Class 10
Reduplication in Harmonic Serialism:
Serial Template Satisfaction (McCarthy, Kimper, & Mullin 2012)

11/30/17

1 Introduction

- McCarthy, Kimper, & Mullin (2012) [MKM] propose "Serial Template Satisfaction" (STS) as the framework for analyzing reduplication in Harmonic Serialism (HS; see McCarthy 2000, 2010, McCarthy & Pater 2016).
- Their main goal is just to show that you can do reduplication in HS (which you might think wouldn't be possible, given the principles of the theory).
  - ○ The basic mechanics work well enough (though it's seriously complicated), if you buy their claims about the typological data on reduplication-phonology interactions.
- But because it's a paper, they need to have some actual results. So they try to manufacture some, but they don't really hold up (see Zukoff 2017a; cf. Somerday 2015).

2 Basics of Harmonic Serialism

- Harmonic Serialism = "Serial OT", in contrast to classical OT = "Parallel OT"

- In OT, GEN provides all conceivable candidates to be evaluated and selected by EVAL.
  - ○ There is a single evaluation, and thus a parallel mapping between input and output.

- In HS, GEN provides only candidates that differ from the input by one change/operation.
  - ⇒ It is thus part of the research program of HS to determine what counts as an operation.
    - ▪ This is also a shortcoming of the literature on the whole, because different papers frequently make different, and perhaps crucially incompatible, assumptions/claims about what is or is not a single operation.
  - ○ Since we obviously observe outputs that differ from the input in more than one way, this process is iterated ("serial").
  - → The candidate selected by EVAL serves as the input to another evaluation, where GEN furnishes all candidates that differ by one operation from the new input.
  - ○ This is repeated until EVAL selects the candidate which matches the input from that step ("convergence").

- Like Parallel OT (unlike Stratal OT), there is a single constraint ranking that is held constant across each evaluation.
  - ⋆ People have proposed "Stratal Harmonic Serialism", because sometimes regular serialism isn't even enough.

- ○ Thus, HS is essentially a way of doing process ordering through constraint ranking rather than rule ordering.
- ○ Changes to the input are only motivated when some $\mathbb{M} \gg \mathbb{F}$.
- ○ The highest ranked $\mathbb{M}$ gets satisfied first, then the second highest ranked $\mathbb{M}$, and so on, until no more changes can be made that don't violate higher ranked $\mathbb{F}$'s.

2.1 Example 1: Epenthesis in Classical Arabic

- Classical Arabic (for the most part) doesn't allow consonant clusters word-initially.
  - ○ It fixes #CC by epenthesizing [ʔi] before the cluster.
- Classical Arabic also doesn't allow onsetless syllables.
  - ○ # repaired by epenthesizing [ʔ] before the V.

- [ʔi] epenthesis can be modeled serially as [i] epenthesis followed by [ʔ] epenthesis (McCarthy 2010:3–4).

- Take an input with an initial cluster: /fʕal/
- In HS, GEN provides candidates which make "one change" (and no change — i.e. the faithful candidate) relative to this input:
  - ○ Deletion of one segment: [ʕal], [fal], etc.
  - ○ Insertion of one segment: [ifʕal], [fiʕal], [ʔfʕal], etc.
  - ○ Change of one(ish) feature: [*u*ʕal], [*fa*al], etc.
  - ○ Metathesis(?): [faʕl], [falʕ](?), etc.
- These are the candidates entered into the initial evaluation.
  - ⋆ The candidate chosen in the POT evaluation (and ultimately in the HS derivation) — ʔifʕal (1✗) — is not available in the initial evaluation, because it makes *two changes* relative to the input.
- Given the ranking in (1), we select the candidate that repairs the initial cluster (satisfying highest ranked *COMPLEXONS) through vowel epenthesis.
  - ○ This introduces a new markedness violation (ONSET), but this is tolerated due to ranking.

(1) Step 1 (vowel prothesis)

/fʕal/	*COMPLEXONS	MAX	CONTIGUITY	ONSET	DEP
a. fʕal	*!				
b. ☞ ifʕal				*	*
c. fiʕal			*!		*
d. ʕal		*!			
✗ ʔifʕal					**

- ⋆ NB: We would seem to need CONTIGUITY to rule out (1c).
  - ○ In at least some work on HS, there is a desire to link faithfulness constraints directly to operations (e.g. MAX = don't perform a deletion operation, DEP = don't perform an insertion operation, etc.).
  - ○ Constraints like CONTIGUITY or ANCHOR don't fit nicely into this logic. They often sweep this under the rug.

- Once Step 1 selects [ifʕal] as the most *most harmonic available candidate*, that candidate becomes the input to a new round of evaluation.
  - The candidate set is different, because it is now derived relative to the new input //ifʕal//, not the original input /fʕal/.
    - Furthermore, faithful violations are assessed differently for equivalent candidates, since they are being reckoned relative to a different input.
  - → Now the ʔ epenthesis candidate is available, because it is only one change away from the new input.
- Since ONSET ≫ DEP, the ONSET violation introduced on the last step can be repaired.

(2) Step 2 (ʔ prothesis)

//ifʕal//	*COMPLEXONS	MAX	CONTIGUITY	ONSET	DEP
a. ifʕal				*	
b. fʕal	*!	*!			
c. ☞ ʔifʕal					*

- This candidate is now the input to another round of evaluation.
  - Given the ranking, there are no more problems that can be fixed.
  - Any additional markedness violations (e.g. *CC) are too low ranked (i.e. below faithfulness) to be able to induce more changes.
  - The faithful candidate is selected, so the derivation converges and ends, and this candidate ends up as the output form.

(3) Step 3 (convergence)

//ʔifʕal//	*COMPLEXONS	MAX	CONTIGUITY	ONSET	DEP	*CC
a. ☞ ʔifʕal						*
b. ifʕal		*!		*		*
c. ʔifiʕal			*!		*	
d. ʔiʕal		*!	*!			

2.2 Example 2: Coda assimilation/deletion

- One result of HS that I kind of like is how it captures the typology of coda assimilation/deletion effects (McCarthy 2008, 2011).

- McCarthy claims that:
  (i) Place assimilation is (virtually) always regressive (i.e. targets codas)
  (ii) Consonant deletion always targets codas over onsets
  (iii) These conditions can be (must be?) reversed when the onset is a laryngeal (which lacks place)

- According to McCarthy, this can be explained more easily (or maybe exclusively) through serial derivation than parallel derivation.

- There is an inherent asymmetry between the markedness of place (or place contrasts) between onset position (i.e. prevocalic) and coda position (i.e. non-pre-vocalic).

(4) CODACONDITION:
Assign one violation mark for every token of Place that is not associated with a segment in the syllable onset. (McCarthy 2008:279)

- McCarthy's claim: you can't delete a segment with a place specification
  - It is one step to delete the place node
  - It is another step to delete a segment without a place node
  - ⋆ This is going to pose a problem for STS for cluster reduction in reduplication (and perhaps onset cluster deletion generally)

- It's marked for segments to not have place:

(5) HAVEPLACE:
Assign one violation mark for every segment that has no Place specification. (McCarthy 2008:279)

- If HAVEPLACE ≫ CODACONDITION, coda consonants will have to be retained faithfully
  - (If CODACONDITION ≫ DEP, you could still get epenthesis under this ranking)
- But if CODACONDITION ≫ HAVEPLACE & MAX[Place], you will get deletion of the place node.
  - The segment deletion candidate is not available yet.

(6) Step 1 (Place deletion)

/patka/			CODACONDITION	DEP	HAVEPLACE	MAX[Place]	MAX
a.		patka	*!				
b.	☞	paHka			*	*	
c.		patHa	*!		*	*	
d.		patika		*!			
✗		paka				*	*

- Once you have a placeless segment in coda position, you can delete that on the next step.

(7) Step 2 (H deletion) [followed by convergence]

//paHka//			CODACONDITION	DEP	HAVEPLACE	MAX[Place]	MAX
a.		paHka			*!		
b.	☞	paka					*
c.		paHika		*!	*		

  - Reversal of HAVEPLACE and MAX yields languages where codas reduce to placeless segments (placeless nasals, and glottal stops/fricatives).

- C+glottal clusters can resolve through deletion of the glottal because glottals lack a place node.
  - Deletion of the glottal lets the coda resyllabify as an onset.
  - If this is about syllable structure and not pre-vocalic licensing, then this requires that syllabification is not an independent operation (i.e. it comes for free when you perform other operations).
  - ⋆ This is the kind of assumption that different HS papers differ on in potentially untenable ways.

(8) /h/-deletion from postconsonantal onsets in Tonkawa (McCarthy 2008:284)

underlying	surface	gloss
nes-**h**e-tsane-oʔs	nesetsnoʔs	‘I cause him to lie down’
nes-**h**a-na-kapa-	nesankapa-	‘to cause to be stuck’

- Place assimilation follows the same first step, but fixes HAVEPLACE through place linkage to the following onset at Step 2.

(9) Step 1 (Place deletion)

/pamta/	CODACOND	DEP	HAVEPLACE	MAX[Place]	MAX	DEP[Link]
a. pam.ta	*!					
b. ☞ paN.ta			*	*		
c. pam.Ha	*!		*	*		
d. pamita		*!				
e. pam͡n.ta	*!					*
✗ pan.ta				*		*

- ○ We might want to worry about [pam.p͡ta], where the onset consonant is doubly articulated to save the coda place. This could probably be ruled out by constraints on complex segments, but might be worrisome for the typology.

(10) Step 2 (Place linking) [followed by convergence]

//paN.ta/	CODACOND	DEP	HAVEPLACE	MAX[Place]	MAX	DEP[Link]
a. paN.ta			*!			
b. ☞ pan.ta						*
c. pata					*!	
d. paNita		*!	*			

3 The mechanics of STS

- STS derives surface patterns of reduplicant shape through a schema consisting of three elements.
  1. Reduplicative morphemes are lexical entries that consist of empty prosodic structure (e.g. foot, syllable, or mora, per the Prosodic Morphology Hypothesis; McCarthy & Prince 1986).
  2. The family of constraints HEADEDNESS(X): these require prosodic constituents (of type X) to have heads (of type X–1).
  3. The operation **Copy(X)** (penalized by the constraint *COPY(X)) that copies a *contiguous string* of constituents of type X from the base, for the purposes of satisfying HEADEDNESS (for the relevant prosodic category).

- Copy(X) competes with an alternative operation which can satisfy HEADEDNESS: **Insert(X)**.
  - ○ Insert(X) inserts an empty prosodic category of the type that can serve as a head for the empty template; for example, an empty syllable can be inserted to provide a head to an empty foot.

  - ○ Based on the operation-to-constraint logic of HS, this operation should be accompanied by a (freely-rankable) constraint *INSERT(X). MKM do not employ such a constraint (nor explore its potential consequences), though they allow that it might exist.
    - ▪ Presumably this is just DEP defined over different structure.

- The surface shape of the reduplicant thus depends on
  (i) The type of underlying prosodic template the morpheme has, and
  (ii) The ranking of the various constraints of the HEADEDNESS(X) and *COPY(X) constraint families.
- As in HS generally, constraint ranking determines the order in which operations apply to the input. That ordered derivation leads to distinctly different shape characteristics of the surface reduplicant.
  - ○ Everything is contingent on prosodic structure and parsing, so I'll be assuming feet throughout.
- (11) and (12) demonstrate the two primary ways of filling a foot-sized reduplicative template.

3.1 Example 1: Manam (syllable copying)

- Manam has a right-edge bimoraic foot template.
- The ranking FOOT-BINARITY ≫ *COPY(σ) forces syllable-copying at Step 1, since it is impossible to satisfy FOOT-BINARITY through a single application of the **Insert** operation.

(11) Syllable copying in Manam: *salaga* → *salaga-<u>laga</u>* (adapted from MKM:182–184)

	FOOT-BIN	HEADED(foot)	HEADED(σ)	*COPY(σ)
ft + *ft* △ σ σ σ △ △ △ sa la ga				
a. ☞ *ft* + *ft* △ △ σ σ σ $\boldsymbol{\sigma}$ $\boldsymbol{\sigma}$ △ △ △ △ △ sa la ga **la ga**				1
b. *ft* + *ft* △ σ σ σ △ △ △ sa la ga	1W	1W		0L
c. *ft* + *ft* △ | σ σ σ σ △ △ △ sa la ga	1W		1W	0L
d. *ft* + *ft* △ | σ σ σ $\boldsymbol{\sigma}$ △ △ △ △ sa la ga **ga**	1W			1

- It is a tacit assumption of MKM that, if the string *laga* were copied as a string of segments rather than a string of syllables, they could not automatically be parsed into syllables that are themselves parsed into the foot in the same derivational step (cf. (12.i.d)).
- That is to say, HEADEDNESS(foot) can never be satisfied by applying the **Copy(seg)** operation; more generally, HEADEDNESS(X) can never be satisfied by applying the **Copy(X–2)** operation.

3.2 Example 2: Balangao (segment copying)

- Balangao has a left-edge disyllabic foot template.
- The ranking *COPY(σ) $\gg$ HEADEDNESS(foot) $\gg$ FOOT-BINARITY favors two-step syllable-insertion to satisfy FOOT-BINARITY.
- The ranking FOOT-BINARITY $\gg$ HEADEDNESS(σ) $\gg$ *COPY(seg) then generates segment-copying to fill the newly created empty syllables.

(12) Segment copying in Balangao: *taynan* → *tayna-taynan* (adapted from MKM:184–186)

i. Step 1: Syllable-insertion

	*COPY(σ)	HD(foot)	FOOT-BIN	HD(σ)	*COPY(seg)
ft + *ft* △ σ σ △ △ tay nan					
a. ☞ *ft* + *ft* │ △ σ σ σ △ △ tay nan			1	1	
b. *ft* + *ft* △ σ σ △ △ tay nan		1W	1	0L	
c. *ft* + *ft* △ △ $\boldsymbol{\sigma}$ $\boldsymbol{\sigma}$ σ σ △ △ △ △ **tay nan** tay nan	1W		0L	0L	
d. *ft* + *ft* △ σ σ △ △ **tay na(n)** tay nan		1W	1		1W

- It is crucial(?) (at least for the typology?) that a candidate that copies segments (d) cannot project new syllables that can be parsed into the foot template, because that would be the optimal candidate right off the bat.
  - I'm not sure if this is actually important, but I don't think they talk about this.

ii. Step 2: Syllable-insertion (again)

ft + *ft* │ △ σ σ σ △ △ tay nan	*COPY(σ)	HD(foot)	FOOT-BIN	HD(σ)	*COPY(seg)
a. ☞ *ft* + *ft* △ △ σ σ σ σ △ △ tay nan				2	
b. *ft* + *ft* │ △ σ σ σ △ △ tay nan			1W	1L	
c. *ft* + *ft* │ △ σ σ σ △ △ △ **ta** tay nan			1W	1L	1W
d. *ft* + *ft* △ △ σ **σ** σ σ △ △ △ **tay** tay nan	1W			1L	

iii. Step 3: Segment-copying

ft + *ft* △ △ σ σ σ σ △ △ tay nan	*COPY(σ)	HD(foot)	FOOT-BIN	HD(σ)	*COPY(seg)
a. ☞ *ft* + *ft* △ △ σ σ σ σ △ △ △ △ **tay na** tay nan					1
b. *ft* + *ft* △ △ σ σ σ σ △ △ tay nan				2W	0L
c. *ft* + *ft* △ △ σ σ σ σ △ △ △ **ta** tay nan				1W	1

- In a language like Balangao which obtains its reduplicated segments via segment copying (after inserting empty syllable nodes into a foot template), the copy operation has some amount of freedom in the string of segments that it copies.
  - While it must be a contiguous string, that string can terminate at any point.
  - That is to say, when filling the empty syllables from (12.ii) through segment copying, there are a number of possible candidates, including those shown in (13):

(13)
  a. **ta.i-tay.nan*
  b. **ta.yn̩-tay.nan*
  c. *tay.na-tay.nan*
  d. **tay.nan-tay.nan*

- **Copy(seg)** is blind to the number of segments copied (MKM:180n.).
  - So these candidates are all equivalent w.r.t. the constraints included in the tableaux.
- There are no built in pressures for smaller reduplicants (given the same prosodic structure) in STS (contrary to a-templatic approaches), so considerations of overall size will not be relevant in selecting between these candidates.
- The choice between the competing segment-copying candidates will be made by phonotactics:
  - **ta.i-tay.nan* violates ONSET/NOHIATUS
  - **ta.yn̩-tay.nan* violates constraints against consonantal nuclei (e.g. HNUC; Prince & Smolensky 1993)
  - (most relevantly) **tay.nan-tay.nan* violates NOCODA twice, as opposed to just once in the desired output *tay.na-tay.nan*
- Since the winner does violate NOCODA, Balangao must have the ranking ONSET, HNUC $\gg$ NOCODA.
  - This seems consistent with the general phonology of the language; codas are generally allowed.

- The activity of NOCODA might lead us to expect coda-less **ta.na-tay.nan*.
  - This should be preferred to the actual winner, which violates NOCODA once.
- For STS, this candidate is unavailable (at this step in the derivation; though, crucially, it *is* available at later stages of the derivation...) due to the nature of the **Copy** operation, as it would have required copying a discontiguous string of segments.

4 Their big result (...is wrong)

4.1 The CVCV pattern

- Based on a survey of disyllabic partial reduplication in Australian and Austronesian languages, MKM find that there are *no* languages that have a pattern equivalent to this unavailable candidate, **ta.na-tay.nan*.
  - They refer to this pattern as the "CVCV" pattern, as it derives a CVCV string even when additional contiguous segmental material that could expand the string is present in the base.
- This pattern, illustrated in (14a), can be contrasted with the "σCV" pattern (14b), attested by Balangao and many others, and the "σCVC" pattern (14c), which MKM claim to also be unattested (and also underivable in STS).

(14) Predicted reduplicant shape patterns (hypothetical)

Root	a. CVCV (unattested)	b. σCV (attested)	c. σCVC (unattested)
mele	mele-mele	mele-mele	mele-mele
kalan	kala-kalan	kala-kalan	kalan-kalan
paltiru	pati-paltiru	palti-paltiru	paltir-paltiru
mikartu	mika-mikartu	mika-mikartu	mikar-mikartu
nampalu	napa-nampalu	nampa-nampalu	nampal-nampalu

- The CVCV pattern is easily derivable in BRCT, due to its fully parallel global evaluation mechanics:
  - The CVCV candidate is available (since all candidates are available), and, given the ranking NoCoda ≫ Contiguity-BR, the CVCV candidate is harmonically superior to corresponding candidates with one or two codas.

(15) CVCV pattern in POT

/RED-paltiru/		Max-IO	NoCoda	Contiguity-BR
a.	palti-paltiru		**!	
b.	paltir-paltiru		**!*	
c. ☞/💣	pati-paltiru		*	*
d.	pati-patiru	*!		

- Given the unavailability of this candidate in STS (at the point of segment copying), MKM claim that STS is incapable of deriving the CVCV pattern.
- Given that the CVCV pattern is apparently unattested (but note that their survey was based on only two language families, Australian and Austronesian), this constitutes a point in favor of STS relative to BRCT, and BRCT overgenerates.
- This claim rests on the assumption that there is no other means in the STS framework of deriving such a pattern.

  ⇒ **This assumption is demonstrably false.**

4.2 Onset skipping in STS predicts coda skipping in STS

- While reduplicant-medial coda skipping may be unattested, there is at least one other type of reduplicant-internal skipping pattern whose existence is not disputed.
- In a number of Indo-European languages (Steriade 1988, Zukoff 2017b, *a.o.*), as well as in Klamath (Barker 1964), and perhaps others, the second member of a base-initial consonant cluster fails to be copied into the reduplicant (see also Fleischhacker 2005).
- MKM illustrate this sort of pattern with data from Sanskrit, as shown in (16):

(16) Skipping in Sanskrit reduplication (MKM:217)

Root			Reduplicated	
√druv	'run'	→	du-druv-	(*dru-druv-)
√jña	'know'	→	ja-jña-	(*jña-jña-)
√psaː	'devour'	→	pa-psaː-	(*psa-psaː-)
√smi	'smile'	→	si-ṣmi-	(*smi-ṣmi-)

- In this pattern, the reduplicant corresponds to a **discontiguous** string of segments in the base.
  - Since copying in STS is limited to contiguous strings, this pattern cannot be generated by a single derivational step.

- The reduplicant cannot be built up through multiple steps of partial copying.
  - Copying just the initial consonant, as in a candidate /σ-druv-/ → *d-druv-*, would fail to satisfy HEADEDNESS(σ), and simultaneously introduce a phonotactically illegal initial geminate.
  - Copying just the vowel, /σ-druv-/ → *u-druv-*, would satisfy HEADEDNESS(σ) (though violating the constraint on locality of copying, which MKM name COPY-LOCALLY; MKM:181), and thus not motivate additional copying at the subsequent step.
  - Copying could in theory be triggered by other markedness constraints, e.g. ONSET, but the fact that onsetless initial syllables are allowed in Sanskrit rules out this possibility.

- Instead of multiple partial copying, this pattern can only be derived through full copying plus subsequent deletion.
- This requires a straightforward implementation of *the emergence of the unmarked*:
  - Complex onsets are permitted in roots: MAXROOT ≫ *COMPLEX.
  - But they must be simplified (through deletion) in affixes: *COMPLEX ≫ MAX(AFFIX).

(17) $/\sigma\text{-druv-}/ \xrightarrow[\text{HD}(\sigma) \gg \text{*COMPLEX}]{\textit{segment copying}} \textit{dru-\{druv-\}}_{\text{ROOT}} \xrightarrow[\text{MAXROOT} \gg \text{*COMPLEX} \gg \text{MAX}]{\textit{non-root complex margin reduction}} [\text{du-\{druv-\}}_{\text{ROOT}}]$

- The introduction of TETU mechanics into the STS system allows for the possibility of reductions of numerous other sorts in post-copying derivational steps.
  - It is a trivial extension to see how this can derive the previously impossible medial coda skipping (CVCV) pattern.
  - All that is required is to replace *COMPLEX with NOCODA:

(18) $/\{\sigma\sigma\}_{ft}\text{-paltiru}/ \xrightarrow[\text{HD}(\sigma) \gg \text{NOCODA}]{\textit{segment copying}} \textit{palti-\{paltiru\}}_{\text{RT}} \xrightarrow[\text{MAXRT} \gg \text{NOCODA} \gg \text{MAX}]{\textit{non-root \textbf{coda} reduction}} \text{*[pati-\{paltiru\}}_{\text{RT}}]$

- Prior to their discussion of Sanskrit-type onset skipping effects, MKM admit that reduplicant-specific constraints could induce skipping effects of various sorts.
  - For example, a NOCODA constraint specifically indexed to the reduplicant could generate the medial coda skipping pattern, but the existence of such constraints was generally controversial (MKM:192).
- They also show that apparent cases of the skipping could arise authentically in their system, but only if they were the result of a phonological process which was generally applicable in the language.
  - They claim that Seediq represents such an example (MKM:189, 220–222).

- However, the TETU scenario in (18) makes use of neither reduplicant-specific constraints nor general phonological processes to derive medial coda skipping.

⇒ Therefore, it must be said that STS actually *does* freely predict medial coda skipping in reduplication.
  - ○ In order to maintain the claim that STS does not predict such a pattern, the TETU mechanics would have to be abandoned.
  - ○ This would leave the onset skipping pattern, and indeed the numerous other well-established cases of the emergence of the unmarked in reduplication, without an explanation in STS, leading to a very serious under-generation problem.

- So, STS and BRCT are the same in this domain.
  - ○ If the typological generalization is right, then they are both guilty of overgeneration.
  - ○ If the typological generalization is wrong, then they can both accurately generate it.

- ...or are they? Consider the onset skipping pattern again, in light of what McCarthy (2008) says about deletion.
  - ○ It seems pretty clear that the problem with the intermediate output *dru-{druv-}*$_{\text{ROOT}}$ is that it has acquired an extra complex onset (or, in contrast licensing terms, a new consonant that is not pre-vocalic).
    - ▪ This is consistent across cluster types.
  - ○ If deleting a place-full segment constitutes two separate operations, it should be impossible to go directly from *dru-druv-* to *du-druv-*
    - ▪ You should have to get there via place deletion, i.e. *dHu-druv-* or *dNu-druv-* (*vel sim.*).
  - ○ But this doesn't solve the problem, it just creates more problems, so you shouldn't do it in the first place.

⇒ So, HS apparently has to give up its solution to coda deletion/assimilation or its solution to onset skipping in reduplication.

- When you look closely at the HS literature, this situation is super common.
  - ○ They claim a result in one domain,
  - ○ but this irreparably conflicts with a claimed result in another domain (whether they realize it or not),
  - ○ yet they continue to use all of the results as evidence in favor of HS as a theory.

- Also, this was completely the wrong analysis of the onset skipping in the first place (Zukoff 2017b).

References

Barker, M. A. R. 1964. *Klamath Grammar*. Berkeley: University of California Press.

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

McCarthy, John J. 2000. Harmonic Serialism and Parallelism. In Masako Hirotani, Andries Coetzee, Nancy Hall & Ji-yung Kim (eds.), *NELS 30: Proceedings of the North East Linguistic Society*, 501–524. Amherst, MA: Graduate Linguistics Student Association. https://works.bepress.com/john_j_mccarthy/79/.

———. 2008. The Gradual Path to Cluster Simplification. *Phonology* 25(2):271–319.

———. 2010. An Introduction to Harmonic Serialism. *Language and Linguistics Compass* 4(10):1001–1018. http://works.bepress.com/john_j_mccarthy/103.

———. 2011. Perceptually Grounded Faithfulness in Harmonic Serialism. *Linguistic Inquiry* 42(1):171–183.

McCarthy, John J., Wendell Kimper & Kevin Mullin. 2012. Reduplication in Harmonic Serialism. *Morphology* 22(2):173–232.

McCarthy, John J. & Joe Pater (eds.). 2016. *Harmonic Grammar and Harmonic Serialism*. UK: Equinox Publishing.

McCarthy, John J. & Alan Prince. 1986. Prosodic Morphology. *Linguistics Department Faculty Publication Series* 13 (1996 version). http://scholarworks.umass.edu/linguist_faculty_pubs/13.

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

Somerday, Megan. 2015. (Some) Partial Reduplication is Full Reduplication. In Thuy Bui & Deniz Özyıldız (eds.), *NELS 45: Proceedings of the Forty-Fifth Annual Meeting of the North East Linguistic Society*, vol. 2, 79–92. Amherst, MA: Graduate Linguistics Student Association.

Steriade, Donca. 1988. Reduplication and Syllable Transfer in Sanskrit and Elsewhere. *Phonology* 5(1):73–155.

Zukoff, Sam. 2017a. Actually, Serial Template Satisfaction Does Predict Medial Coda Skipping in Reduplication. In Karen Jesney, Charlie O'Hara, Caitlin Smith & Rachel Walker (eds.), *Supplemental Proceedings of the 2016 Annual Meetings on Phonology*, Washington, DC: Linguistic Society of America. http://journals.linguisticsociety.org/proceedings/index.php/amphonology/article/view/3983.

———. 2017b. Indo-European Reduplication: Synchrony, Diachrony, and Theory. MIT, PhD Dissertation. http://web.mit.edu/szukoff/www/pdfs/Zukoff2017Dissertation.pdf.