Cyclic opacity facilitates phonological interpretation

Ewan Dunbar
Laboratoire de Sciences Cognitives et Psycholinguistique, ENS/EHESS/CNRS

April 4, 2014
GLOW 37, Brussels

1 Some quirky properties of human language

(1) **Cyclic computation**

(i) ***Inside-outness***

Phonology:

[[[noun]	[class]]$_N$	[[sci.ence]	[news]]$_N$]$_{NP}$
[[[1]	[1]]$_N$	[1 0]	[1]$_N$]$_{NP}$
[[[1]	[2]]$_{\mathbf{N}}$	[[1 0]	[2]$_{\mathbf{N}}$]$_{NP}$
[[[2]	[3]]$_{\mathbf{N}}$	[[1 0]	[3]$_{\mathbf{N}}$]$_{\mathbf{NP}}$

but not

[[[noun]	[class]]$_N$	[[sci.ence]	[news]]$_N$]$_{NP}$
[[[1]	[1]]$_N$	[[1 0]	[1]$_N$]$_{NP}$
[[[2]	[2]]$_{\mathbf{N}}$	[[1 0]	[1]$_{\mathbf{N}}$]$_{\mathbf{NP}}$
[[[2]	[2]]$_{\mathbf{N}}$	[[1 0]	[3]$_{\mathbf{N}}$]$_{NP}$

Syntax:

T	*v*	V	NP
		[V	NP]
	[*v*+*V*	[~~V~~	NP]]
[T+*v*+*V*	[~~*v*+*V*~~	[~~V~~	NP]]]

but not

T	*v*	V	NP
	[*v*		]
[T+*v*	[~~*v*~~		]]
		[V	NP]
[T+*v*	[~~*v*~~	[V	NP]]]

(ii) ***Cyclic opacity***

Phonology:

	[[tórna] u]	[[[triómf] ál] ízm]
	[[tórna] u]	**[[[triómf] ál]** ízm]
Glide (after vowel)	**[**tórna**] w]**	–
Destress (if not last str.)	–	[[[tri**o**mf] ál] ízm]
Reduction (unstressed)	**[**tórnə] w]	[[[tri**u**mf] ál] ízm]
		[[[triumf] ál] ízm]
GF	***note!*** →	–
Des.		**[[[**triumf] al] ízm]
Red.		**[[[**triumf] əl] ízm]
	tórnəw	triumfəlízm

not

[[[triómf] ál] ízm]
[[[triómf] ál] ízm]
–
[[[tri**o**mf] ál] ízm]
[[[tri**u**mf] ál] ízm]
[[[triumf] ál] ízm]
[[[tri**w**mf] al] ízm]
[[[triwmf] al] ízm]
[[[triwmf] əl] ízm]
triwmfəlízm

Strict Cycle Condition in phonology (Kean 1974, Mascaró 1976): *"No reaching back"* (more later)

Syntax/morphology:
Strict Cycle Condition (Chomsky 1973), extension of the root (Chomsky 1995), i.e., matters of timing: "don't operate *completely* inside material you've already dealt with"; **not** locality effects like Subjacency, although these are generally folded under one single condition in syntax (e.g., Conditions on Transformations (59) [Strict Cycle] + (80) [Subjacency] = (81); Chomsky 1999 PIC)

(2) **Computationally distinct modules**
Phonology and syntax have very different properties, looked at from the point of view of what family of functions they belong to

(i) *Syntactic computations are mildly context sensitive*
e.g,. Cross-serial dependencies in Swiss German (Shieber 1985)

Jan	*säit*	*das*	*mer*	**em Hans**	*es huus*	**hälfed**	*aastriiche*
Jan	says	that	we	**DATIVE Hans**	ACCUS house	**helped**	paint

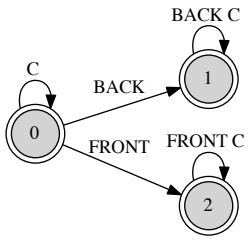
"Jan says that we helped Hans paint the house"

Jan	*säit*	*das*	*mer*	**de Hans**	*es huus*	**lönd**	*aastriiche*
Jan	says	that	we	**ACCUS Hans**	ACCUS house	**let**	paint

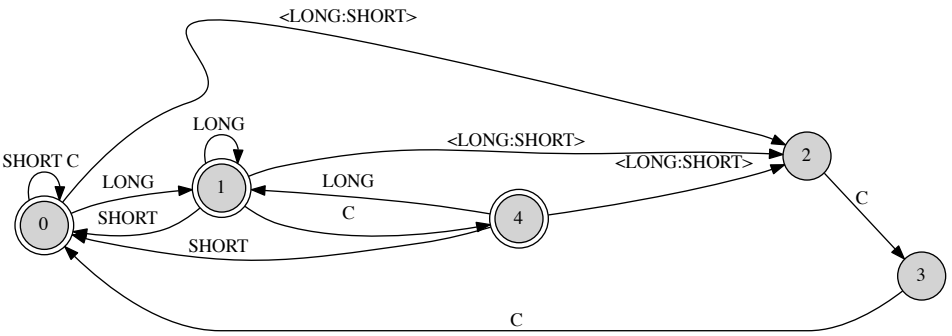
"Jan says that we let Hans paint the house"

blah	*blah*	*blah*	*we*	**N**	**N**	**V**	**V**
blah	blah	blah	we	1	2	1	2

(ii) *Phonological computations are always (sub) regular ("finite-state")*
(ii-a) static generalizations about surface forms (e.g., vowel harmony, like "don't have front vowels anywhere after back vowels and don't have back vowels anywhere after front vowels")



(ii-b) underlying–surface mappings (e.g., "shorten vowels in closed syllables"—or, simplifying a bit, "before two consonants")



(ii-c) any derivational combination of a **finite** number of these mappings (e.g., combine vowel shortening with consonant deletion)

	pi :tsa	panta	pi :nta
V→V̆/ — CC	p**i**tsa	–	p**i**nta
n → ϕ / — t	–	pa**<n>**ta	pi**<n>**ta
	pitsa	pata	pita

	pi :tsa	panta	pi :nta
n → ϕ / — t	–	pa**<n>**ta	pi :**<n>**ta
V→V̆/ — CC	p**i**tsa	–	–
	pitsa	pata	pi :ta

Finite-state tranducers are closed under finite composition:

1. Convert each rule to a transducer like the one in (ii-b)
2. Generate a **single big finite state transducer** that corresponds to feeding the output of R1 to R2 and so on

You can always do this and still have a finite-state transducer. Pulling out the rules implicit in the transducers corresponding to the two orders above, for example, we get:

(Both orders) n → ϕ / — t

(Left order) V→V̆/ — $\left\{ \begin{array}{ll} C\left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle t & \\ \left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle t & \\ n & \text{any C but t or } \left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle \\ \text{any C but n} & C \end{array} \right\}$

(Right order) V→V̆/ — $\left\{ \begin{array}{ll} C\left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle t & \\ \left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle t\left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle t & \\ \left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle tC & \\ n & \text{any C but t or } \left\langle \begin{matrix} n \\ \emptyset \end{matrix} \right\rangle \\ \text{any C but n} & C \end{array} \right\}$

(3) **Limits to regular computations**
Finite-state devices only have finite memory. So there isn't any finite-state device that can do the tasks of, e.g.,
Matching arbitrary pairs of brackets:
[. . . [. . . [. . .] . . .] . . .] , [. . . [. . . [. . . [. . .] . . .] . . .] . . .] , etc.,
vs. *[. . . [. . . [. . .] . . .] . . . , *[. . . [. . . [. . .] . . .] . . .] . . .]

Tracking arbitrary nested dependencies:
[The man [[the dog [the cat bit <the dog>]] chased <the man>]] cried at his misfortune

anti-missile missile , anti [*anti-missile missile*] *missile , anti* [*anti* [*anti-missile missile*] *missile*] *missile , . . .*

Reversing or copying arbitrary length strings:
pittip , pitaatip , etc., vs. *pitip , *pitatip , *pitaip

Tracking arbitrary cross-serial dependencies:
e.g., Swiss German above

(4) **Phonology is regular:** every known phonological pattern is regular (assuming reduplication is an extra-phonological operation): Johnson 1972, Kaplan and Kay 1994, Heinz 2007, Heinz 2011, Heinz and Idsardi 2011; thus by the above every known phonological **grammar** is regular

(5) **Except that:** if you were to take inside-out (cyclic) application seriously, then you would **not** have a finite composition of finite-state transducers; you would have an unbounded composition of finite state transducers, and it would not necessarily be regular anymore (the problem is with cases where the same rule could reapply to its output from a previous cycle—examples below); Kaplan and Kay pointed this out and left it as an open question as to how to deal with it

(6) **Possible clues to biological underpinnings:** Birdsong appears to be strictly regular (Berwick et al. 2011); the corresponding artificial grammar learning experiments showing failure to learn context-free (i.e., non-regular) patterns by birds and monkeys, but not humans, are still the subject of debate; see Fitch and Friederici 2012 for an optimistic discussion, Hochmann, Azadpour, and Mehler, 2008 for criticism

(7) **Today's (weak) claim:** Adding inside-outness to regular relations would make them super-regular; **the Strict Cycle Condition turns the composition of a potentially unbounded number of transducers into the composition of a finite number of transducers**

2 The phonological cycle

Chomsky, Halle, and Lukoff 1956, Chomsky and Halle 1968:

(8)

	[[a nec dote] al]
	[**[0 0 0]** 0]
Main Stress Rule	[[0 0 1] 0]
Alternating Stress Rule	[[1 0 2] 0]
Bracket Erasure	[1 0 2 0]
	[1 0 2 0]
Main Stress Rule	[2 0 1 0]
Alternating Stress Rule	–
Bracket Erasure	2 0 1 0
Vowel Reduction	à nəc dót əl
	à nəc dót əl

In parallel, Kiparsky (1973) noted that some processes seem to banned unless they appear in a "derived" environment:

(9) **Ruki rule (Sanskrit):** $s \rightarrow ṣ$ / $\{r, u, k, i\}$ —

/si + snih/ →[siṣnih], "be sticky (perf.)"
(*morpheme boundary*)

/śaːs + ta/ → [śista] (ablaut) → [śiṣta] (ruki) → [śiṣṭa] "taught"
(*environment changed by another process triggered by a morpheme boundary*)

/bisa/ → [bisa]
(*no ruki*)

Kean (1974) and Mascaró (1976), inspired by Chomsky (1973) connected this with the phonological cycle:

(10) **Strict Cyclicity (1974 version):**

Suppose we have $[_j[_i$ P X $_i]$ $Z_j]$. The domain of the *j*'th cycle is PXZ, where Z is the only material uniquely in *j*. No rule *R* may apply on the *j*th cycle unless it makes crucial reference to Z

Strict Cyclicity (1976 addendum):

. . . or information assigned on cycle *j* by a rule applying before *R*.

(11)

	[[tórna] u]	[raím + ét]	[[[triómf] ál] ízm]	[nó [ínstár]]
	[**[tórna]** u]	**[raím + ét]**	[[**[triómf]** ál] ízm]	[nó **[ínstár]**]
Glide (unstressed after vowel)	–	–	–	–
Destress (if not last stress in word)	–	[raim + ét]	–	[nó [instár]]
Reduction (if unstressed)	–	[rəim + ét]	–	–
	[tórna u]	rəim + ét	[**[triómf ál]** ízm]	**[nó instár]**
GF	[tórna w]		–	[nó jnstár]
Des.	–		[[triomf ál**]** ízm]	–
Red.	**[**tórnə w**]**		[[triumf ál**]** ízm]	–
	tórnə w		**[triumf ál ízm]**	nó jnstár
GF			–	
Des.			**[**triumf al ízm**]**	
Red.			**[**triumf əl ízm**]**	
	tórnəw	rəimét	triumfəlízm	nójnstár

(12) **Lexical phonology:** assumed all this; also proposed a division between the cyclic phonology and the post-cyclic/post-lexical phonology, where the cyclic phonology always had to obey an even stronger version of the SCC which would block any rule activity properly within a cycle (cycle 0 activity was somehow special, but the ideas are not quite aligned, since by now the system was understood to be "in the lexicon")

(13) **Modern times:** a move away from lexicalism, and back to cyclic operations "on the PF side"; cyclic morphological operations/Vocabulary Insertion assumed/argued for in Embick 2010, Bobaljik 2012

(14) **To syntacticians:** Remember, SCC (even SCC74) is a weaker condition than the opacity+locality principles in syntax that say "don't do anything down there at all"; Embick (2013) points to several phonological examples of reaching "down there" which are fine by the SCC but not if we add "escape hatch only" type locality.

(15) **Trigger "down there":**
Turkish Backness Harmony (left to right): [[[$\sqrt{\text{Root}}$] *n*] *a*], *n* is a phase head, $\sqrt{\text{Root}}$ is the complement of *n*, and the Phase Impenetrability Condition says "the complement of *n* is invisible to operations in the domain of *a*," yet, lo and behold:

[[[merhamet] ø] sIz] → merhametsiz, "compassionless"
[[[bilim] ø] sAl] → bilimsel, "scientific"

(16) **Target "down there":**
Turkana ATR Harmony (right to left): [[⋯ [[$\sqrt{\text{Root}}$] *v*]] *n*]

[[e [k [[ɪmʊj] ø]]] e] → ekimuje, "way of eating"
[[e [[cɪlɪcɪl] ø]] e] → ecilicile, "way of scratching"

(17) **Clearing up a point of possible confusion.** Although it looks like we are treating all brackets as equal, (you can see that) phonologists too distinguish cyclic/phasal nodes and treat the rest of the morphosyntactic structure as phonologically largely inactive (cf., sub-cyclic +/= boundary symbols used in SPE, affix levels in Lexical Phonology); this is orthogonal to the theoretical argument I'm about to make

3 Phonology as regular relations

(18) **Simplified Catalan.**

Glide Formation *Destressing*

$$\text{u}\rightarrow\text{w} \;/\; \left\{ \begin{matrix} \text{á}, \\ \text{a}, \\ \text{ú}, \\ \text{u} \end{matrix} \right\} \text{—} \qquad \begin{matrix} \text{ú} \rightarrow \text{u} \\ \text{á} \rightarrow \text{a} \end{matrix} \;/\; \text{—} \left\langle \left\{ \begin{matrix} \text{a}, \\ \text{u}, \\ \text{C} \end{matrix} \right\rangle_0 \left\{ \begin{matrix} \text{á}, \\ \text{ú} \end{matrix} \right\}$$

(19) **GF > Des.** These are two regular relations. Compose GF ∘ Des. The result (G) is still regular.

(20) **Two-level grammar.** Now that one process feeds another, we need to look at the "two-level grammar" to understand what is going on, as before (Kaplan and Kay 1994, Karttunen and Beesley 1992; see there also for the "same-length relation" trick, which avoids the problem whereby regular relations are not closed under intersection):

Glide Formation *Destressing*

$$\text{u}\rightarrow\text{w} \;/\; \left\{ \begin{matrix} \text{á}, \\ \text{a}, \\ \text{ú}, \\ \text{u}, \\ \left\langle \begin{matrix} \text{ú} \\ \text{u} \end{matrix} \right\rangle, \\ \left\langle \begin{matrix} \text{á} \\ \text{a} \end{matrix} \right\rangle \end{matrix} \right\} \text{—} \qquad \begin{matrix} \text{ú} \rightarrow \text{u} \\ \text{á} \rightarrow \text{a} \end{matrix} \;/\; \text{—} \left\langle \left\{ \begin{matrix} \text{a}, \\ \text{u}, \\ \left\langle \begin{matrix} \text{ú} \\ \text{u} \end{matrix} \right\rangle, \\ \left\langle \begin{matrix} \text{á} \\ \text{a} \end{matrix} \right\rangle, \\ \left\langle \begin{matrix} \text{u} \\ \text{w} \end{matrix} \right\rangle, \\ \text{C} \end{matrix} \right\rangle_0 \left\{ \begin{matrix} \text{á}, \\ \text{ú} \end{matrix} \right\}$$

In other words, we've broken down the relation into two regular languages of *strings of pairs* (i.e., input–output correspondences), i.e. the languages where the pairs $\left\langle \begin{matrix} \text{u} \\ \text{w} \end{matrix} \right\rangle$ / $\left\langle \begin{matrix} \text{á} \\ \text{a} \end{matrix} \right\rangle$ / $\left\langle \begin{matrix} \text{ú} \\ \text{u} \end{matrix} \right\rangle$ occur (as opposed to pairs where the left side does not change to the given segment) if and only if they appear in the given environment—where now the environment can have other non-identity pairs in it. We get back *G* just by taking the intersection of the two.

(21) **A note on interactions:** We have a change $\left\langle \begin{matrix} \text{u} \\ \text{w} \end{matrix} \right\rangle$, but not $\left\langle \begin{matrix} \text{ú} \\ \text{w} \end{matrix} \right\rangle$, because of the order of rule application (the only potential interaction here)

(22) **A note on reapplication:** Destressing is a "left to right reapplying" rule (because it affects *all* the non-final stressed vowels, not just one: see Kenstowicz and Kisseberth 1979); this means that one *single* application (within a single cycle) actually needs to have pairs in its environment (that's where $\left\langle \begin{matrix} \text{á} \\ \text{a} \end{matrix} \right\rangle, \left\langle \begin{matrix} \text{ú} \\ \text{u} \end{matrix} \right\rangle$ come from). One-shot reapplication does *not* cause the kind of problems that cyclic reapplication does.

(23) **A note for phonologists:** These two-level grammar rules can be seen as filters on input-output correspondences. In this respect two-level grammars are a lot like OT constraints; the only thing that differs is the mode by which they are combined (OT doesn't use intersection: see Eisner 1997, Riggle 2009). Yet we can state any phonological grammar like this, with or without opaque interactions. That's because isn't the "state things as two-level filters" part of OT that gives problems with opacity; the culprit is the (informal and sometimes weakened) "use *output* filters" part of "Classical" OT (see Buccola 2013 for discussion).

(24) **The trouble with the cycle.** The SPE (unrestricted) cycle, given a relation R:

1. Modify R so that it only affects content between the innermost pair(s) of brackets, inclusive (and otherwise leaves everything as is) $\rightarrow R_0$
2. E is Bracket Erasure, which deletes (or marks as invisible) any pair of brackets with no brackets inside
3. Cycle 0 is just R_0; cycle 1 is R_0ER_0; cycle 2 is $R_0ER_0ER_0$; and so on

(25) **The most important thing.** Matching the brackets is not the problem—this is not our job, this is the job of the morphology/syntax—

(26) **But.** There is still a problem.

(27) **Example.** b→c / —$\left\{ \begin{array}{c} \text{c} \\ \text{a} \end{array} \right\}$

[$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a] *[ba]

[b $\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a] *[bba], *[$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle \left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a], *[b $\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle \left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a], *[$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle \left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle \left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a]

[b $\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ [$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a]] *[bb [$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a]]

[b [$\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle \left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a]] *[b [b $\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ a]]

OK to do: $R_1 \cup R_2 \cup \cdots \cup R_k$
Not OK in general: $R_1 \cup R_2 \cup \cdots$
In this case, cyclic reapplication forces us to match the $\left\langle \begin{array}{c} \text{b} \\ \text{c} \end{array} \right\rangle$ elements to the bracketing in the input; can easily show that this language is not regular.

(28) **0-cyclic convergence.** $Cyc_0(R,i)$ is $R_0(ER_0)^i$ as above. $Cyc_0(R)$ is the countable union of the $Cyc_0(R,i)$ which leave no nested brackets behind (i.e., leaving out the sets of pairs where $Cyc_0(R,i)$ applies but there are $> i$ cycles in the input). R is 0-cyclic convergent if $R_0ER_0 = ER_0$.

(29) ***Claim.*** 0-cylic convergent regular relations have regular $Cyc_0(R)$. *Proof.* Suppose $R_0ER_0 = ER_0$. Provided the domain of R is not restricted (i.e., it is Σ^*), so that every output of ER_0 is also a valid input for R_0, we have $Cyc_0(R,2) = R_0ER_0ER_0 = ER_0ER_0 = EER_0$. By an easy induction $Cyc_0(R,i) = E^iR_0$. Since the countable union of E^i over inputs with i cycles just deletes all but the outermost brackets, it obviously corresponds to a regular relation E_0, and $Cyc_0(R) = E_0R_0$.

(30) **Brackets.** Obviously erasing brackets will not do if we are working with derived environment effects, where grammars are sensitive to the morphological structure. The SCC presupposes a slightly different version of the cycle. However, as long as we *sometimes* erase brackets, we can get a similar result.

(31) **k-cyclic application.** Define $Cyc_k(R,i)$ as $R_k(ER_k)^i$, where R_k is R applied to only the innermost $k+1$ levels of nesting. The way k-cyclic application works, rather than deleting all but the outer $1(=0+1)$ pairs of brackets, we delete brackets—the *one* innermost pair—only after the k'th cycle, and leave behind the outer $k+1$ pairs of brackets for R_k to see.

(32) **k-cyclic convergence.** $Cyc_k(R)$ is the countable union (over i, not k) of the subsets of $Cyc_k(R,i)$ which leave no nested brackets behind, analogous to the above. R is k-cyclic convergent if $R_k(ER_k)^i = R_{k+i}E^i$.

(33) ***Note.*** k-cyclic convergent regular relations have regular $Cyc_k(R)$. Notice that k is fixed and, for the strings in $Cyc_k(R)$, $k+i$ is always the outermost cycle and E^i simply deletes all but the outermost k brackets.

(34) **What this means.** We can construct grammars which make crucial use of up to k brackets, for some fixed k; as long as they are k-cyclic convergent, then there is a corresponding cyclic grammar. We need $k = n$ to get a not-too-fuzzy statement of the SCC, where n is the number of distinct segmental non-identity input–output pairs licensed by the grammar

(35) **Example.** b→c / —[$\left\{ \begin{matrix} \text{c} \\ \text{a} \\ \left\langle \begin{matrix} \text{b} \\ \text{c} \end{matrix} \right\rangle \end{matrix} \right\}$

[ba] *[$\left\langle \begin{matrix} \text{b} \\ \text{c} \end{matrix} \right\rangle$ a]

[[$\left\langle \begin{matrix} \text{b} \\ \text{c} \end{matrix} \right\rangle$] a] *[[b] a]

[$\left\langle \begin{matrix} \text{b} \\ \text{c} \end{matrix} \right\rangle$ [$\left\langle \begin{matrix} \text{b} \\ \text{c} \end{matrix} \right\rangle$ [a]]

4 Formalizing strict cyclicity

(36) **Strict Cyclicity:**

Given some relation Q (the original grammar), get R (the SCC-compatible grammars) as follows:

1. Consider Q as a two-level grammar. Let T_0 be the set of all non-identity pairs τ in any pair-string in the language. For each τ, E^1_τ is the set of all pair-strings containing a special symbol $\star$ where, if the pair τ occurs, it is always in the position of $\star$ in some pair-string of E^0_τ, and, if any element of E^1_τ occurs as a substring (i.e., with any material whatsoever after the left or right edges) with some pair σ having the same upper symbol of τ, a, in place of $\star$, then $\sigma = \tau$.
2. Let $n = |T_0|$. **Construct T as the set of all non-identity pairs in any pair-string in Q^i for any $i \leq n$.** Construct E^i_τ as the set of all non identity pairs in any pair string in Q^i for any $1 < i \leq n$. No new pairs (note: pairs—not environments) will be found in Q^i for $i > n$, because additional changes not present in Q can only arise as the result of τ being composed with some other change σ, or with the composition with two changes, etc., but no more than $n-1$ can be distinct. Add generalizations to any sequences in an environment drawn from Q^i wzx, where and zx but not wzx is in Q^{i-1}, and w differs from z only by containing $\left\langle \begin{matrix} ? \\ a \end{matrix} \right\rangle$ where z has $\left\langle \begin{matrix} a \\ a \end{matrix} \right\rangle$ (on the right; or vice versa on the left): generalize by adding wx to the set of environments. Now, for each τ:
   (a) for all the environments e^1 in E^1_τ, let e^1_λ be the string concatenated with a left bracket at the left edge and no other left brackets appearing on the left side of $\star$, and at least one left bracket on the right side of $\star$ ...
   (b) ... e^1_ρ the string concatenated with a right bracket at the right edge, with no other right brackets to the right of $\star$ and at least one right bracket on the left side of $\star$
   (c) for all the environments e^2 in E^2_τ, let e^2_λ be the string with at least one left bracket inserted to the left of $\star$, but not only at the left edge of the string, and left brackets freely ignored on the right side of $\star$; for e^i in E^i_τ, require at least $i-1$ left brackets ...

(d) … let e^2_ρ be the string with at least one right bracket (at least $i-1$ for e^i) inserted to the right of $\star$, but not only at the right edge of the string, and right brackets freely ignored on the left side of $\star$.

(e) *(To handle the phonological derived environment cases):* for each of the environments in E^1_τ which itself contains a non-identity mapping σ at position $\star\star$, add an environment which consists of e^1_λ / e^1_ρ, except with no requirement for a bracket on the inner side, and with both $\star$ and $\star\star$ on the inside of the necessary bracket, intersected with each of the (c–d) environments for σ. The resulting environments always fall into case (c–d), so this does not change the proof.

3. **The union of all these is E_τ.**
4. **R is the intersection of all the languages derived from E_τ as in 1.**

(37) **Proof of n-cyclic convergence.** Not actually required to show regularity, but important nevertheless: in (very) short, we essentially need to show that if $\left\langle \begin{smallmatrix} w \\ x \end{smallmatrix} \right\rangle$ (here I now mean strings rather than just symbols) and $\left\langle \begin{smallmatrix} axb \\ cyd \end{smallmatrix} \right\rangle$ then $\left\langle \begin{smallmatrix} awb \\ cyd \end{smallmatrix} \right\rangle$. This has nothing to do with brackets: the changes in $\left\langle \begin{smallmatrix} w \\ x \end{smallmatrix} \right\rangle$ cannot be affected by their surroundings because of the way we have set up the language (the change(s) in $\left\langle \begin{smallmatrix} w \\ x \end{smallmatrix} \right\rangle$ need to have environments within $\left\langle \begin{smallmatrix} w \\ x \end{smallmatrix} \right\rangle$ that are satisfied, and would be satisfied *no matter what else we surround them with*); so $\left\langle \begin{smallmatrix} w \\ y \end{smallmatrix} \right\rangle$ only if there are some other changes surrounding the ones we already knew about which are licensed by some outside material. Whatever those other changes are, they must necessarily be the same ones we find in $\left\langle \begin{smallmatrix} x \\ y \end{smallmatrix} \right\rangle$, because otherwise we could not have y on the bottom. That means the whole central part is licit in the given environment. Once we know that the changes in the middle are licit, however, it means the whole string must be, otherwise we would never have obtained this environment for any of the middle part from applying the original grammar.

(38) **Intuition.** The construction is frankly not that interesting either. In fact, were we to remove the "generalize" clause from 2, we would not need derived environment blocking, because the environments we obtain for a given τ would only go up to those we find up to cycle k, which gives us a finite intersection. Funnily, we add this clause, we do not particularly *need* derived environment blocking either, because we have actually just created a reapplication (i.e., unbounding spreading) rule—

(39) **However** —this would then be at odds with the local pattern we have within stems (given Q). We observe that certain patterns are only possible via cyclic reapplication of the grammar (i.e., reapplication triggered by the addition of morphemes), and SCC is a way of capturing this without running afoul of regularity. The mechanism is merely to look at the morphological structure we are given in the input rather than trying to count reapplications (which we can't do).

(40) **When would this matter?** If, for example, spreading is only observed locally within words but to 2 segments when it crosses a morpheme boundary, then the learner will ceteris paribus allow it to occur across arbitrarily many morpheme boundaries—and with the (e) clause, it becomes unbounded within morphemes after cycle 0; if it is observed more than locally word-internally, then the learner may simply generalize it to unbounded spreading (within Q).

5 Some thoughts

(41) **Why SCC?** There are certainly other compatible solutions; for example, we might have derived environment blocking only for problematic processes! If we could show that the strategy was nevertheless in some sense "optimal" then we would have a very good candidate for an explanation: cyclic opacity exists to make phonology possible

(42) **Two possibilities for integrating the analysis with syntax:**
(i) *analogy:* there is a similar reason for cyclic opacity (or even locality) effects in syntax—making do with the mechanisms we have available—(distinct from "reduction of computational burden" in the normal sense)—which similarly allows us a "moving window" analysis; but the need for phonological interpretation is not the limiting factor
(ii) *reduction:* cyclic opacity effects in syntax are *driven* by the to phonological interpretation: phonological interpretation requires opacity, thus syntax requires opacity

(43) **The trouble with reduction.** Syntax is not regular, and so the operations that need restricting (movement) are not even in the domain of what we are talking about

(44) **The trouble with analogy.** There is then no real reason to think that the cyclic nodes should be in any way the same in morphosyntax/phonology (which is maybe fine)

Acknowledgments

Work was supported in part by the European Research Council under the European Union's Seventh Framework Programme (FP/2007-2013) / ERC Grant Agreement ERC-2011-AdG-295810 BOOTPHON, from the Agence Nationale pour la Recherche (ANR-2010-BLAN-1901-1 BOOTLANG) and from the Fondation de France, ANR-10-IDEX-0001-02 and ANR-11-LABX-0087.

References

Berwick, Robert C., Kazuo Okanoya, Gabriel J.L. Beckers, and Johan J. Bolhuis (Mar. 2011). "Songs to syntax: the linguistics of birdsong". In: *Trends in Cognitive Sciences* 15.3, pp. 113–121. ISSN: 13646613. DOI: 10.1016/j.tics.2011.01.002.

Bobaljik, Jonathan David (2012). *Universals in Comparative Morphology*. Cambridge, MA: MIT Press.

Buccola, Brian (2013). *Two proofs that classic Optimality Theory is expressively weaker than ordered rewrite rules*. Unpublished ms, McGill. Montreal.

Chomsky, Noam (1973). "Conditions on transformations". In: *A Festschrift for Morris Halle*. Ed. by Stephen Anderson and Paul Kiparsky. New York: Holt, pp. 232–286.

– (1995). *The Minimalist Program*. Cambridge, MA: MIT Press.

– (1999). "Derivation by phase". In: *MIT Occasional Papers in Linguistics* 18.

Chomsky, Noam and Morris Halle (1968). *The Sound Pattern of English*. Harper and Row.

Chomsky, Noam, Morris Halle, and Fred Lukoff (1956). "On accent and juncture in English". In: *For Roman Jakobson*, 65–80.

Eisner, Jason (1997). "Efficient generation in primitive Optimality Theory". In: *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 313–320.

Embick, David (2010). *Localism versus Globalism in Morphology and Phonology*. Cambridge, MA: MIT Press.

– (2013). “Phase cycles, φ-cycles, and phonological (in) activity”. In: *Manuscript, University of Pennsylvania. Forthcoming in Festschrift for Jean Lowenstamm*.

Fitch, W. T. and A. D. Friederici (July 2012). “Artificial grammar learning meets formal language theory: an overview”. In: *Philosophical Transactions of the Royal Society B: Biological Sciences* 367.1598, pp. 1933–1955. ISSN: 0962-8436, 1471-2970. DOI: 10.1098/rstb.2012.0103.

Heinz, Jeffrey (2007). “Inductive learning of phonotactic patterns”. PhD Thesis. UCLA.

– (Apr. 2011). “Computational Phonology - Part I: Foundations”. In: *Language and Linguistics Compass* 5.4, pp. 140–152. ISSN: 1749818X. DOI: 10.1111/j.1749-818X.2011.00269.x.

Heinz, Jeffrey and William Idsardi (July 2011). “Sentence and Word Complexity”. In: *Science* 333.6040, pp. 295–297. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1210358.

Hochmann, Jean-Remy, Mahan Azadpour, and Jacques Mehler (Sept. 2008). “Do Humans Really Learn An Bn Artificial Grammars From Exemplars?” In: *Cognitive Science: A Multidisciplinary Journal* 32.6, pp. 1021–1036. ISSN: 0364-0213. DOI: 10.1080/03640210801897849.

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

Kaplan, Ronald M. and Martin Kay (1994). “Regular models of phonological rule systems”. In: *Computational Linguistics* 20.3, 331–378.

Karttunen, Lauri and Kenneth Beesley (1992). *Two-level rule compiler*. Xerox PARC Technical Report ISTL-92-2. Palo Alto, California.

Kean, Mary-Louise (1974). “The Strict Cycle in Phonology”. In: *Linguistic Inquiry* 5, pp. 179–203.

Kenstowicz, Michael and Charles Kisseberth (1979). *Generative Phonology*. San Diego: Academic Press.

Kiparsky, Paul (1973). “Phonological representations”. In: *Three Dimensions of Linguistic Theory*. Ed. by Osamu Fujimura. Tokyo: TEC.

Mascaró, Joan (1976). “Catalan Phonology and the Phonological Cycle”. PhD Thesis. Massachusetts Institute of Technology.

Riggle, Jason (2009). “Violation semirings in Optimality Theory”. In: *Language and Computation* 7, pp. 1–12.

Shieber, Stuart (1985). “Evidence against the context-freeness of natural language”. In: *Linguistics and Philosophy* 8, pp. 333–343.