Locality in Phonological Interactions: A Computational Account

Jane Chandlee

Haverford College

University of California - San Diego
January 18, 2017

Introduction

- Many phonological input-output maps can be characterized by a small, computationally-restrictive and learnable class of functions, including **opaque interactions**.
- This result is part of a larger research program that aims to understand the computational nature of phonological patterns and how their computational properties can inform how they are learned.

Outline

- Establish a framework for classifying phonological maps based on their computational complexity.
  - Subregular hierarchy of function classes: ISL, OSL, TSL
- Use this framework to present a computational analysis of opaque map interactions.
  - 7 distinct cases of opacity are shown to be ISL functions.

Computational complexity of phonological patterns

- What do we know about the computational complexity of phonotactics and phonological maps?
- Starting point: both are *regular* (i.e., finite state describable)
- Proposal: both are in fact *subregular* (i.e., describable with proper subsets of regular)
- Why is this restriction desirable?
  - Better fit to the typology
  - Learnability advantages

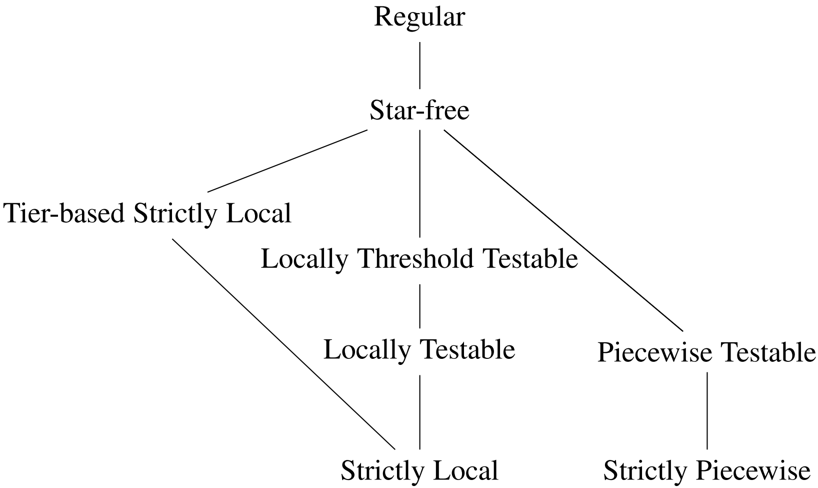
Formal languages and phonotactics

- A formal language is a set of strings.
- A phonotactic constraint can be modeled with the set of strings that do not violate it.

(1) { amba, nanta, ama, ... }

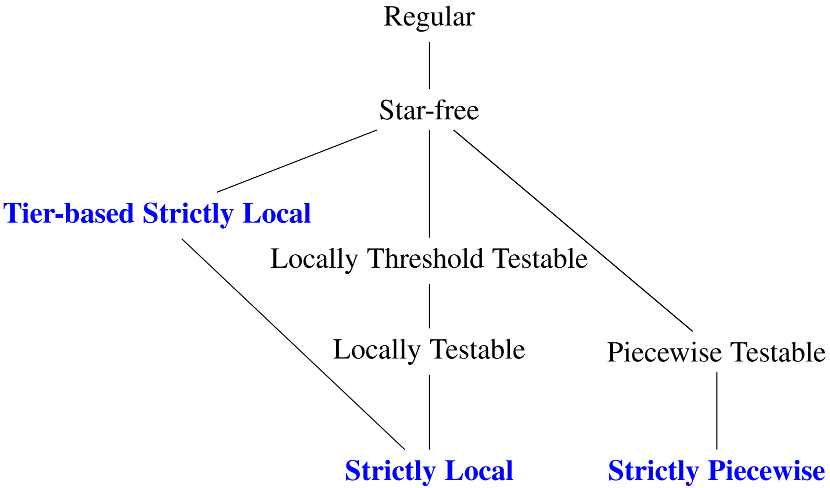
- What is the *complexity* of the formal languages that model phonotactics?

Subregular hierarchy of languages



(Rogers and Pullum, 2011; Rogers et al., 2013; Heinz, 2010; Heinz et al., 2011; McMullin, 2016)

Subregular hierarchy of languages



(Rogers and Pullum, 2011; Rogers et al., 2013; Heinz, 2010; Heinz et al., 2011; McMullin, 2016)

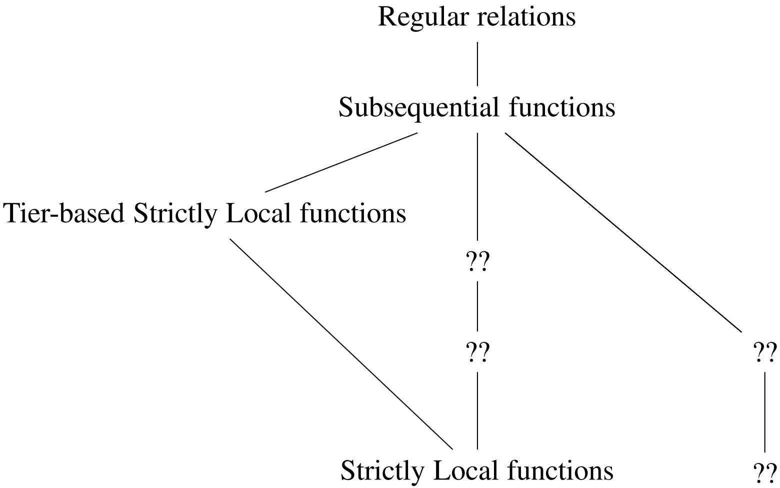
Phonological maps

- A relation/function/map is a set of string *pairs*.
- A phonological map (i.e., ‘process’) is a (UR, SR) pair.

(2) { (anba, amba), (nanta, nanta), (ama, ama), ... }

- What is the *complexity* of phonological maps?

Subregular hierarchy of maps



(Johnson, 1972; Kaplan and Kay, 1994; Mohri, 1997; Chandlee, 2014; Chandlee et al., 2017)

Computational complexity of phonological maps

REGULAR RELATIONS (Johnson, 1972; Kaplan and Kay, 1994)

↓

SUBSEQUENTIAL FUNCTIONS (Mohri, 1997)

↓

TIER-BASED STRICTLY LOCAL FUNCTIONS (Chandlee et al., 2017)

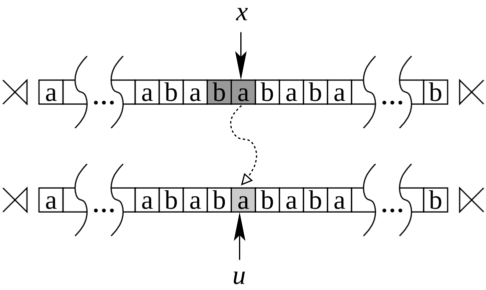
↓

STRICTLY LOCAL FUNCTIONS (Chandlee, 2014)

Strictly Local functions

- Output string is computed *locally*, depending only on a bounded number of previous segments.
- Two varieties:
  - Input Strictly Local: output depends on previous input symbols
  - Output Strictly Local: output depends on previous output symbols

Input Strictly Local (ISL) function



Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] ‘rice water’

$\rtimes$	p	a	p	m	u	l	$\ltimes$
	λ						

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] ‘rice water’

$\rtimes$	p	a	p	m	u	l	$\ltimes$
	λ	pa					

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] ‘rice water’

$\rtimes$	p	a	p	m	u	l	$\ltimes$
	λ	pa	λ				

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] ‘rice water’

$\rtimes$	p	a	p	m	u	l	$\ltimes$
	λ	pa	λ	mm			

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] ‘rice water’

⋊	p	a	p	m	u	l	⋉
	λ	pa	λ	mm	u		

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] 'rice water'

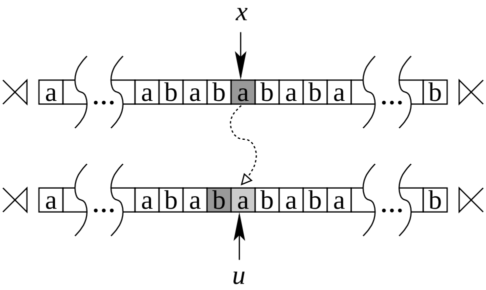
$\rtimes$	p	a	p	m	u	l	$\ltimes$
	λ	pa	λ	mm	u	l	

Example: Input Strictly Local (ISL) function

(3) Korean (Lee and Pater, 2008)
/pa**pm**ul/ $\mapsto$ [pa**mm**ul] 'rice water'

Window size is 2: this map is 2-ISL.

Output Strictly Local function



Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p									

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e								

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ							

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã						

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃					

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃	ã				

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃	ã	s			

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ $\mapsto$ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃	ã	s	a		

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ ↦ [pəŋãw̃ãsan] ‘supervision’

⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃	ã	s	a	n	

Example: Output Strictly Local (OSL) function

(4) Johore Malay (Onn, 1980)
/pəŋawasan/ $\mapsto$ [pəŋãw̃ãsan] ‘supervision’

Window size is 2: this map is 2-OSL.

Long-distance (unbounded) assimilation

(5) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ $\mapsto$ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ $\mapsto$ [kudu**m**ukisi**n**a] ‘to cause to jump for’

Long-distance (unbounded) assimilation

(5) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ ↦ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ ↦ [kudu**m**ukisi**n**a] ‘to cause to jump for’

⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n	i	k	i	n		

⋊	k	u	d	u	m	u	k	i	s	i	l	a	⋉
	k	u	d	u	m	u	k	i	s	i	l		

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ $\mapsto$ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ $\mapsto$ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

$\rtimes$	t	u	n	i	k	i	d	i	$\ltimes$
	t								

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ ↦ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ ↦ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

⋊	t	u	n	i	k	i	d	i	⋉
	t	u							

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ ↦ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ ↦ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n						

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ $\mapsto$ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ $\mapsto$ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n	i					

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ $\mapsto$ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ $\mapsto$ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

$\rtimes$	t	u	n	i	k	i	d	i	$\ltimes$
	t	u	n	i	k				

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ ↦ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ ↦ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n	i	k	i			

Example: Tier-based Strictly Local (TSL) function

(6) Kikongo (Meinof, 1932; Odden, 1994; Rose and Walker, 2004)

a. /tu**n**iki**d**i/ ↦ [tu**n**iki**n**i] ‘we ground’

b. /kudu**m**ukisi**l**a/ ↦ [kudu**m**ukisi**n**a] ‘to cause to jump for’

- Designate a subset of the alphabet as the *tier*: T = {n, m, d, l}

⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n	i	k	i	n		

Long-distance assimilation with blocking

Slovenian (Jurgec, 2011; McMullin, 2016)

a. **s**pi**ʃ** ↦ **ʃ**pi**ʃ** ‘(you) sleep’

b. **z**aklɔni**ʃtʃ**e ↦ **ʒ**aklɔni**ʃtʃ**e ‘bomb shelter’

c. na**s**i**t**i**ʃ** ↦ na**s**i**t**i**ʃ** ‘(you) feed’

- The blocking segments go on the tier.

⋊	n	a	s	i	t	i	ʃ	⋉
							ʃ	

Long-distance assimilation with blocking

Slovenian (Jurgec, 2011; McMullin, 2016)

a. **s**pi**ʃ** ↦ **ʃ**pi**ʃ** ‘(you) sleep’

b. **z**aklɔni**ʃtʃ**e ↦ **ʒ**aklɔni**ʃtʃ**e ‘bomb shelter’

c. na**s**i**t**i**ʃ** ↦ na**s**i**t**i**ʃ** ‘(you) feed’

- The blocking segments go on the tier.

⋊	n	a	s	i	t	i	ʃ	⋉
						i	ʃ	

Long-distance assimilation with blocking

Slovenian (Jurgec, 2011; McMullin, 2016)

a. **s**pi**ʃ** ↦ **ʃ**pi**ʃ** ‘(you) sleep’

b. **z**aklɔni**ʃtʃ**e ↦ **ʒ**aklɔni**ʃtʃ**e ‘bomb shelter’

c. na**s**i**t**i**ʃ** ↦ na**s**i**t**i**ʃ** ‘(you) feed’

- The blocking segments go on the tier.

⋊	n	a	s	i	t	i	ʃ	⋉
					t	i	ʃ	

Long-distance assimilation with blocking

Slovenian (Jurgec, 2011; McMullin, 2016)

a. **s**pi**ʃ** ↦ **ʃ**pi**ʃ** ‘(you) sleep’

b. **z**aklɔni**ʃtʃ**e ↦ **ʒ**aklɔni**ʃtʃ**e ‘bomb shelter’

c. na**s**i**t**i**ʃ** ↦ na**s**i**t**i**ʃ** ‘(you) feed’

- The blocking segments go on the tier.

⋊	n	a	s	i	t	i	ʃ	⋉
				i	t	i	ʃ	

Long-distance assimilation with blocking

Slovenian (Jurgec, 2011; McMullin, 2016)

a. **s**pi**ʃ** ↦ **ʃ**pi**ʃ** ‘(you) sleep’

b. **z**aklɔni**ʃtʃ**e ↦ **ʒ**aklɔni**ʃtʃ**e ‘bomb shelter’

c. na**s**i**t**i**ʃ** ↦ na**s**i**t**i**ʃ** ‘(you) feed’

- The blocking segments go on the tier.

⋊	n	a	s	i	t	i	ʃ	⋉
			s	i	t	i	ʃ	

Example classifications

- ISL and OSL are argued to include all local phonological processes (assimilation, dissimilation, insertion, deletion, metathesis) (Chandlee, 2014; Chandlee et al., 2015)
- Many morphological ‘maps’ are also ISL (Chandlee, under review)
- TSL is conjectured to include long-distance processes

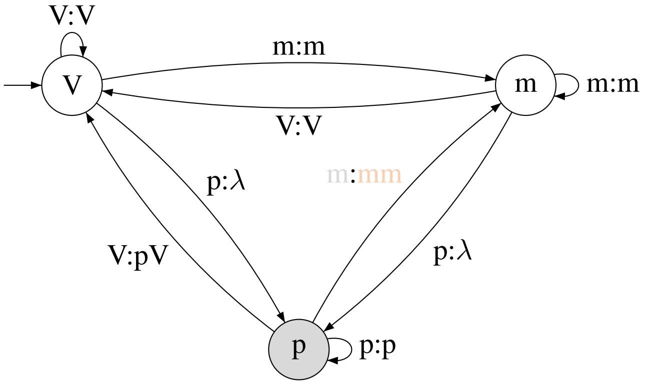
Characterizations

Class	Finite-state	Language-theoretic	Logical
ISL	✓	✓	✓
OSL	✓	✓	
TSL	✓	✓	

(Chandlee (2014); Chandlee et al. (2014, 2015, 2017), Chandlee and Lindell (to appear))

Input Strictly Local (ISL): FST characterization

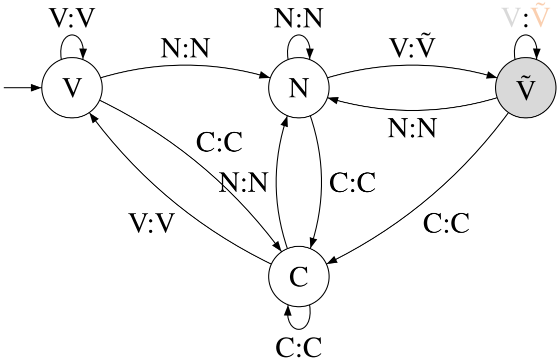
(7) /pa**pm**ul/ $\mapsto$ [pa**mm**ul]



⋊	p	a	p	m	u	l	⋉
	λ	pa	λ	mm			

Output Strictly Local (OSL): FST characterization

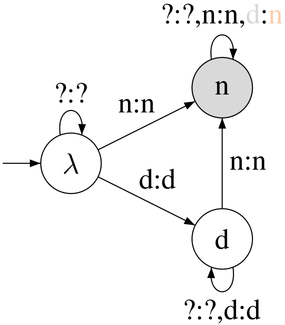
(8) /pəŋawasan/ ↦ [pəŋãw̃ãsan]



⋊	p	e	ŋ	a	w	a	s	a	n	⋉
	p	e	ŋ	ã	w̃					

Tier-based Strictly Local (TSL): FST characterization

(9) /tu**n**iki**d**i/ $\mapsto$ [tu**n**iki**n**i]



⋊	t	u	n	i	k	i	d	i	⋉
	t	u	n	i	k	i	n		

FST Characterizations

REGULAR RELATIONS (describable with FSTs)

↓

SUBSEQUENTIAL FUNCTIONS (describable with deterministic FSTs)

↓

TIER-BASED STRICTLY LOCAL FUNCTIONS (states are $T^{\leq k-1}$)

↓

STRICTLY LOCAL FUNCTIONS (transitions follow recent input/output)

Map interactions

What about map interactions?

Case study: opaque phonological generalizations

- Opaque phonological generalizations result from the interaction of two processes, one of which is not ‘surface-true’.
- The 7 distinct types of opacity from McCarthy (2007); Baković (2007); Kavitskaya and Staroverov (2010) can all be described with ISL maps.
- The needed information for determining the correct output at any point in the computation is found in a contiguous window of the input.

Case study: opaque phonological generalizations

(10) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [$-$high]

b. Shortening: V $\rightarrow$ [$-$long] / __ C #

(11) Yokuts (McCarthy, 1999)
/ʔili:l/ $\mapsto$ [ʔilel], 'might fan'

- Lowering over-applies.

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [$-$high]

b. Shortening: V $\rightarrow$ [$-$long] / ___ C #

(13) Yokuts (McCarthy, 1999)

/ʔili:l/ $\mapsto$ [ʔilel], ‘might fan’

⋊	ʔ	i	l	i:	l	⋉
	ʔ					

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] → [−high]

b. Shortening: V → [−long] / ___ C #

(13) Yokuts (McCarthy, 1999)
/ʔiliːl/ ↦ [ʔilel], ‘might fan’

⋊	ʔ	i	l	iː	l	⋉
	ʔ	i				

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [$-$high]

b. Shortening: V $\rightarrow$ [$-$long] / ___ C #

(13) Yokuts (McCarthy, 1999)
/ʔili:l/ $\mapsto$ [ʔilel], ‘might fan’

⋊	ʔ	i	l	i:	l	⋉
	ʔ	i	l			

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [$-$high]

b. Shortening: V $\rightarrow$ [$-$long] / __ C #

(13) Yokuts (McCarthy, 1999)
/ʔili:l/ $\mapsto$ [ʔilel], ‘might fan’

$\rtimes$	ʔ	i	l	i:	l	$\ltimes$
	ʔ	i	l	λ		

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [−high]

b. Shortening: V $\rightarrow$ [−long] / ___ C #

(13) Yokuts (McCarthy, 1999)

/ʔili:l/ $\mapsto$ [ʔilel], ‘might fan’

⋊	ʔ	i	l	i:	l	⋉
	ʔ	i	l	λ	λ	

Case study: opaque phonological generalizations

(12) Counterbleeding

a. Lowering: [+long] $\rightarrow$ [$-$high]
b. Shortening: V $\rightarrow$ [$-$long] / ___ C #

(13) Yokuts (McCarthy, 1999)
/ʔili:l/ $\mapsto$ [ʔilel], 'might fan'

$\rtimes$	ʔ	i	l	i:	l	$\ltimes$
	ʔ	i	l	λ	λ	el

Case study: opaque phonological generalizations

(14) Counterfeeding on focus

a. Deletion: $i \rightarrow \emptyset$ / ___ CV
b. Raising: $a \rightarrow i$ / ___ CV

(15) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], ‘he wrote’

- Deletion underapplies.

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / __ CV

b. Raising: a $\rightarrow$ i / __ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], 'he wrote'

⋊	k	a	t	a	b	⋉
	k					

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / __ CV
b. Raising: a $\rightarrow$ i / __ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], 'he wrote'

$\rtimes$	k	a	t	a	b	$\ltimes$
	k	λ				

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / ___ CV

b. Raising: a $\rightarrow$ i / ___ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], 'he wrote'

$\rtimes$	k	a	t	a	b	$\ltimes$
	k	λ	λ			

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / ___ CV

b. Raising: a $\rightarrow$ i / ___ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], 'he wrote'

$\rtimes$	k	a	t	a	b	$\ltimes$
	k	λ	λ	it		

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / ___ CV

b. Raising: a $\rightarrow$ i / ___ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], ‘he wrote’

$\rtimes$	k	a	t	a	b	$\ltimes$
	k	λ	λ	it	λ	

Case study: opaque phonological generalizations

(16) Counterfeeding on focus

a. Deletion: i $\rightarrow \emptyset$ / ___ CV
b. Raising: a $\rightarrow$ i / ___ CV

(17) Bedouin Arabic (McCarthy, 1999)
/katab/ $\mapsto$ [kitab], 'he wrote'

$\rtimes$	k	a	t	a	b	$\ltimes$
	k	λ	λ	it	λ	ab

Case study: opaque phonological generalizations

(18) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(19) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

- Deletion destroys the environment for epenthesis.

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], ‘your baby’

$\rtimes$	b	e	b	e	k	+	n	$\ltimes$
	b							

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

$\rtimes$	b	e	b	e	k	+	n	$\ltimes$
	b	e						

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

⋊	b	e	b	e	k	+	n	⋉
	b	e	b					

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

⋊	b	e	b	e	k	+	n	⋉
	b	e	b	e				

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

$\rtimes$	b	e	b	e	k	+	n	$\ltimes$
	b	e	b	e	λ			

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

⋊	b	e	b	e	k	+	n	⋉
	b	e	b	e	λ	λ		

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], 'your baby'

$\rtimes$	b	e	b	e	k	+	n	$\ltimes$
	b	e	b	e	λ	λ	λ	

Case study: opaque phonological generalizations

(20) Self-destructive feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / C(+)__C#

b. Deletion: k $\rightarrow \emptyset$ / V__+V

(21) Turkish (Sprouse, 1997)
/bebek+n/ $\mapsto$ [bebein], ‘your baby’

$\rtimes$	b	e	b	e	k	+	n	$\ltimes$
	b	e	b	e	λ	λ	λ	in

Summary: ISL Opaque Maps

Opacity Type	**Language**	**k-value**
cross-derivational feeding	Lithuanian	$k = 2$
counterbleeding	Yokuts	$k = 3$
fed counterfeeding	Tundra Nenets	$k = 3$
counterfeeding on environment	Bedouin Arabic	$k = 3$
counterfeeding on focus	Bedouin Arabic	$k = 3$
non-gratuitous feeding	Classical Arabic	$k = 4$
self-destructive feeding	Turkish	$k = 5$

Implications for phonological learning

- The regular relations are not learnable from positive data...
- but the ISL functions are (Chandlee et al., 2014; Jardine et al., 2014)

Conclusion

- There is an increasing amount of evidence that phonological maps are subregular in nature, and this property is not dependent on analyses of individual generalizations.
- In particular, opaque map interactions belong to one computationally restrictive and learnable subregular class, the ISL functions.

Future work

- Complete the hierarchy of subregular maps, particularly the regions for long-distance phenomena
- Further explore the logical characterizations of these classes
- Investigate other types of map interaction
  - Blocking
  - Co-existing long-distance maps with and without distinct tiers

Acknowledgements

- Jeffrey Heinz (University of Delaware)
- Adam Jardine (Rutgers University)
- Jim Rogers (Earlham College)
- Rémi Eyraud (Laboratoire D'Informatique Fondamentale de Marseille)
- Kevin McMullin (University of Ottawa)
- Steven Lindell (Haverford College)

Appendix: Additional examples of opacity

(22) Counterfeeding on environment

a. Raising: a $\rightarrow$ i / __ CV

b. Vocalization: G $\rightarrow$ V / C __ #

(23) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], 'Bedouin'

- Raising underapplies.

Appendix: Additional examples of opacity

(24) Counterfeeding on environment

a. Raising: $a \rightarrow i$ / ___ CV

b. Vocalization: $G \rightarrow V$ / C ___ #

(25) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], 'Bedouin'

⋊	b	a	d	w	⋉
	b				

Appendix: Additional examples of opacity

(24) Counterfeeding on environment

a. Raising: a $\rightarrow$ i / ___ CV

b. Vocalization: G $\rightarrow$ V / C ___ #

(25) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], ‘Bedouin’

$\rtimes$	b	a	d	w	$\ltimes$
	b	λ			

Appendix: Additional examples of opacity

(24) Counterfeeding on environment

a. Raising: a $\rightarrow$ i / ___ CV

b. Vocalization: G $\rightarrow$ V / C ___ #

(25) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], ‘Bedouin’

$\rtimes$	b	a	d	w	$\ltimes$
	b	λ	λ		

Appendix: Additional examples of opacity

(24) Counterfeeding on environment

a. Raising: a $\rightarrow$ i / ___ CV

b. Vocalization: G $\rightarrow$ V / C ___ #

(25) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], ‘Bedouin’

$\rtimes$	b	a	d	w	$\ltimes$
	b	λ	λ	ad	

Appendix: Additional examples of opacity

(24) Counterfeeding on environment

a. Raising: a $\rightarrow$ i / __ CV

b. Vocalization: G $\rightarrow$ V / C __ #

(25) Bedouin Arabic (McCarthy, 1999)
/badw/ $\mapsto$ [badu], ‘Bedouin’

$\rtimes$	b	a	d	w	$\ltimes$
	b	λ	λ	ad	u

Appendix: Additional examples of opacity

(26) Non-gratuitous feeding (Baković, 2007)

a. Vowel epenthesis: $\emptyset \rightarrow V_i$ / # __ CCV_i

b. Glottal epenthesis: $\emptyset \rightarrow$ ʔ / #__V

(27) Classical Arabic (McCarthy, 2007)
/ktub/ $\mapsto$ [ʔuktub], ‘write.MascSg!’

$\rtimes$	k	t	u	b	$\ltimes$
	λ				

Appendix: Additional examples of opacity

(26) Non-gratuitous feeding (Baković, 2007)

a. Vowel epenthesis: $\emptyset \rightarrow V_i$ / # ___ CCV_i

b. Glottal epenthesis: $\emptyset \rightarrow$ ʔ / #___V

(27) Classical Arabic (McCarthy, 2007)
/ktub/ $\mapsto$ [ʔuktub], 'write.MASCSG!'

⋊	k	t	u	b	⋉
	λ	λ			

Appendix: Additional examples of opacity

(26) Non-gratuitous feeding (Baković, 2007)

a. Vowel epenthesis: $\emptyset \rightarrow V_i$ / # ___ CCV_i

b. Glottal epenthesis: $\emptyset \rightarrow$ ʔ / #___V

(27) Classical Arabic (McCarthy, 2007)
/ktub/ $\mapsto$ [ʔuktub], ‘write.MascSg!’

⋊	k	t	u	b	⋉
	λ	λ	ʔuktu		

Appendix: Additional examples of opacity

(28) Cross-derivational feeding (Baković, 2007)

a. Epenthesis: $\emptyset \rightarrow$ i / K_i __ K_i

b. Assimilation: K $\rightarrow$ [+voice] / __ D

(29) Lithuania (Odden, 2005)

a. /at-taiki:ti/$\rightarrow$[atitaiki:ti], 'to make fit well'

b. /ap-gauti/ $\mapsto$ [abgauti], 'to deceive'

c. /ap-berti/ $\mapsto$ [apiberti], 'to strew all over'

$\rtimes$	a	p	b	e	r	t	i	$\ltimes$
	a	λ	pib					

Appendix: Additional examples of opacity

(30) Fed Counterfeeding (Kavitskaya and Staroverov, 2010)

a. Glottalization: $\{$t, d, s, n, ŋ$\} \rightarrow$ ʔ / __ #

b. Deletion: ʌ $\rightarrow \emptyset$ / __ (ʔ) #

(31) Tundra Nenets

a. /tasʌ/ $\mapsto$ [tas], ‘whole’

b. /tʲimjʌs/ $\mapsto$ [tʲimjʔ], ‘it rotted’

$\rtimes$	t	a	s	ʌ	$\ltimes$
	t	a	λ	λ	s

$\rtimes$	tʲ	i	m	j	ʌ	s	$\ltimes$
	tʲ	i	m	j	λ	λ	ʔ

References I

Baković, E. (2007). A revised typology of opaque generalisations. *Phonology*, 24(2):217–259.

Chandlee, J. (2014). *Strictly Local Phonological Processes*. PhD thesis, University of Delaware.

Chandlee, J., Eyraud, R., and Heinz, J. (2015). Output strictly local functions. In *Proceedings of the 14th Meeting on the Mathematics of Language (MOL 2015)*.

Chandlee, J., Heinz, J., and Eyraud, R. (2014). Learning strictly local subsequential functions. *Transactions of the Association for Computational Linguistics*, 2:491–503.

Chandlee, J., Heinz, J., Jardine, A., and McMullin, K. (2017). Modeling long-distance alternations with tier-based strictly local functions. Talk given at LSA 2017.

Chandlee, J. and Lindell, S. (to appear). A logical characterization of input strictly local functions. to appear in Doing Computational Phonology, edited by Jeffrey Heinz.

References II

Heinz, J. (2010). Learning long-distance phonotactics. *Linguistic Inquiry*, 41(4):623–661.

Heinz, J., Rawal, C., and Tanner, H. G. (2011). Tier-based Strictly Local constraints for phonology. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 58–64, Portland, Oregon, USA. Association for Computational Linguistics.

Jardine, A., Chandlee, J., Eyraud, R., and Heinz, J. (2014). Very efficient learning of structured classes of subsequential functions from positive data. In *Proceedings of the 12th International Conference on Grammatical Inference (ICGI 2014)*, JMLR Workshop Proceedings, pages 94–108.

Johnson, C. (1972). *Formal Aspects of Phonological Description*. Mouton, The Hague.

Jurgec, P. (2011). *Feature spreading 2.0: a unified theory of assimilation*. PhD thesis, University of Tromsø.

References III

Kaplan, R. and Kay, M. (1994). Regular models of phonological rule systems. *Computational Linguistics*, (20):371–387.

Kavitskaya, D. and Staroverov, P. (2010). When an interaction is both opaque and transparent: the paradox of fed counterfeeding. *Phonology*, 27:255–288.

Lee, S. and Pater, J. (2008). Phonological inference and word recognition: Evidence from korean. Ms., Korea University and University of Massachusetts, Amherst.

McCarthy, J. J. (1999). Sympathy and phonological opacity. *Phonology*, 16:331–399.

McCarthy, J. J. (2007). *Hidden Generalizations: Phonological Opacity in Optimality Theory*. London: Equinox.

McMullin, K. (2016). *Tier-based Locality in Long-distance Phonotactics: Learnability and Typology*. PhD thesis, University of British Columbia.

References IV

Meinof, C. (1932). *Introduction to the phonology of the Bantu languages*. Berlin: Dietrich Reimer/Ernst Vohsen. Trans. by N. J. van Warmelo.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, (23):269–311.

Odden, D. (1994). Adjacency parameters in phonology. *Language*, 70(2):289–330.

Odden, D. (2005). *Introducing phonology*. Cambridge: Cambridge University Press.

Onn, F. M. (1980). *Aspects of Malay Phonology and Morphology: A Generative Approach*. Kuala Lumpur: Universiti Kebangsaan Malaysia.

Rogers, J., Heinz, J., Fero, M., Hurst, J., Lambert, D., and Wibel, S. (2013). Cognitive and sub-regular complexity. In Morrill, G. and Nederhof, M.-J., editors, *Formal Grammar, Lecture Notes in Computer Science*, volume 8036, pages 90–108. Springer.

References V

Rogers, J. and Pullum, G. (2011). Aural pattern recognition experiments and the subregular hierarchy. *Journal of Logic, Language and Information*, (20):329–342.

Rose, S. and Walker, R. (2004). A typology of consonant agreement as correspondence. *Language*, 80:475–531.

Sprouse, R. (1997). A case for enriched inputs.