LIN 629 Learnability Presenter: Ji Yea Kim
November 28, 2017

Input Strictly Local Opaque Maps

Chandlee, Heinz, and Jardine (to appear in *Phonology*)

1. Introduction

- Extension: opaque generalizations provided by Bakovic (2007)
- Proposal: - opaque generalizations have the property of being Input Strictly Local (ISL).
  - computational nature of opacity is as simple as single processes.
  - by transformation (relating underlying representations to surface representations; input-out map)

2. Background

2.1 Intensional and extensional descriptions

2.2 Input Strictly Local (ISL) functions

2.3 Finite-state characterization of ISL functions

- Finite-state transducer (FST): start state, transitions, final state (e.g., final deletion)
  e.g., FST mapping *tat* to *tat* (5) and *tata* to *tat* (6): five states needed (λ, t, a, ⋊, ⋉) in Fig. 3
  - If *a* is read, it is not written right away because it is unknown whether that *a* is word-final.
  - The ability to write λ allows the ISL FST to deterministically decide whether to write a V.

2.4 Properties of ISL functions

2.5 Relevance to phonological theory

- Restrictive class of maps--subclasses of both the regular and subsequential classes of maps

3. Opaque ISL maps

- 4 categories of opaque maps can be modeled with ISL functions (direct input/output map).
- Different *k*-values to account for the data
  - Cross-derivational feeding in Lithuanian: 2-ISL
  - Counterbleeding in Yowlumne: 3-ISL
  - Non-gratuitous feeding in Classical Arabic: 3-ISL
  - Fed Counterfeeding in Tundra Nenets: 3-ISL

3.1 Input/output tables

- Weakness of FST: as the number of states and transitions ↑, FST harder to read graphically
- Instead, an input/output table (list of the transitions of the ISL FST) used with same purpose

	1-suffix	Input	Output		1-suffix	Input	Output		1-suffix	Input	Output
a.	λ	⋊	⋊	e.	t	⋉	⋉	i.	a	t	at
b.	⋊	⋉	⋉	f.	t	t	t	j.	a	a	a
c.	⋊	t	t	g.	t	a	λ				
d.	⋊	a	λ	h.	a	⋉	⋉				

Table 1: Full input/output table for the ISL FST in Figure 3.

(1c)

⋊	t	a	t	⋉
⋊	t			

(1g)

⋊	t	a	t	⋉
⋊	t	λ		

(1i)

⋊	t	a	t	⋉
⋊	t	λ	at	

(1e)

⋊	t	a	t	⋉
⋊	t	λ	at	⋉

Table 2: Example derivation using an input/output table (the first step is omitted)

- Abbreviated input/output table: only shows cases where inputs do not match the outputs

(7) Danish lowering: $\varepsilon \rightarrow$ æ/ r __ ; e $\rightarrow \varepsilon$ / r __ ; i $\rightarrow$ e / r __

	1-suff	Input	Output
a.	r	i	e
b.	r	e	ɛ
c.	r	ɛ	æ

Table 4: Abbreviated input/output table for Danish lowering

⋊	i	r	i	⋉	⋊	i	r	i	⋉	⋊	i	r	i	⋉	⋊	i	r	i	⋉
⋊	i				⋊	i	r			⋊	i	r	e		⋊	i	r	e	⋉
												(4a)							

Table 5: Example derivation using Table 4

3.2 Cross-derivational feeding in Lithuanian (2-ISL: scanning a window of size 2)

- Opacity in which one process applies in order to avoid a derivation in which another process would create a marked structure
  (8a) [i]-epenthesis occurs between identical obstruents.
  (9a) Voiceless obstruents assimilate to following voiced obstruents.

(8) a. Epenthesis in Lithuanian (Baković, 2007)
$\emptyset \rightarrow$ i / K_1__K_2, where $K_1 = K_2$
b. /at-taiki:ti/→[atitaiki:ti][9], 'to make fit well'

(9) a. Voicing assimilation in Lithuanian (Baković, 2007; Odden, 2005)
K → [+voice] / __ D
b. /ap-gauti/ ↦ [abgauti], 'to deceive'

(10) Cross-derivational feeding in Lithuanian (Baković, 2005; Odden, 2005)
/ap-berti/ ↦ [apiberti], 'to strew all over'; *[abberti],*[abiberti]

→The application of epenthesis and not assimilation: overapplication to avoid *DD

- Solution to the interaction of epenthesis and assimilation: a single 2-ISL map
  - When K appears in the input, its output is delayed until it is determined for voicing assimilation (i.e., depends on the following segment). (6a, 6b, 6e, 6f)
  - If following segment is D of a *diff*. place: output as D & no epenthesis (6l, 6p)
    (i.e., voicing assimilation in heterorganic segments)
  - If following segment is obs. of the *same* place: both K and [i] are output (6q, 6s)
    (i.e., V-epenthesis of identical segments)
  - When a K is followed by a homorganic D: output as KiD (6r, 6t)

	1-suff	Input	Output		1-suff	Input	Output		1-suff	Input	Output
a.	⋊	t	λ	h.	V	b	b	o.	p	t	p
b.	⋊	p	λ	i.	t	V	tV	p.	p	d	bd
c.	⋊	d	d	j.	t	⋉	t⋉	q.	t	t	ti
d.	⋊	b	b	k.	t	p	t	r.	t	d	tid
e.	V	t	λ	l.	t	b	db	s.	p	p	pi
f.	V	p	λ	m.	p	V	pV	t.	p	b	pib
g.	V	d	d	n.	p	⋉	p⋉				

Table 6: Abbreviated input/output table for Lithuanian cross-derivational feeding

3.3 Counterbleeding in Yowlumne (3-ISL: scanning a window of size 3)

- (11a) All long Vs become [-high]; (11b) Vs in closed syllables shorten (i.e., (11b) bleeds (11a).)

(11) a. [+long] → [−high]
b. V → [−long] / __C {C,⋉}

- (12) Overapplication of (11a) because application of (11b) removes the triggering long V.

(12) Yowlumne (McCarthy, 1999)
/mi:k-hin/ ↦ [mekhin], 'swallowed'

- Solution to the interaction of lowering and shortening: a single 3-ISL map
  - When a high long V is read, the output is delayed (8a-h) until it is determined whether the following input is:
    - C: output still cannot be determined (8l-n)
    - C⋉ or CC: both lower and shorten the V (8p, 8q); output as [eC]
    - something else (not a shortening environment): output as [e:] (lowering only) (8i-k)
    - /V/ following /i:C/ (not a shortening environment): output as [e:CV] (8o)
      (i.e., the output [e:C] of /i:C/ 2-suff. concatenated to the output [V] for the new input)
    - Non-high long Vs (i.e., /V:/) are shortened without lowering (8r-8y).

	2-suff	Input	Output		2-suff	Input	Output		2-suff	Input	Output
a.	⋊C	i:	λ	j.	Ci:	V	e:V	s.	CV	V:	λ
b.	CC	i:	λ	k.	Vi:	V	e:V	t.	VC	V:	λ
c.	VC	i:	λ	l.	⋊i:	C	λ	u.	VC	V:	λ
d.	i:C	i:	λ	m.	Ci:	C	λ	v.	CV:	C	λ
e.	⋊V	i:	λ	n.	Vi:	C	λ	w.	VV:	C	λ
f.	CV	i:	λ	o.	i:C	V	e:CV	x.	V:C	C	VCC
g.	VV	i:	λ	p.	i:C	C	eCC	y.	V:C	⋉	VC
h.	i:V	i:	λ	q.	i:C	⋉	eC⋉				
i.	⋊i:	V	e:V	r.	CC	V:	λ				

Table 8: Abbreviated input/output table for Yowlumne counterbleeding

3.4 Non-gratuitous feeding in Classical Arabic (CA) (3-ISL: scanning a window of size 3)

- Structural description of one process is obscured by a second process that is fed by the first.
- In CA, (14a) vowel epenthesis before an initial CC.
  (14b) glottal stop epenthesis before initial vowels ((14a) feeds (14b).)

(13) Classical Arabic (McCarthy, 2007)
/ktub/ ↦ [ʔuktub], 'write.MascSg!'

(14) a. ∅ → V / # __ CC
b. ∅ → ʔ / #__V

- Solution to the interaction of V-epenthesis and ʔ-epenthesis: a single 3-ISL map
  - If the suffix is ⋊: input /V/ is output as [ʔV] (environment for ʔ-epenthesis) (10a)
    input /C/ is output as λ (delayed) (10b)
  - If the suffix is ⋊C: if /V/ follows, [CV] (V-epenthesis not applied) (10c)
    if /C/ follows, both V and ʔ epenthesis [ʔVCC] (10d)

	2-suff	Input	Output		2-suff	Input	Output
a.	⋊	V	ʔV	c.	⋊C	V	CV
b.	⋊	C	λ	d.	⋊C	C	ʔVCC

Table 10: Abbreviated input/output table for Classical Arabic epenthesis

3.5 Fed Counterfeeding in Tundra Nenets (3-ISL: scanning a window of size 3)

- (16a) Debuccalization (loss of original POA and becomes [ʔ]); (16b) vowel deletion

(15) Tundra Nenets (Kavitskaya and Staroverov, 2010)

a. /tasʌ/ ↦ [tas], 'whole'

b. /tʲimjʌs/ ↦ [tʲimjʔ], 'it rotted'

(16) a. {t, d, s, n, ŋ} → ʔ / __ #

b. ʌ → ∅ / __ (ʔ) #

- The same two rules can exhibit both feeding ((16a) feeds (16b) for (15b)) and counterfeeding ((16b) counterfeeds (16a) for (15a) *taʔ), depending on the input form.

- Solution to the interaction of V-epenthesis and ʔ-epenthesis: a single 3-ISL map
  - If the input is T = {t, d, s, n, ŋ}, output delayed (uncertain whether word-final) (12f-12i)
  - If the input is ʌ, output delayed (uncertain whether word-final) (12k)
  - If the 2-suffix ends in T, and the input is T: output as T (because non-final) (12j)
    the input is ʌ: also output as T (because non-final) (12e)
  - V-deletion applied when in word-final position (12t-12w) or before final ʔ (12s)
  - Both deletion and debuccalization (12n) (like the feeding relation between (16a) and (16b))
  - Only debuccalizaiton (12o-12r): 2-suffixes CT, VT, TT, ʔT followed by ⋉ (like (16a))
  - Only deletion (12s): because V deletes before the final ʔ (like (16b))

	2-suff	Input	Output		2-suff	Input	Output		2-suff	Input	Output
a.	XC	ʌ	λ	i.	Xʌ	T	λ	q.	TT	⋉	ʔ⋉
b.	XV	ʌ	λ	j.	XT	T	T	r.	ʔT	⋉	ʔ⋉
c.	Xʔ	ʌ	λ	k.	Xʌ	ʔ	λ	s.	ʌʔ	⋉	ʔ⋉
d.	Xʌ	ʌ	ʌ	l.	Xʌ	C	ʌC	t.	Cʌ	⋉	⋉
e.	XT	ʌ	T	m.	Xʌ	V	ʌV	u	Vʌ	⋉	⋉
f.	XC	T	λ	n.	ʌT	⋉	ʔ⋉	v.	Tʌ	⋉	⋉
g.	XV	T	λ	o.	CT	⋉	ʔ⋉	w.	ʔʌ	⋉	⋉
h.	Xʔ	T	λ	p.	VT	⋉	ʔ⋉				

Table 12: Abbreviated input/output table for Tundra Nenets fed counterfeeding

3.6 Discussion

- Two important questions
  - What will the *k*-value of the map that describes their interaction be? Can we predict the *k*-value of the interaction of two ISL processes?
    - No, the composition of two ISL processes is not simply the largest of their respective *k*-values, nor their sum.

§	Opacity Type	Language	Process A	Process B	Interaction
3.2	cross-derivational feeding	Lithuanian	$k = 2$	$k = 2$	$k = 2$
3.3	counterbleeding	Yowlumne	$k = 1$	$k = 3$	$k = 3$
3.4	non-gratuitous feeding	Classical Arabic	$k = 3$	$k = 2$	$k = 3$
3.5	fed counterfeeding	Tundra Nenets	$k = 2$	$k = 3$	$k = 3$
5.1	counterfeeding on environment	Bedouin Arabic	$k = 3$	$k = 3$	$k = 3$
5.2	counterfeeding on focus	Bedouin Arabic	$k = 3$	$k = 3$	$k = 3$
5.3	self-destructive feeding	Turkish	$k = 3$	$k = 4$	$k = 5$

Table 14: The *k*-values for the ISL maps analyzed in §3

    - Is the class of ISL functions closed under composition? (= Is the composition of any two ISL functions guaranteed to also be ISL?)
        - No, subsets of ISL class may be...

4. Comparison to other theories of phonology
4.1 Points of comparison: generation and recognition, learnability, typology
4.2 Generation and recognition

- Whether the correct output is generated from a given input, and vice versa.
- Rule-based: have solutions (evidence: all maps for phonology can be expressed as a list of ordered SPE-style rewrite rules.)
- OT: have solutions provided some conditions are met.; lack a comprehensive solution

4.3. Learnability

- Rule-based: some results but not strong; more phonology-specific information needed
- OT: despite variants of the core OT theory, no results for the diversity of opaque maps

4.4 Typology

- Predictions for the kinds of maps that should and should not occur
- Rule-based: do not undergenerate; regular; describe with an ordered list of rules
- OT: undergenerate many opaque maps → many adjustment to OT made
- ISL undergenerates: Iterative spreading and unbounded C and V harmony (solution: Output SL)
  LD phonotactic phenomena (solution: SP, TSL) … and combination
    - Compared to OT: no special modification needed; subregular properties such as input strict locality speak directly to the computational nature of phonology