Triggers and Blockers as Domain Edges

Marjorie Leduc
Rutgers University
Logical Phonology Workshop 2026

March 13, 2026

Introduction

Purpose of the talk:

- Propose that triggers and blockers are domain edges carving out an area within which phonological change occurs
- Show that variations in harmony patterns falls out from how processes interact with those edges.

Empirical Domain: [+ATR] Harmony

	[+ATR]	
	Front	Back
High	i	u
Mid	e	o
low		ə

	[−ATR]	
	Front	Back
High	ɪ	ʊ
Mid	ɛ	ɔ
Low		a

Vowel inventory split into two sets

Common in African languages

Introduction

Standard harmony:
/ɛ. . . ɛ. . . i/ → [e. . . e. . . i]

Introduction

/a/ as blocker:

- Does not harmonize
- Prevent harmony from extending beyond it

/ɛ...a...i/ → [ɛ...a...i]

Maasai

This is the pattern that we see in Maasai (Eastern Nilotic; Tucker & Mpaayei 1955)

[+ATR] vowels cause [−ATR] vowels to surface as [+ATR]

	UR	SR	Gloss
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]	'wash-INTR-APPL'
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-iʃo-re]	'2SG-cut-INTR-APPL.'

Maasai

When /a/ is present, however, this vowel neither undergoes harmony nor allows harmony to proceeds past itself.

	UR	SR	Gloss
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]	'wash-INTR-APPL'
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-iʃo-re]	'2SG-cut-INTR-APPL.'

Maasai

In this sense, the behavior of the segments in Maasai follows a very standard categorization:

Triggers: [+ATR] vowels (*e.g.*, /i/)
Targets: Non-low [−ATR] vowels (*e.g.*, /e/)
Blocker: low [−ATR] vowels (/a/)

Maasai

However, there exists certain patterns that do not follow this categorization:

Icy Targets
Use It or Lose It
Sour Grapes

Maasai

Note: To keep things straightforward, I will adapt the Maasai pattern shown above into different toy languages.

Icy Targets

Icy Targets (Jurgec 2011)

In this first pattern, an icy target is a specific segment that assumes *both* the identity of a target and a blocker.

	Maasai-IT	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-əs-iʃo-re]

Icy Targets

In this example, the low vowel /a/ does undergo [+ATR] harmony, but the feature doesn't affect the vowel /ɪ/ that precedes it, despite /ɪ/ being a licit target.

	Maasai-IT	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-əs-iʃo-re]

Icy Targets

It simultaneously block and undergoes the harmony process

	Maasai-IT	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-əs-iʃo-re]

Use It or Lose It

Use It or Lose It (Mullin & Pater 2015)

> Use It or Lose It is a pattern where a *trigger* responsible for spreading a feature becomes the harmonizing value in the presence of a blocker.

Use It or Lose It

Use It or Lose It (Mullin & Pater 2015)

> [+ATR] harmony proceeds successfully so long as the feature can spread throughout the entire domain

	Maasai-UIoLI	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-ɪʃɔ-rɛ]

Use It or Lose It

Use It or Lose It (Mullin & Pater 2015; UIoLI)

The presence of a blocker anywhere in the string, however, causes that [+ATR] vowel to surface as [−ATR]

	Maasai-UIoLI	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-ɪʃɔ-rɛ]

Sour Grapes

Sour Grapes (Padgett 1995; Wilson 2003)

In Sour Grapes patterns, the presence of a blocker anywhere in the string causes [+ATR] harmony to "give up"

	Maasai-SG	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-ɪʃɔ-re]

Sour Grapes

Sour Grapes (Padgett 1995; Wilson 2003)

In SG patterns, the blocker does not have to intervene between the trigger and the target.

Its presence *anywhere* in the string is sufficient to halt the propagation.

	Maasai-SG	
	UR	SR
a.	/ɪsʊj-ɪʃɔ-re/	[isuj-iʃo-re]
b.	/ɪ-as-ɪʃɔ-re/	[ɪ-as-ɪʃɔ-re]

Interim Summary

Within a broader pattern of [+ATR] harmony, the difference between these patterns is how the presence/absence of a blocker affects the behavior of [+ATR] harmony

Pattern	Surface Form	Difference
Full Harmony	[i-əs-iʃo-re]	Everything harmonizes
Partial Harmony	[ɪ-as-iʃo-re]	/a/ is inert and blocks harmony
Icy Targets	[ɪ-əs-iʃo-re]	/a/ is both a target and a blocker
UIoLI	[ɪ-as-ɪʃɔ-rɛ]	[+ATR] vowel surfaces as [−ATR]
Sour Grapes	[ɪ-as-ɪʃɔ-re]	Nothing Happens

Interim Summary

The "incomplete harmony" patterns have been treated as quite different and independent from each other

Pattern	Surface Form	Difference
Full Harmony	[i-əs-iʃo-re]	Everything harmonizes
Partial Harmony	[ɪ-as-iʃo-re]	/a/ is inert and blocks harmony
Icy Targets	[ɪ-əs-iʃo-re]	/a/ is both a target and a blocker
UIoLI	[ɪ-as-ɪʃɔ-rɛ]	[+ATR] vowel surfaces as [−ATR]
Sour Grapes	[ɪ-as-ɪʃɔ-re]	Nothing Happens

Interim Summary

But they share a lot of things!

Pattern	Surface Form	Difference
Partial Harmony	[ɪ-as-iʃo-re]	/a/ is inert and blocks harmony
Icy Targets	[ɪ-əs-iʃo-re]	/a/ is both a target and a blocker
UIoLI	[ɪ-as-ɪʃɔ-rɛ]	[+ATR] vowel surfaces as [−ATR]
Sour Grapes	[ɪ-as-ɪʃɔ-re]	Nothing Happens

Interim Summary

And so the broader question that motivated this presentation is "can we find something that unify these processes, or are the similarities purely coincidental?"

Pattern	Surface Form	Difference
Full Harmony	[i-əs-iʃo-re]	Everything harmonizes
Partial Harmony	[ɪ-as-iʃo-re]	/a/ is inert and blocks harmony
Icy Targets	[ɪ-əs-iʃo-re]	/a/ is both a target and a blocker
UIoLI	[ɪ-as-ɪʃɔ-rɛ]	[+ATR] vowel surfaces as [−ATR]
Sour Grapes	[ɪ-as-ɪʃɔ-re]	Nothing Happens

Proposal

My proposal is that if we reconceptualize the notion of blockers and triggers as domain edges carving a subdomain within which phonological changes occur, we can eventually arrive at a formal classification of these patterns that unifies their similarities into something more than a descriptive category or a coincidence.

Proposal

Part of my ongoing work, then, is to see and show how that conceptualization is met and represented in different frameworks

I do it here in LP

Representation

In LP, a domain would be defined and carved both by the Search (identifying the edges) and Change (carving out the application space) rule.

In this context, the typology falls out from three parameters:

Representation

In this context, the typology falls out from three parameters:

1. Is there a left *and* right SEARCH required (contradirectional)?

Pattern	Contradirectional
Full Harmony	No
Partial Harmony	No
Icy Targets	No
UIoLI	Yes
SG	Yes

Representation

In this context, the typology falls out from three parameters:

2. Is SUBTRACTION required?

Pattern	Contradirectional	Subtraction
Full Harmony	No	No
Partial Harmony	No	No
Icy Targets	No	Yes
UIoLI	Yes	Yes
SG	Yes	No

Representation

In this context, the typology falls out from three parameters:

3. Does the order of the contradirectional function matters?

Pattern	Contradirectional	Subtraction	Ordered
Full Harmony	No	No	—
Partial Harmony	No	No	—
Icy Targets	No	Yes	—
UIoLI	Yes	Yes	No
SG	Yes	No	Yes

Representation

Full harmony and partial harmony are redundant.

This assumes that the same SEARCH and CHANGE parameters should be able to yield both.

Partial/Full harmony:

SEARCH

INR $\supseteq$ [+SYL]
DIR : Right
TRM $\supseteq$ [αATR]

CHANGE

INR $\sqcup$ {αATR}

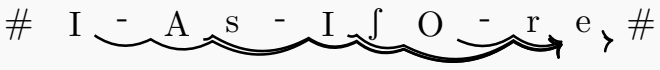
Representation

Full Harmony

a. Input

I - A s - I ʃ O - r e

b. Search



c. Unify

d. Output

i - ə s - i ʃ o - r e

Partial Harmony

a. Input

I - a s - I ʃ O - r e

b. Search

c. Unify

d. Output

ɪ - a s - i ʃ o - r e

The only difference is in the specification of the low vowel.

In the Full Harmonty pattern, /A/ is underspecified for [Atr], and receives its specification from the subsequent unification process

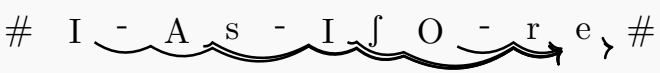
Representation

Full Harmony

a. Input

I - A s - I ʃ O - r e

b. Search



c. Unify

d. Output

i - ə s - i ʃ o - r e

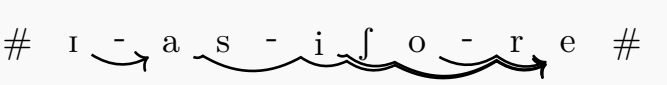
Partial Harmony

a. Input

I - a s - I ʃ O - r e

b. Search

c. Unify



d. Output

ɪ - a s - i ʃ o - r e

In the Partial Harmony process, /a/ is *already* specified for [−Atr], becomes a TRM and is also incapable of undergoing a Change

Representation : Icy Targets

If you add subtraction to the Change operation, you become capable of modifying that /a/ vowel, yielding the icy target effect

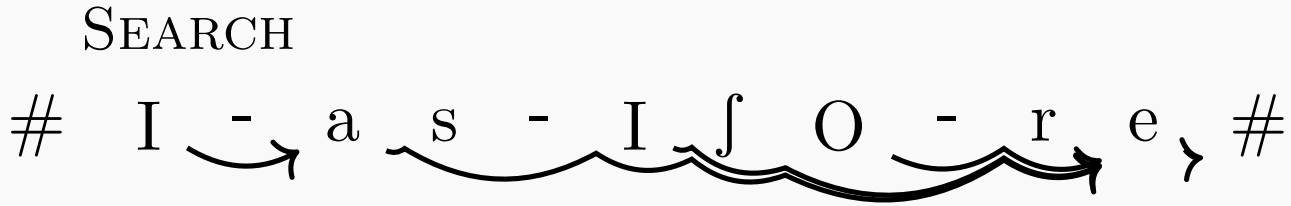
Icy Target

- SEARCH
  - INR $\supseteq$ [+Syl]
  - DIR : Right
  - TRM $\supseteq$ [αATR]
- Change
  - INR $\setminus$ {$-$Atr}
  - INR $\sqcup$ {αATR}

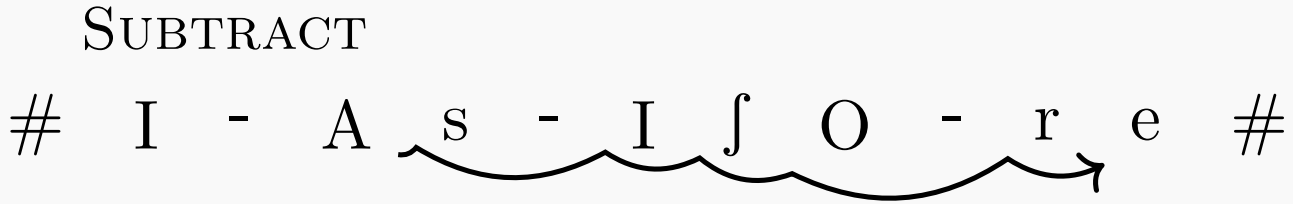
Input

I - a s - I ∫ O - r e

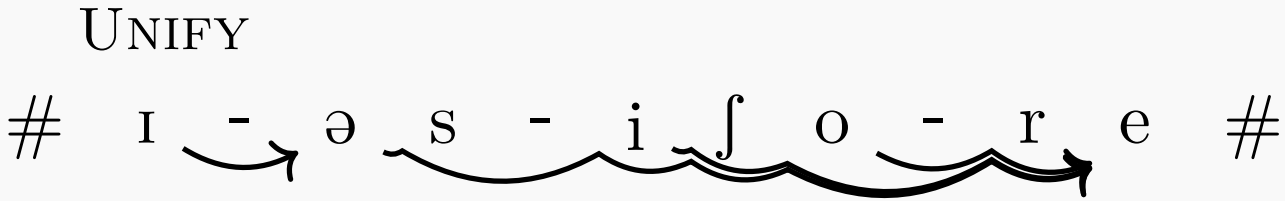
Representation: Icy Targets



Representation: Icy Targets



Representation: Icy Targets



Representation: Icy Targets

Output

ɪ - ə s - i ʃ o - r e

Representation

The distinction, then, between full harmony, partial harmony and icy target is the nature of the underlying representation combined with what the process targets

* In partial harmony, the presence of another fully-specified segment creates a secondary edge that can both initiate and terminate a SEARCH function, but cannot undergo a change of feature due to the lack of SUBTRACTION operation.

Representation

The distinction, then, between full harmony, partial harmony and icy target is the nature of the underlying representation combined with what the process targets

* In Icy Targets, the addition of a SUBTRACTION allows that /a/ to terminate a SEARCH function while undergoing a change of feature.

Representation: Contradirectionality

Once again, expanding from this initial rule, we can yield use it or lose it *and* Sour Grapes by adding a second SEARCH and CHANGE operation proceeding in the opposite direction

Representation: Use It or Lose It

Use It or Lose It

Rule 1: Partial/Full harmony:

INR $\supseteq$ [+Syl]
DIR : Right
TRM $\supseteq$ [αATR]

Change

INR $\sqcup$ {αATR}

Representation: Use It or Lose It

Rule 2: Subtraction and Unification rule:

SEARCH

INR $\supseteq$ [+Syl]
DIR : Left
TRM $\supseteq$ [$-$ATR]

CHANGE

SUBTRACTION : INR $\setminus$ {+ATR}
UNIFY : INR $\sqcup$ {$-$ATR}

Representation: Use It or Lose It

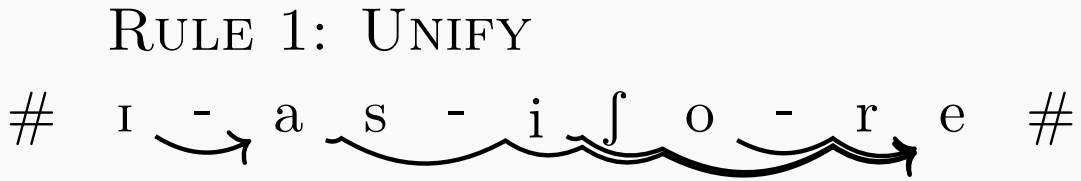
Rule 1 : Input

I - a s - I ʃ O - r e

Representation: Use It or Lose It

Rule 1: Search

Representation: Use It or Lose It



Representation: Use It or Lose It

Rule 1: Output & Rule 2: Input

ɪ - a s - i ʃ o - r e

Rule 2: Search

Representation: Use It or Lose It

Rule 2 : Subtraction

Representation: Use It or Lose It

Rule 2 : Unify

Representation: Use It or Lose It

Rule 2 : Output

ɪ - a s - ɪ ʃ ɔ - r ɛ

Representation: Sour Grapes

Sour Grapes

Rule 1

SEARCH

INR $\supseteq$ [+Syl]

DIR : Left

TRM $\supseteq$ [αATR]

CHANGE

INR $\sqcup$ {αATR}

Representation: Sour Grapes

Rule 2

SEARCH

$\textsc{Inr} \supseteq [+\textsc{Syl}]$

DIR : Right

TRM : $\supseteq [\alpha\text{ATR}]$

Change

INR $\sqcup \{\alpha\text{ATR}\}$

Representation: Sour Grapes

Rule 1: Input

I - a s - I ʃ O - r e

Representation: Sour Grapes

Rule 2: Search

Representation: Sour Grapes

Rule 3 Unify

Representation: Sour Grapes

	Rule 1: Output	&	Rule 2: Input								
#	I	-	a	s	-	ɪ	ʃ	ɔ	-	r	e	#

Representation: Sour Grapes

Rule 2: Search

Representation: Sour Grapes

Rule 2 : Unify

Representation: Sour Grapes

Rule 2 : Output

ɪ - a s - ɪ ʃ ɔ - r e

Conclusion

Pattern	contradirectional	Subtraction	Ordered	/ɪ-as-ɪʃɔ-re/
Partial Harmony	No	No	—	[ɪ-as-iʃo-re]
Icy Targets	No	Yes	—	[ɪ-əs-iʃo-re]
UIoLI	Yes	Yes	No	[ɪ-as-ɪʃɔ-rɛ]
SG	Yes	No	Yes	[ɪ-as-ɪʃɔ-re]

Conclusion

At the surface, they look like different phenomena

Pattern	contradirectional	Subtraction	Ordered	/I-as-IʃO-re/
Partial Harmony	No	No	—	[ɪ-as-iʃo-re]
Icy Targets	No	Yes	—	[ɪ-əs-iʃo-re]
UIoLI	Yes	Yes	No	[ɪ-as-ɪʃɔ-rɛ]
SG	Yes	No	Yes	[ɪ-as-ɪʃɔ-re]

Conclusion

In all four patterns, /a/ and /e/ are both carving the same subodmain. The difference is what the grammar does with the material inside that subdomain and at its edges

Pattern	contradirectional	Subtraction	Ordered	/I-as-IʃO-re/
Partial Harmony	No	No	—	[ɪ-as-iʃo-re]
Icy Targets	No	Yes	—	[ɪ-əs-iʃo-re]
UIoLI	Yes	Yes	No	[ɪ-as-ɪʃɔ-rɛ]
SG	Yes	No	Yes	[ɪ-as-ɪʃɔ-re]

Proposal

And from there, those four patterns (minus Full Harmony) emerge from how the process interacts with that structure

Conclusion

If we focused on the identity of the segments as "blockers", "triggers" and "targets", it forces us to treat them as independent things.

Conclusion

It also requires us to explain and describe behaviors that don't fit neatly in those categories, and come up with more labels to describe their behavior appropriately.

Conclusion

But if we also start asking "what does this segment *do* in this particular configuration", then we're able to understand why they can pair together and why they involve the same structure.

Conclusion

While in LP, the answer is situated within the way you can modify your Search (which determines the edges of your domain) and Change (which determines the application space) component by varying three parameters, this conceptualization is, ultimately, framework-agnostic.

Conclusion

Future Work:

I've worked backward from patterns I knew to determine the parameters, but if I combine parameters differently, what is going to come out of that? Does it generate an attested pattern? an unattested one? a pathological one?

Conclusion

Future Work:

Implement this analysis in other frameworks: how does this behave in OT, for example, or BMRS? Can I yield the same logic? Is there a genuine difference in what we consider a "blocker" or a "trigger" that is masked by the way LP works that can be revealed in other frameworks; and what would that mean?

Conclusion

Thank you.