Long Distance Processes in Tone and Subsequentiality

Tajudeen Mamadou

Rutgers

NECPhon 2020

Roadmap

- What is the right characterization for tonal processes?
  - The key notion here is determinism
  - Introduce a new class of functions (IML functions)
  - Model them with a restrictive type of finite state transducers
  - Show the new class's empirical coverage
- Carve out a deterministic class for tone processes, excluding known pathologies

[Tonal functions are maximally Input or Output Melody Local]

Background

- Most phonological processes are **Subsequential:**
  - A property of **string functions**

[Payne, 2017, Chandlee, 2014, Heinz and Lai, 2013, Mohri, 1997]

Background

- A majority of phonological processes are **Subsequential:**
  - A property of **string functions**
  - Characterizes processes with trigger(s) on **one end** of and/or **local** to a given domain

$\underbrace{\boxed{a} \longrightarrow \text{TARGETS}}_{\textbf{Left-Subseq}}$ *or* $\underbrace{\text{TARGETS} \longleftarrow \boxed{a}}_{\textbf{Right-Subseq}}$ ✓

[Payne, 2017, Chandlee, 2014, Heinz and Lai, 2013, Mohri, 1997]

Background

- A majority of phonological processes are **Subsequential:**
  - A property of **string functions**
  - Characterizes processes with trigger(s) on **one end** of and/or **local** to a given domain

$\underbrace{\boxed{a} \longrightarrow \text{TARGETS}}_{\textbf{Left-Subseq}}$ *or* $\underbrace{\text{TARGETS} \longleftarrow \boxed{a}}_{\textbf{Right-Subseq}}$ ✓

- Computed **deterministically** with Finite State Transducers

[Payne, 2017, Chandlee, 2014, Heinz and Lai, 2013, Mohri, 1997]

Background

- A majority of phonological processes are **Subsequential:**
  - A property of **string functions**
  - Characterizes processes with trigger(s) on **one end** of and/or **local** to a given domain

$$\underbrace{\boxed{a} \longrightarrow \text{TARGETS}}_{\textbf{Left-Subseq}} \quad \textit{or} \quad \underbrace{\text{TARGETS} \longleftarrow \boxed{a}}_{\textbf{Right-Subseq}} \qquad \checkmark$$

- Computed **deterministically** with Finite State Transducers

$$\underbrace{\boxed{a} \longrightarrow \text{TARGETS} \longleftarrow \boxed{b}}_{\textbf{Non-Subseq}} \qquad ✗$$

[Payne, 2017, Chandlee, 2014, Heinz and Lai, 2013, Mohri, 1997]

Tone & Non-subsequentiality

- Tone includes Unbounded Circumambient processes (UC)

[Hyman, 2011, Jardine, 2016]

Tone & Non-subsequentiality

- Tone includes Unbounded Circumambient processes (UC)
  - Triggers can be unboundedly far away in both directions

Tone & Non-subsequentiality

- Tone includes Unbounded Circumambient processes (UC)
  - Triggers can be unboundedly far away in both directions
  - E.g: UTP in Luganda

a.	/mutéma+bisikí/	[mutémá+bísíkí]	‘log chopper’
b.	Unattested	*[mutéma+bisikí]	_

LHLLLLL	→	L**H**LLLLL
LHLL(...)LHL	→	L**HHH(...)HH**L

[Hyman and Katamba, 2010]

Tone & Non-subsequentiality

a ⟶ TARGETS ⟵ *b*

Non-Subseq

- Tone includes Unbounded Circumambient processes (UC)
  - Triggers can be unboundedly far away in both directions
  - E.g: UTP in Luganda

a.	/mutéma+bisikí/	[mutémá+bísíkí]	‘log chopper’
b.	Unattested	*[mutéma+bisikí]	_

LHLLLLL	→	L**H**LLLLL
LHLL(...)LHL	→	L**HHH(...)HH**L

- UC processes are non-subsequential

[Hyman and Katamba, 2010, Jardine, 2016]

Tone & Non-subsequentiality

a ⟶ TARGETS ⟵ *b*

Non-Subseq

- Tone includes Unbounded Circumambient processes (UC)
  - Triggers can be unboundedly far away in both directions
  - E.g: UTP in Luganda

a.	/mutéma+bisikí/	[mutémá+bísíkí]	'log chopper'
b.	Unattested	*[mutéma+bisikí]	_

LHLLLLL	→	L**H**LLLLL
LHLL(...)LHL	→	L**HHH(...)HH**L

- UC processes are non-subsequential
- We can make them deterministic with a two tier representation

[Hyman and Katamba, 2010, Jardine, 2016]

Input Melody Local (IML)

- IML functions extend subsequentiality to UTP-like processes, a.o

Input Melody Local (IML)

- IML functions extend subsequentiality to UTP-like processes, a.o
- Represent a sub-class of a previously introduced Multi-Input Strictly Local (MISL) class

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020]

Input Melody Local (IML)

- IML functions extend subsequentiality to UTP-like processes
- Represent a sub-class of a recently introduced Multi-Input Strictly Local (MISL) class
- Intuitively, we are enriching the representation while maintaining a notion of autosegmental locality

[Odden, 1994]

Input Melody Local (IML)

- **Two key components**:
  - **A Melody Function**: Only retains one symbol in each span of tones (like the OCP); assumes underlying associations

    `mel`(LHLLLLH)=LHLH

[Jardine, 2020]

Input Melody Local (IML)

- **Two key components**:
  - **A Melody Function**: Only retains one symbol in each span of tones (like the OCP); assumes underlying associations

    `mel`(LHLLLLH)=LHLH

  - **The IO Function**: Takes a combination of timing and melody tier symbols as input

Input Melody Local (IML)

- **Two key components**:
  - **A Melody Function**: Only retains one symbol in each span of tones (like the OCP); assumes underlying associations

    `mel`(LHLLLLH)=LHLH

  - **The IO Function**: Takes a combination of timing and melody tier symbols as input

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

IML Functions

- Intuitively, we want our function to be computed deterministically as follows:

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : L

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHHHH

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHHHH

- In general, multi-tier function takes an input of the type $\langle \texttt{mel}(w), w \rangle$:

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHHHH

- In general, multi-tier function takes an input of the type $\langle \texttt{mel}(w), w \rangle$:
- E.g: $f_{UTP}(\langle \texttt{mel}(LHLH), LHLLLLLH \rangle) = LHHHHHHH$

Input Melody Local (IML)

- Intuitively, we want our function to be computed deterministically as follows

Input

melody	L	H	L					H
timing	L	H	L	L	L	L	L	H

Output : LHHHHHHH

- In general, multi-tier function takes an input of the type $\langle \texttt{mel}(w), w \rangle$:
- E.g: $f_{UTP}(\langle \texttt{mel}(LHLH), LHLLLLLH \rangle) = LHHHHHHH$
- Note also that any string function can be converted into a multi-tier one

IML Functions and Automata

IML are modeled with asynchronous 1-way Deterministic Multi-tape Finite State Transducers (DMFSTs)

[Rawski and Dolatian, 2020, Furia, 2012, Elgot and Mezei, 1965, Rabin and Scott, 1959]

IML Functions and Automata

IML are modeled with asynchronous 1-way Deterministic Multi-tape Finite State Transducers (DMFSTs)

- FST with a set of states and transitions

[Rawski and Dolatian, 2020, Furia, 2012, Elgot and Mezei, 1965, Rabin and Scott, 1959]

IML Functions and Automata

IML are modeled with asynchronous 1-way Deterministic Multi-tape Finite State Transducers (DMFSTs)

- FST with a set of states and transitions
- transitions are of the form: $\delta = (\text{p}, \text{X}|\text{Y}, \text{Z}, \text{q})$

[Rawski and Dolatian, 2020, Furia, 2012, Elgot and Mezei, 1965, Rabin and Scott, 1959]

IML Functions and Automata

IML are modeled with asynchronous 1-way Deterministic Multi-tape Finite State Transducers (DMFSTs)

- FST with a set of states and transitions
- transitions are of the form: δ = (p, X|Y, Z, q)

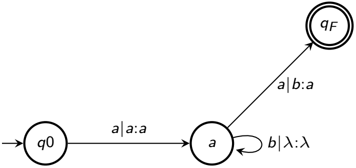


Figure: A Toy DM-FST.

[Rawski and Dolatian, 2020, Furia, 2012, Elgot and Mezei, 1965, Rabin and Scott, 1959]

IML Functions and Automata

- Additional constraints on transitions are needed:

IML Functions and Automata

- Additional constraints on transitions are needed:
  - **Determinism:**
    - transitions out of the same state cannot differ only by replacing λ (for the same Y, $X_1 \neq X_2 \neq \lambda$)
    - both input symbols cannot be empty strings (X or Y $\neq \lambda$)

IML Functions and Automata

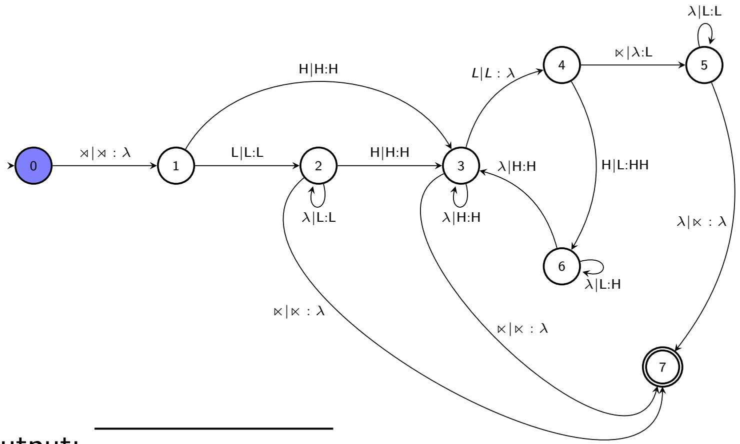
- Additional constraints on transitions are needed:
  - **Determinism:**
    - transitions out of the same state cannot differ only by replacing λ (for the same Y, $X_1 \neq X_2 \neq \lambda$)
    - both input symbols cannot be empty strings (X or Y $\neq \lambda$)
  - **Locality:**
    - each state represents j-1 and k-1 factors, respectively on the melody and timing tapes

IML Functions and Automata

- Additional constraints on transitions are needed:
  - **Determinism:**
    - transitions out of the same state cannot differ only by replacing λ (for the same Y, $X_1 \neq X_2 \neq \lambda$)
    - both input symbols cannot be empty strings (X or Y $\neq \lambda$)
  - **Locality:**
    - each state represents j-1 and k-1 factors, respectively on the melody and timing tapes
  - **Melody:**
    - both the melody and timing tapes share the same input alphabet ($X, Y \in \Sigma \cup \{\lambda\}$, where Σ={H,L})

Luganda Derivation: With Tone Plateauing

Input	mel(w)	⋊	L	H	L			H	⋉
	w	⋊	L	H	L	L	L	H	⋉

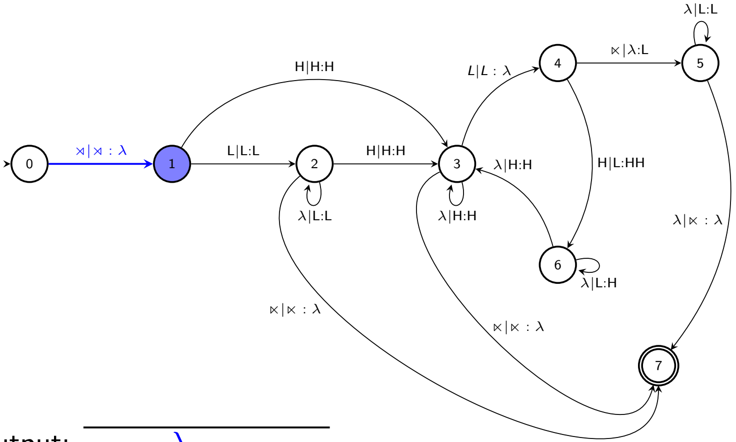


Output: ______________

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

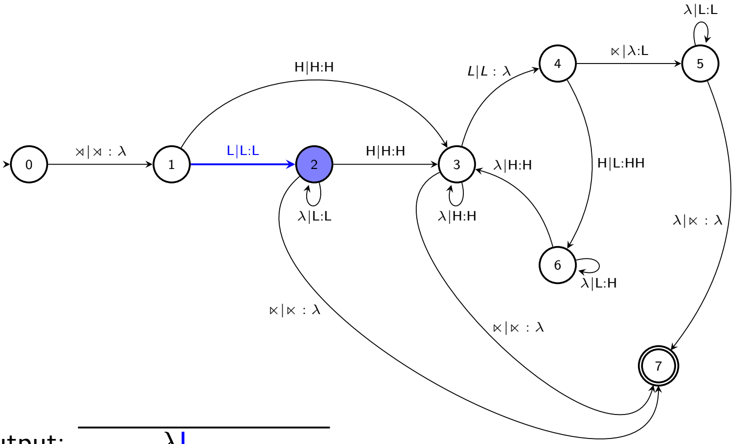


Output: λ

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

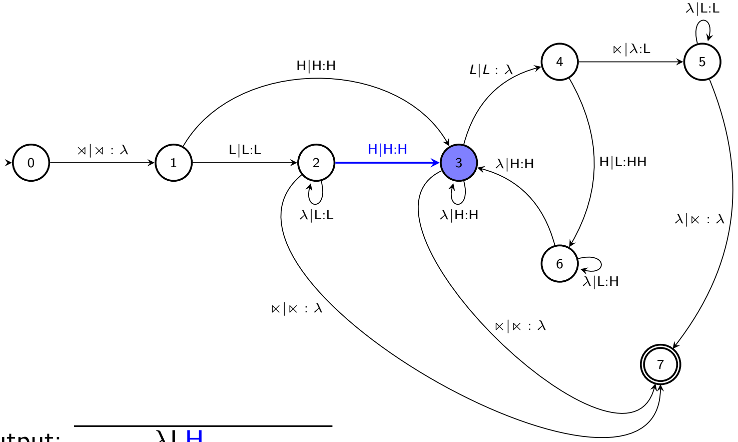


Output: λL

Luganda Derivation: With Tone Plateauing

Input:

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

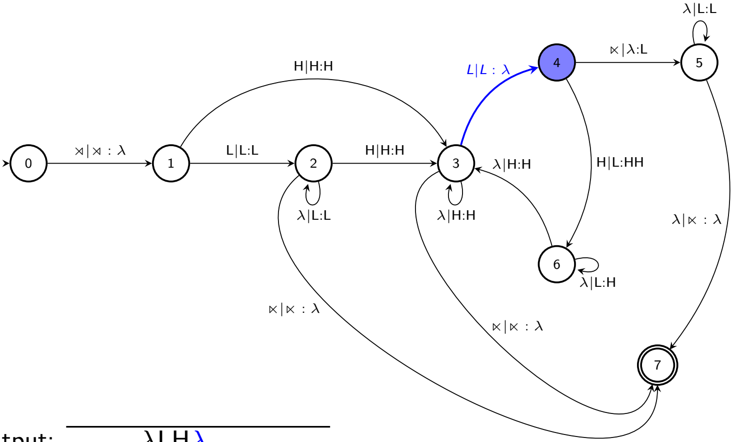


Output: λLH

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

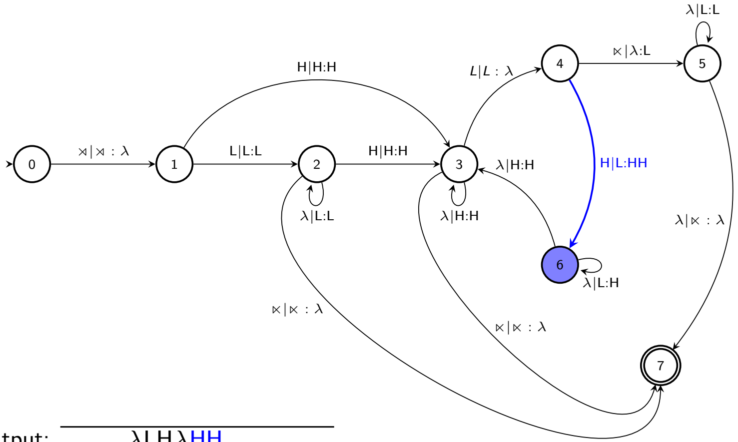


Output: λLHλ

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

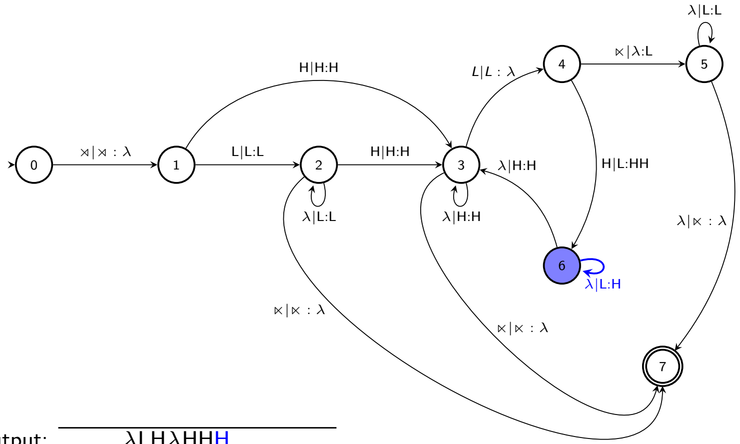


Output: λLHλHH

Luganda Derivation: With Tone Plateauing

Input

mel(w)	$\rtimes$	L	H	L			H	$\ltimes$
w	$\rtimes$	L	H	L	L	L	H	$\ltimes$

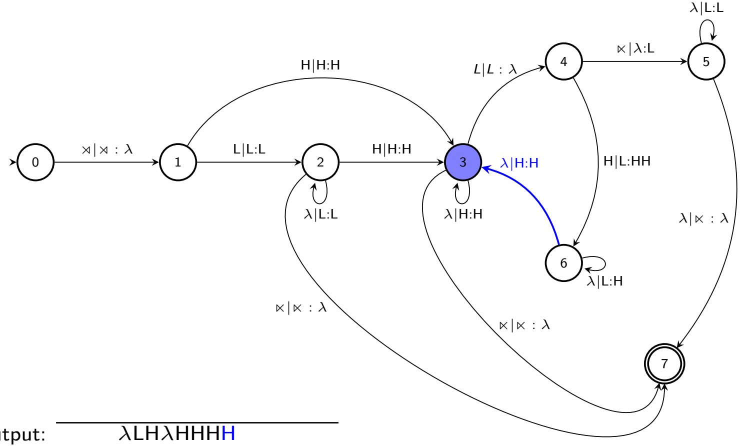


Output: λLHλHHH

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

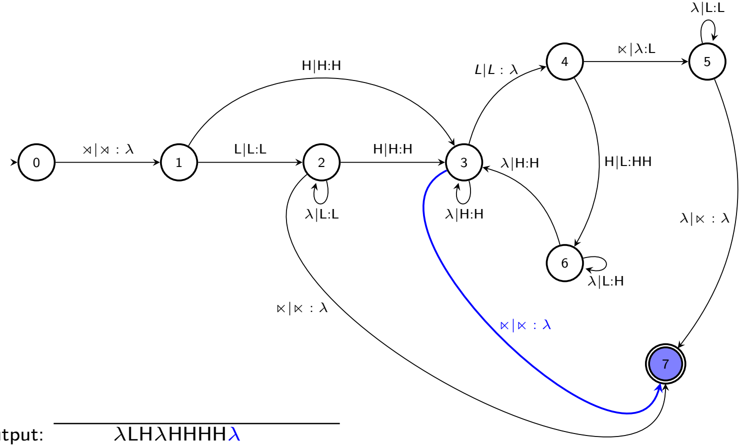


Output: λLHλHHHH

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

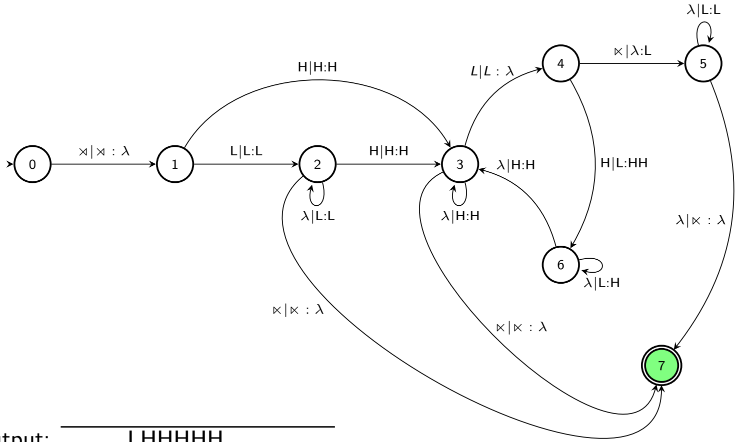


Output: λLHλHHHHλ

Luganda Derivation: With Tone Plateauing

Input

mel(w)	⋊	L	H	L			H	⋉
w	⋊	L	H	L	L	L	H	⋉

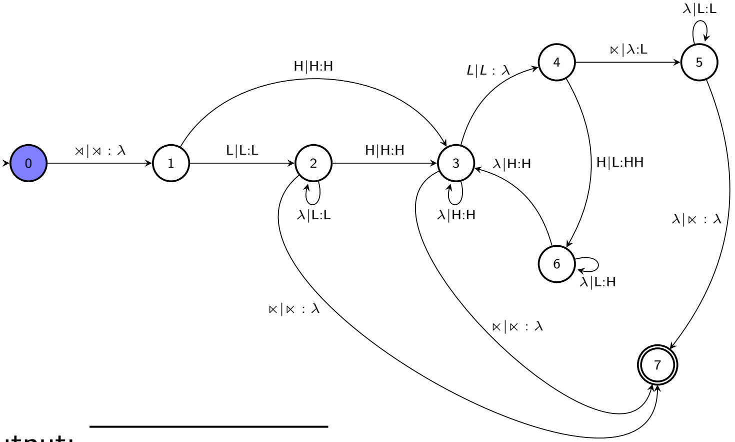


Output: LHHHHH

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

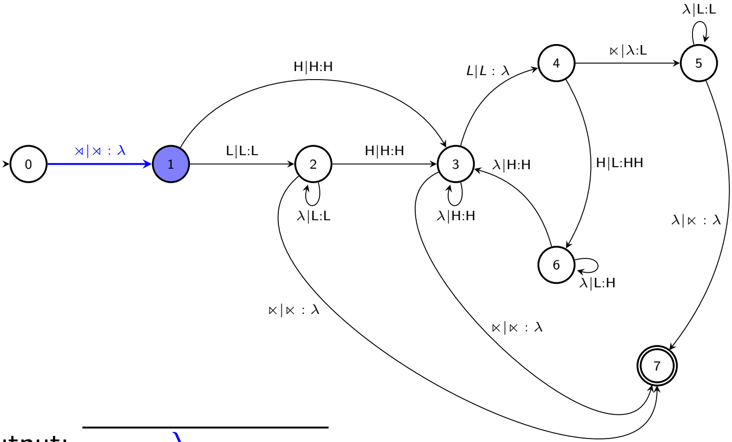


Output: ____________________

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

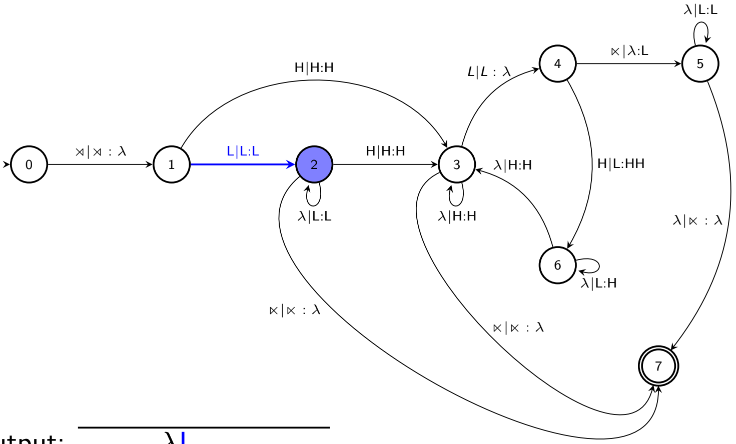


Output: λ

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

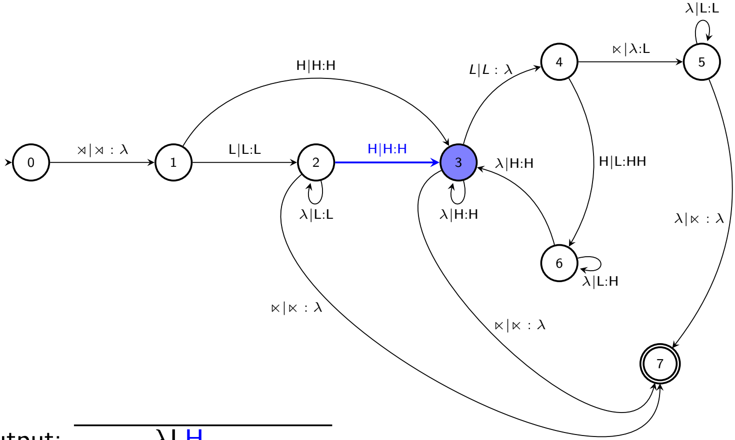


Output: λL

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

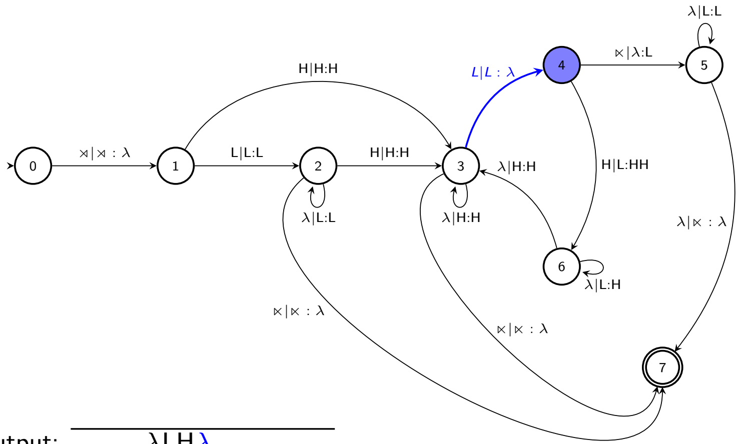


Output: λLH

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

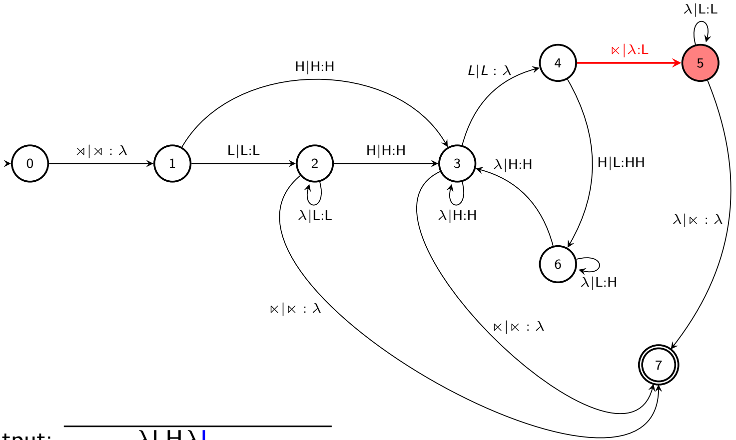


Output: λLHλ

Luganda Derivation 2: Without Tone Plateauing

Input

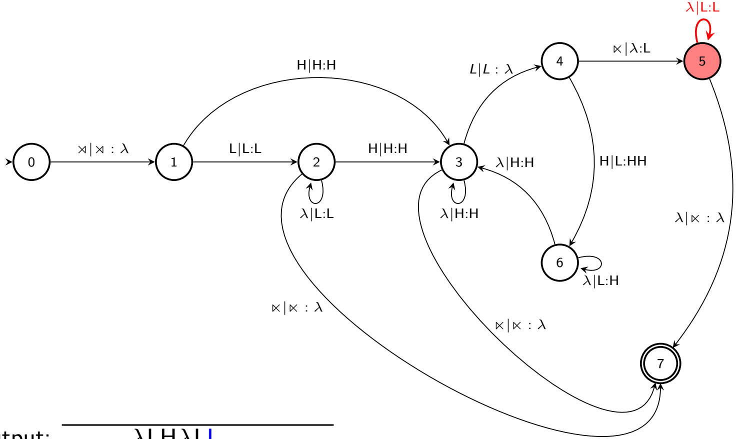
mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉



Output: λLHλL

Luganda Derivation 2: Without Tone Plateauing

Input								
mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

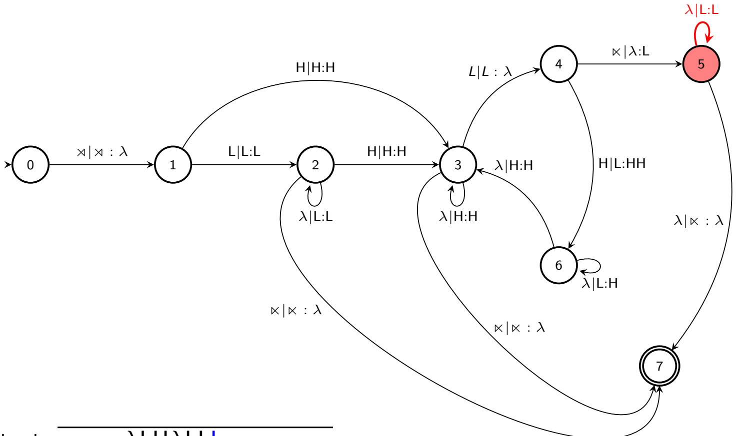


Output: λLHλLL

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

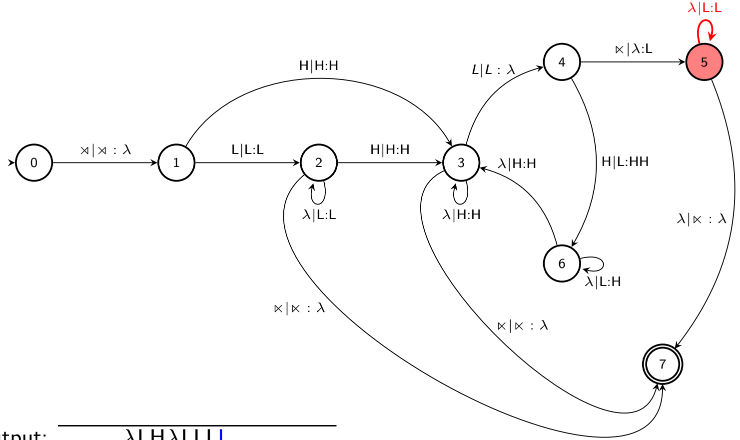


Output: λLHλLLL

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

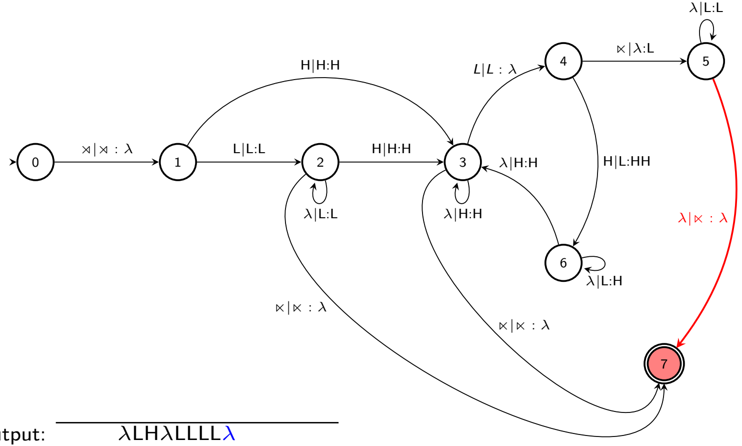


Output: λLHλLLLL

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉

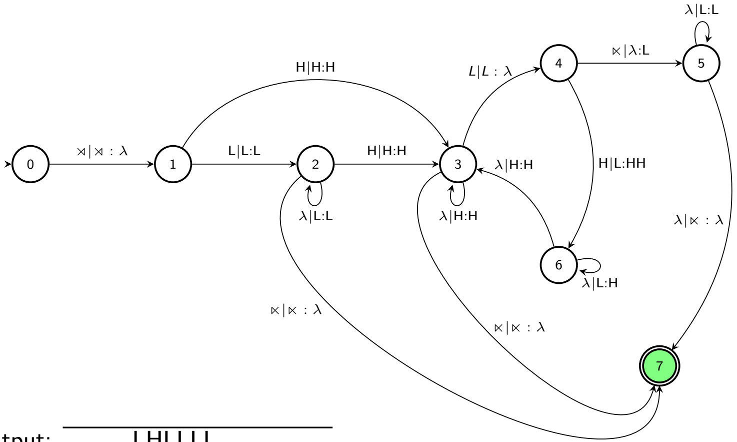


Output: λLHλLLLLλ

Luganda Derivation 2: Without Tone Plateauing

Input

mel(w)	⋊	L	H	L				⋉
w	⋊	L	H	L	L	L	L	⋉



Output: LHLLLL

Empirical Summary and Comparative Table

Tone Patterns and their Subregular Classes							
Patterns	Languages	ISL	OSL	A-ISL	Subseq	MISL	IML
Bounded shift	Rimi	✓	✗	✓	✓	✓	✓
Bounded Spread	Bemba	✓	✓	✓	✓	✓	✓
Bounded Meussen's Rule	Luganda	✓	✓	✗	✓	✓*	✓
Unbounded Shift	Zigula	✗	✗	✓	✓	✓	✓
Unbounded Spread	Ndebele	✗	✓	✗	✓	✓	✓
Unbounded Deletion	Arusa	✗	✗	✓	✓	✓	✓
Anticipatory downstep	Tiriki	✗	✓	✓	✓	-	✓
Anticipatory Upstep	Amo	✗	✗	✓(?)	✗	-	✓
UTP	Luganda	✗	✗	✗	✗	✓	✓
SG-like	C. Bemba	✗	✗	✗	✗	✓	✓
AMR	Shona	✗	✓	✗	✓	✗	✗
*Majority Rule[1]	-	-	-	-	-	-	✗
*Midpoint Pathology[2]	-	-	-	-	-	-	✗

[1] [Baković, 2000, Heinz and Lai, 2013]

[2] [Eisner, 1997]

Discussion

- IML functions were developed in parallel with MISL
  - which are also computed by deterministic asynchronous Multi-tape FSTs

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020]

Discussion

- IML functions were developed in parallel with MISL
  - which are also computed by deterministic asynchronous Multi-tape FSTs
- Crucially, IML and MISL functions differ in two important regards:

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020]

Discussion

- IML functions were developed in parallel with MISL
  - which are also computed by deterministic asynchronous Multi-tape FSTs
- Crucially, IML and MISL functions differ in two important regards:
  - Unlike with MISL, the input tapes of IML-computing DMFSTs are connected via a shared alphabet

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020]

Discussion

- IML functions were developed in parallel with MISL
  - which are also computed by deterministic asynchronous Multi-tape FSTs
- Crucially, IML and MISL functions differ in two important regards:
  - Unlike with MISL, the input tapes of IML-computing DMFSTs are connected via a shared alphabet
  - Any IML function is also MISL, but not vice versa

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020]

Discussion

- The empirical results of the IML functions speak to 'phonological directionality' as well

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020, Zoll, 2003]

Discussion

- The empirical results of the IML functions speak to 'phonological directionality' as well
  - All of the patterns investigated fall in the intersection of L-IML and R-IML[3]

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020, Zoll, 2003]

[3] Though AU in Amo may be an exception.

Discussion

- The empirical results of the IML functions speak to 'phonological directionality' as well
  - All of the patterns investigated fall in the intersection of L-IML and R-IML[4]
  - Suggesting that directionality needs not be encoded in the grammar

[Rawski and Dolatian, 2020, Dolatian and Rawski, 2020, Zoll, 2003]

[4] Though AU in Amo may be an exception.

Discussion

- The empirical results of the IML functions speak to 'phonological directionality' as well
  - All of the patterns investigated fall in the intersection of L-IML and R-IML[5]
  - Suggesting that directionality needs not be encoded in the grammar
- The Autosegmental Theory's Well-formedness conditions are preserved (for free):

[Zoll, 2003, Goldsmith, 1976, Williams, 1976]

[5] Though AU in Amo may be an exception.

Discussion

- The empirical results of the IML functions speak to 'phonological directionality' as well
  - All of the patterns investigated fall in the intersection of L-IML and R-IML[6]
  - Suggesting that directionality needs not be encoded in the grammar
- The Autosegmental Theory's Well-formedness conditions are preserved (for free):
  - *No-gapping* constraint is satisfied by the melody function
  - *No-crossing* constraint is also satisfied through determinism and the melody

[Zoll, 2003, Goldsmith, 1976, Williams, 1976]

[6] Though AU in Amo may be an exception.

Future Research

- Expressivity of multi-tape FSTs
  - Their formal characterization is in progress

Future Research

- Expressivity of multi-tape FSTs
  - Their formal characterization is in progress
- The Alternating Meussen's Rule is not IML, we conjecture it is Ouput Melody Local (OML)
  - OML is yet to be defined but can be based on [Chandlee, 2014] 's ISL-OSL

Future Research

- Expressivity of multi-tape FSTs
  - Their formal characterization is in progress
- The Alternating Meussen's Rule is not IML, we conjecture it is Ouput Melody Local (OML)
  - OML is yet to be defined but can be based on [Chandlee, 2014] ' ISL/OSL
  - Suggests that tone functions are not all IML, but rather ML (IML&OML)
- IML functions, as currently formulated, work best for languages with underlying associations

Take-away Message

[Enriching the representation allows for a deterministic characterization of UC processes, a.o.]

Thanks!

Appendix 1: The two IML component functions

- **A Melody Function:** (Adapted from Jardine, 2020a)

$$\texttt{mel}(w) \stackrel{\text{def}}{=} \lambda \quad \text{if } w = \lambda,$$
$$\stackrel{\text{def}}{=} \texttt{mel}(v)\sigma \quad \text{if } w = v\sigma^n,\ v \neq u\sigma \text{ forsome} u \in \Sigma^*$$

- **The IO Function**: (e.g: UTP)

$$f_{utp}(\langle \texttt{mel}(w), w \rangle) \stackrel{\text{def}}{=} \text{L}^m H^{(2n+o)} \quad \text{if } w = \text{L}^m H^n L^o H^n,\ \texttt{mel}(w) = \text{(L)H(L)H};\ m \text{ \& o} \geq 0, \text{n} = 1$$
$$\stackrel{\text{def}}{=} w \quad \text{elsewhere.}$$

Appendix 2: A DMFST for Bounded Tone Shift in Rimi

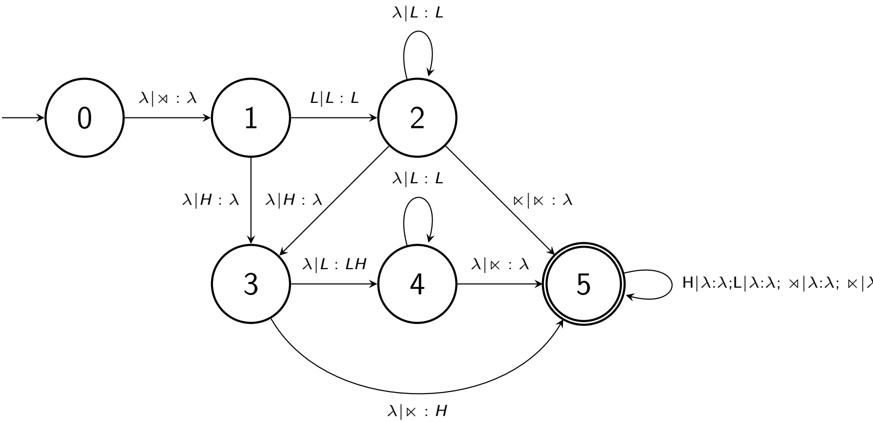


Figure: A 2-tape DM-FST for the one-step (bounded) tone shift in Rimi.

Appendix 3: Alternating Meussen's Rule in Shona

- Examples:

a.	/né-hóvé/	[né-hòvè]	'with-fish'
b.	/né-é-hóvé/	[né-è-hóvé]	'with-of-fish'
c.	/sé-né-é-hóvé/	[sé-nè-é hòvè]	'like-with-of-fish'

- AMR is not IML because the states of the DM-FST computing it do not represent $j-1$ and $k-1$ (input) factors.

[Odden, 1986, Chandlee and Jardine, 2019]

Appendix 4: Derivation Table for Luganda UTP

- For UTP, consider $w = \rtimes\mathrm{LHLLLH}\ltimes$ and melody $\mathtt{mel}(w) = \rtimes\mathrm{LHLH}\ltimes$

Step	Current state	Melody tape	Timing tape	Transition	Dest. state	Output
1.	*q0*	$\underline{\rtimes}\mathrm{LHLH}\ltimes$	$\underline{\rtimes}\mathrm{LHLLLH}\ltimes$	$\rtimes\|\rtimes{:}\lambda$	*q1*	
2.	*q1*	$\rtimes\underline{\mathrm{L}}\mathrm{HLH}\ltimes$	$\rtimes\underline{\mathrm{L}}\mathrm{HLLLH}\rtimes$	L|L:L	*q2*	L
3.	*q2*	$\rtimes\mathrm{L}\underline{\mathrm{H}}\mathrm{LH}\ltimes$	$\rtimes\mathrm{L}\underline{\mathrm{H}}\mathrm{LLLH}\rtimes$	H|H:H	*q3*	LH
4.	*q3*	$\rtimes\mathrm{LH}\underline{\mathrm{L}}\mathrm{H}\ltimes$	$\rtimes\mathrm{L}\underline{\mathrm{H}}\mathrm{LLLH}\rtimes$	$\mathrm{H}\|\lambda{:}\lambda$	*q4*	LH
5.	*q4*	$\rtimes\mathrm{LHL}\underline{\mathrm{H}}\ltimes$	$\rtimes\mathrm{LH}\underline{\mathrm{L}}\mathrm{LLH}\rtimes$	H|L:H	*q6*	LHH
6.	*q6*	$\rtimes\mathrm{LHL}\underline{\mathrm{H}}\ltimes$	$\rtimes\mathrm{LHL}\underline{\mathrm{L}}\mathrm{LH}\rtimes$	$\lambda\|\mathrm{L{:}H}$	*q6*	LHHH
7.	*q6*	$\rtimes\mathrm{LHL}\underline{\mathrm{H}}\ltimes$	$\rtimes\mathrm{LHLL}\underline{\mathrm{L}}\mathrm{H}\rtimes$	$\lambda\|\mathrm{L{:}H}$	*q6*	LHHHH
8.	*q6*	$\rtimes\mathrm{LHL}\underline{\mathrm{H}}\ltimes$	$\rtimes\mathrm{LHLLL}\underline{\mathrm{H}}\rtimes$	$\lambda\|\mathrm{H{:}H}$	*q3*	LHHHHH
9.	*q3*	$\rtimes\mathrm{LHLH}\underline{\ltimes}$	$\rtimes\mathrm{LHLLLH}\underline{\rtimes}$	$\ltimes\|\ltimes{:}\lambda$	*q7*	LHHHHH
10.	*q7*	$\rtimes\mathrm{LHLH}\ltimes$	$\rtimes\mathrm{LHLLLH}\rtimes$			LHHHHH

Baković, E. (2000).
Harmony, dominance and control.
PhD thesis, Rutgers University.

Chandlee, J. (2014).
Strictly Local Phonological Processes.
PhD thesis, University of Delaware.

Chandlee, J. and Jardine, A. (2019).
Autosegmental input-strictly local functions.
Transactions of the Association for Computational Linguistics, 7:157–168.

Dolatian, H. and Rawski, J. (2020).
Multi-input strictly local functions for templatic morphology.
Proceedings of the Society for Computation in Linguistics (SCiL), pages 282–296.

Eisner, J. (1997).

Efficient generation in primitive Optimality Theory.
In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 313–320, Madrid.

Elgot, C. C. and Mezei, J. E. (1965).
On relations defined by generalized finite automata.
IBM Journal of Research and Development, 9:47—68.

Furia, C. A. (2012).
A survey of multi-tape automata.
Technical report.

Goldsmith, J. (1976).
Autosegmental Phonology.
PhD thesis, Massachussets Institute of Technology.

Heinz, J. and Lai, R. (2013).
Vowel harmony and subsequentiality.

In Kornai, A. and Kuhlmann, M., editors, *Proceedings of the 13th Meeting on Mathematics of Language*, Sofia, Bulgaria.

Hyman, L. M. (2011).
Tone: Is it different.
The handbook of phonological theory, 2.

Hyman, L. M. and Katamba, F. X. (2010).
Tone, syntax, and prosodic domains in luganda.
UC Berkeley PhonLab Annual Report, 6(6).

Jardine, A. (2016).
Computationally, tone is different.
Phonology, 33(2):247–283.

Jardine, A. (2020).
Melody learning and long-distance phonotactics in tone.
Natural Language and Linguistic Theory, pages 1–51.

Mohri, M. (1997).

Finite-state transducers in language and speech processing.
Computational Linguistics, 23(2):269–311.

Odden, D. (1986).
On the role of the Obligatory Contour Principle in phonological theory.
Language, 62(2):353–383.

Odden, D. (1994).
Adjacency parameters in phonology.
Language, pages 289–330.

Payne, A. (2017).
All dissimilation is computationally subsequential.
Language: Phonological Analysis, 93(4):e353–371.

Rabin, M. O. and Scott, D. (1959).
Finite automata and their decision problems.
IBM Journal of Research and Development, 3(2):114–125.

Rawski, J. and Dolatian, H. (2020).
Multi-input strict local functions for tonal phonology.
Proceedings of the Society for Computation in Linguistics, 3(1):245–260.

Williams, E. S. (1976).
Underlying tone in margi and igbo.
Linguistic Inquiry, pages 463–484.

Zoll, C. (2003).
Optimal tone mapping.
Linguistic Inquiry, 34(2):225–268.