The Typology of Repetition Avoidance Patterns in Indo-European Reduplication

Sam Zukoff, Leipzig University
samuel.zukoff@uni-leipzig.de · www.samzukoff.com

Linguistics Research Seminar, University of Lausanne
May 4, 2021

1 Introduction

- A number of the ancient Indo-European (IE) languages display a typologically unusual alternation in reduplication, as exemplified by the data from Gothic in (1), relating to the treatment of cluster-initial bases:
  - Bases beginning in obstruent-sonorant (TR) clusters copy just the **first consonant** (1a)
  - Bases beginning in *s*-obstruent (ST) clusters *do something else*; in Gothic, they copy **both consonants** (1b)

(1) **Reduplicated cluster-initial bases in Gothic** (Lambdin 2006:115)

a. **C_1-copying reduplication** ⇔ *obstruent-sonorant (TR) clusters*

	Infinitive	Preterite	
'weep'	*grēt-an*	*ge-grōt*	(not *[gre-grōt])
'sleep'	*slēp-an*	*se-slēp*	(not *[sle-slēp])
'bewail'	*flōk-an*	*fe-flōk*	(not *[fle-flōk])
'tempt'	*frais-an*	*fe-frais*	(not *[fre-frais])

b. **Cluster-copying reduplication** ⇔ *sibilant-stop (ST) clusters*

	Infinitive	Preterite	
'possess'	*stald-an*	*ste-stald*	(not *[se-stald])
'divide'	*skaið-an*	*ske-skaiθ*	(not *[se-skaiθ])

- When looking around the IE languages, we find two dimensions of variability relating to this kind of reduplicative alternation:

(2) **Dimensions of variation**
  a. What alternative (i.e. non-C_1-copying) reduplication pattern do the ST-clusters show?
  b. Which cluster types pattern with the ST-clusters and which pattern with the TR-clusters?

→ In this talk, I'll explore the first of these questions, and develop explanations for the resulting (micro-)typology:

§3 Explain the main alternative patterns that arise, motivated primarily by a new constraint: ***PCR**.
§4 Confirm that the factorial typology of just a few Optimality-Theoretic constraints provides a good fit to the IE data, including capturing two attested patterns of non-alternation.
§5 Sketch the analyses of two other IE reduplication patterns, which both involve infixation driven by *PCR. [*time permitting*]

⋆ On the second question, see Zukoff (2017a:Ch. 6):
  - The different cluster-wise distributions across the languages is explained by formalizing *PCR in terms of acoustic/auditory cues to contrast (see Wright 2004), namely, *intensity rise* (Parker 2002, 2008).

2 A brief introduction to OT and Correspondence Theory

- Before moving on to the analysis, I will introduce and clarify the mechanics of Optimality Theory (OT; Prince & Smolensky [1993] 2004), which I will be using to analyze the reduplication patterns.
  - I will first give an overview of the basic components of the theory.
  - And then I will review the extension of this theory used for the analysis of reduplication, namely, Base-Reduplicant Correspondence Theory (McCarthy & Prince 1995, 1999).

2.1 Basics of OT

- OT is a theory where rules are replaced with constraints and constraint interaction/evaluation.
  - ∗ Strictly speaking, it is not a theory of *phonology*, but rather a theory of *computation*.

- There are three main architectural components of the theory:

(3) **GEN ("generator"):**
Produces all possible outputs related to the input.

(4) **CON ("constraints"):**
An ordered ("ranked") list of constraints regulating output structures (6) and input-output mappings (7).

(5) **EVAL ("evaluator"):**
Assigns constraint violations to outputs, and selects the output that has the least bad violation profile.

- There are two main types of constraints:

(6) **Markedness constraints:** penalize a specific structure in the output

(7) **Faithfulness constraints:** penalize a specific change between input and output

- Phonological processes (≈ changes to the input) occur only when a **markedness** constraint outranks a **faithfulness** constraint.

- Take, for example, an epenthesis process that splits up consonant clusters:

(8) **Cluster-breaking epenthesis**
   a. Ø → i / C_C
   b. CC → CiC

- The way that this process would be expressed in OT is that constraint against consonant clusters (9a) *outranks* the constraint against epenthesis (9b).

(9) **Constraints for cluster-breaking epenthesis**
   a. ***CLUSTER (*CC)** [≈ *COMPLEX] (*Don't have clusters!*)
   Assign a violation mark * for each sequence of two consonants in the output.
   b. **DEP-IO** [technically short for "Dependence: Input-Output"] (*Don't epenthesize!*)
   Assign a violation mark * for each output segment without a correspondent in the input.
   c. **Ranking:** *CC ≫ DEP-IO

- The analysis is demonstrated using a "tableau" as in (10):

(10) **Tableau for cluster-breaking epenthesis**

/ptako/	*CC	DEP-IO
a. ptako	*!	
b. ☞ p**i**tako		*

- Candidate (10a) is faithful to the input and retains the consonant cluster.
  - This causes a violation of *CC.
- Candidate (10b) is unfaithful to input because it epenthesizes into the cluster.
  - This causes a violation of DEP-IO.
- → Because *CC ≫ DEP-IO, (10a)'s *CC violation is worse than (10b)'s DEP-IO violation, and (10b) is selected as the optimal output.

2.2 Alternative repairs and factorial typology

- In OT, it is not sufficient to only consider the faithfulness constraints relating to the attested repair.
  - → We also need to consider faithfulness constraints relating to other possible ways of fixing the markedness problem.
- The main other way of fixing a cluster is deletion, which is regulated by the faithfulness constraint MAX-IO:

(11) **MAX-IO** [technically short for "Maximality: Input-Output"] (*Don't delete!*)
Assign a violation mark * for each input segment without a correspondent in the output.

- The repair attested by a phonological process is the one that violates the *lowest ranked faithfulness constraint*. This means that, in our hypothetical language, MAX-IO ≫ DEP-IO:

(12) **Tableau for cluster-breaking epenthesis, now with MAX-IO**

/ptako/	MAX-IO	*CC	DEP-IO
a. ptako		*!	
b. ☞ p**i**tako			*
c. tako	*!		

- In OT, the best way to demonstrate that you are using the right constraints is to consider the "factorial typology" (see, e.g., Kager 1999:34ff.).
  - The basic premise of OT is that languages vary principally in the ranking of their constraints.
  - It follows that all ranking permutations are possible, and should be evidenced by real languages.
  - → Therefore, if all of the languages predicted by the factorial permutation of your constraints are attested, then you've probably done a good job at defining your constraints.

- Taking our example about clusters, the factorial typology predicts three different languages (the relative ranking of the top two constraints never makes a difference):

(13) **Factorial typology of *CC, DEP-IO, and MAX-IO**

a. **Epenthesis languages:** /ptako/ → [pitako]
Rankings: {Max-IO ≫ *CC ≫ DEP-IO}, {*CC ≫ Max-IO ≫ DEP-IO} (DEP-IO lowest)

b. **Deletion languages:** /ptako/ → [tako]
Rankings: {Dep-IO ≫ *CC ≫ MAX-IO}, {*CC ≫ Dep-IO ≫ MAX-IO} (MAX-IO lowest)

c. **Cluster languages:** /ptako/ → [ptako]
Rankings: {Max-IO ≫ DEP-IO ≫ *CC}, {Dep-IO ≫ MAX-IO ≫ *CC} (*CC lowest)

- When we look at the languages of the world, we find all three of these types of languages:
  - (i) Languages that fix clusters through epenthesis,
  - (ii) Languages that fix clusters through deletion, and
  - (iii) Languages that tolerate clusters.

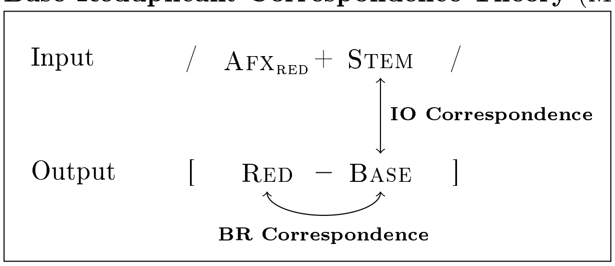
⋆ This means that our constraints accurately predict the typology in this domain, which is a good argument that this is the right sort of analysis.

→ In Section 3–4 below, I'll show that the factorial typology of the constraints I employ in the analysis of the IE reduplication patterns is a good match to the attested typology.

2.3 Basics of BRCT

- Thus far, the faithfulness constraints being considered have all been of the "Input-Output" variety, regulating changes between the input and the output.
- McCarthy & Prince (1995, 1999) proposed "Base-Reduplicant Correspondence Theory" (BRCT), which asserts that there are equivalent faithfulness constraints that regulate changes between base and reduplicant.
  - This is conceptualized in terms of "correspondence relations", as shown in (14):

(14) **Base-Reduplicant Correspondence Theory** (McCarthy & Prince 1995:4)



- All of these correspondence relations have the *same faithfulness constraints*, just defined over different relations. For example, faithfulness constraints over the BR-correspondence relation include:

(15) **BR-faithfulness constraints**

a. **MAX-BR:**
Assign a violation * for each segment in the base without a correspondent in the reduplicant.

b. **DEP-BR:**
Assign a violation * for each segment in the reduplicant without a correspondent in the base.

⋆ This allows for "the emergence of the unmarked" (TETU; McCarthy & Prince 1994) in reduplication:

→ Marked structures which are tolerated in bases can be repaired in reduplicants.

- Tableau (16) illustrates this with a hypothetical language that tolerates clusters outside of reduplication (MAX-IO, DEP-IO ≫ *CC), but fixes them with epenthesis in the reduplicant (*CC ≫ DEP-BR).
  - ∗ This is exactly the pattern I reconstruct for the precursor of "Attic Reduplication" in Pre-Greek (Zukoff 2017a,b). (I will briefly discuss this in Section 6.)

(16) **Reduplicant-internal epenthesis**

/RED, ptako/			MAX-IO	DEP-IO	*CC	DEP-BR
a.		pta-ptako			*!*	
b.	☞	p**i**ta-ptako			*	*
c.		pita-p**i**tako		*!		
d.		ta-tako	*!			

→ A number of aspects of the various IE reduplication patterns can be conceived of as this sort of TETU.

3 The typology of repetition avoidance patterns in IE reduplication

- Proto-Indo-European (PIE) expressed the verbal PERFECT by prefixal reduplication (see generally, e.g., Fortson 2010:103–104; for details, see Keydana 2006, Zukoff 2017a, *a.o.*).
  - ∗ PIE also had reduplication in other categories, but I will focus on the perfect.

- In all the daughter languages that retain this reduplication (as either the PERFECT or the PRETERITE), single-consonant-initial roots show a prefixal reduplicant in CV.
  - ◦ The consonant always corresponds to the base-initial consonant (C_1).
  - ◦ The languages differ on the nature of the vowel (more on this below).

(17) **Example of C_1V reduplication to C_1VX– root in Ancient Greek**
$\sqrt{}$*dō*- 'give' → PERF *de*-*dō*- 'have given'

⋆ However, the daughter languages show significant divergence in the behavior of cluster-initial roots.
→ In this section, I will show that we can model the full range of patterns using just *five* OT constraints.

3.1 Non-alternating patterns

- While most of the IE languages show differences in the behavior of different clusters, I will start by looking at patterns where all cluster types are treated the same, first schematically and then with the real data.
  - ◦ I will use these patterns to introduce the relevant constraints and show how they work.
  - ◦ I will then proceed to the patterns which show cluster-type sensitivity in the next subsection.

3.1.1 Across-the-board cluster-copying: Hittite

- The conceptually simplest reduplication pattern attested among the IE languages is what I will call "across-the-board cluster-copying", which copies the first base vowel and all consonants that come before it (18).
  - → This pattern is attested in Hittite (Dempsey 2015, Zukoff 2017a:Ch. 3, Yates & Zukoff 2018).

▷ In (18), subscripts in the "Red. Shape" column indicate which number segment of the base, counting from the left, each reduplicated segment corresponds to (via Base-Reduplicant correspondence; cf. McCarthy & Prince 1995, 1999).

(18) **Across-the-board cluster-copying**

	Base Type	Root		Reduplicated	Red. Shape
a.	Singleton	$\sqrt{}$*mako*	→	*ma*-*mako*	C_1V_2
b.	Stop-sonorant	$\sqrt{}$*prako*	→	*pra*-*prako*	$C_1C_2V_3$
c.	*s*-obstruent	$\sqrt{}$*stako*	→	*sta*-*stako*	$C_1C_2V_3$

- In the constraint system to be proposed, a CV reduplicant to a CVX– (i.e. singleton-initial) base is virtually perfect (i.e. no violations).

⋆ We only start encountering violations when we consider the actual and possible candidate outputs for cluster-initial bases.

- While the CCV reduplicants perfectly match their bases, they display a marked syllable structure, namely, a complex onset. In syllable-neutral terms, the constraint *CLUSTER (*CC) encodes this markedness.[1]

(19) ***CLUSTER (*CC)** (*Don't have clusters!*)
Assign a violation mark * for each sequence of two consonants in the output.

[1] In some of the full analyses in Zukoff (2017a), the effect of *CC is instead enforced by left-oriented alignment constraints (McCarthy & Prince 1993).

- For a language with the across-the-board cluster-copying pattern, this constraint must be *low*-ranked, because it is violated by the actual output (cf. (21a) below).
  → This means that there must be *high*-ranked constraint(s) that promote this kind of candidate.

- I employ two Base-Reduplicant (BR) faithfulness constraints that fit the bill:
  - Contiguity-BR (20a) requires contiguous copying from the base.
  - Anchor-L-BR (20b) requires copying that begins at the left edge of the base.

(20) **BR-faithfulness constraints that promote cluster-copying**

a. **Contiguity-BR** (*Copy a contiguous string!*)
Assign one violation mark * for each pair of segments that are adjacent in the reduplicant but have non-adjacent correspondents in the base (i.e. no $\underline{X_1X_3}$-$X_1X_2X_3$).

b. **Anchor-L-BR** (*Copy from the left edge!*)
Assign a violation mark * if the segment at the left edge of the reduplicant does not stand in correspondence with the segment at the left edge of the base.

- The *CC violation incurred by copying the whole cluster can be avoided by copying only one member of the cluster: either the first consonant (21b) or the second consonant (21c).

(21) **Generating across-the-board cluster-copying**

/red, prako/	Contiguity-BR	Anchor-L-BR	*CC
a. ☞ pra-prako			**
b. pa-prako	*!		*
c. ra-prako		*!	*

- However, each option violates one of these two constraints:
  - Candidate (21b) copies a discontiguous string, and thus violates Contiguity-BR.[2]
  - Candidate (21c) doesn't copy the leftmost segment of the base, and thus violates Anchor-L-BR.

→ Therefore, as long Contiguity-BR, Anchor-L-BR ≫ *CC, we select cluster-copying (21a) even though it violates *CC an extra time.

> ∗ In all the IE languages, consonant clusters are allowed outside of reduplication. Therefore, Max-IO and Dep-IO outrank *CC, and it is never optimal to repair the base-initial cluster. This means optimal candidates (such as (21a)) will always have at least one *CC violation.

- Hittite displays the across-the-board cluster-copying pattern (22). (Prothesis in STVX– bases (22b) is a general process in the language and not specific to reduplication.)

(22) **Across-the-board cluster-copying in Hittite** (Zukoff 2017a:Ch. 3, Yates & Zukoff 2018)

a. **TRVX– bases → cluster-copying**

Root		Reduplicated stem	
√*par(a)i-*	'blow'	*parip(p)ar(a)i-*	[pri-pːr(a)i-]
√*ḫal(a)i-*	'kneel'	*ḫaliḫal(a)i-*	[χli-χl(a)i-]

b. **STVX– bases → cluster-copying**

Root		Reduplicated stem	
√*stu-*	'become evident'	*išdušduške-*	[istu-stu-]

[2] This requires that the base vowel and the reduplicant vowel stand in correspondence, i.e., that the vowel not be a morphologically-fixed segment, as in Ancient Greek (see below).

3.1.2 Across-the-board C_1-copying: Old Irish (and elsewhere)

- The other across-the-board reduplicative behavior attested among the IE languages is "across-the-board C_1-copying": all reduplicants surface as CV, where the consonant corresponds to the base-initial C.
  - This pattern, which is equivalent to candidate (b) in tableau (21), is schematized in (23).

(23) **Across-the-board C_1-copying**

	Base Type	Root		Reduplicated	Red. Shape
a.	Singleton	√*mako*	→	*ma-mako*	C_1V_2
b.	Stop-sonorant	√*prako*	→	*pa-prako*	C_1V_3
c.	*s*-obstruent	√*stako*	→	*sa-stako*	C_1V_3

- This pattern is derived by simply swapping the ranking of *CC and CONTIGUITY-BR (24).
  - This ranking means that avoiding the extra cluster (24a) is worth doing discontiguous copying (24b).

(24) **Generating across-the-board C_1-copying**

/RED, prako/			ANCHOR-L-BR	*CC	CONTIGUITY-BR
a.		pra-prako		**!	
b.	☞	pa-prako		*	*
c.		ra-prako	*!	*	

- Across-the-board C_1-copying is attested in Old Irish (25). (The root-initial stops in the TRVX– roots undergo lenition (spirantization), but this is not transferred to the reduplicant.)

(25) **Old Irish reduplicated preterites** (Thurneysen [1946] 1980:424–428/§687–691)

a. **TRVX– roots → C_1-copying**

Root		Reduplicated preterite	
√*-glenn-*	'learn'	*-geglann*	[-ge-ɣlənn]
√*-grenn-*	'persecute'	*-gegrann*	[-ge-ɣrənn]
√*brag-*	'bleat'	*bebrag-*	[be-vrəɣ-]
√*klad-*	'dig'	*cechlad-*	[ke-xləð-]

b. **STVX– roots → C_1-copying**

Root		Reduplicated preterite	
√*skenn-*	'fly off'	*sescann-*	[se-skənn]

⋆ This pattern is also reconstructible to Pre-Greek (Zukoff 2017a:Ch. 2), and potentially other prior stages within the Indo-European family, including possibly PIE itself (Zukoff 2017a:Ch. 7).

3.1.3 Across-the-board C_2-copying: Unattested

- There is one more pattern that can be generated by permuting the ranking of these three constraints:[3]
  → The ranking *CC, CONTIGUITY-BR ≫ ANCHOR-L-BR predicts "across-the-board C_2-copying" (26), as demonstrated in (27).

[3] This again requires BR-correspondence for the vowels.

(26) **Across-the-board C_2-copying**

	Base Type	Root		Reduplicated	Red. Shape
a.	Singleton	√*mako*	→	*ma-mako*	C_1V_2
b.	Stop-sonorant	√*prako*	→	*ra-prako*	C_2V_3
c.	*s*-obstruent	√*stako*	→	*ta-stako*	C_2V_3

(27) **Generating across-the-board C_2-copying**

/RED, prako/		CONTIGUITY-BR	*CC	ANCHOR-L-BR
a.	pra-prako		**!	
b.	pa-prako	*!	*	
c. ☞	ra-prako		*	*

⋆ This is the only pattern predicted by the factorial typology not attested in IE (see Section 4 below).

3.2 Cluster-dependent copying patterns

- In the patterns discussed thus far, all base-initial clusters behave identically. While formally simplest and perhaps typologically most common, this behavior is somewhat atypical of the IE languages.

⋆ In Gothic, Sanskrit, and Ancient Greek, different types of initial clusters trigger different copying patterns.

- In all of these languages, TRVX– (i.e. obstruent-sonorant-initial) bases exhibit the C_1-copying pattern:
  - $\mathbf{T_1R_2VX}$– → $\mathbf{\underline{T_1V}\text{-}T_1R_2VX}$– (like Old Irish does for all clusters)
- However, for STVX– bases, they all have some other copying pattern:
  - Cluster-copying in Gothic (Section 3.2.2)
  - C_2-copying in Sanskrit (Section 3.2.3)
  - Non-copying in Ancient Greek (Section 3.2.4)

→ **My proposal:** These divergent copying behaviors are triggered by ***PCR**, a constraint that places restrictions on *consonant repetitions*, i.e. sequences of identical C's separated only by a vowel ($C_\alpha VC_\alpha$).

3.2.1 The repetition avoidance constraint: *PCR

- In Zukoff (2017a:Ch. 6), I develop a repetition avoidance analysis of these patterns based on the distribution and perception of acoustic/auditory cues to particular consonantal contrasts.
  → I call this approach the NO POORLY-CUED REPETITIONS constraint (*PCR).

⋆ For today's purposes, I will use a simplified version of this constraint, which militates against locally repeated consonants in *pre-obstruent position*, as defined in (28):

(28) **NO POORLY-CUED REPETITIONS (*PCR)** [$\approx *C_\alpha VC_\alpha/\_C_{[\text{-sonorant}]}$]
For each sequence of repeated identical consonants separated by a vowel ($C_\alpha VC_\alpha$), assign a violation * if that sequence immediately precedes an obstruent.

- *PCR penalizes C_1-copying to STVX– (i.e. *s*-obstruent-initial) bases, but not to TRVX– bases:

(29) **Repetitions and satisfaction/violation of *PCR**

	Base type	C_1-copying	Repetition	Context	Satisfied?
a.	**TRVX–**	[*pa-pr*]*ako*	*pap*	/ _*r* (sonorant)	✓
b.	**STVX–**	[*sa-st*]*ako*	*sas*	/ _*t* (obstruent)	✗

- Since TRVX– bases *do* show C_1-copying in all these languages, we can understand these systems as follows:

(30) **Logic of cluster-dependent copying systems**

a. They prefer to reduplicate base-initial clusters with C_1-copying (and do so for TRVX– bases).

b. This is **blocked** for STVX– bases by high-ranked *PCR, diverting derivation to another pattern.

↪ The resulting pattern is determined by the relative ranking of the other relevant constraints.

⋆ I will now demonstrate how this derives the distributions in Gothic, Sanskrit, and Ancient Greek.

3.2.2 TRVX– C_1-copying, STVX– cluster-copying: Gothic

- One way to avoid a *PCR violation is to copy the entire base-initial cluster (as in the across-the-board cluster-copying pattern in Hittite).
  - By doing this, the copy of the root-second consonant intrudes into the consonant repetition (31c).

(31) **TRVX– C_1-copying, STVX– cluster-copying**

	Base Type	Root		Reduplicated	Red. Shape	
a.	Singleton	√*mako*	→	*ma-mako*	C_1V_2	
b.	Stop-sonorant	√*prako*	→	*pa-prako*	C_1V_3	
c.	*s*-obstruent	√*stako*	→	*sta-stako*	$C_1C_2V_3$	(****sa-stako***)

∗ Note that, in (31c), both base-initial consonants have a nearby copy in the reduplicant. However, each repetition is separated by both a vowel and a consonant, which evidently is sufficient to avoid a *PCR violation.

- To generate C_1-copying in the basic case (i.e. TRVX–), we need the ranking **Anchor-L-BR, *CC ≫ Contiguity-BR** (cf. (24) above for Old Irish), demonstrated in (32):

(32) **Generating TRVX– C_1-copying**

/red, prako/	*PCR	Anchor-L-BR	*CC	Contig-BR
a. pra-prako			**!	
b. ☞ pa-prako			*	*
c. ra-prako		*!	*	

- Then, in order to motivate diversion from the C_1-copying pattern just for STVX– bases, ***PCR must dominate *CC**, as shown in (33).
  - Anchor-L-BR must also dominate *CC, so that cluster-copying (33a) is selected as the new repair, and not C_2-copying (33c).

(33) **Generating STVX– cluster-copying alongside TRVX– C_1-copying**

/red, stako/	*PCR	Anchor-L-BR	*CC	Contig-BR
a. ☞ sta-stako			**	
b. sa-stako	*!		*	*
c. ta-stako		*!	*	

→ In other words, it is generally preferable to avoid creating a consonant cluster in the reduplicant, but this is tolerated if it allows a pre-obstruent repetition to be avoided.

- Gothic illustrates this pattern perfectly (also Proto-Anatolian; Zukoff 2017a:Ch. 4, Yates & Zukoff 2018).
  → TRVX– bases show default C_1-copying pattern (34a), while STVX– bases show cluster-copying (34b).

(34) **Class VII preterites in Gothic** (forms from Lambdin 2006:115)

a. **TRVX– roots → C_1-copying preterites**

Root	Infinitive		Preterite		
‘to weep’	*gretan*	[grēt-an]	*gaigrot*	[ge-grōt]	(not **[gre-grōt])

b. **STVX– roots → cluster-copying preterites**

Root	Infinitive		Preterite		
‘to possess’	*staldan*	[stald-an]	*staistald*	[ste-stald]	(not **[se-stald])
‘to divide’	*skaidan*	[skaið-an]	*skaiskaiþ*	[ske-skaiθ]	(not **[se-skaiθ])

3.2.3 TRVX– C_1-copying, STVX– C_2-copying: Sanskrit

- Another way to avoid a *PCR violation is to copy C_2 rather than C_1, as shown in (35c).
  - This is the cluster-dependent version of the unattested across-the-board C_2-copying pattern (Section 3.1.3).

(35) **TRVX– C_1-copying, STVX– C_2-copying**

	Base Type	Root		Reduplicated	Red. Shape	
a.	Singleton	√*mako*	→	*ma-mako*	C_1V_2	
b.	Stop-sonorant	√*prako*	→	*pa-prako*	C_1V_3	
c.	*s*-obstruent	√*stako*	→	*ta-stako*	C_2V_3	(****sa*-*stako***)

- Since this pattern shows the same C_1-copying behavior for TRVX– bases as the previous case, we can begin by importing the TRVX– C_1-copying ranking from (32): Anchor-L-BR, *CC ≫ Contiguity-BR.

(36) **Generating TRVX– C_1-copying**

/red, prako/		*PCR	*CC	Anchor-L-BR	Contig-BR
a.	pra-prako		**!		
b. ☞	pa-prako		*		*
c.	ra-prako		*	*!	

- The only difference from the STVX– cluster-copying pattern that is required to generate STVX– C_2-copying is to reverse the role of Anchor-L-BR and *CC.
  - The ranking ***CC ≫ Anchor-L-BR** prefers mis-anchoring the reduplicant (37c) to copying the cluster (37a).

(37) **Generating STVX– C_2-copying alongside TRVX– C_1-copying**

/red, stako/		*PCR	*CC	Anchor-L-BR	Contig-BR
a.	sta-stako		**!		
b.	sa-stako	*!	*		*
c. ☞	ta-stako		*	*	

- The TRVX– C_1-copying with STVX– C_2-copying pattern is instantiated in Sanskrit for cluster-initial roots:

(38) **Perfects to cluster-initial roots in Sanskrit** (forms from Whitney 1885)

a. **TRVX– roots → C_1-copying perfects**

Root		Perfect Tense	
√*b*^{h}*raj-*	'shine'	*ba-b*^{h}*rāj-a*	(not ***ra-b*^{h}*rāj-a*)
√*prac*^{h}*-*	'ask'	*pa-prāc*^{h}*-a*	(not ***ra-prāc*^{h}*-a*)
√*dru-*	'run'	*du-druv-ē*	(not ***ru-druv-ē*)
√*tviṣ-*	'be stirred up'	*ti-tviṣ-ē*	(not ***vi-tviṣ-ē*)

b. **STVX– roots → C_2-copying perfects**

Root		Perfect Tense	
√*sparç-*	'touch'	*pa-spr̥ç-ē*	(not ***sa-spr̥ç-ē*)
√*st*^{h}*ā-*	'stand'	*ta-st*^{h}*ā-u*	(not ***sa-st*^{h}*ā-u*)
√*stamb*^{h}*-*	'prop'	*ta-stamb*^{h}*-a*	(not ***sa-stamb*^{h}*-a*)

3.2.4 TRVX– C_1-copying, STVX– non-copying: Ancient Greek

- The last remaining basic *PCR-avoidance strategy attested among the IE languages is to copy no consonant at all ("non-copying"), as schematized in (39c):

(39) **TRVX– C_1-copying, STVX– non-copying**

	Base Type	Root		Reduplicated	Red. Shape	
a.	Singleton	√*mako*	→	*m-e-mako*	C_1-V	
b.	Stop-sonorant	√*prako*	→	*p-e-prako*	C_1-V	
c.	*s*-obstruent	√*stako*	→	*e-stako*	Ø-V	(****s-e-stako***)

- This pattern is attested in Ancient Greek, as shown in (40):

(40) **TRVX– C_1-copying, STVX– non-copying in Ancient Greek**

a. **TRVX– roots → C_1-copying perfects**

Root		Perfect Tense		
√*kri-*	'decide'	κέκριμαι	[k-e-kri-mai]	(not **[e-kri-mai])
√*pneu-*	'breathe'	πέπνυμαι	[p-e-pnū-mai]	(not **[e-pnū-mai])
√*tla-*	'suffer, dare'	τέτληκα	[t-e-tlē̄-k-a]	(not **[e-tlē̄-k-a])

b. **STVX– roots → Non-copying perfects**

Root		Perfect Tense		
√*stel-*	'prepare'	ἔσταλκα	[e-stal-k-a]	(not **[s-e-stal-k-a])
√*strat-eu-*	'wage war'	ἐστρατευμαι	[e-strat-eu-mai]	(not **[s-e-strat-eu-mai])

⋆ This pattern is derivable with the constraints employed thus far (plus one more); but it requires a different treatment of the reduplicative vowel: as an underlying "**fixed segment**", rather than a copy.

- The patterns of reduplicant vocalism in the IE languages vacillate between two descriptive types:

(41) **Type of reduplicant vocalism**

a. *Copy vocalism*: the reduplicative vowel is always (partially) identical to the base vowel.
b. *Fixed vocalism*: the reduplicative vowel has a consistent value (doesn't co-vary with base vowel).

- Following Alderete et al. (1999), fixed vocalism (more generally, fixed segmentism) comes in two types:

(42) **Types of fixed vocalism**

a. *Phonologically fixed*: the reduplicative vowel copies (i.e. corresponds with) the base vowel but is consistently reduced to satisfy markedness constraints (McCarthy & Prince 1994, 1995).
b. *Morphologically fixed*: the reduplicative vowel is specified in the underlying representation, and thus not a "copy" at all.

⋆ The Ancient Greek-type STVX– non-copying pattern requires a **morphological** fixed segmentism analysis, because of the way that BR-correspondence works (see Zukoff 2017a:Ch. 2 for detailed argumentation).

- If the reduplicative vowel stands in correspondence with the base vowel, non-copying will violate Anchor-L-BR (43i.b), just as C_2-copying candidate violates Anchor-L-BR (43i.a). The tie is broken by Onset (44) in favor of C_2-copying.
- On the other hand, if the reduplicative vowel does *not* stand in correspondence with the base vowel, there is *no* reduplicant proper in the non-copying candidate (43ii.b), and Anchor-L-BR is vacuously satisfied. Given the ranking Anchor-L-BR ≫ Onset, we can now properly select non-copying ((43ii.b) ≻ (43ii.a)).

(43) **Anchor-L-BR violations by vocalism type**

i. *Copy vocalism* or *phonologically-fixed vocalism*

/red, stako/	Anchor-L-BR	Onset
a. 💣 ta-stako	*	
b. ☹ a-stako	*	*!

ii. *Morphologically-fixed vocalism*

/red, e, stako/	Anchor-L-BR	Onset
a. t-e-stako	*!	
b. ☞ _-e-stako		*

- The constraint which is violated in service of *PCR by non-copying is Onset:

(44) **Onset:** Assign a violation mark * for each onsetless syllable. (*Have an onset!*)

→ The ranking that generates the Ancient Greek pattern is: ***PCR, Anchor-L-BR, *CC ≫ Onset**[4]

- Onset enforces C_1-copying for TRVX– bases because non-copying confers no benefit in this case (45):

(45) **Generating TRVX– C_1-copying (with a morphologically fixed vowel)**

/red, e, prako/	*PCR	Anchor-L-BR	*CC	Onset
a. pr-e-prako			**!	
b. ☞ p-e-prako			*	
c. r-e-prako		*!	*	
d. _-e-prako			*	*!

- But, again, *PCR blocks C_1-copying for STVX– bases by *PCR (46b); since Onset is lowest ranked (and the vowel is morphologically fixed vowel), non-copying (46d) now becomes the optimal strategy:

(46) **Generating STVX– C_2-copying alongside TRVX– C_1-copying**

/red, e, stako/	*PCR	Anchor-L-BR	*CC	Onset
a. st-e-stako			**!	
b. s-e-stako	*!		*	
c. t-e-stako		*!	*	
d. ☞ _-e-stako			*	*

[4] Contiguity-BR is not relevant because the reduplicative vowel doesn't correspond with the base vowel.

4 Factorial typology

- Holding left-edge positioning constant, the factorial typology of the five constraints employed in Section 3, shown in (47), yields six possible reduplication systems (confirmed by OTSoft; Hayes, Tesar, & Zuraw 2013).

▷ Each entry in the factorial typology is notated with:
  - The behavior of TRVX– roots and STVX– roots
  - The type(s) of vocalism which is compatible with the pattern
  - A language which displays the pattern
  - One possible ranking that generates the pattern.

∗ Constraints in parentheses are ones which have no impact on the ranking because of the required vocalism.

(47) **Factorial typology of constraints in Sections 3.1–3.2**

i. *Across-the-board copying patterns*

a. **Across-the-board cluster-copying [C_1C_2V-C_1C_2VX–]**
TRVX– behavior: Cluster-copying *pra-prako*
STVX– behavior: Cluster-copying *sta-stako*
Vocalism: Copy
Language: **Hittite** STVX– example: *istu-stu-*
Ranking: Anchor-L-BR, Contig-BR, (Onset) $\gg$ *CC, *PCR

b. **Across-the-board C_1-copying [C_1V-C_1C_2VX–]**
TRVX– behavior: C_1-copying *pa-prako*
STVX– behavior: C_1-copying *sa-stako*
Vocalism: Copy or Morphologically fixed
Language: **Old Irish** STVX– example: *se-skann*
Ranking: Anchor-L-BR, Onset, *CC $\gg$ *PCR, Contig-BR

c. **Across-the-board C_2-copying [C_2V-C_1C_2VX–]**
TRVX– behavior: C_2-copying *ra-prako*
STVX– behavior: C_2-copying *ta-stako*
Vocalism: Copy
Language: **Unattested** STVX– example: (hypothetical) *ta-sta-*
Ranking: Contig-BR, *CC, (Onset) $\gg$ Anchor-L-BR, *PCR

ii. *Cluster-dependent copying patterns*

d. **TRVX– C_1-copying [T_1V-T_1R_2VX–], STVX– cluster-copying [S_1T_2V-S_1T_2VX–]**
TRVX– behavior: C_1-copying *pa-prako*
STVX– behavior: Cluster-copying *sta-stako*
Vocalism: Copy or Morphologically fixed
Language: **Gothic** STVX– example: *ste-stald*
Ranking: *PCR, Anchor-L-BR, Onset $\gg$ *CC $\gg$ Contig-BR

e. **TRVX– C_1-copying [T_1V-T_1R_2VX–], STVX– C_2-copying [T_2V-S_1T_2VX–]**
TRVX– behavior: C_1-copying *pa-prako*
STVX– behavior: C_2-copying *ta-stako*
Vocalism: Copy or Morphologically fixed
Language: **Sanskrit** STVX– example: *ta-stamb^h-*
Ranking: *PCR, Onset, *CC $\gg$ Anchor-L-BR $\gg$ Contig-BR

f. **TRVX– C_1-copying [T_1V-T_1R_2VX–], STVX– non-copying [V-S_1T_2VX–]**
TRVX– behavior: C_1-copying *p-e-prako*
STVX– behavior: Non-copying *_-e-stako*
Vocalism: Morphologically fixed
Language: **Ancient Greek** STVX– example: *e-stal-*
Ranking: *PCR, Anchor-L-BR, *CC $\gg$ Onset, (Contig-BR)

⋆ As can be seen in (47), five of the six predicted systems are indeed attested within the IE language family.[5]
  ◦ The across-the-board C_2-copying pattern (47c) is the only pattern not attested in IE (nor, to my knowledge, anywhere else).
- This is, admittedly, an argument against the current approach (see also Kim 2020).
  ◦ Nevertheless, it may be reasonable to view this is an accidental gap, given the scarcity of languages that could be expected to display the conditions necessary for such a pattern.

⋆ Determining the nature of this gap, and whether there is a fix within the given proposal, is an important question for further consideration.
  → Pending this question, the factorial typology demonstrates that the basic constraint types employed in this analysis of IE reduplication lead to a good fit with the attested patterns.

5 *PCR and infixal reduplication in IE (time permitting)

- In this section, I will present two additional IE reduplication patterns, both of which are infixal and give additional evidence for *PCR:

(48) **Infixal IE reduplication patterns**
  a. Latin perfect reduplication with *PCR-driven infixation for STVX– bases (Section 5.1)
  b. Desiderative reduplication in Classical Sanskrit, which shows infixation for vowel-initial roots, where the precise position of the reduplicant is driven by *PCR (Section 5.2)

5.1 Latin infixing perfect reduplication for STVX– bases

- Among the IE languages, Latin displays a unique reduplicative behavior for its STVX– bases in the perfect:
  → *PCR violations are avoided by *infixing* the reduplicant (cf. Fleischhacker 2005, DeLisi 2015).
  ◦ In this pattern, the reduplicant retains its target shape *CV*, but deviates from its target position at the left edge by placing the reduplicant *after* the root-initial *s*, as shown in (49).

(49) **Latin infixing perfect reduplication to STVX– bases** (forms from Weiss 2009:410)

Root		Perfect		
√*st*	'stand/stop'	*s-<u>te</u>-t-ī*	(not ***<u>se</u>-st-ī*)	[but present *<u>si</u>-st-ō*]
√*spond*	'promise'	*s-<u>po</u>-pond-ī*	(not ***<u>so</u>-spond-ī*)	
√*scid*	'cut'	*s-<u>ci</u>-cid-ī*	(not ***<u>si</u>-scid-ī*)	

- Infixation here is triggered by *PCR, because, as before, it penalizes prefixal C_1-copying (e.g. ***<u>si</u>-scid-ī*).

⋆ What is different is which constraints are lowest ranked, and thus can be violated in service of *PCR: in Latin, it is two constraints which, in effect, prefer the reduplicant to surface as a prefix:
  ◦ Align-Red-L (50a) wants the reduplicant to be as close to the left edge as possible.
  ◦ Contiguity-IO (50b) wants nothing to end up inside the root.

(50) a. **Align-Red-L:** Assign one violation mark * for each segment intervening between the left edge of the reduplicant and the left edge of the word. (*Prefix the reduplicant!*)
  b. **Contiguity-IO:** Assign one violation mark * for each pair of segments which are adjacent in the input that have non-adjacent correspondents in the output. (*Don't infix!*)

[5] Kim (2020:11–12) rightly notes that the across-the-board C_1-copying pattern (47b) could logically be grouped with the cluster-dependent copying patterns (47d–f) in that they all represent patterns with C_1-copying for TRVX– bases, and some pattern for STVX– bases. He is incorrect, however, in stating that the "typological calculation has not considered all logically possible types", because it has already been included in the factorial typology, just under a different heading.

→ If these constraints are dominated by *PCR, Anchor-L-BR, and *CC, infixation will be selected as the optimal pattern for STVX– bases, as shown in (51).
- ∗ This alignment approach correctly predicts that infixation is minimal: (51d) ≻ (51e).
- ∗ The base of reduplication must be the string to the right of the reduplicant.

(51) **Infixing reduplication in Latin STVX– bases to avoid *PCR violation**

/red, scid, ī/	*PCR	Anchor-L-BR	*CC	Contig-IO	Align-Red-L
a. si-scid-ī	*!		*		
b. ci-scid-ī		*!	*		
c. sci-scid-ī			**!		
d. ☞ s-ci-cid-ī			*	*	*
e. sc-id-id-ī			*	*	**!

- This analysis predicts that TRVX– roots should exhibit C_1-copying pattern, because infixation is triggered by *PCR-violating repetitions: hypothetical √*plen-* → *pe-plen-*, not *p-le-len-*. Unfortunately, Latin doesn't have any reduplicated forms to TRVX– roots (Cser 2009), so we can't test this prediction.[6]

5.2 Sanskrit infixing desiderative reduplication for vowel-initial bases

- In addition to the perfect reduplication pattern discussed in Section 3.2.3, Sanskrit also shows reduplication in a number of other verbal categories (consult Kulikov 2005).
- One such category is the desiderative (see Whitney 1889:372–374/§1026–1031), which is marked by:
  - Prefixal reduplication, with a fixed [+high] vowel that matches the base vowel in [±round], and
  - A suffix *-(i)ṣa-*, which attaches immediately after the root.
- For consonant-initial roots, the distribution of reduplicant shape is the same as in the perfect:
  - C_1-copying to TRVX– roots (52a)
  - C_2-copying to STVX– roots (52b)

(52) **Sanskrit desiderative reduplication to cluster-initial bases**

a. √*tvar* 'hasten' → desiderative *ti-tvar-iṣa-*, perfect *ta-tvar-*
b. √*stamb*^{h} 'prop' → desiderative *ti-stamb*^{h}*-iṣa-*, perfect *ta-stamb*^{h}*-*

- Vowel-initial roots, however, do something different. According to the Classical Sanskrit grammarians, vowel-initial roots build the desiderative with infixal reduplication (53) (forms from Whitney 1885):

(53) **Classical Sanskrit infixing desiderative reduplication to vowel-initial roots**

	Root shape	Root		Desiderative	
a.	VC	√aɟ	'drive'	a-ɟi-ɟ-iṣa-	(not **aɟ-aɟ-iṣa-)
		√īḍ	'praise'	ī-ḍi-ḍ-iṣa-	(not **īḍ-īḍ-iṣa-)
		√ēd$^{\mathrm{h}}$	'thrive'	ē-di-d$^{\mathrm{h}}$-iṣa-	(not **ēd-ēd$^{\mathrm{h}}$-iṣa-)
b.	VCT	√arc	'praise'	ar-ci-c-iṣa-	(not **a-ri-rc-iṣa-)
		√ubɟ	'force'	ub-ɟi-ɟ-iṣa-	(not **u-bi-bɟ-iṣa-)
		√aɲɟ	'anoint'	aɲ-ɟi-ɟ-iṣa-	(not **a-ɲi-ɲɟ-iṣa-)
c.	Vkṣ	√akṣ	'attain'	ā-ci-kṣ-iṣa-	(not **āk-ṣi-ṣ-iṣa-)
		√īkṣ	'see'	ī-ci-kṣ-iṣa-	(not **īk-ṣi-ṣ-iṣa-)

6 My constraints can't generate across-the-board infixation for cluster-initial roots without also predicting it for CVX– roots.

∗ Today I will not be concerned with *what triggers infixation* in the first place for vowel-initial roots (see Zukoff 2017a:Ch. 6.6.2 for the full analysis).

→ I will instead focus on the *position of the infixed reduplicant* for different post-vocalic cluster types, because this alternation is driven by *PCR.

- This is an infixal reduplication pattern, so Align-Red-L must be violated (also Contig-IO).
  - Because Align-Red-L assigns violations *gradiently* (i.e., the farther the reduplicant is from the left, the more violations it gets), we predict that the reduplicant should surface after the root-initial vowel.
- For the VCT– roots (53b), this prediction is *incorrect*:
  → The reduplicant surfaces after the second segment (54b), not after the first (54b).

⋆ **The reason:** for roots with post-vocalic CT-clusters, infixing after the V would cause a *PCR violation (54a).

(54) **Non-minimal infixation to *PCR-violating cluster (VCT– roots)**

/RED, ubɟ, -iṣa-/	*PCR	Align-Red-L
a. u-bi-bɟ-iṣa-	*!	*
b. ☞ ub-ɟi-ɟ-iṣa-		**
c. ubɟ-iṣ-iṣa-		***!

- The minimal infixation candidate (54a) contains the sequence *-bibɟ-*, where the consonant repetition (*bib*) surfaces before an obstruent (*ɟ*), and thus violates *PCR.
- On the other hand, infixing past the first consonant (54b) causes the repeated sequence (*ɟiɟ*) to end up pre-vocalic position (before the suffix vowel *i*).
- Infixing past the 2nd consonant (54c) also satisfies *PCR, but incurs an extra Align-Red-L violation.

- The most interesting thing about the Sanskrit desiderative is that we observe something different just in case the post-vocalic cluster is /kṣ/ (53c).
  → In /Vkṣ/ roots, infixation *does* land after the vowel (55a), rather than after the first consonant (55b).

⋆ **The reason:** the sequence *-cVkṣ-* doesn't violate *PCR.

(55) **Minimal infixation to Vkṣ roots**

/RED, ākṣ, -iṣa-/	*PCR	Align-Red-L
a. ☞ ā-ci-kṣ-iṣa-		*
b. āk-ṣi-ṣ-iṣa-		**!
c. ākṣ-iṣ-iṣa-		**!*

- In Sanskrit, base velar consonants, e.g. /k/, always reduplicates as a palatal, e.g. [c], due to a semi-productive palatalization process.
- This means that, in just this case, the base and reduplicant consonants don't constitute an identical repetition, and thus satisfy *PCR vacuously.[7]

→ Without the *PCR violation eliminating the minimal infixation candidate (55a), Align-Red-L can now eliminate the non-minimal infixation candidates (55b,c).

∗ This analysis predicts that VTR roots would reduplicate like Vkṣ roots, showing minimal infixation: e.g. hypothetical $\sqrt{}$*atr* → desiderative *a-ti-tr-iṣa-* (**at-ri-r-iṣa-*). Unfortunately, no such roots are attested.

[7] Even if this did still count as an "identical" consonant repetition, under the more precise version of *PCR for Sanskrit laid out in Zukoff (2017a:Ch. 6), -TVTS- sequences do not violate *PCR, because a TS sequence includes an intensity rise.

⋆ In order to reconcile the analysis of V-initial roots with C-initial roots, we need to use *#CC (56a) as the markedness constraint motivating reduplicant-cluster reduction (58) instead of the more general *CC (57).

- ○ The constraint *C#V (56b) stands in for the cue-based constraint proposed in Zukoff (2017a:Ch. 6.6.2) to motivate infixation to vowel-initial roots.

(56) a. ***#CC:** Assign one violation * for each word-initial sequence of two consonants in the output. (= *ComplexOns)
b. ***C#V:** Assign one violation * for each root-initial vowel preceded by a reduplicant consonant.

(57) **Ranking paradox with *CC**

/red, ubɟ, -iṣa-/	*C#V	*CC	Anchor-L-BR	Align-Red-L
a. ☞ ub-ɟi-ɟ-iṣa-		*		**
b. u-ɟi-bɟ-iṣa-		*	*!	*
c. ubɟ-ubɟ-iṣa-	*!	**!		
/red, stan, -iṣa-/	***C#V**	***CC**	**Anchor-L-BR**	**Align-Red-L**
a. ☹ ti-stan-iṣa-		*	*!	
b. 💣 s-ti-tan-iṣa-		*		*
c. sti-stan-iṣa-		**!		

(58) **Ranking paradox resolved with *#CC**

/red, ubɟ, -iṣa-/	*C#V	*#CC	Anchor-L-BR	Align-Red-L
a. ☞ ub-ɟi-ɟ-iṣa-				**
b. u-ɟi-bɟ-iṣa-			*!	*
c. ubɟ-ubɟ-iṣa-	*!			
/red, stan, -iṣa-/	***C#V**	***#CC**	**Anchor-L-BR**	**Align-Red-L**
a. ☞ ti-stan-iṣa-			*	
b. s-ti-tan-iṣa-		*!		*
c. sti-stan-iṣa-		*!		

5.3 Local summary

- This section showed that there are several infixal reduplication patterns attested among the IE languages, and that these patterns also respond to *PCR.
- The next step would be to try to integrate Align-Red-L into the factorial typology and see whether that continues to give a good fit to the available data.

→ One other place to look for infixed reduplicated forms is Northwest Germanic (Jasanoff 2007, Zukoff 2017a:159–161), but the data is quite messy.

6 Conclusion

- In this talk, I have shown that a relatively small number of constraints can do a good job modeling the diversity of reduplication patterns among the IE languages.
  - → Using factorial typology, I showed that the core constraints lead to only one unattested system.
- The main contribution of this work is the introduction of the constraint *PCR, which militates against certain kinds of consonant repetitions, repeated in (59):

(59) **No Poorly-Cued Repetitions (*PCR)** $[\approx *C_\alpha VC_\alpha/\_C_{[\text{-sonorant}]}]$
For each sequence of repeated identical consonants separated by a vowel ($C_\alpha VC_\alpha$), assign a violation * if that sequence immediately precedes an obstruent.

∗ Additional work is required to fully understand how *PCR is to be defined, because the formulation used here doesn't fully account for the cluster-wise distributions of *PCR effects in the IE reduplicative systems.

- → See Zukoff (2017a:Ch. 6) for further details and a more fleshed out proposal.

- Beyond the reduplication patterns predicted by the main relevant constraints, I also discussed two infixal reduplication patterns: the Latin perfect and the Sanskrit desiderative.
  - These can be analyzed by allowing violation of ALIGN-RED-L in service of *PCR and other constraints.

→ In Zukoff (2017a), I also discuss two additional reduplication(-related) patterns in the IE languages which have evolved from earlier stages with transparent *PCR effects:

- One is Ancient Greek's "Attic Reduplication" pattern, exemplified in (60): an irregular pattern of VC- copying to vowel-initial roots. (The regular pattern for vowel-initial roots is essentially non-copying.)
  - In Zukoff (2017a,b), I argue that this arose from a *PCR-like restriction on the repetition of "laryngeals". (The laryngeals are a series of weak consonants reconstructed for PIE; de Saussure 1879; see Fortson 2010:62–64.)

(60) **"Attic Reduplication" in Ancient Greek:**
Ancient Greek *ag-āger-* 'have gathered' < Pre-Greek **h₂əg-e-h₂ger-* (***h₂-e-h₂ger-*)

- The other is Sanskrit's "CēC" pattern, exemplified in (61): in certain inflected forms in the perfect (when the suffix is accented), rather than exhibiting reduplication, the stem appears to have a long vowel [ē].
  - In Zukoff (2017a:Ch. 5), I show that these allomorphs appear just in case C_1-copying reduplication would yield a *PCR violation (or a violation of other high-ranked markedness), e.g. **[sa-sp-úr].
  - It is likely that this allomorphy arose from deletion with compensatory lengthening of originally reduplicated forms (e.g. **sa-sp-úr*), subject to the same markedness triggers, including *PCR.

(61) **"CēC" perfect weak stems in Sanskrit:**
√*sap-* 'serve' → perfect plural *sēp-úr* (***sa-sp-úr*)

∗ The Germanic Class V preterites in CēC, and possibly other "long-vowel preterites" around IE (cf. Schumacher 2005, *a.o*) likely arose in the same fashion (though independently).

⋆ The next step will be to see how well *PCR can explain patterns outside of IE.
  - In Zukoff (2017a:Ch. 6.6.3), I show that Klamath (isolate, Oregon) has a pattern equivalent to Gothic, but with a version of *PCR identical to Ancient Greek, despite a much richer cluster inventory.
  - We also find cluster-dependent copying effects in Gbe (Atlantic-Congo, Benin; Capo 1989, Ameka 1991), again with a pattern that looks like Gothic, but with different cluster types.

→ This broader look at cluster-dependent copying patterns may tell us more about what's actually going on in Indo-European.

References

Alderete, John, Jill Beckman, Laura Benua, Amalia Gnanadesikan, John McCarthy & Suzanne Urbanczyk. 1999. Reduplication with Fixed Segmentism. *Linguistic Inquiry* 30(3):327–364.

Ameka, Felix Kofi. 1991. Ewe: Its grammatical constructions and illocutionary devices. PhD Dissertation, Australian National University Canberra.

Capo, Hounkpati Bamikpo Christophe. 1989. *Précis phonologique du Gbe: Une perspective comparative*. Labo Gbe.

Cser, András. 2009. The History of Perfective Reduplication and Stem-Initial Patterns in Latin. *Acta Antiqua Academiae Scientiarum Hungaricae* 49(2):107–115.

de Saussure, Ferdinand. 1879. *Mémoire sur le système primitif des voyelles dans les langues indo-européennes*. Leipzig: Teubner.

DeLisi, Jessica. 2015. Sonority Sequencing Violations and Prosodic Structure in Latin and Other Indo-European Languages. *Indo-European Linguistics* 3(1):1–23.

Dempsey, Timothy Richard. 2015. Verbal Reduplication in Anatolian. PhD Dissertation, UCLA.

Fleischhacker, Heidi Anne. 2005. Similarity in Phonology: Evidence from Reduplication and Loan Adaptation. PhD Dissertation, UCLA.

Fortson, Benjamin W. 2010. *Indo-European Language and Culture: An Introduction*. 2nd edn. Malden, MA: Wiley-Blackwell Publishing.

Hayes, Bruce, Bruce Tesar & Kie Zuraw. 2013. OTSoft 2.5, software program http://www.linguistics.ucla.edu/people/hayes/otsoft/.

Jasanoff, Jay H. 2007. From Reduplication to Ablaut: The Class VII Strong Verbs of Northwest Germanic. *Historische Sprachforschung* 120(1):241–284.

Kager, René. 1999. *Optimality Theory*. Cambridge: Cambridge University Press.

Keydana, Götz. 2006. Die indogermanische Perfektreduplikation. *Folia Linguistica Historica* 27(1–2):61–116.

Kim, Hyung-Soo. 2020. A New Look at Onset Transfer in Indo-European Reduplication: Dissimilation of Consonant Clusters. *Language Research* 56(1):1–27. doi:10.30961/lr.2020.56.1.1.

Kulikov, Leonid. 2005. Reduplication in the Vedic Verb: Indo-European Inheritance, Analogy, and Iconicity. In Bernard Hurch (ed.), *Studies on Reduplication*, 431–454. New York: Mouton de Gruyter.

Lambdin, Thomas O. 2006. *An Introduction to the Gothic Language*. Eugene, Oregon: Wipf & Stock Publishers.

McCarthy, John J. & Alan Prince. 1993. Generalized Alignment. In Geert Booij & Jaap van Marle (eds.), *Yearbook of Morphology 1993*, 79–153. Kluwer. doi:10.1007/978-94-017-3712-8_4.

———. 1994. The Emergence of the Unmarked: Optimality in Prosodic Morphology. In Mercè Gonzàlez (ed.), *NELS 24: Proceedings of the North East Linguistic Society*, 333–379. Amherst, MA: Graduate Linguistics Student Association. http://works.bepress.com/john_j_mccarthy/43.

———. 1995. Faithfulness and Reduplicative Identity. In Jill Beckman, Suzanne Urbanczyk & Laura Walsh Dickey (eds.), *Papers in Optimality Theory* (University of Massachusetts Occasional Papers in Linguistics 18), 249–384. Amherst, MA: Graduate Linguistics Student Association. http://works.bepress.com/john_j_mccarthy/44.

———. 1999. Faithfulness and Identity in Prosodic Morphology. In René Kager, Harry van der Hulst & Wim Zonneveld (eds.), *The Prosody-Morphology Interface*, 218–309. Cambridge: Cambridge University Press. http://works.bepress.com/john_j_mccarthy/77.

Parker, Steve. 2002. Quantifying the Sonority Hierarchy. PhD Dissertation, University of Massachusetts, Amherst.

———. 2008. Sound Level Protrusions as Physical Correlates of Sonority. *Journal of Phonetics* 36(1):55–90.

Prince, Alan & Paul Smolensky. [1993] 2004. *Optimality Theory: Constraint Interaction in Generative Grammar*. Malden, MA: Blackwell Publishing.

Schumacher, Stefan. 2005. 'Langvokalische Perfekta' in indogermanischen Einzelsprachen und ihr grundsprachlicher Hintergrund. In Gerhard Meiser & Olav Hackstein (eds.), *Sprachkontakt und Sprachwandel. Akten der XI. Fachtagung der Indogermanischen Gesellschaft, 17–23. September 2000, Halle an der Salle*, 591–626. Wiesbaden: Reichert.

Thurneysen, Rudolf. [1946] 1980. *A Grammar of Old Irish*. Dublin: Dublin Institue for Advanced Studies.

Weiss, Michael. 2009. *Outline of the Historical and Comparative Grammar of Latin*. Ann Arbor: Beech Stave Press.

Whitney, William Dwight. 1885. *The Roots, Verb-Forms, and Primary Derivatives of the Sanskrit Language: A Supplement to his Sanskrit Grammar*. New Haven: American Oriental Society.

———. 1889. *Sanskrit Grammar*. 2nd edn. Cambridge, MA: Harvard University Press.

Wright, Richard. 2004. A Review of Perceptual Cues and Cue Robustness. In Bruce Hayes, Robert Kirchner & Donca Steriade (eds.), *Phonetically Based Phonology*, 34–57. Cambridge: Cambridge University Press.

Yates, Anthony D. & Sam Zukoff. 2018. The Phonology of Anatolian Reduplication: Synchrony and Diachrony. *Indo-European Linguistics* 6(1):201–270. doi:10.1163/22125892-00601001.

Zukoff, Sam. 2017a. Indo-European Reduplication: Synchrony, Diachrony, and Theory. PhD Dissertation, MIT. https://www.samzukoff.com/zukoffdiss.

———. 2017b. The Reduplicative System of Ancient Greek and a New Analysis of Attic Reduplication. *Linguistic Inquiry* 48(3):459–497. doi:10.1162/ling_a_00250.