Are Phonological Functions Total or Partial?

Jane Chandlee

Haverford College

AMP 2019 - Stony Brook
11 October 2019

Formalizing phonological processes

- Standard assumption in computational phonology: phonological processes are total functions (Johnson 1972, Kaplan & Kay 1994, Heinz 2011, Chandlee 2014, Bale & Reiss 2018).

  (1) Schwa-epenthesis in word-initial obstruent clusters
  {(spaɪ, səpaɪ), (staɪ, sətaɪ), (skaɪ, səkaɪ), (saI, saI), (wwww, wwww) ... }

- Domain is Σ^*
- Aligns with the Richness of the Base (Prince & Smolensky 1993).

Limiting the domain

- Morpheme Structure Constraints (Chomsky & Halle 1968)
- Domain is $D \subsetneq \Sigma^*$

{(spaɪ, səpaɪ), (staɪ, sətaɪ), (skaɪ, səkaɪ), (saI, saI), (wwww, wwww)
... }

Vacuous rule application

UR	/spaɪ/	/plen/
Epenthesis	səpaɪ	—
SR	[səpaɪ]	[plen]

UR	/spaɪ/	/plen/
Epenthesis	səpaɪ	plen
SR	[səpaɪ]	[plen]

{(spaɪ, səpaɪ), (staɪ, sətaɪ), (skaɪ, səkaɪ), (sal, sal), (wwww, wwww)
... }

Function composition

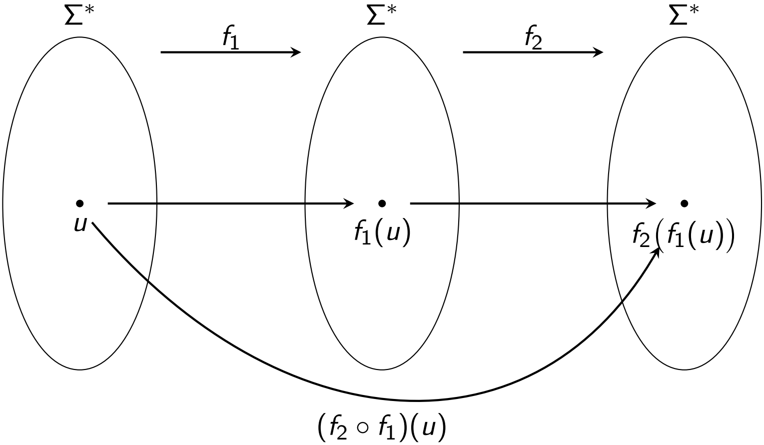
- Processes are combined via function composition (Johnson 1972, Kaplan & Kay 1994).

UR	$/u/$
f_1	x
f_2	y
f_3	z
SR	$[z]$

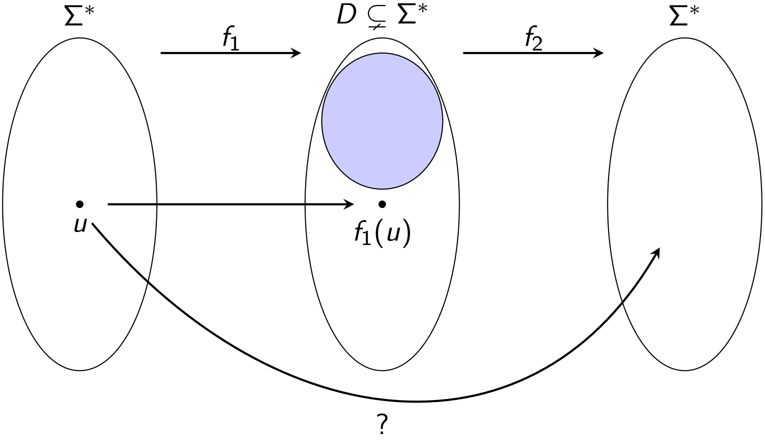
$$f_3(f_2(f_1(u))) = z$$

$$(f_3 \circ f_2 \circ f_1)(u) = z$$

Total functions simplify composition



Total functions simplify composition



Proposal

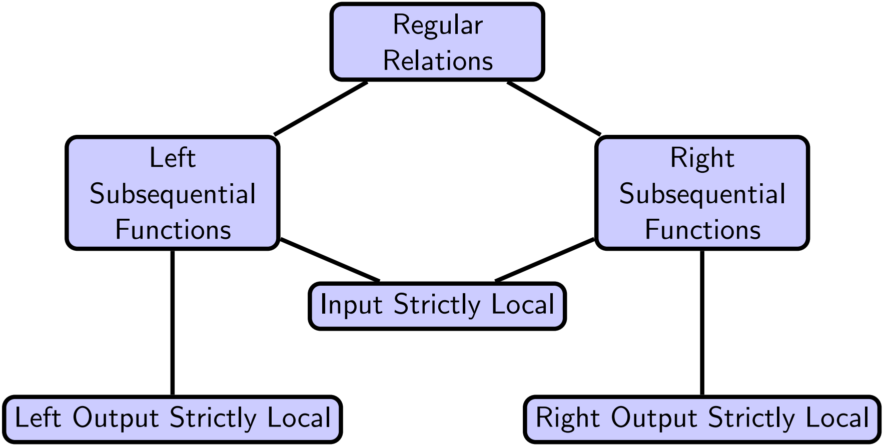
- In certain cases, a partial function analysis has advantages:
  - Better characterization of computational complexity
  - Learnability advantage
- These cases include 'non-local' phenomena with a bounded number of intervening segments.

(2) $a \rightarrow b \ / \ \_\_ \ X_0 \ d$

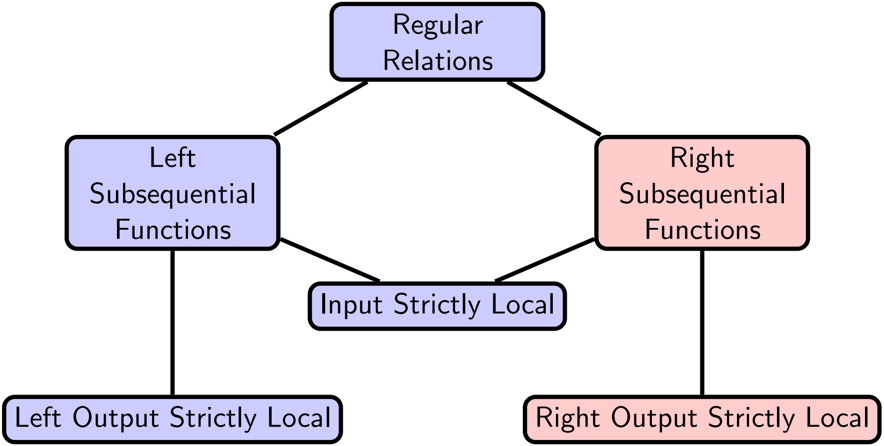
- As partial functions, such processes are computationally distinct from those described with rules like

(3) $a \rightarrow b \ / \ \_\_ \ \ldots \ d$

Hierarchy of functions



Hierarchy of functions



Right output strict locality

$a \to b \;/\; \_\_ \; b$	$a \to b \;/\; \_\_ \; a$
aaab $\mapsto$ bbbb	aaaa $\mapsto$ baba
# a a a b #	# a a a a #
b b b b	b a b a

Right subsequentiality

$$a \rightarrow b \ / \ \_\_ \ldots \ a \ \#$$

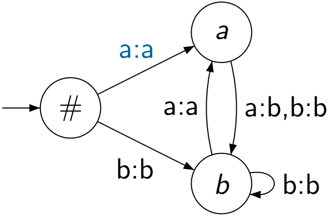
$$abba \mapsto bbba$$

#	a	b	b	a	#
	b	b	b	a	

FST characterizations

Right OSL

a → b / __ a

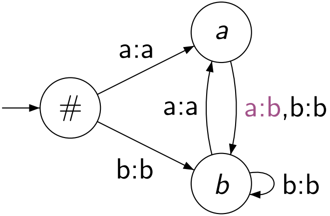


Input string:	#	a	a	a	a	#
Output string:					a	

FST characterizations

Right OSL

a → b / __ a

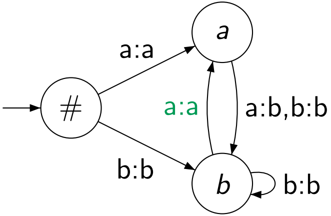


Input string:	#	a	a	a	a	#
Output string:				b	a	

FST characterizations

Right OSL

a → b / __ a

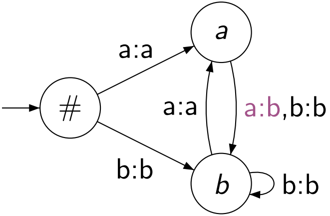


Input string:	#	a	a	a	a	#
Output string:			a	b	a	

FST characterizations

Right OSL

a → b / __ a

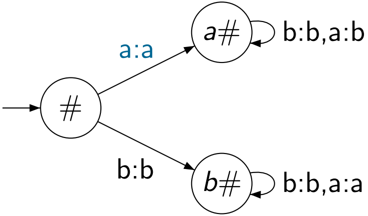


Input string:	#	a	a	a	a	#
Output string:		b	a	b	a	

FST characterizations

Right subsequential

a → b / ___ ... a#

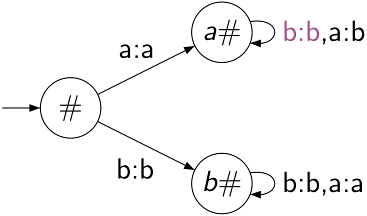


Input string:	#	a	b	b	a	#
Output string:					a	

FST characterizations

Right subsequential

a → b / __ … a#

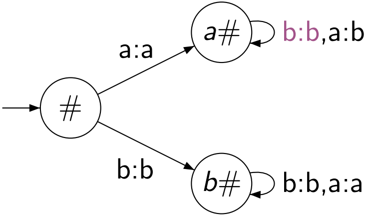


Input string:	#	a	b	b	a	#
Output string:				b	a	

FST characterizations

Right subsequential

a → b / ___ ... a#

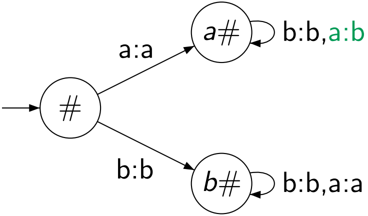


Input string:	#	a	b	b	a	#
Output string:			b	b	a	

FST characterizations

Right subsequential

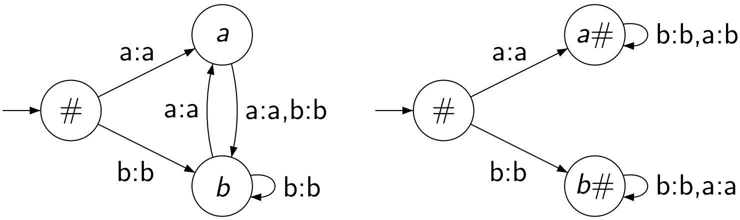
a → b / ___ ... a#



Input string:	#	a	b	b	a	#
Output string:		b	b	b	a	

FST characterizations

Right OSL　　　　Right subsequential



Isolating processes

- Computational classifications reflect the amount of information needed to model the input-output correspondence of a process.
- Total functions are meant to focus the classification on the process in question and nothing else.

#	a	w	a	n	#
	ã	w̃	ã	n	

#	a	n	n	n	#
	ã	n	n	n	

Vowel dissimilation

(4) Woleaian (Austronesian) (Sohn 1975)

a. /a/ $\rightarrow$ [e] / __ #

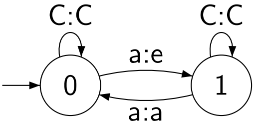
b. /a/ $\rightarrow$ [e] / __ C_0 a

(5)

a.	/matala/	[metale]	‘his eyes’
b.	/marama/	[merame]	‘moon’
c.	/yafaramami/	[yefaremami]	‘our shoulders’

Vowel dissimilation: total function

$\Sigma = \{C, a\}$
Right subsequential



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

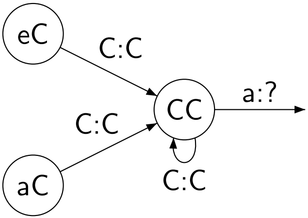
Vowel dissimilation: total function

OSL fails!

#	C	a	C	a	C	a	#
				a	C	e	

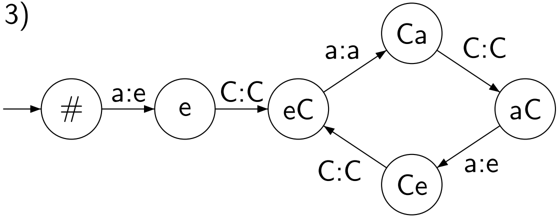
#	C	a	C	a	C	C	a	#
				?	C	C	e	

#	C	a	C	a	C	C	C	a	#
				?	C	C	C	e	



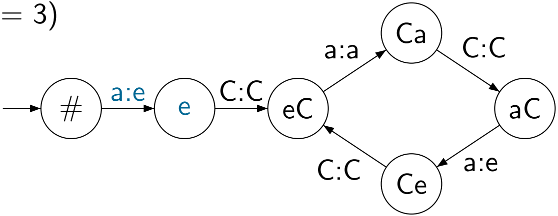
Vowel dissimilation: partial function

Domain: (Ca)*
Right OSL ($k = 3$)



Vowel dissimilation: partial function

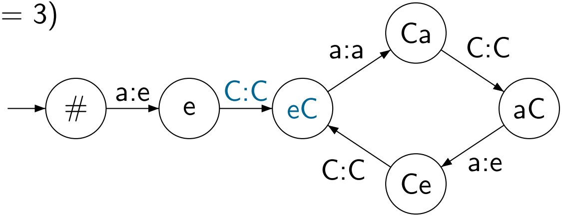
Domain: $(Ca)^*$
Right OSL ($k = 3$)



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Vowel dissimilation: partial function

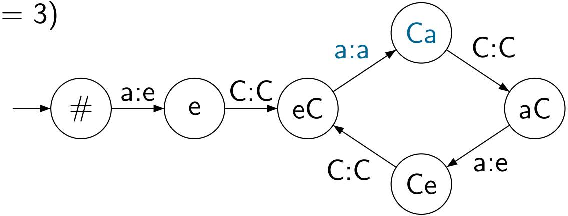
Domain: $(Ca)^*$
Right OSL ($k = 3$)



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Vowel dissimilation: partial function

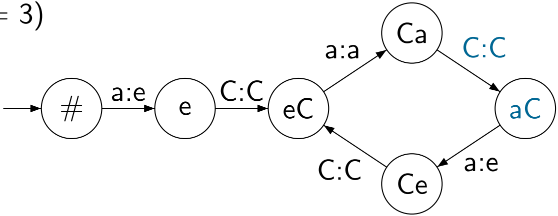
Domain: (Ca)*
Right OSL ($k = 3$)



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Vowel dissimilation: partial function

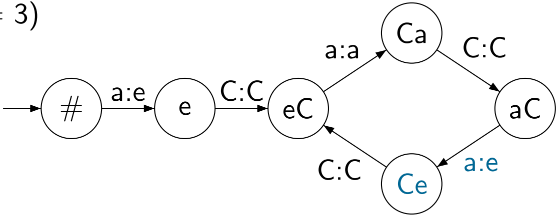
Domain: $(Ca)^*$
Right OSL $(k = 3)$



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Vowel dissimilation: partial function

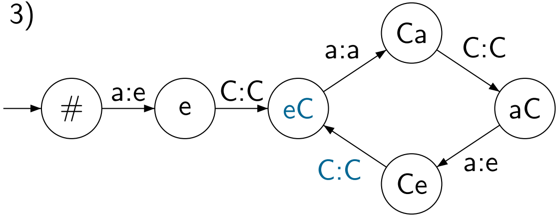
Domain: $(Ca)^*$
Right OSL ($k = 3$)



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Vowel dissimilation: partial function

Domain: (Ca)*
Right OSL ($k = 3$)



#	m	a	t	a	l	a	#
	m	e	t	a	l	e	

Computational characterization

- This process is *output-oriented* and its trigger and target are included in a *bounded substring*.
- Classifying it as subsequential is only necessary to maintain the assumption of a total function.

Vowel harmony

(6) $V \rightarrow V_{\alpha F} \;/\; \_\_ \; C_0 \; V_{\alpha F}$

- Subsequential as a total function: one-sided context (Gainor et al. 2012, Heinz & Lai 2013)
    - Circumambient harmony is not subsequential (McCollum et al. (under review))
- As a partial function,
    - Without neutral vowels: OSL
    - With opaque vowels: OSL
    - With transparent vowels: Subsequential

Learning OSL functions

- Output Strictly Local Function Inference Algorithm (OSLFIA) (Chandlee et al. 2015)
- Proof of correctness assumes the target is a total function.

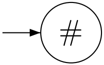
OSLFIA

$\Sigma = \{a,\ b\}$

$a \rightarrow b \ / \ b$ __

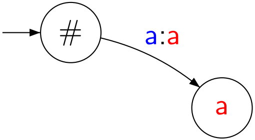
data:

{(a,a),(b,b),(aa,aa),(ab,ab),(ba,bb),(bb,bb),(aaa,aaa),(aab,aab),
(aba,abb),(abb,abb),(baa,bbb),(bab,bbb),(bba,bbb),(bbb,bbb)}



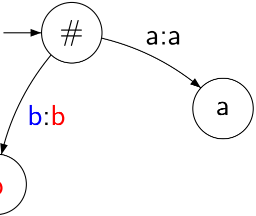
OSLFIA

$\Sigma = \{a, b\}$
$a \rightarrow b \;/\; b\;\_\_$
data:
{(a,a),(b,b),(aa,aa),(ab,ab),(ba,bb),(bb,bb),(aaa,aaa),(aab,aab),
(aba,abb),(abb,abb),(baa,bbb),(bab,bbb),(bba,bbb),(bbb,bbb)}



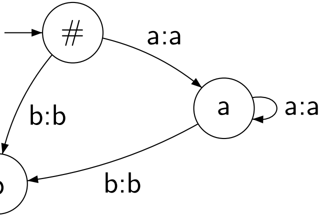
OSLFIA

$\Sigma = \{$a, b$\}$
a $\rightarrow$ b / b ___
data:
{(a,a),(b,b),(aa,aa),(ab,ab),(ba,bb),(bb,bb),(aaa,aaa),(aab,aab),
(aba,abb),(abb,abb),(baa,bbb),(bab,bbb),(bba,bbb),(bbb,bbb)}



OSLFIA

$\Sigma = \{a, b\}$
$a \rightarrow b \ / \ b \ \_\_$
data:
{(a,a),(b,b),(aa,aa),(ab,ab),(ba,bb),(bb,bb),(aaa,aaa),(aab,aab),
(aba,abb),(abb,abb),(baa,bbb),(bab,bbb),(bba,bbb),(bbb,bbb)}



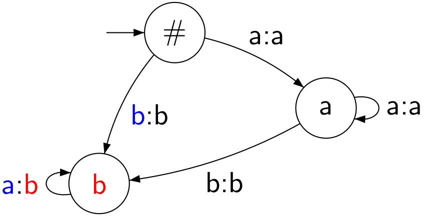
OSLFIA

$\Sigma = \{a, b\}$

a $\rightarrow$ b / b __

data:

{(a,a),(b,b),(aa,aa),(ab,ab),(ba,bb),(bb,bb),(aaa,aaa),(aab,aab),
(aba,abb),(abb,abb),(baa,bbb),(bab,bbb),(bba,bbb),(bbb,bbb)}

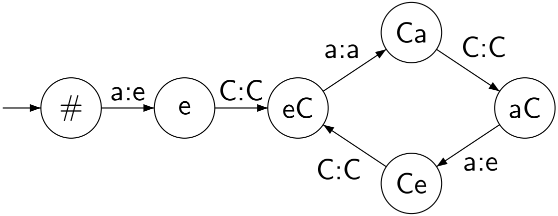


Data required to learn total functions

- For a given Σ and k, the sample must include input strings for all possible sequences from Σ up to length $k+2$
- For example, if $\Sigma = \{C, a\}$ and $k = 3$, needs to see inputs like CaCa, but also CCCCC, aaaC, etc.

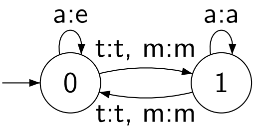
Learning partial functions

- Given strings of up to three syllables from $(Ca)^*$, with $C = \{t, m\}$, OSLFIA returns a partial function FST that is equivalent to the target:



Learning partial functions

- Onward Subsequential Transducer Inference Algorithm (OSTIA) (Oncina et al. 1993) learns total subsequential functions.
- OSTIA-D (Oncina & Varó 1996)) learns partial functions given an FSA representation of the domain.
- Given the same data sample as OSLFIA, *both* OSTIA and OSTIA-D return:



Learning partial functions

- When targeted as an OSL function, vowel dissimilation is learnable with less and more plausible data.
- Targeting it as an OSL function is only possible if it's a partial function.
- For future work:
  - Formally, what types of partial functions are learnable by OSLFIA?
  - Why doesn't domain information help OSTIA in these cases?

When to use a partial function?

- What are the formal properties of maps for which a partial function analysis should be considered?
  - Always iterative/output-oriented?
  - What is the nature of the limit on the domain?
- What about composition?

Locality and representation

- Additional function classes have been proposed for non-local processes, including A-ISL (Chandlee & Jardine 2019) and O-TSL (Burness & McMullin 2019).
- These classes are subsets of subsequential, but more powerful than OSL.
- Partial functions are a way to preserve locality without adding complexity.

Acknowledgements

For helpful discussion and feedback thanks to Jeffrey Heinz, Adam Jardine, and members of the University of Delaware Phonology Group. Thanks also to the organizers of AMP 2019!

References

- Bale, Alan and Charles Reiss. 2018. *Phonology: A formal introduction*. MIT Press.
- Burness, Phillip and Kevin McMullin. 2019. Efficient learning of Output Tier-based Strictly 2-Local functions. In *Proceedings of the 16th Meeting on the Mathematics of Language*, 78-90.
- Chandlee, Jane, Rémi Eyraud, and Jeffrey Heinz. 2015. Output strictly local functions. In Marco Kuhlmann, Makoto Kanazawa, and Gregory M. Kobele, editors, *Proceedings of the 14th Meeting on the Mathematics of Language (MoL 2015)*, 112-125.
- Chandlee, Jane. 2017. Computational locality in morphological maps. *Morphology 27*, 599-641.
- Chandlee, Jane and Adam Jardine. 2019. Autosegmental input strictly local functions. *Transactions of the Association for Computational Linguistics 7*, 157-158.
- Chomsky, Noam and Morris Halle. 1968. *The Sound Pattern of English*. MIT Press.

References

- Gainor, Brian, Regine Lai, and Jeffrey Heinz. 2012. Computational characterizations of vowel harmony patterns and pathologies. In *The Proceedings of the 29th West Coast Conference on Formal Linguistics*, 63-71.
- Heinz, Jeffrey. 2011. Computational phonology part II: Grammars, learning, and the future. *Language and Linguistics Compass*, 5 (4): 153-168.
- Heinz, Jeffrey and Regine Lai. 2013. Vowel harmony and subsequentiality. In Andras Kornai and Marco Kuhlmann, editors, *Proceedings of the 13th Meeting on the Mathematics of Language (MoL 13)*, 52-63.
- Johnson, C. Douglas. 1972. *Formal Aspects of Phonological Description*. The Hague: Mouton.
- Kaplan, Ronald, and Martin Kay. 1994. Regular models of phonological rule systems. *Computational Linguistics 20*. 331-78.

References

- McCollum, Adam G., Eric Baković, Anna Mai, and Eric Meinhardt. (under review) The expressivity of segmental phonology and the definition of weak determinism.
- Oncina, José, Pedro García, and Enrique Vidal. 1993. Learning subsequential transducers for pattern recognition interpretation tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence 15*(5): 448-457.
- Oncina, José and Miguel A. Varó. 1996. Using domain information during the learning of a subsequential transducer. *Lecture Notes in Computer Science - Lecture Notes in Artificial Intelligence*, 313-325.
- Prince, Alan, and Smolensky, Paul. 1993. Optimality Theory: Constraint interaction in generative grammar. Technical Report CU-CS-696-93, Department of Computer Science, University of Colorado at Boulder.
- Sohn, Ho-min. 1975. *Woleaian Reference Grammar*. University of Hawaii Press.