Very efficient learning of structured classes of subsequential functions from positive data

Adam Jardine (Delaware)

Jane Chandlee (Delaware)

Rémi Eyraud (Marseilles)

Jeffrey Heinz (Delaware)

The 12th International Conference of Grammatical Inference

University of Kyoto, Japan

September 18, 2014

The researchers from Delaware acknowledge support from NSF#1035577.

This paper

1. We present the Structured Onward Subsequential Function Inference Algorithm (SOSFIA), which identifies proper subclasses of subsequntial functions in linear time and data.

2. The key to this result is *a priori* knowledge regarding the *common structure* shared by every function in the class.

3. At least one of these classes appears to be quite natural. The Input Strictly Local class of functions adapts the notion of Strictly Local stringsets [MP71] to mappings [Cha14, CEH14].

4. Demonstrations in phonology and morphology where such structural knowledge plausibly exists *a priori*.

Part 1: Background

1. Longest Common Prefix
2. Subsequential transducers
3. Subsequential functions
4. Onwardness
5. OSTIA, OSTIA-D, OSTIA-R

Longest Common Prefix

1. Let $\texttt{sh\_pref}(S)$ denote the *shared prefixes* of a stringset S.

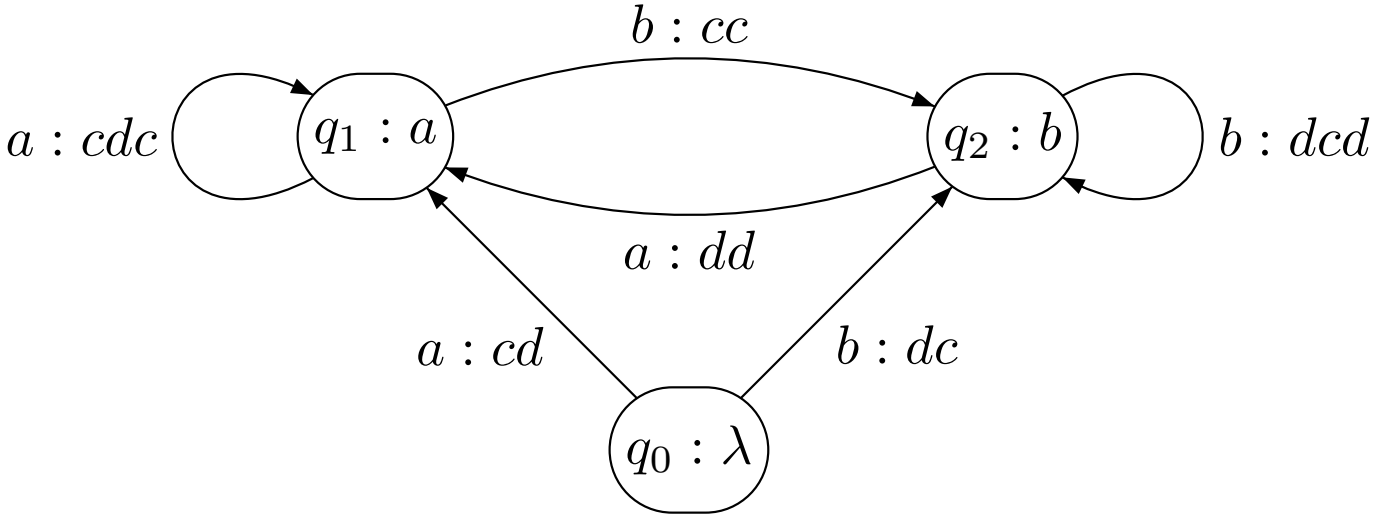
$$\texttt{sh\_pref}(S) = \Big\{ u \mid (\forall s \in S)(\exists v \in \Sigma^*)[s = uv] \Big\}$$

2. The *longest common prefix* ($\texttt{lcp}$) of a stringset S is

$$\begin{aligned} \texttt{lcp}(S) \quad = \quad & w \in \{u \in \texttt{sh\_pref}(S)\} \text{ and} \\ & \Big(\forall u' \in \texttt{sh\_pref}(S)\Big) \Big[|w| \geq |u'| \Big] \end{aligned}$$

We set the $\texttt{lcp}(\varnothing) = \lambda$.

Subsequential Finite State Transducers (SFSTs)



Informally, SFSTs are weighted deterministic transducers where the strings are weights and multiplication is concatenation.

$$t(aba) = cdccdda \text{ because } \begin{array}{ccccccccc} & & a & & b & & a & & \\ q_0 & \to & q_1 & \to & q_2 & \to & q_1 & \to & \\ & & cd & & cc & & dd & & a \end{array}$$

Subsequential functions

1. The *tails* of $w \in \Sigma^*$ with respect to $t : \Sigma^* \to \Delta^*$ is

$$\texttt{tails}_t(w) = \Big\{(x, v) \mid t(wx) = uv \text{ and } u = \texttt{lcp}(t(w\Sigma^*))\Big\}.$$

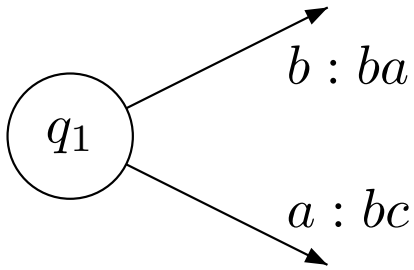
2. If $\texttt{tails}_t(w) = \texttt{tails}_t(w')$ then w, w' are *tail-equivalent* with respect to t, written $w \sim_t w'$.

3. A function $t : \Sigma^* \to \Delta^*$is *subsequential* if $\sim_t$ partitions Σ^* into *finitely* many blocks.

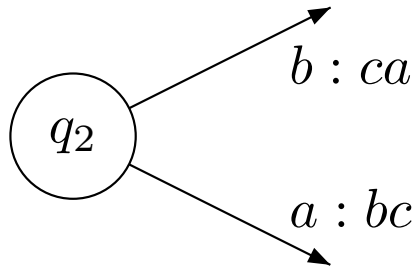
Onwardness

Informally, a SFST τ is *onward* if the longest common prefix of the outgoing transitions of each noninitial state is the empty string.

$$onward(\tau) \stackrel{\text{def}}{=} \big(\forall q \in (Q - q_0)\big) \Big[\texttt{lcp}\Big\{w \in \Sigma^* | (\exists a \in \Sigma, r \in Q)[(q, a, w, r) \in \delta]\Big\} = \lambda\Big]$$



Not Onward



Onward

OSTIA

Theorem 1 ([OG91]) *Every subsequential function has a canonical form given by an onward subsequential transducer.*

Theorem 2 ([OGV93]) *Total subsequential functions are identifiable in the limit from positive data in cubic time.*

- An interesting corollary is that *partial* subsequential functions are identifiable in this weak sense:

  If t is the target function and h is the hypothesis OSTIA returns, then, for all w where $t(w)$ is defined, it is the case that $h(w) = t(w)$. But if t is not defined on w, h may be!

OSTIA-D and OSTIA-R [OV96, CVVO98]

1. OSTIA-D assumes a priori knowledge of the *domain* of the target function, given as a DFA.
2. OSTIA-R assumes a priori knowledge of the *range* of the target function, given as a DFA.
3. Both *add* steps and checks to OSTIA's state-merging procedures to ensure that the merges are consistent with the domain and range DFA, respectively.
4. Therefore, their time complexity is at least cubic.

Our result, in contrast

1. Our result is most like OSTIA-D. As you will see, the a priori knowledge we consider structures the *domain*.
2. However, we show both *linear* time *and data* complexity in the sense of de la Higuera (1997).
3. This is possible because if the structure is known, there is no reason to merge states at all!

Part 2: Theoretical Results

1. Delimited SFSTs
2. Output-empty subsequential transducers
3. `min_change`
4. SOSFIA
5. Strong learning in polynomial time and data
6. Theorems and proofs

Delimited SFSTs (DSFSTs)

A DSFST $\tau = \langle Q, q_0, q_f, \Sigma, \Delta, \delta \rangle$ where

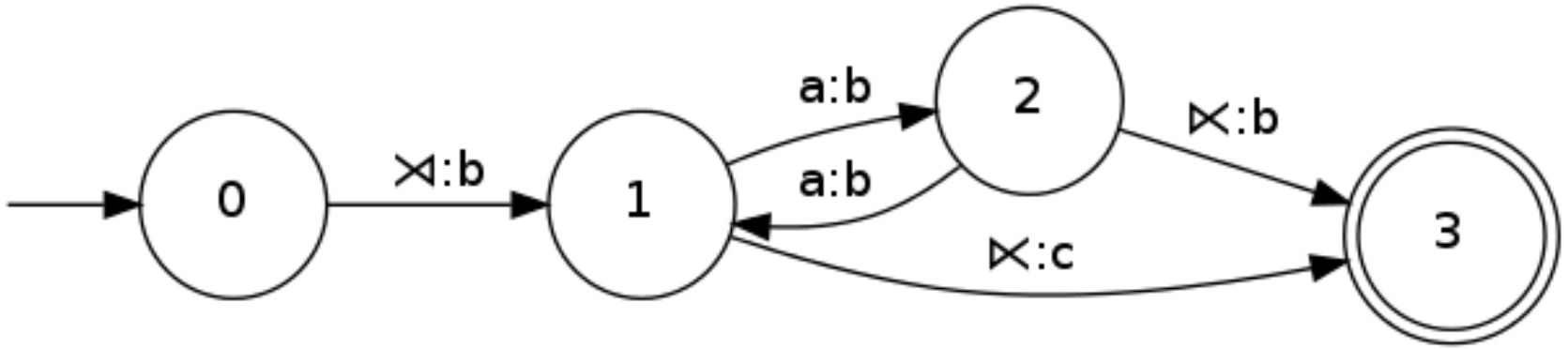
1. Q is a finite set of states
2. $q_0, q_f \in Q$ are the initial and final states, respectively
3. Σ and Δ are finite alphabets of symbols
4. $\delta \subseteq Q \times (\Sigma \cup \{\rtimes, \ltimes\}) \times \Delta^* \times Q$ is the transition function where

   $\rtimes, \ltimes \notin \Sigma$ are special symbols indicating 'start of the input' and 'end of the input', respectively.
5. q_0 has no incoming transitions and exactly one outgoing transition whose input label is $\rtimes$ which leads to a non-final state; and
6. q_f has no outgoing transitions and every incoming transition has input label $\ltimes$; and
7. It is deterministic on the input

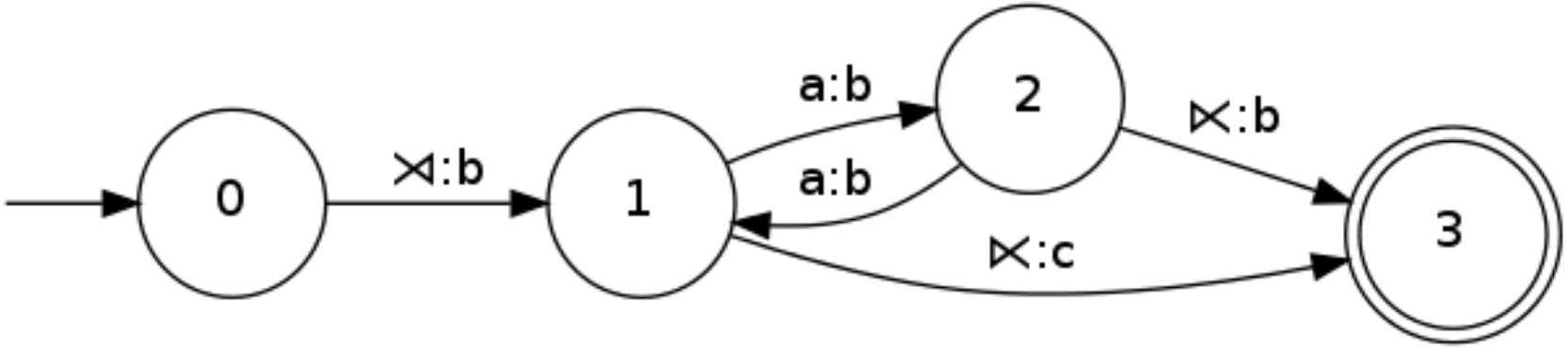
Functions recognizable by DSFSTs

The function recognized by a DSFST τ is

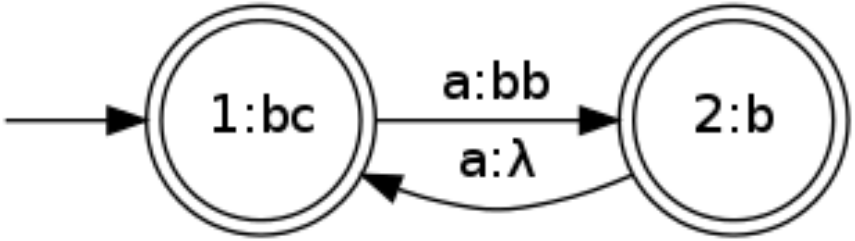
$$R(\tau) = \left\{(w, v) \mid (q_0, \rtimes w \ltimes, v, q_f) \in \delta^*\right\}$$



Comparison to typical SFSTs



The DSFST from the previous slide



A SFST from [OG91] recognizing the same function.

Theorems about DSFSTs

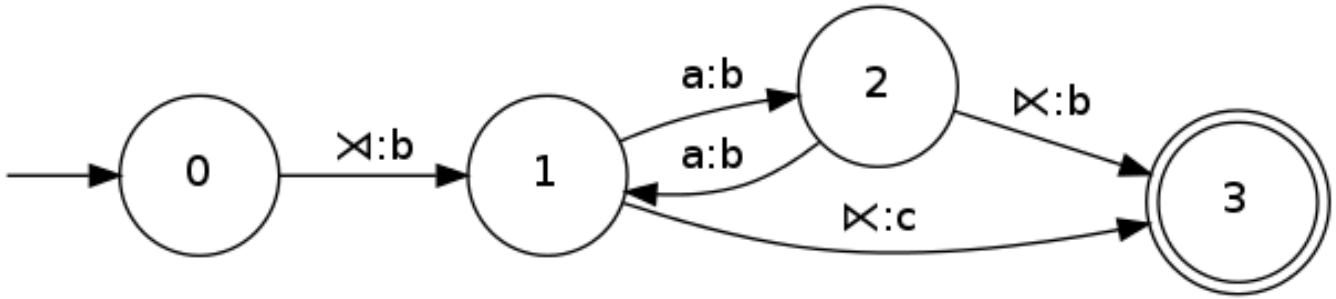
Theorem 3 (Co-incidence with Subsequential Functions) *The class of subsequential functions and the class of functions representable with DSFSTs coincide exactly.*

Theorem 4 (Canonical DSFSTs) *For every subsequential function t, there is a unique, smallest, onward DSFST representing it.*

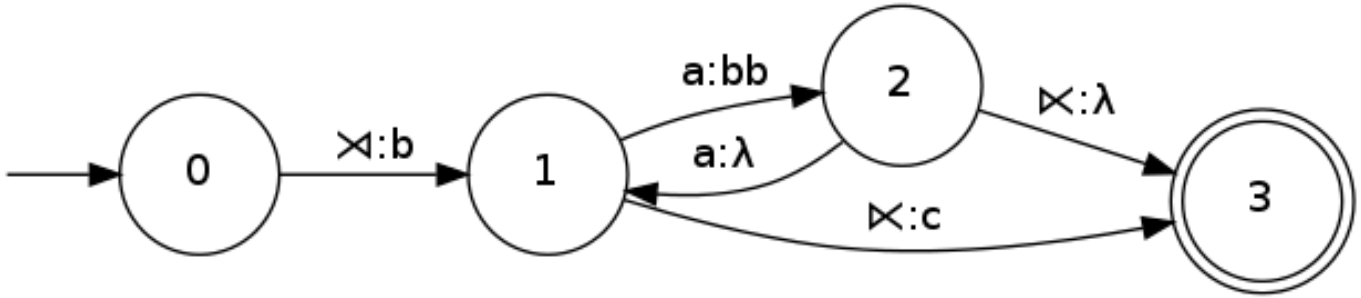
Theorem 5 (Structure Preserving Onward Transformations) *Every DSFST can be made onward* only by *changing the output transitions; the rest of the structure is preserved.*

Example of how to make DSFSTs onward

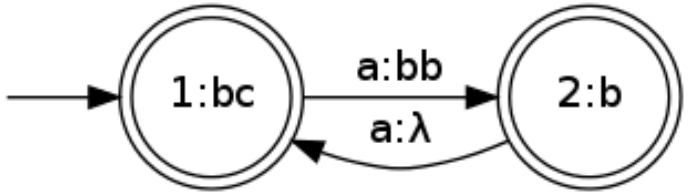
The proof of the last theorem makes use of a function $\texttt{push\_lcp}(\tau, q)$ which returns a transducer τ' in which the longest common prefix of the outputs of the transitions leaving q is pushed as a suffix onto the outputs of the transitions entering q (if they exist).



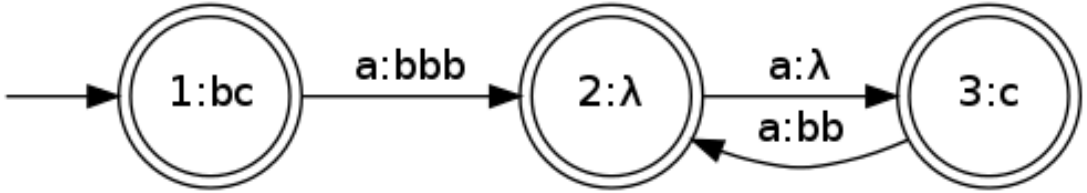
The DSFST from before (above) and its onward version (below)



In contrast, standard SFSTS may have to add an initial state [OG91]



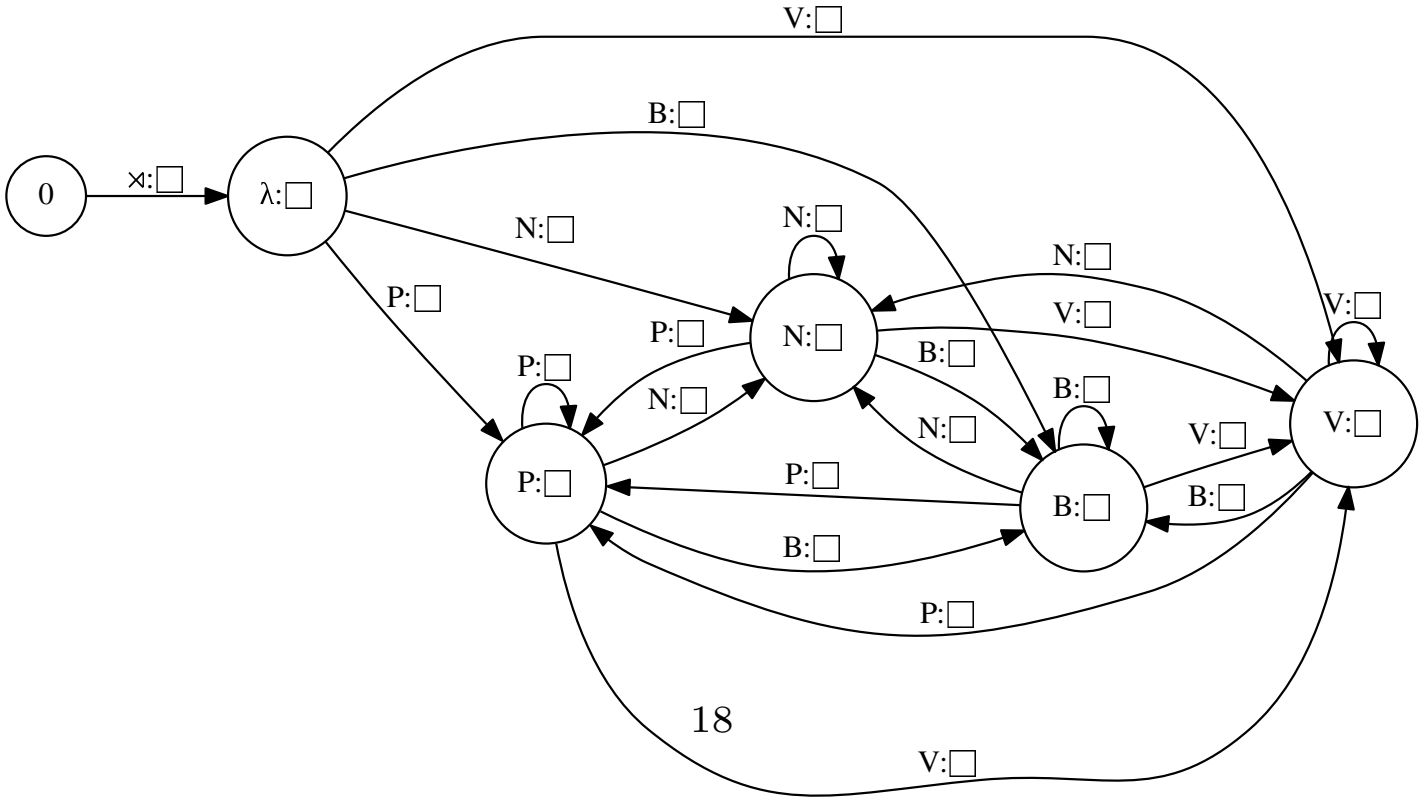
The standard SFST from before



A canonical standard SFST recognizing the same function

Target classes and Output-Empty DSFSTs

1. A DSFST is *output-empty* if all of its transition outputs are blanks (□).
2. An output-empty transducer $\tau_{\square}$ defines a *class* of functions $\mathcal{T}$ which is exactly the set of functions which can be created by taking the states and transitions of $\tau_{\square}$ and replacing the blanks with output strings, maintaining onwardness.



SOSFIA Overview

1. The input to SOSFIA is an output-empty transducer $\tau_{\square}$ and a finite sample $S \subset \Sigma^* \times \Delta^*$ generated from one of the functions in $\mathcal{T}_{\tau_{\square}}$.
2. SOSFIA iterates through the states of $\tau_{\square}$. At each state, it sets the output of each outgoing transition to be the *minimal change in the output* generated by this transition, according to S.

Min Change (`min_change`)

1. The *common output* of an input prefix w in a sample $S \subset \Sigma^* \times \Delta^*$ for t is the `lcp` of all $t(wv)$ that are in S:

$$\texttt{common\_out}_S(w) = \texttt{lcp}\Big(\{u \in \Delta^* \mid \exists v \text{ s.t. } (wv, u) \in S\}\Big)$$

2. The minimal change in the output is then simply the difference between the common outputs of w and $w\sigma$.

3. The *minimal change in the output* in $S \subset \Sigma^* \times \Delta^*$ from w to $w\sigma$ is:

$$\texttt{min\_change}_S(\sigma, w) =$$

$$\begin{cases} \texttt{common\_out}_S(\sigma) & \text{if } w = \lambda \\ \texttt{common\_out}_S(w)^{-1}\texttt{common\_out}_S(w\sigma) & \text{otherwise} \end{cases}$$

Example illustrating `min_change`

If

$$S = \left\{ \begin{array}{lcllcl} (anpa & , & ama), & & (anpo & , & amo), \\ (ana & , & ana), & & (ano & , & ano), \\ (anda & , & anda), & & (ando & , & ando) \end{array} \right\}$$

Then

1. $\texttt{common\_out}_S(a) = a$
2. $\texttt{common\_out}_S(an) = a$
3. $\texttt{min\_change}_S(n, a) = \lambda$
4. $\texttt{min\_change}(p, an) = m$
5. $\texttt{min\_change}(a, an) = na$
6. $\texttt{min\_change}(d, an) = nd$

SOSFIA

- `min_change` gives us exactly the output needed to maintain onwardness, which will in turn guarantee that the SOSFIA converges to the correct function, provided that the sample contains enough information. Note that the minimal change is calculable for S because it is finite.
- SOSFIA proceeds through the states of the output-empty transducer in lexicographic order.
  1. Each state q is associated with the shortest string w which leads to it.
  2. For each transition $(q, a, \square, r) \in \delta$, SOSFIA sets the output label of this transition to $\texttt{min\_change}(a, w)$.

The Learning Paradigm [dlH97, ?]

Let $\mathbb{T}$ be a class of functions and $\mathbb{R}$ a class of representations for $\mathbb{T}$.

Definition 1 (Strong characteristic sample) *For a* $(\mathbb{T}, \mathbb{R})$*-learning algorithm* $\mathfrak{A}$*, a sample* CS *is a* strong characteristic sample *of a representation* $r \in \mathbb{R}$ *if for all samples* S *for* $\mathcal{L}(r)$ *such that* $CS \subseteq S$*,* $\mathfrak{A}$ *returns* r*.*

Definition 2 (Strong identification in polynomial time and data) *A class* $\mathbb{T}$ *of functions is strongly identifiable in polynomial time and data if there exists a* $(\mathbb{T}, \mathbb{R})$*-learning algorithm* $\mathfrak{A}$ *and two polynomials* $p()$ *and* $q()$ *such that:*

1. *For any sample* S *of size* m *for* $t \in \mathbb{R}$*,* $\mathfrak{A}$ *returns a hypothesis* $r \in \mathbb{R}$ *in* $\mathcal{O}(p(m))$ *time.*
2. *For each representation* $r \in \mathbb{R}$ *of size* k*, there exists a strong characteristic sample of* r *for* $\mathfrak{A}$ *of size at most* $\mathcal{O}(q(k))$*.*

Main Result

Theorem 6 *For every output-empty transducer $\tau_\square$, the SOSFIA strongly identifies $\mathcal{T}_{\tau_\square}$ in linear time and data.*

Notes:

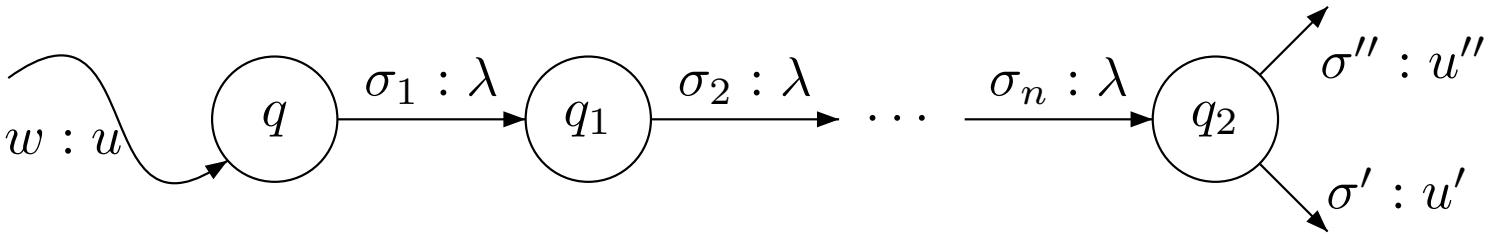
1. The size of a sample S is the sum of the length of the strings it is composed of: $|S| = \sum_{(w,v)\in S} |w| + |v|$.
2. The size of a DSFST $\tau = \langle Q, q_0, q_f, \Sigma, \Delta, \delta \rangle$ is $|\tau| = |Q| + \sum_{(q,\sigma,u,q')\in\delta} |u|$.

Proof Sketch

1. A strong characteristic sample exists by essentially including for each state q (reachable by a shortest prefix $w \in \Sigma^*$) and each $\sigma \in \Sigma$ the pairs

$$\Big(w, t(w)\Big), \Big(w\sigma, t(w\sigma)\Big) \in S$$

and for all situations like this:



$$\big(w\sigma_1 \cdots \sigma_n\sigma', uu'\big), \big(w\sigma_1 \cdots \sigma_n\sigma'', uu''\big) \in S$$

Proof sketch (con't)

2. For a target DSFST τ, the data complexity is $O(|\tau|)$.

   Each of these string pairs can have a left projection of length at most $|Q|$ and a right projection of length at most $\sum_{(q,\sigma,u,q')\in\delta} |u|$, which yields $O(|Q| \cdot (|Q| + |\tau|)) = O(|\tau|)$ since $|Q|$ is a constant for each target class.

3. The time complexity is $O(n \cdot m)$ where $|S| = n$ and m equals the length of the longest string in the right projection of S.

   In the worst case, the algorithm launches `min_change` for each transition, which corresponds to the computation of two `lcp`. Each of these calculations is doable in $O(n \cdot m)$. Each state is considered exactly once, as is each transition, but since $|Q|$ and $card(\delta)$ are fixed, the time complexity depends only on n and m.

Part 3: Demonstrations

1. Input Strictly Local Phonological Transformations
2. Long-distance Phonological Transformations
3. Morpheme Identification (PF/SF Mappings)

Input Strictly Local Functions (Chandlee 2014)

Definition 3 (Input Strictly Local Function, [Cha14, CEH14]) *A function f is Input Strictly Local (ISL) if there is a k such that for all $w_1, w_2 \in \Sigma^*$, if $\texttt{suff}^{k-1}(w_1) = \texttt{suff}^{k-1}(w_2)$ then $\texttt{tails}_f(w_1) = \texttt{tails}_f(w_2)$.*

1. ISL functions are Markovian: the output written upon reading σ depends only on the previous $k-1$ input symbols (cf. Strictly Local stringsets [MP71]).
2. **Lemma:** ISL functions are a proper subclass of subsequential functions.
3. **Theorem:** For each k, there is a unique empty-output DSFST $\tau_\square$ such that $\mathcal{T}_{\tau_\square}$ coincides exactly with the class of ISL_k functions.

Phonology

The foundational hypothesis at the center of modern generative phonology is that there is a phonological mapping from abstract, lexical 'underlying' representations of words and morphemes to their concrete surface pronunciations.

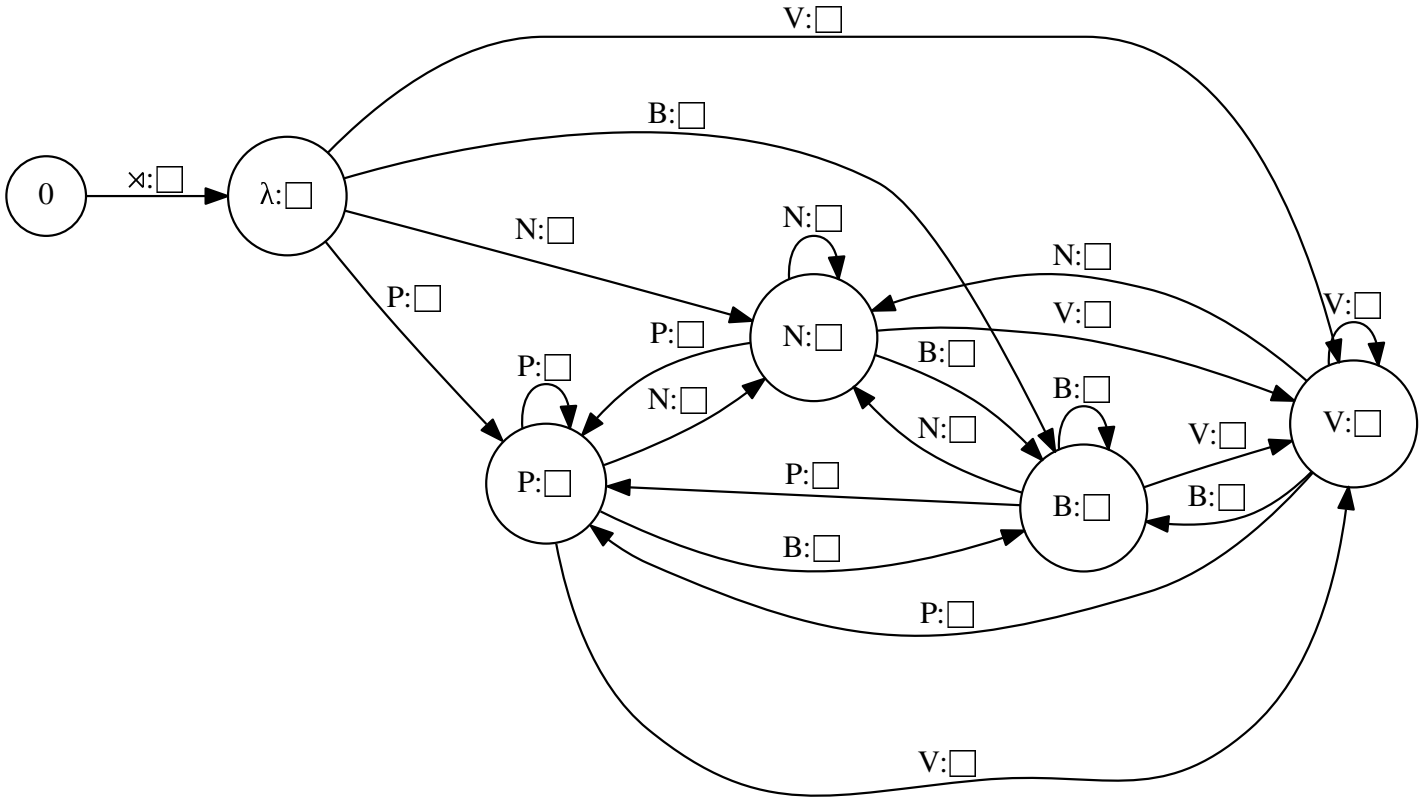
Of all the logically possible mappings, what kind are the humanly possible phonological ones?

Strictly Local Phonological Mappings

1. Chandlee (2014) establishes that phonological mappings which are not long-distance nor 'iterative spreading' can be modeled with ISL functions.
2. In a database of over 4000 phonological processes (P-base, Mielke 2008), she shows at least 94% are ISL.
3. These include substitution (letter-change), epenthesis (insertion), deletion, and bounded metathesis (letters change positions).
4. In phonological terms: rewrite rules of the form $CAD \rightarrow CBD$ where CAD is a *finite* stringset, which apply simultaneously.
5. Much stronger computational characterization than the one suggested by Kaplan and Kay (1994)

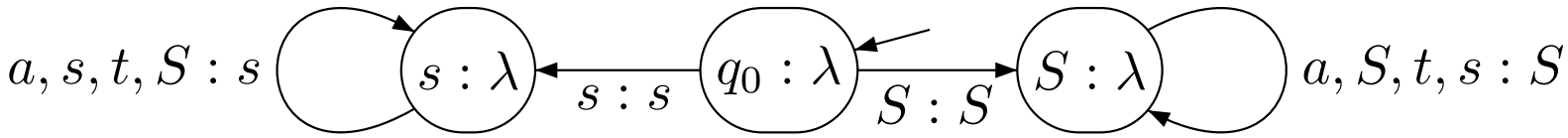
Demonstration #1: *NC̥ Repair Typology

1. Fusion (Indonesian): /məŋ+pilih/ ↦ [məmilih], ‘to choose’
2. Voicing (Quechua): /kam+pa/ ↦ [kamba], ‘yours’
3. Denasalization (Toba Batak):
   /maŋinum tuak/ ↦ [maŋinup tuak], ‘drink palm wine’



Long-distance phonology

1. Sibilant Harmony (Samala)
   /hasxintilawaʃ/ ↦ [haʃxintilawaʃ] ‘his former gentile name’
2. Provided the right structure is known a priori, classes of functions which can model long-distance phonology can be learned.
3. We conjecture there is a class of mappings that is to the Strictly Piecewise stringsets what ISL functions are to the SL stringsets, perhaps in a compact, factorized manner [RHB+10, HR13] and which thus have a common underlying structure.
4. For proof-of-concept, we demonstrated learning consonantal harmony using a particular transducer.



Matching Semantic Forms with Phonetic Forms (Morpheme Identification)

Verbal constructions in Swahili:

ni +	me +	ni +	penda
1-acc	perf	1st-nom	like

'I have liked myself'

Input Data to SOSFIA

⟨`1-nom`⟩⟨`perf`⟩⟨`1-acc`⟩⟨`like`⟩, nimenipenda

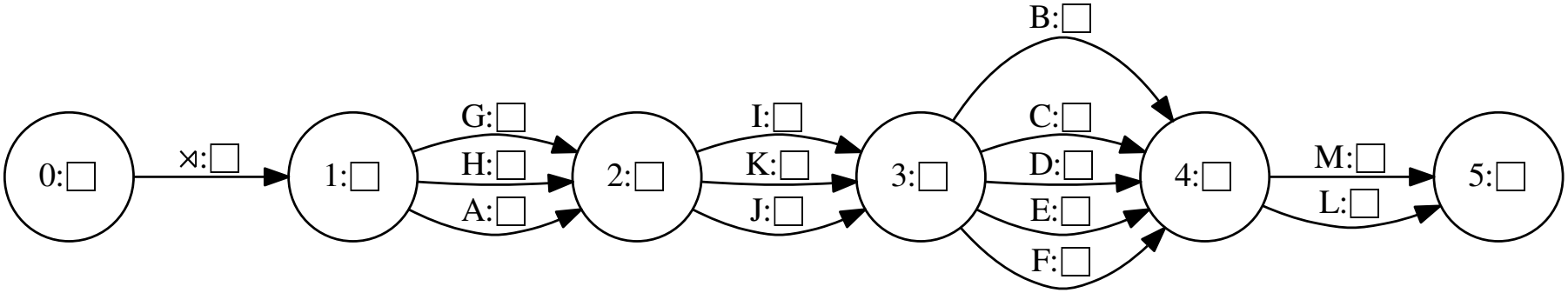
⟨`3-nom`⟩⟨`pres`⟩⟨`1-acc`⟩⟨`like`⟩, ananipenda

⟨`2-nom`⟩⟨`perf`⟩⟨`1-pl-acc`⟩⟨`beat`⟩, umetupiga

...

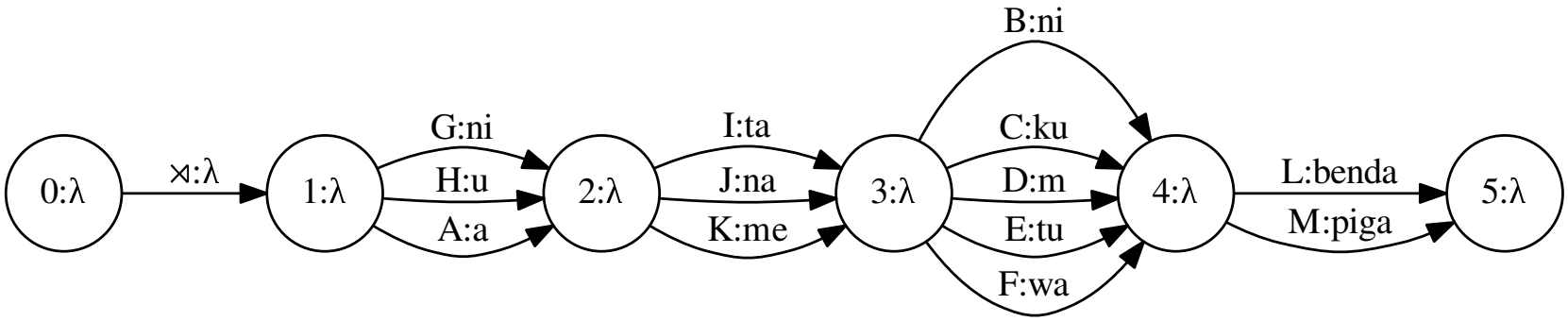
Morpheme Identification 2

Structure given to SOSFIA



Morpheme Identification 3

Structure given to SOSFIA



The point is that once the structure is known, onwardness provides a principled way to align the elements which make up the input and output strings.

Future Work

- Develop ways to address the issue when samples don't include every shortest prefix...
- Develop a similar result which similarly compares favorably to OSTIA-R.
- Develop a similar result for semi-deterministic transducers (previous talk)
- Develop probabilistic versions for each of the above and successfully apply to practical applications (NLP tasks like P2G, G2P, transliteration, etc).

Conclusion

1. Subclasses with a common structure can be learned *very efficiently* by SOSFIA, which takes this common structure as *a priori* knowledge.
2. Some of these subclasses (like ISL) are natural from the perspective of formal language theory.
3. Such subclasses of subsequential transductions are of interest in certain domains (like phonology).

Thank you.

References

[CEH14] Jane Chandlee, Rémi Eyraud, and Jeffrey Heinz. Learning strictly local subsequential functions. *Transactions of the Association for Computational Linguistics*, 2014.

[Cha14] Jane Chandlee. *Strictly Local Phonological Processes*. PhD thesis, The University of Delaware, 2014.

[CVVO98] Antonio Castellanos, Enrique Vidal, Miguel A. Varó, and José Oncina. Language understanding and subsequential transducer learning. *Computer Speech and Language*, 12:193–228, 1998.

[dlH97] Colin de la Higuera. Characteristic sets for polynomial grammatical inference. *Machine Learning*, 27(2):125–138, 1997.

[HR13] Jeffrey Heinz and James Rogers. Learning subregular classes of languages with factored deterministic automata. In Andras Kornai and Marco Kuhlmann, editors, *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, pages 64–71, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

[MP71] Robert McNaughton and Seymour Papert. *Counter-Free Automata*. MIT Press, 1971.

[OG91] Jose Oncina and Pedro Garcia. Inductive learning of subsequential functions. Technical Report DSIC II-34, University Politécnia de Valencia, 1991.

[OGV93] José Oncina, Pedro García, and Enrique Vidal. Learning subsequential transducers for pattern recognition tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:448–458, May 1993.

[OV96] José Oncina and Miguel A. Varò. Using domain information during the learning of a subsequential transducer. *Lecture Notes in Computer Science - Lecture Notes in Artificial Intelligence*, pages 313–325, 1996.

[RHB+10] James Rogers, Jeffrey Heinz, Gil Bailey, Matt Edlefsen, Molly Visscher, David Wellcome, and Sean Wibel. On languages piecewise testable in the strict sense. In Christian Ebert, Gerhard Jäger, and Jens Michaelis, editors, *The Mathematics of Language*, volume 6149 of *Lecture Notes in Artifical Intelligence*, pages 255–265. Springer, 2010.