Reduplication in Tawala isn't Base-Dependent

Companion Handout

Sam Zukoff, USC

AMP 2021 · Toronto, ON · Oct 1–3, 2021

1 Data and Analysis Overview

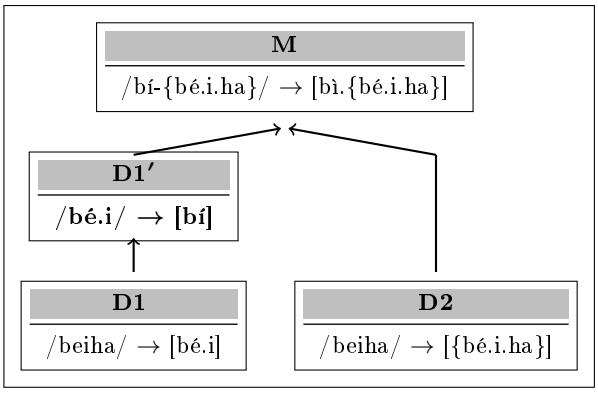
- Tawala has four distinct, predictable reduplicant shapes:
  - $C_1V_1.V_2$-initial bases copy $\underline{C_1V_2}$- (**Type A**)
  - CVCV-initial bases copy $\underline{\text{CVCV}}$- (**Type B**)
  - VC-initial bases copy $\underline{\text{VC}}$- (**Type C**)
  - Bases that begin in a repeated CV sequence locally double the first root vowel (**Type D**)

	Base	Reduplicated
Type A:	*bé.i.ha*	→ *bì.bé.i.ha*
Type B:	*hu.né.ya*	→ *hù.ne.hu.né.ya*
Type C:	*a.tú.na*	→ *à.ta.tú.na*
Type D:	*gu.gú.ya*	→ *gù.u.gú.ya*

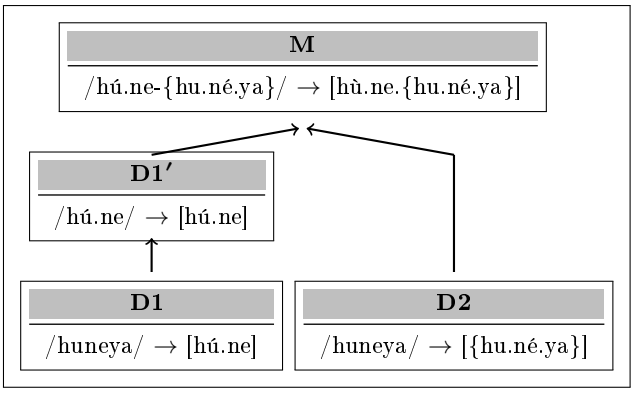
- The MDT derivations are given below.

- **Main components of analysis:**
  1. Truncation in D1
  2. Prosodic Root ({...}) parsing in D2
  3. Hiatus resolution in M, sometimes blocked by FAITH
  4. V_1-deletion under hiatus in D1′

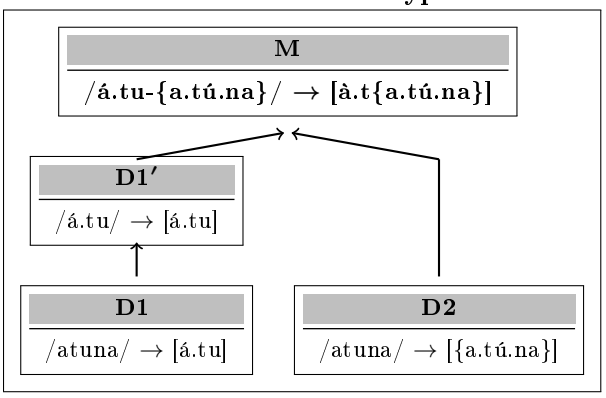
MDT derivation of Type A



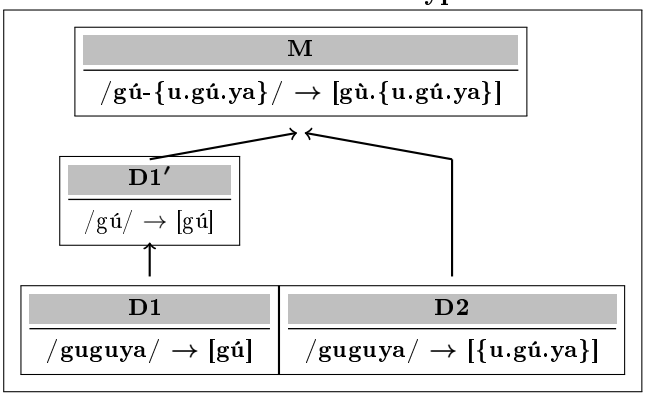
MDT derivation of Type B



MDT derivation of Type C



MDT derivation of Type D



2 Truncation in D1

- First step in MDT analysis (IZ:95, HHK): D1 preferentially truncates input down to two syllables (1A,B,C).
  - This is effectuated by the ranking in (2) (constraints defined in (3)), and demonstrated for Type B in (4).

  → It is the combined effect of StressL and NonFinality that requires at least two syllables in the truncatum. (This also correctly places stress on the first syllable.)

(1) **Output of D1**

	Input	[D1]
A	*beiha*	→ *bé.i*
B	*huneya*	→ *hú.ne*
C	*atuna*	→ *á.tu*
D	*guguya*	→ *gú*

(2) **Ranking for 2σ truncation**
StressL, NonFin ≫ *Struc) ≫ Max

(3)
a. **StressL**: Assign one violation if the leftmost syllable is unstressed. (*[#$\breve{\sigma}$])
b. **NonFinality**: Assign one violation if the rightmost syllable is stressed. (*[$\acute{\sigma}$#])
c. ***Struc**: Assign one violation for each segment in the output.
d. **Max-IO**: Assign one violation for each input segment without an output correspondent

(4) **D1 derivation of Type B (same for Types A & C)**

/huneya/	StressL	NonFin	*Struc	Max-IO
a. hú.ne.ya [100]			6!	
b. ☞ hú.ne [10]			4	2
c. hú [1]		*!	2	4
d. hu [0]	*!		2	4

- However, Type D, the crucial case in HHK's argument, is truncated to a single syllable.
  - This is effectuated by ranking *Repeat(initial) (Zukoff 2021; cf. HK) (5) and StressL above NonFin.
  - This ranking is shown in (6) and demonstrated for Type D in (7).

(5) ***Repeat(initial)**: Assign one violation if the first two syllables are identical. (*[#$\sigma_\alpha\sigma_\alpha$])

(6) **Ranking for 1σ truncation in Type D**: StressL, *Repeat(init) ≫ NonFin

(7) **D1 derivation of Type D**

/gu_1gu_2ya/	StressL	*Repeat(init)	NonFin
a. $gú_1.gu_2$.ya [100]		*!	
b. $gú_1.gu_2$ [10]		*!	
c. ☞ $gú_1$ [1]			*
d. gu_1 [0]	*!		

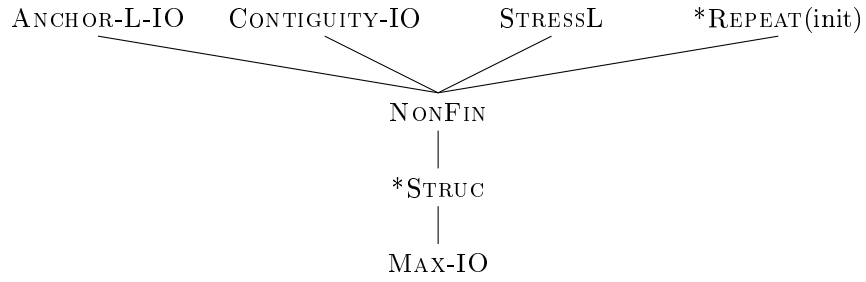
- High-ranked Anchor-L (8a) and Contiguity (8b) ensure that other 2σ alternatives are not viable (9).

(8)
a. **Anchor-L-IO**: Assign one violation if the leftmost segment in the input does not correspond to the leftmost segment in the output.
b. **Contiguity-IO**: Assign one violation for each pair of adjacent segments in the output which were not adjacent in the input.

(9) **D1 derivation of Type D**

/gu$_1$gu$_2$ya/		Anchor-L	Contiguity	NonFin
a. ☞ gú$_1$	[1]			*
b. ú$_1$.gu$_2$	[10]	*!		
c. gú$_1$.ga	[10]		*!	
d. gú$_1$.ya	[10]		*!	
e. gú$_2$.ya	[10]	*!		

(10) **Ranking summary for D1**



3 Type D C_1-deletion in D2

- D2 parses all its segment into an output Prosodic Root (cf. IZ:140; Downing 1998a,b).
  - This is effectuated with the constraint Max–Input-PRoot (11), as shown in (12).

(11) **Max-IP**: Assign one violation for each input segment which does not have an output correspondent contained within a Prosodic Root.

(12) **Max-IP induces PRoot parsing, e.g. for Type A**

/beiha/	Max-IP
a. bé.i.ha	5!
b. {bé.i}.ha	2!
c. ☞ {bé.i.ha}	

⋆ **Note:** it is crucial that D1 not output a PRoot.
- Therefore, D1 needs to have a constraint against PRoot's in the output (e.g. *PRoot) or against parsing segments into a PRoot (unclear how this would work), which outranks Max-IP.
- D2 needs to have the reverse ranking: Max-IP ≫ *PRoot

- The only constraint which forces violation of Max-IP in D2 is *Repeat(init).
  - If *Repeat(init) outranks Max-IP *and* Max-IO (13), we generate initial-C deletion for Type D only (14).
  - Since there is not general truncation in D2, we know that *Struc ranks below at least one of the Max constraints.

(13) **Ranking for initial-C deletion in Type D**: *Repeat(init) ≫ Max-IP, Max-IO

(14) ***Repeat(init)-driven deletion in D2 for Type D**

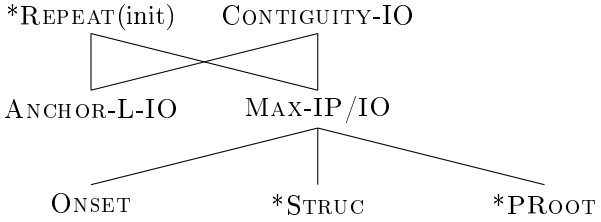
/gu$_1$gu$_2$ya/	*Repeat(init)	Max-IP	Max-IO	*Struc
a. {gu$_1$.gú$_2$.ya}	*!			6
b. g{u$_1$.gú$_2$.ya}	*!	*		6
c. ☞ {u$_1$.gú$_2$.ya}		*	*	5

- The fact that the *Repeat(init) violation is repaired by *initial* deletion and not medial deletion shows that Contiguity-IO ≫ Anchor-L-IO (15).
  - At least the higher-ranked Max constraint must also outrank Onset, because initial-C deletion (15b) is preferred to initial-CV deletion (15c).
  - The low ranking of Onset in D2 also explains why hiatus is tolerated stem-internally in D2 (e.g. {bé.i.ha}).

(15) ***Repeat(init)-driven deletion in D2 for Type D**

/gu_1gu_2ya/	*Repeat(init)	Contig-IO	Anch-L	Max-IP/IO	Onset
a. {gu_1.$gú_2$.ya}	*!				
b. ☞ {u_1.$gú_2$.ya}			*	*	*
c. {$gú_2$.ya}			*	**!	
d. {$gú_1$.ya}		*!		**	

(16) **Ranking summary for D2**



- D2 also enforces the language's normal stress pattern (see Ezard 1997):
  - Right-to-left trochees (i.e., penult primary stress + alternating stress leftward from the penult).
  - Unless the penult lacks an onset and the antepenult has a higher vowel, in which case stress retracts to the antepenult, as in *bé.i.ha.*

⋆ I will not spell out the stress constraints here, except to note that StressL must be low-ranked, unlike in D1.

4 Restricted hiatus resolution in M

- The only process in M is deletion to repair hiatus, which occurs at the reduplicant-base juncture for Type C (19).
  - This can be modeled using the ranking Onset ≫ Max-IO (18).
  - To prevent deletion of reduplicant-initial vowels in Type C, we could include Anchor-L-IO ≫ Onset in the rankings, or we could use a constraint protecting stressed vowels (17), which will be needed independently below.

(17) **MaxV́-IO**: Assign one violation for each stressed vowel in the input that lacks a correspondent in the output.

(18) **Ranking for hiatus resolution in M**: MaxV́-IO ≫ Onset ≫ Max-IO[1]

(19) **Onset-driven reduplicant-V_2 deletion in M for Type C**

/á.tu-{a.tú.na}/	MaxV́-IO	Max-PO	Onset	Max-IO
a. à.tu.{a.tú.na}			**!	
b. à.tu.{tú.na}		*!	*	*
c. ☞ à.t{a.tú.na}			*	*
d. t{a.tú.na}	*!			*

[1] If we assume that adjacency relations are established in the input across morpheme boundaries, then Contiguity-IO would have to rank below Onset as well to allow deletion in Type C.

- Crucially, hiatus is left unresolved in Type D. This can be derived from the combined effect of two special MAX constraints, both of which are sensitive to input properties of segments:
  - A constraint protecting underlyingly stressed vowel: MAXV́-IO (17).
  - A constraint protecting underlying PRoot segments: MAX-PO (20).

(20) **MAX–PROOT-OUTPUT [MAX-PO]**: Assign one violation for each vowel segment which was part of a PRoot in the input that does not have an output correspondent.

- These constraints, unlike general MAX-IO, rank *above* ONSET (21), and thus can block hiatus resolution.
  - They succeed in doing so across the juncture for Type D (22), because the lefthand vowel is stressed and the righthand vowel belongs to a PRoot.

(21) **Full ranking for hiatus (non-)resolution in M**: MAXV́-IO, MAX-PO ≫ ONSET ≫ MAX-IO

(22) **Hiatus tolerance across the juncture for Type D in M**

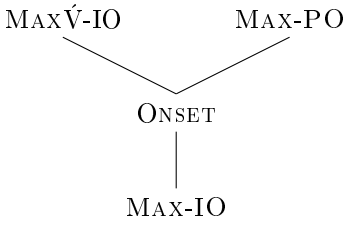
/gú-{u.gú.ya}/	MAXV́-IO	MAX-PO	ONSET	MAX-IO
a. ☞ gù.{u.gú.ya}			*	
b. gù.{gú.ya}		*!		*
c. g{u.gú.ya}	*!			*

- MAX-PO also explains why there is no hiatus-driven deletion base-internally in, e.g., Type A:

(23) **Hiatus tolerance base-internally for Type A in M**

/bí-{bé.i.ha}/	MAXV́-IO	MAX-PO	ONSET	MAX-IO
a. ☞ bì.{bé.i.ha}			*	
b. bì.{bé.ha}		*!		*

(24) **Ranking summary for M**



- The only other piece of phonology in M is demotion of the reduplicant's underlying primary stress to secondary stress.

5 An extra node for Type A: Hiatus-resolution in D1′

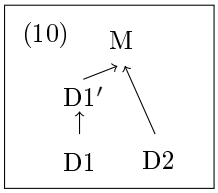
∗ There is one problem remaining with this analysis with respect to Type A (which was suppressed in (23)):
  - Given that the output of D1 for Type A is [bé.i], we predict that the unstressed /i/ should delete under hiatus in M, yielding *[bè.{bé.i.ha}] (25b).
  - If deletion of the /i/ were blocked by the fact that this would introduce an initial repetition (a *REPEAT(init) violation), the alternative would be *no deletion* [bè.i.{bé.i.ha}] (25a), not deletion of stressed /é/ (25c,d).
  - And even if something could prefer /é/ deletion, we would still need something else to force the addition of stress to the /i/ in order to overcome (25c).

(25) **Incorrect prediction for Type A in M**

/bé.i-{bé.i.ha}/	(*Rpt(init))	MaxV́-IO	Max-PO	Onset	Max-IO	Dep[stress]-IO
a. (💣) bè.i.{bé.i.ha}				*(!)		
b. 💣 bè.{bé.i.ha}	(*!)				*	
c. bi-{bé.i.ha}		*!			*	
d. ☹ bì-{bé.i.ha}		*!			*	*

- The simplest solution (following HHK:15–16) is to introduce an additional, semantically vacuous node [D1′] (26) between D1 and M that maps /bé.i/ → [bí] (28).

(26) **MDT derivational structure with D1′**



- The way to do this is as follows:
  - Have Onset outrank MaxV́-IO and Max-IO (27a) to make sure stressed vowel deletion will be tolerated as a means of avoiding hiatus.
  - Also have Onset outrank Contiguity-IO to allow for the discontiguous mapping resulting from V_1-deletion, and outrank Dep[stress]-IO to allow for restressing the /i/ (27a).
  - Have Anchor-R-IO outrank MaxV́-IO and Contiguity-IO (27b) to prefer keeping the rightmost vowel instead of the stressed one, which is also the one that's adjacent to the stem-initial consonant.
  - StressL must outrank Dep[stress]-IO (27c) to favor inserting stress onto the newly leftmost /i/, and also outrank NonFin (27c) (just as in D1) because this means that the final vowel will now be stressed.

(27) **Rankings in D1′ to fix Type A**

a. Onset ≫ MaxV́-IO, Max-IO, Contiguity-IO, Dep[stress]-IO [(28d) ≻ (28a)]
b. Anchor-R-IO ≫ MaxV́-IO, Contiguity-IO [(28d) ≻ (28b)]
c. StressL ≫ Dep[stress]-IO, NonFin [(28d) ≻ (28c)]

(28) **Fixing Type A in D1′**

/bé.i/	Onset	Anchor-R	StressL	MaxV́	Contiguity	Dep[stress]-IO	NonFin
a. bé.i	*!						
b. bé		*!					*
c. bi			*!	*	*		
d. ☞ bí				*	*	*	*

- We'll also need Anchor-L-IO to outrank Onset (29) in order to avoid deleting the stem-initial vowel in Type C (30).

(29) **Ranking for Type C in D1′**: Anchor-L-IO ≫ Onset

(30) **Not messing up Type C in D1′**

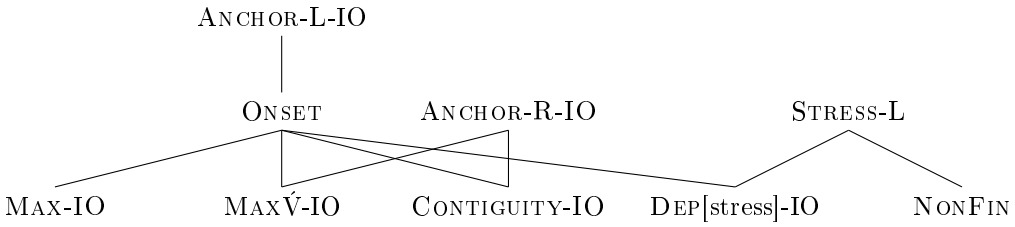
/á.tu/	Anchor-L	Onset	Anchor-R	MaxV́-IO	Contiguity
a. ☞ á.tu		*			
b. tú	*!			*	

- Once we implement these rankings in D1′, we achieve the desired result for Type A in M (31); in fact, there are no obvious competitor candidates.

(31) **Correct result for Type A in M**

/**bí**-{bé.i.ha}/	MaxV́-IO	Max-PO	Onset	Max-IO
a. ☞ bì-{bé.i.ha}				

(32) **Ranking summary for D1′**



6 Conclusion

- This analysis, while consistent, requires four substantially different cophonologies.
  - The processes that we need to posit in the different nodes bear little resemblance to one another, and sometimes even make contradictory demands.

- MDT provides us with the tools we need to fix all the problems.
  - But the facility with which we can use disparate tools in an ad hoc fashion should give us pause about the restrictiveness of the theory.

References

Downing, Laura J. 1998a. On the Prosodic Misalignment of Onsetless Syllables. *NLLT* 16(1):1–52.

———. 1998b. Prosodic Misalignment and Reduplication. In Geert Booij & Jaap van Marle (eds.), *Yearbook of Morphology 1997*, 83–120. Kluwer.

Ezard, Bryan. 1980. Reduplication in Tawala. *Kivung: Journal of the Linguistic Society of Papua New Guinea* 12(2):145–160. https://www.langlxmelanesia.com/llm-archive.

———. 1997. *A Grammar of Tawala: An Austronesian Language of the Milne Bay Area, Papua New Guinea.* Canberra: Pacific Linguistics.

Haugen, Jason D. & Cathy Hicks Kennard. 2011. Base-Dependence in Reduplication. *Morphology* 21(1):1–29.

Hicks Kennard, Catherine. 2004. Copy but Don't Repeat: The Conflict of Dissimilation and Reduplication in the Tawala Durative. *Phonology* 21(3):303–323.

Inkelas, Sharon & Cheryl Zoll. 2005. *Reduplication: Doubling in Morphology.* Cambridge, UK: Cambridge University Press.

McCarthy, John J. & Alan Prince. 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.

Zukoff, Sam. 2021. Contiguity in Tawala Reduplication. Paper Presented at Mfm 28, Manchester, UK. https://www.samzukoff.com/mfm2021handout.