text{a} & (\notin). \end{array}$$

Now consider two $\text{coTSL}^{\{a,b\}}$ stringsets, one which requires that some "ab" occurs on the projection to $\{a,b\}$, and another that requires that some "ba" occurs on this projection. Their intersection (requiring both to occur) is not coTSL, as the following violates PIC:

$$\begin{array}{ll} w = \text{a}\overbrace{\text{b}..\text{b}}^{k-1}\text{a} & (\in) \\ S = \{\text{a}\,\text{b}..\text{b}, & \\ \quad \text{b}..\text{b}\,\text{a}\} & (\text{each } \notin), \end{array}$$

since the collective factorset of S is a superset of the factors of w. And of course, the complement of the first of these stringsets, banning occurrences of "ab" on the $\{a,b\}$-projection, is also not coTSL, because, as shown in section 3.2, the stringset of the projection is not coSL.

From this we have shown that TLTT^T and TLT^T are closed under all Boolean operations, while TSL^T and coTSL^T are closed only under intersection and union, respectively. We have also shown that intersection and union closures hold only when both stringsets have the same set of salient symbols.

6 Algebra

Many of the algorithms that decide whether a given DFA represents a stringset from a particular class actually make use of the *syntactic semigroup* associated with the

DFA. Given a complete minimal DFA $A = \langle \delta, q_0, F \rangle$, recall that $\delta\colon \Sigma \times Q \to Q$ can be viewed as $\hat{\delta}\colon \Sigma \to (Q \to Q)$ and define $\gamma\colon \Sigma^* \to (Q \to Q)$ as follows:

$$\gamma(w) \stackrel{\text{def}}{=} \begin{cases} \gamma(v) \circ \hat{\delta}(\sigma) & \text{if } w = \sigma v \text{ for some } \sigma \in \Sigma \text{ and } v \in \Sigma^* \\ \text{id} & \text{otherwise.} \end{cases}$$

Here, 'id' refers to the identity function. The syntactic semigroup is then the semigroup under flipped composition of the following functions:

$$S(A) \stackrel{\text{def}}{=} \{\gamma(w) : w \in \Sigma^+\}.$$

Then if $a = \gamma(u)$ and $b = \gamma(v)$, we have $ab = b \circ a = \gamma(v) \circ \gamma(u) = \gamma(uv)$. Since $\gamma(u)\gamma(v) = \gamma(uv)$, the semigroup operation is a homomorphism.

The star-free stringsets are those whose syntactic semigroup is finite and has no nontrivial subgroups (Pin 1997).

Recall from section 4 that the set of nonsalient symbols, $\complement T$, is the set of symbols that form self-loops on every state of a DFA. Translated to the algebraic domain, these are all and only those symbols σ for which $\gamma(\sigma) = \text{id}$. Sometimes we denote id by 1. If $1 \in S$, then we also say S is a monoid.

Lemma 1 *If S is the syntactic semigroup of a star-free stringset and $a, b \in S$ are elements such that $ab = 1$, then $a = b = 1$.*

Proof Suppose a and b are elements of S such that $ab = 1$, and that S is the syntactic semigroup of a star-free stringset. Then $1 = ab = (a(ab))b$, and by continuing this process we see that $a^n b^n = 1$ for all n. Schützenberger (1965) proves that because S is star-free, there exists some m such that $a^m = a^{m+1}$. Then we have $1 = a^m b^m = a^{m+1} b^m = a a^m b^m = a1 = a$, and by a similar argument we see that $b = 1$ as well. Therefore $a = b = 1$. □

This means that if S corresponds to a star-free stringset, then every w such that $\gamma(w) = \text{id}$ is composed entirely of symbols from $\complement T$. We now define the *projected subsemigroup* as

$$X(A) \stackrel{\text{def}}{=} \{\gamma(w) : w \in T^+\}.$$

Theorem 4 *If S corresponds to a star-free stringset, then the projected subsemigroup X of S is equal to $S \setminus \{1\}$.*

Proof Suppose $s \in S$. Then $s = \gamma(w)$ for some $w \in \Sigma^*$, and by definition if w is the string $\sigma_1 \ldots \sigma_n$ then $s = \gamma(\sigma_n) \circ \cdots \circ \gamma(\sigma_1)$.

Suppose $s \neq 1$. Because whenever $\sigma \in \complement T$ it holds that $\gamma(\sigma) = 1$, it follows that $\gamma(w) = \gamma\big(\pi_T(w)\big)$. By definition, if $|\pi_T(w)| \neq 0$ then $s \in X$ as well. If the length of this projection were 0, then $s = \gamma\big(\pi_T(w)\big)$ would be 1, and since by assumption $s \neq 1$, this cannot be. Therefore, $s \in X$.

On the other hand, suppose $s = 1$. Then by Lemma 1, we have $\{\sigma_1, \ldots, \sigma_n\} \subseteq \complement T$. Then $w = \sigma_1 \ldots \sigma_n$ is not in T^* and by definition is excluded from X.

Thus if S is star-free, X contains all and only the nonidentity elements of S. □

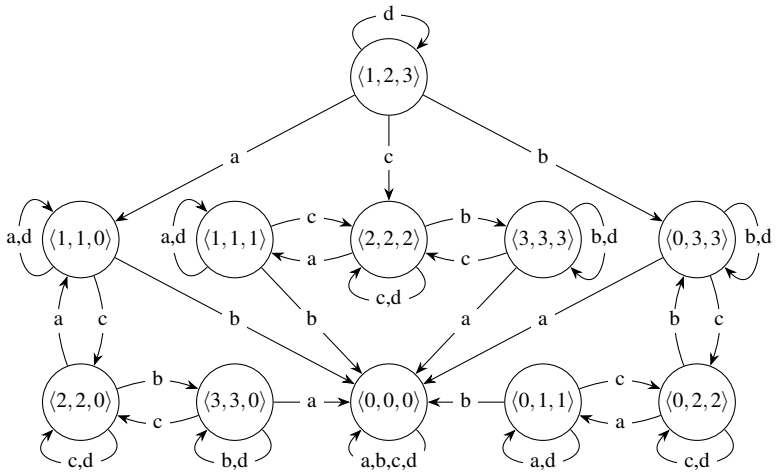


Fig. 9 The syntactic semigroup corresponding to the DFA of Figure 3. Each function is represented by the image of $\langle 1,2,3\rangle$. (Each tuple additionally has a fourth entry that is always labeled 0 and is omitted here for brevity.) The corresponding projected subsemigroup lacks the identity as well as every edge labeled d.

Because S and X are finite and generated by Σ or T, respectively, we can visualize them as edge-labeled directed graphs just like a DFA. If A is the DFA shown in Figure 3 on page 13 augmented with a nonaccepting sink labeled 0, then S is the semigroup shown in Figure 9.

6.1 Strictly local stringsets and their complements

The SL and coSL classes (and their relativizations) cannot be characterized algebraically solely on the basis of their syntactic semigroups. Because the syntactic semigroup is generated exclusively from the transition function δ of a DFA, an automaton and its complement are associated with the same semigroup. This fact has been noted previously by McNaughton (1974). However, if information describing whether each state is accepting or rejecting (state parity) is retained then an algebraic characterization is possible.

De Luca and Restivo (1980) provide a characterization based on Schützenberger's (1975) concept of a *constant*. Let A be a subset of some semigroup S, and let $c \in S$. Then c is a constant for A if for all $p,q,r,s \in S \cup 1$ it holds that whenever $pcq \in A$ and $rcs \in A$ it follows that $pcs \in A$. If S is freely generated from some finite alphabet, De Luca and Restivo prove that A is SL iff all sufficiently long words of S are constants for A. This is equivalent to the suffix-substitution closure discussed in section 3.1. Of course if all such words are instead constants for the complement of A, then A is coSL rather than SL.

In this case, A is the stringset being tested, the language generated by the syntactic semigroup under observation. If instead these properties hold for the projected subsemigroup, then the stringset is TSL or coTSL, respectively. But again, no algorithm

can distinguish a stringset and its complement given only an unadorned syntactic semigroup.

6.2 Locally testable stringsets

The characterization for LT and therefore TLT is due to Brzozowski and Simon (1973). First, note that an element x of a semigroup S is called *idempotent* iff $x = xx$. An *idempotent semigroup* is a semigroup such that all of its elements are idempotent. Given the syntactic semigroup S of a stringset L, Brzozowski and Simon proved that L is LT iff for all idempotent elements e of S it holds that the subsemigroup eSe is a commutative idempotent monoid. Note that the monoid requirement is immaterial, as eSe will always be a semigroup with identity. Namely, for any $s \in S$, we see that $eee \cdot ese = (ee)(ee)se = eese = ese$ and similarly $ese \cdot eee = ese$, so eee (which is itself simply e) is the identity.

Theorem 5 *A stringset L is* TLT *iff for all idempotent elements e of X, the projected subsemigroup of its syntactic semigroup, it is the case that eXe is a commutative idempotent semigroup.*

This holds because the projected subsemigroup is equivalent to the syntactic semigroup of the projection, and a stringset is TLT iff its projection is LT by definition.

6.3 Locally threshold testable stringsets

The characterization for LTT and therefore TLTT is due to Beauquier and Pin (1989). They define a variety (collection) of semigroups $\mathbb{V}$ to contain all and only those aperiodic semigroups S such that if e and f are idempotent, $p = es_1f$, $q = fs_2e$, and $r = es_3f$, it holds that $pqr = rqp$. They then prove that a language L is LTT iff its syntactic semigroup S is a member of $\mathbb{V}$. By expansion, this means that for all idempotents e, f of S and for all $s_1, s_2, s_3 \in S$, it holds that $es_1fs_2es_3f = es_3fs_2es_1f$.

The aperiodicity requirement is meaningful, as there do exist stringsets that satisfy the latter requirement without even being star-free. Consider the set of strings of even length over a unary alphabet, $(aa)^*$. Its syntactic semigroup consists of two elements, 1 and a, such that $aa = 1$. It necessarily holds that $es_1fs_2es_3f = es_3fs_2es_1f$ under any instantiation of these variables, because this is a commutative monoid.

Theorem 6 *Let L be a stringset and X be the projected subsemigroup of its syntactic semigroup. Then L is* TLTT *iff for all idempotent elements e, f of X, and for all s_1, s_2, s_3 in X, it holds that $es_1fs_2es_3f = es_3fs_2es_1f$.*

This holds because the projected subsemigroup is equivalent to the syntactic semigroup of the projection, and a stringset is TLTT iff its projection is LTT by definition.

7 Conclusion

We introduced several new classes to the piecewise-local subregular hierarchy building from the model-theoretic characterization of TSL by Lambert and Rogers (2020) and noting that projective relativization does not affect the expressivity of classes based on general precedence. We provided model-, language-, and automata-theoretic as well as algebraic characterizations for each new class. This establishes a clearer notion of relativized adjacency based on the salience of individual symbols, and informs linguistic theory as it pertains to long-distance dependencies in phonology.

The topic of learning is not covered in this work. But we would be remiss to ignore that the unifying concept of relativized adjacency provides a straightforward mechanism to extend known learning results for the TSL class to TLT and TLTT (see McMullin 2016; Jardine and Heinz 2016; Jardine and McMullin 2017).

A Haskell library[4] containing the automata-theoretic and algebraic decision algorithms has been created. With this one can construct regular and subregular stringsets from factors, or import OpenFST-format automata, and determine which, if any, subregular classes the result occupies. The factor-based constructors make use of the alphabet-agnostic automata described in section 4.2.

The class formed by intersecting multiple TSL constraints over different tier alphabets, MTSL (De Santo and Graf 2019), is not explored here. Under this model-theoretic view, the relativized successor relation $\lhd^T$ is parameterized by the alphabet over which it is relativized, so different instantiations of this relation are as different as the $\lhd$ and $<$ relations. With this in mind, future work in understanding these interactions will shed light on the strictly piecewise-local class discussed in Rogers and Lambert (2019b), and vice-versa. Moreover, Aksënova and Deshmukh (2018) discuss attested interactions of multiple tiers; one might ask which kinds of combinations (if any) allow intersection-closure to be maintained. Counterexamples exist for each type of interaction (disjoint, overlapping, and subset configurations), but perhaps some smaller portion of the constraint space may combine more readily.

The characterizations provided here of the relativized variants of the subregular classes rely heavily on the fact that $\lhd^T$ and $<^T$ are the results of a projective relativization. Another direction for future work then would be accounting for arbitrary nonprojective relations, which would improve our understanding of De Santo and Graf's structure-sensitive TSL class and its (threshold) testable extensions, or Graf's (2017) domain- and interval-based strictly piecewise classes.

Finally, some of the subregular classes have been explored in application to trees (Rogers 1997), and some extended to transducers (Chandlee 2014; Ji and Heinz 2020). It would be interesting to see how relativization applies to these cases as well.

Conflict of interest

The author has no conflicts of interest to declare that are relevant to the content of this article.

[4] Software available at `https://github.com/vvulpes0/Language-Toolkit-2`

References

Aho, A. V., Garey, M. R., Ullman, J. D. (1972). The transitive reduction of a directed graph. *SIAM Journal on Computing* 1(2):131–137, https://doi.org/10.1137/0201008

Aksënova, A., Deshmukh, S. (2018). Formal restrictions on multiple tiers. In *Proceedings of the Society for Computation in Linguistics*, Salt Lake City, Utah, vol 1, (pp. 64–73), https://doi.org/10.7275/R5K64G8S

Beauquier, D., Pin, J.-E. (1989). Factors of words. In G. Ausiello, M. Dezani-Ciancaglini, S. Ronchi Della Rocca (Eds.) *Automata, Languages and Programming: 16th International Colloquium*, Lecture Notes in Computer Science, vol 372, Springer Berlin / Heidelberg (pp. 63–79), https://doi.org/10.1007/BFb0035752

Beesley, K. R., Karttunen, L. (2003). *Finite State Morphology*. CSLI Publications

Brzozowski, J. A., Simon, I. (1973). Characterizations of locally testable events. *Discrete Mathematics* 4(3):243–271, https://doi.org/10.1016/S0012-365X(73)80005-6

Caron, P. (1998). LANGAGE: A Maple package for automaton characterization of regular languages. In D. Wood, S. Yu (Eds.) *Automata Implementation*, Lecture Notes in Computer Science, vol 1436, Springer Berlin / Heidelberg (pp. 46–55), https://doi.org/10.1007/BFb0031380

Chandlee, J. (2014). Strictly local phonological processes. PhD thesis, University of Delaware, https://chandlee.sites.haverford.edu/wp-content/uploads/2015/05/Chandlee_dissertation_2014.pdf

Cser, A. (2010). The -alis/-aris allomorphy revisited. In F. Rainer, W. Dressler, D. Kastovsky, H. C. Luschützky (Eds.) *Variation and Change in Morphology: Selected Papers from the 13th International Morphology Meeting*, John Benjamins Publishing Company, Vienna, Austria (pp. 33–52), https://doi.org/10.1075/cilt.310.02cse

De Luca, A., Restivo, A. (1980). A characterization of strictly locally testable languages and its application to subsemigroups of a free semigroup. *Information and Control* 44(3):300–319, https://doi.org/10.1016/S0019-9958(80)90180-1

De Santo, A., Graf, T. (2019). Structure sensitive tier projection: Applications and formal properties. In R. Bernardi, G. Kobele, S. Pogodalla (Eds.) *Formal Grammar 2019*, Lecture Notes in Computer Science, vol 11668, Springer Verlag (pp. 35–50), https://doi.org/10.1007/978-3-662-59648-7_3

Edlefsen, M., Leeman, D., Myers, N., Smith, N., Visscher, M., Wellcome, D. (2008). Deciding strictly local (SL) languages. In J. Breitenbucher (Ed.) *Proceedings of the 2008 Midstates Conference for Undergraduate Research in Computer Science and Mathematics*, (pp. 66–73)

Gold, E. M. (1967). Language identification in the limit. *Information and Control* 10(5):447–474, https://doi.org/10.1016/S0019-9958(67)91165-5

Graf, T. (2017). The power of locality domains in phonology. *Phonology* 34(2):385–405, https://doi.org/10.1017/S0952675717000197

Haines, L. H. (1969). On free monoids partially ordered by embedding. *Journal of Combinatorial Theory* 6(1):94–98, https://doi.org/10.1016/s0021-9800(69)80111-0

Heinz, J., Rawal, C., Tanner, H. G. (2011). Tier-based strictly local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Short Papers*, Association for Computational Linguistics, Portland, Oregon, vol 2, (pp. 58–64), https://aclweb.org/anthology/P11-2011

Hulden, M. (2009). Finite-state machine construction methods and algorithms for phonology and morphology. PhD thesis, The University of Arizona

Jardine, A., Heinz, J. (2016). Learning tier-based strictly 2-local languages. *Transactions of the Association for Computation in Linguistics* 4:87–98, https://doi.org/10.1162/tacl_a_00085

Jardine, A., McMullin, K. (2017). Efficient learning of tier-based strictly k-local languages. In F. Drewes, C. Martín-Vide, B. Truthe (Eds.) *Language and Automata Theory and Applications: 11th International Conference*, Lecture Notes in Computer Science, vol 10168, Springer, Cham (pp. 64–76), https://doi.org/10.1007/978-3-319-53733-7_4

Ji, J., Heinz, J. (2020). Input strictly local tree transducers. In A. Leporati, C. Martín-Vide, D. Shapira, C. Zandron (Eds.) *Language and Automata Theory and Applications: Proceedings of the 14th International Conference, LATA 2020*, Springer International Publishing, Cham, Switzerland, Theoretical Computer Science and General Issues, vol 12038, (pp. 369–381), https://doi.org/10.1007/978-3-030-40608-0_26

Knuth, D. E., Morris, J. H., Pratt, V. R. (1977). Fast pattern matching in strings. *SIAM Journal on Computing* 6(2):323–350, https://doi.org/10.1137/0206024

Lambert, D., Rogers, J. (2020). Tier-based strictly local stringsets: Perspectives from model and automata theory. In *Proceedings of the Society for Computation in Linguistics*, New Orleans, Louisiana, vol 3, (pp. 330–337), https://doi.org/10.7275/2n1j-pj39

Libkin, L. (2004). *Elements of Finite Model Theory*. Texts in Theoretical Computer Science, Springer Berlin / Heidelberg, https://doi.org/10.1007/978-3-662-07003-1

McMullin, K. (2016). Tier-based locality in long-distance phonotactics: Learnability and typology. PhD thesis, University of British Columbia

McNaughton, R. (1974). Algebraic decision procedures for local testability. *Mathematical Systems Theory* 8(1):60–76, https://doi.org/10.1007/bf01761708

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

Nerode, A. (1958). Linear automaton transformations. In *Proceedings of the American Mathematical Society*, American Mathematical Society, vol 9, (pp. 541–544), https://doi.org/https://doi.org/10.1090/s0002-9939-1958-0135681-9

Pin, J.-E. (1997). Syntactic semigroups. In G. Rozenberg, A. Salomaa (Eds.) *Handbook of Formal Languages: Volume 1 Word, Language, Grammar*, Springer-Verlag, Berlin (pp. 679–746), https://doi.org/10.1007/978-3-642-59136-5_10

Rogers, J. (1996). A model-theoretic framework for theories of syntax. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, Association for Computational Linguistics, Santa Cruz, CA, (pp. 10–16), https://doi.org/10.3115/981863.981865

Rogers, J. (1997). Strict LT_2 : regular :: local : recognizable. In C. Retoré (Ed.) *Logical Aspects of Computational Linguistics: First International Conference, LACL '96 (Selected Papers)*, Springer-Verlag, Berlin, Lecture Notes in Computer Science, vol 1328, (pp. 366–385), https://doi.org/10.1007/BFb0052167

Rogers, J. (1998). *A Descriptive Approach to Language-Theoretic Complexity*. (Monograph.) Studies in Logic, Language, and Information, CSLI Publications

Rogers, J., Lambert, D. (2019a). Extracting Subregular constraints from Regular stringsets. *Journal of Language Modelling* 7(2):143–176, https://doi.org/10.15398/jlm.v7i2.209

Rogers, J., Lambert, D. (2019b). Some classes of sets of structures definable without quantifiers. In *Proceedings of the 16th Meeting on the Mathematics of Language*, Association for Computational Linguistics, Toronto, Canada, (pp. 63–77), https://doi.org/10.18653/v1/W19-5706

Rogers, J., Pullum, G. K. (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information* 20(3):329–342, https://doi.org/10.1007/s10849-011-9140-2

Rogers, J., Heinz, J., Bailey, G., Edlefsen, M., Visscher, M., Wellcome, D., Wibel, S. (2010). On languages piecewise testable in the strict sense. In C. Ebert, G. Jäger, J. Michaelis (Eds.) *The Mathematics of Language: Revised Selected Papers from the 10th and 11th Biennial Conference on the Mathematics of Language*, LNCS/LNAI, vol 6149, FoLLI/Springer (pp. 255–265), https://doi.org/10.1007/978-3-642-14322-9_19

Schützenberger, M.-P. (1965). On finite monoids having only trivial subgroups. *Information and Control* 8(2):190–194, https://doi.org/10.1016/s0019-9958(65)90108-7

Schützenberger, M.-P. (1975). Sur certaines opérations de fermeture dans les langages rationnels. *Symposia Mathematica* 15:245–253

Simon, I. (1975). Piecewise testable events. In H. Brakhage (Ed.) *Automata Theory and Formal Languages*, Lecture Notes in Computer Science, vol 33, Springer-Verlag, Berlin (pp. 214–222), https://doi.org/10.1007/3-540-07407-4_23