The Role of Strict Locality in Computing Phonology: Implications for Typology and Learning

Jane Chandlee

University of Delaware | Nemours Biomedical Research

April 15, 2015

Objectives

- Present some basics of phonology and introduce a learning problem.
- Provide a computational solution to this problem.
- Discuss some of the larger implications for natural language phonology.

Phonological Processes

- English plural suffix: /z/

shoes	[ʃuz]
lids	[lɪdz]

- Sometimes pronounced [s]

hats	[hæts]

Phonological Processes

How should we describe this fact of English?

- Option 1: English has more than one plural.
- Option 2: English has one plural and a **process** for determining how it is pronounced.

Phonological Processes

Option 2: English has one plural and a process for determining how it is pronounced in any given word.

Generative linguistics argues that the best explanation for systematic variation in pronunciations is to posit an abstract, underlying form (e.g., /z/) and a **transformation** to a surface, pronounced form (e.g., /z/ $\mapsto$ [s]).

Phonological Processes

hats	[hæts]
lids	[lɪdz]

Generalization:

- When the preceding sound is voiced, $/\text{z}/ \mapsto$ [z].
- When the preceding sound is voiceless, $/\text{z}/ \mapsto$ [s].

Phonological Processes

- A process can be represented with a *function*:

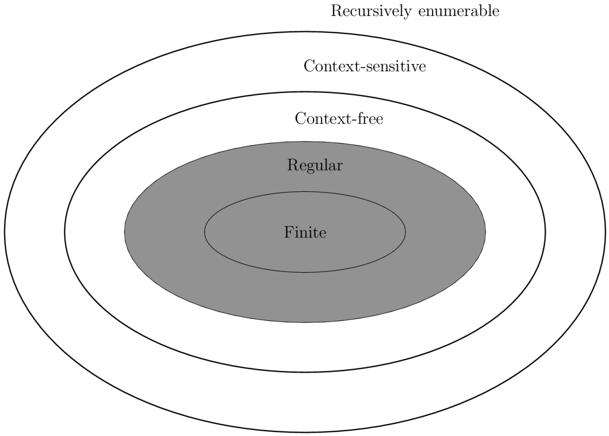
(1)
a. f(lɪdz) = lɪdz
b. f(hætz) = hæts
c. f(ʃuz) = ʃuz

- This function is *infinite* and will be defined for all *possible* words, including those the speaker has no prior experience with.

Phonological Functions

- Functions can be grouped into classes that are distinguished by their computational complexity.
- How can we define the class of *phonological* functions?

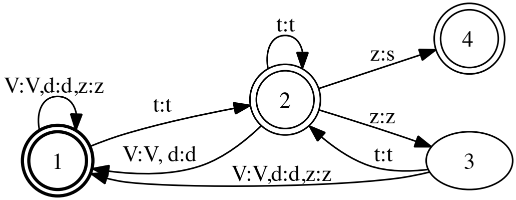
Chomsky Hierarchy



Regular Relations

- Phonological transformations are regular relations (Johnson 1972, Koskenniemi 1983, Kaplan & Kay 1994).
- This means they can be represented with Finite State Transducers (FSTs).

Regular Relations

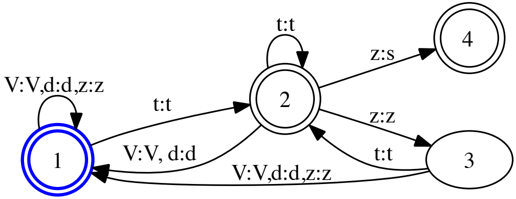


Regular Relations

(2) $f(\text{datz}) = \text{dats}$

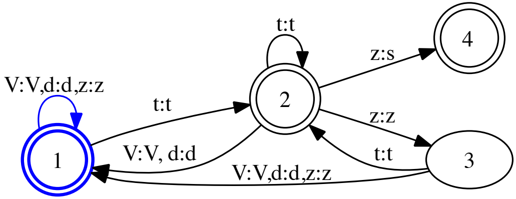
- Does the FST accept the pair (datz, dats)?

Regular Relations



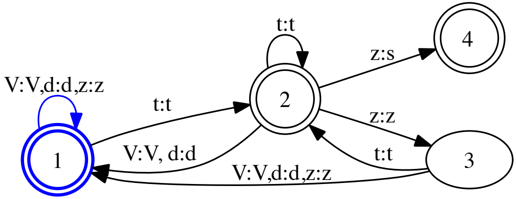
d	ɑ	t	z
d	ɑ	t	s

Regular Relations



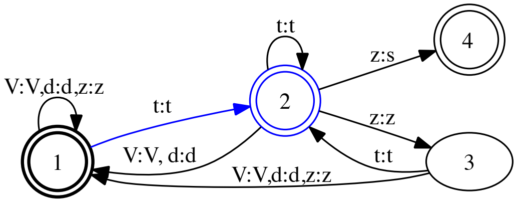
d	ɑ	t	z
d	ɑ	t	s

Regular Relations



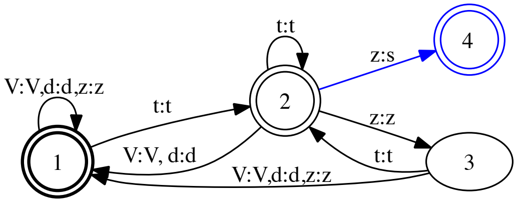
d a t z
d a t s

Regular Relations



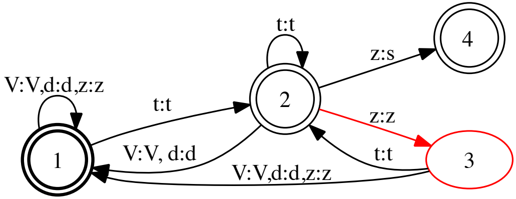
d a t z
d a t s

Regular Relations



d	ɑ	t	z
d	ɑ	t	s

Regular Relations



d a t z
d a t z

Regular Relations

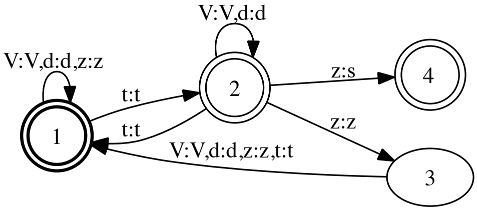
- Phonological functions are regular relations, but...
- ...two indications that we can ‘do better’:
  1. The class of regular relations cannot be learned from positive data in the sense of (Gold 1967).
  2. Predicts the existence of processes that are intuitively ‘odd’ based on what we know about phonology.

Non-phonological regular relations

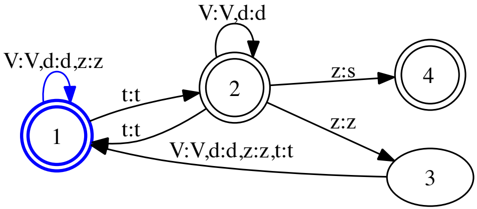
(3) ‘odd’ phonological function:
z $\mapsto$ s, but only at the end of a word that contains an odd number of t’s

- Still a regular relation!

Non-phonological regular relations

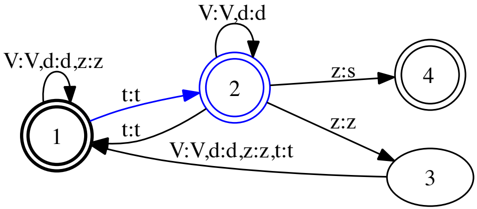


Non-phonological regular relations



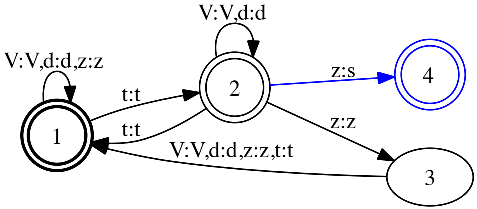
d	a	t	z
d	a	t	s

Non-phonological regular relations



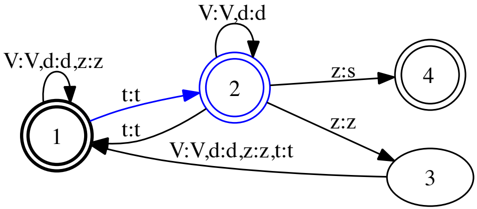
d	ɑ	t	z
d	ɑ	t	s

Non-phonological regular relations



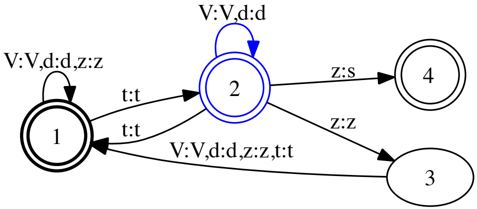
d	ɑ	t	z
d	ɑ	t	s

Non-phonological regular relations



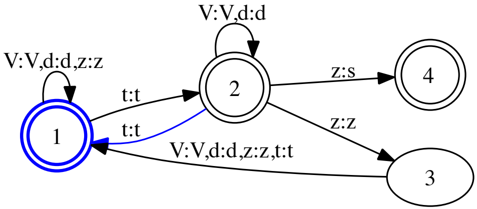
t	a	t	z
t	a	t	z

Non-phonological regular relations



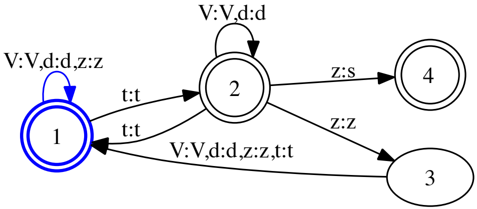
t ɑ t z
t ɑ t z

Non-phonological regular relations



t	a	t	z
t	a	t	z

Non-phonological regular relations



t	ɑ	t	z
t	ɑ	t	z

Phonological functions

Phonological functions are properly *subregular*.

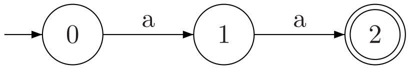
Questions

1. What subregular class corresponds to phonological functions?
2. How do humans learn these functions from positive data?

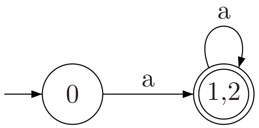
Questions

1. What subregular class corresponds to phonological functions?
2. **How do humans learn these functions from positive data?**

State Merging



$\{$aa$\}$



aa*

(Angluin 1982, Oncina et al. 1993, Heinz 2009, de la Higuera 2010)

State Merging

Demonstration: German final devoicing

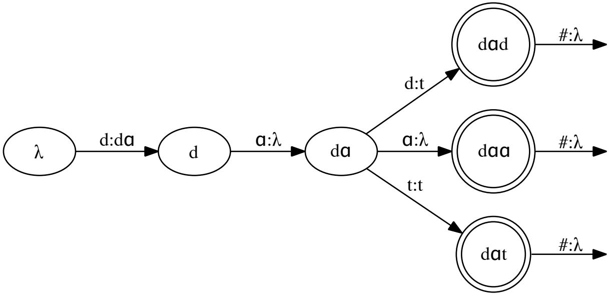
(4)	a.	/bɑd/ $\mapsto$ [bɑt], ‘bath’
	b.	/sɑg/ $\mapsto$ [sɑk], ‘say’

The Learning Problem

Given this data: {(dɑd, dɑt), (dɑt, dɑt), (dɑɑ, dɑɑ...}

How can we learn the function (i.e., the FST)?

State Merging



State Merging

What states should be merged?

State Merging

What states should be merged?

Hypothesis: phonological computations require only 'local' information.

State Merging

What states should be merged?

Hypothesis: phonological computations require only 'local' information.

How can we quantify 'local'?

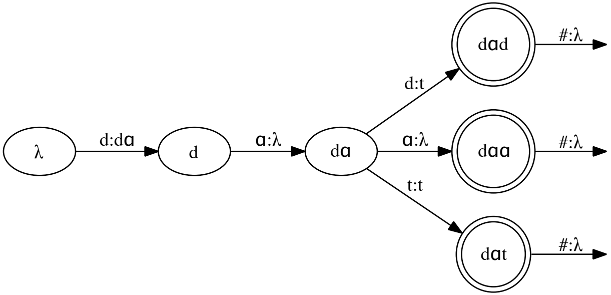
Strictly Local Languages

- The Strictly Local (SL) Languages are a subregular class of formal languages.
- The constraints on the strings in a SL-k language are expressible as substrings of length k.

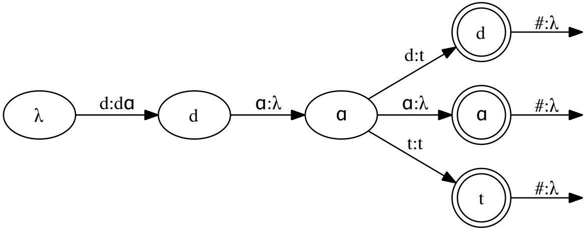
(5) German is SL-2, since $|\text{d}\#| = 2$.

- A SL-k language can be learned by merging states that have the same suffix of length $k-1$ (Heinz 2009).

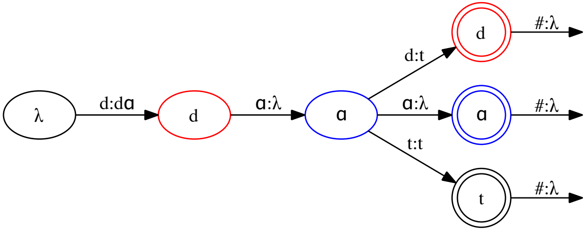
Strictly Local State Merging



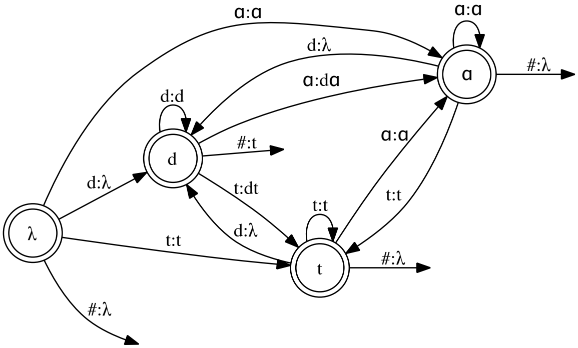
Strictly Local State Merging



Strictly Local State Merging



Strictly Local State Merging



Strictly Local State Merging

What kind of function is this (i.e., what can we learn this way)?

Input Strictly Local Functions

Definition

A function is Input Strictly k-Local if the output of every input string $a_0 a_1 \cdots a_n$ is $u_0 u_1 \cdots u_n$ where $a_i \in \Sigma$ and u_i is a string which only depends on a_i and the $k-1$ symbols before a_i in the input string.

(Chandlee 2014, Chandlee & Heinz to appear)

ISL function learning algorithm (ISLFLA)

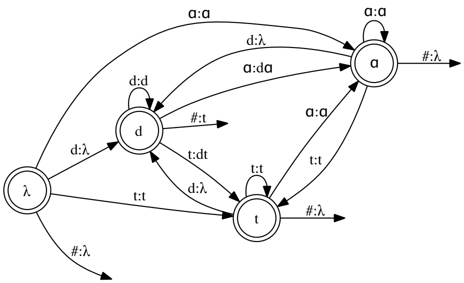
- ISLFLA can learn all and only *k*-ISL functions, in quadratic time and data (Chandlee et al. 2014).
- ISL is a *proper* subclass of regular relations, which rules out many non-phonological relations.

(Chandlee et al. 2014)

Non-phonological functions

- Consider an alternative version of German final devoicing:
    - Devoicing final /d/ only if the word contains an even number of d's.

Strictly Local State Merging



- This FST cannot distinguish /dɑd/ (even number of d's) from /dɑdɑd/ (odd number of d's).

Phonological typology

How many phonological processes are ISL functions?

Phonological typology

- A review of the phonological pattern database P-Base (v1.95, Mielke 2008) revealed that 95% are ISL.
- This includes substitution processes like voicing assimilation and final devoicing, as well as insertion, deletion, and metathesis.

Phonological typology

(6) Dutch epenthesis ($k = 2$)

a. mɛlk $\mapsto$ mɛlək, ‘milk’

(7) Tagalog deletion ($k = 4$)

a. bukasin $\mapsto$ buksin, ‘was opened’

(8) Rotuman metathesis ($k = 3$)

a. hosa $\mapsto$ hoas, ‘flower’

Non-ISL phonological processes

What phonological processes *aren't* ISL functions?

Non-ISL phonological processes

Two main categories:

1. Iterative processes:

   (9) Johore Malay nasal spreading

   a. pəŋawasan ↦ pəŋãw̃ãsan, ‘supervision’

2. Long-distance processes:

   (10) Kikongo nasal assimilation

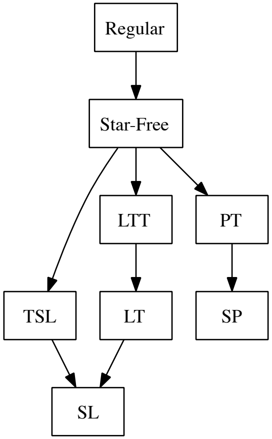
   a. tunikidi ↦ tunikini, ‘we ground’

(Onn 1980, Rose & Walker 2004)

What processes *aren't* ISL functions?

- Iterative processes are Output Strictly Local (OSL) functions (Chandlee et al. 2015).
- Long-distance processes are likely still subregular, but require a different notion of locality (Heinz 2010).

Subregular hierarchy of languages



Strictly Piecewise Functions?

- ISL functions are based on SL languages and correspond to SL constraints: *tz#, *CV#, *lk, etc.
- Long-distance processes correspond to constraints like *n...d, which are SP constraints.

Beyond phonology

The ISL subclass may lead to more efficient algorithms for other string-to-string transductions:

- Grapheme-to-Phoneme conversion

(11) knight $\mapsto$ [nɑɪt]

- Pronunciation variation in speech recognition and synthesis

(12) ‘bouquet’

a. boʊkeɪ $\mapsto$ buwkeɪ

Conclusions

- The ISL functions formalize the role of locality in the input-output transformations allowed in phonology.
- These functions are expressive enough to model a significant range of phonological processes and restrictive enough to be efficiently learned from positive data.
- Current and future work is focused on addressing certain types of non-ISL phonological processes and exploring HLT applications of the ISL class.

Selected references

- Angluin, D. (1982). Inference of reversible languages. *Journal for the Association of Computing Machinery 29* (3), 741-765.
- Chandlee, J. (2014). Strictly Local Phonological Processes. PhD thesis, University of Delaware.
- Chandlee, J., R. Eyraud, and J. Heinz. (2014). Learning Strictly Local subsequential functions. *Transactions of the Association for Computational Linguistics 2*, 491-503.
- Chandlee, J. and J. Heinz. (to appear). Strict Locality and Phonological Maps. *Linguistic Inquiry*, under revision.
- Gold, E.M. (1967). Language identification in the limit. *Information and Control 10*:447-474.
- Heinz, J. (2009). On the role of locality in learning stress patterns. *Phonology 26*, 303-351.

Selected references

- Heinz, J. (2010). Learning long-distance phonotactics. *Linguistic Inquiry 41*(4):623-661.
- de la Higuera, C. (2010). *Grammatical Inference: Learning Automata and Grammars.* Cambridge University Press.
- Johnson, C. Douglas. (1972). *Formal Aspects of Phonological Description*. The Hague: Mouton.
- Kaplan, R. M. and M. Kay (1994). Regular models of phonological rule systems. *Computational Linguistics 20*, 371-387.
- Koskenniemi, K. (1983). *Two-Level Morphology: A general computational model for word-form recognition and production*. University of Helsinki, Department of General Linguistics.
- Mielke, J. (2008). *The Emergence of Distinctive Features.* Oxford: Oxford University Press.
- Oncina, J., J. García, and E. Vidal. (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*(5), 448-457.