Compiling Search & Change Rules into Subsequential Finite State Transducers

Malek Azadegan

Workshop on Logical Phonology
CUNY Graduate Center
Spring 2026

Outline

Valued Features

Assumptions (Dabbous et al., 2025)

- Binary feature values: $\mathcal{W} = \{+, -\}$
- Finite feature set: $\mathcal{F} = \{\textsc{Voice}, \textsc{Round}, \ldots\}$

Definition

Valued Feature. A valued feature is an element of $\mathcal{W} \times \mathcal{F}$, e.g. $+\textsc{Voice}$, $-\textsc{Round}$, . . .

Segments

Definition

Segment. A segment is a set of valued features (i.e., elements of $\mathcal{W} \times \mathcal{F}$) that is *consistent*, meaning that both valued-features aren't present in the set (Bale et al., 2014).

Examples:

$$
\begin{aligned}
/\text{u}/: &\quad \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}, +\textsc{Back}, +\textsc{Round}\} \\
/\text{i}/: &\quad \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}, -\textsc{Back}, -\textsc{Round}\} \\
/\text{b}/: &\quad \{-\textsc{Sonorant}, +\textsc{Labial}, -\textsc{Continuant}, +\textsc{Voice}\} \\
/\text{p}/: &\quad \{-\textsc{Sonorant}, +\textsc{Labial}, -\textsc{Continuant}, -\textsc{Voice}\}
\end{aligned}
$$

Underspecified segments are licit in LP, denoted using capital letters:

$$/\text{U}/: \quad \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}\}$$

Natural Classes

Definition

Natural Class. A set of segments that share certain features (Bale and Reiss, 2015).

Take /u/ and /i/:

$$/u/: \quad \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}, +\textsc{Back}, +\textsc{Round}\}$$
$$/i/: \quad \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}, -\textsc{Back}, -\textsc{Round}\}$$

Shared feature set can be computed by $/u/ \cap /i/$ (Gorman and Reiss, 2025). A natural class containing these is:

$$\{x \mid x \supseteq \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}\}\}$$

Using square brackets as shorthand for natural classes (Bale and Reiss, 2015), this can be written more compactly as:

$$[+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}] = \{x \mid x \supseteq \{+\textsc{Syllabic}, +\textsc{High}, -\textsc{Low}\}\}$$

Words

Definition

Word. Words are finite ordered sequences (strings) of segments.

The word cat → /kæt/ → $\langle \ltimes, \text{k}, \text{æ}, \text{t}, \rtimes \rangle$:

$$\left\langle \ltimes, \begin{Bmatrix} -\textsc{Syllabic} \\ -\textsc{Coronal} \\ +\textsc{High} \\ -\textsc{Voice} \\ -\textsc{Continuant} \\ \dots \end{Bmatrix}, \begin{Bmatrix} +\textsc{Syllabic} \\ -\textsc{Round} \\ +\textsc{Tense} \\ +\textsc{Voice} \\ +\text{ATR} \\ \dots \end{Bmatrix}, \begin{Bmatrix} -\textsc{Syllabic} \\ +\textsc{Coronal} \\ +\textsc{Anterior} \\ -\textsc{Voice} \\ -\textsc{Continuant} \\ \dots \end{Bmatrix}, \rtimes \right\rangle$$

Segment Modification

Segments are sets of valued features, so modification is done with set-theoretic operations.

Set subtraction removes features:

$$+\textsc{Voice} \notin (/\mathrm{b}/ \setminus \{+\textsc{Voice}\})$$

Naïve union can create inconsistent segments:

$$/\mathrm{b}/ \cup \{-\textsc{Voice}\} \Rightarrow \{\ldots, +\textsc{Voice}, -\textsc{Voice}, \ldots\}$$

To avoid this, use **set unification** (Bale et al., 2014; Gorman and Reiss, 2025):

$$A \sqcup B = \{c\mathrm{F} \mid c\mathrm{F} \in B \wedge -c\mathrm{F} \notin A\}$$

$$(/\mathrm{b}/ \setminus \{+\textsc{Voice}\}) \sqcup \{-\textsc{Voice}\} = /\mathrm{p}/$$

What is Search & Change?

Definition

Search & Change (S&C). “a model of phonological computation based on rules characterized by a small set of parameters” (Dabbous et al., 2021).

A S&C rule is a 5-tuple $\langle \textsc{Inr}, \textsc{Trm}, \textsc{Out}, \textsc{Dir}, \textsc{Cnd} \rangle$:

- $\textsc{Inr} \subseteq \Sigma$ (Search initiating natural class)
- $\textsc{Trm} \subseteq \Sigma$ (Search terminating natural class)
- $\textsc{Out} \in S \to S$ (Output function)
- $\textsc{Dir} \in \{\mathrm{L}, \mathrm{R}\}$ (Search direction)
- $\textsc{Cnd} \subseteq \Sigma$ (Change licensing natural class)

INR and TRM

Segments in INR **initiate** searches and are candidates for change. Segments in TRM are those that **terminate** searches. These are defined using square bracket notation:

1. [−SYLLABIC, +VOICE] ⟨⋉, ɪ, m, p, ɑ, s, ə, b, ə, l, ⋊⟩
2. [−SYLLABIC] ⟨⋉, ɪ, m, p, ɑ, s, ə, b, ə, l, ⋊⟩
3. [ ] ⟨⋉, ɪ, m, p, ɑ, s, ə, b, ə, l, ⋊⟩

Universal Natural Class (Bale and Reiss, 2015)

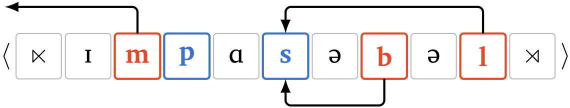
The natural class that contains all segments: $[\ \ ] = \{x \mid x \supseteq \emptyset\}$

DIR

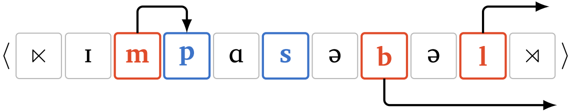
DIR specifies the search direction taken by segments in INR.

INR = [−SYLLABIC, +VOICE]   TRM = [−SYLLABIC, −VOICE]

- DIR = L



- DIR = R



Searches end at the nearest segment in TRM!

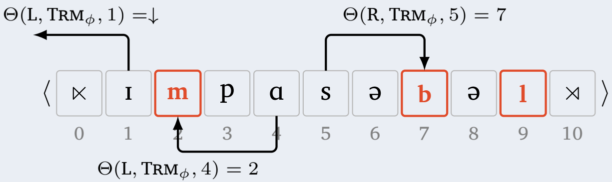
Search Definition

The *search* portion of S&C can be defined as follows:

$$\Theta(\textsc{Dir}, \textsc{Trm}, i) = \begin{cases} \min\{j > i \mid s_j \in \textsc{Trm}\} & \text{if } \textsc{Dir} = R \wedge \exists j > i\,(s_j \in \textsc{Trm}) \\ \max\{j < i \mid s_j \in \textsc{Trm}\} & \text{if } \textsc{Dir} = L \wedge \exists j < i\,(s_j \in \textsc{Trm}) \\ \downarrow & \text{otherwise} \end{cases}$$

Example

If $\textsc{Trm} = [-\textsc{Syllabic}, +\textsc{Voice}]$, then:



OUT

Definition

OUT is a function of type $S \to S$ that maps a segment to a (possibly) modified segment.

Consider the following segments:

$$/\mathrm{b}/ = \{-\text{SYLLABIC},\ -\text{SONORANT},\ +\text{LABIAL},\ +\text{VOICE},\ \ldots\}$$

$$/\mathrm{U}/ = \{+\text{SYLLABIC},\ +\text{SONORANT},\ +\text{VOICE},\ \ldots\} \qquad (\text{i.e., } /\mathrm{u}/ \cap /\mathrm{i}/)$$

- *Devoicing*: $\text{OUT} = \lambda s \in S.\ \big(s \setminus \{+\text{VOICE}\}\big) \sqcup \{-\text{VOICE}\}$

$$\text{OUT}(\mathrm{b}) = \big(\mathrm{b} \setminus \{+\text{VOICE}\}\big) \sqcup \{-\text{VOICE}\} = \mathrm{p}$$

- *Feature-filling*: $\text{OUT} = \lambda s \in S.\ \big(s \sqcup \{+\text{ROUND}, +\text{BACK}\}\big)$

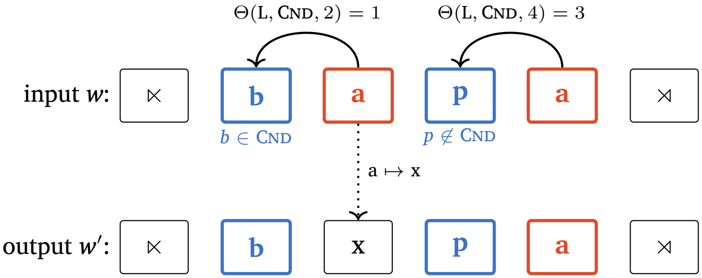
$$\text{OUT}(\mathrm{U}) = \big(\mathrm{U} \sqcup \{+\text{ROUND}, +\text{BACK}\}\big) = \mathrm{u}$$

CND

Definition

CND is a natural class of segments that license transformations.

INR = [+SYLLABIC] TRM = [−SYLLABIC] CND = [+VOICE] DIR = L



Full Specification

Given a $\textsc{Rule} = \langle \textsc{Inr}, \textsc{Trm}, \textsc{Dir}, \textsc{Out}, \textsc{Cnd} \rangle$, the S&C specification is:

S&C Specification

$$s'_i(\textsc{Rule}) = \begin{cases} \textsc{Out}(s_i) & \text{if } s_i \in \textsc{Inr} \wedge \Theta(\textsc{Dir}, \textsc{Trm}, i) \neq \downarrow \wedge s_{\Theta(\textsc{Dir}, \textsc{Trm}, i)} \in \textsc{Cnd} \\ s_i & \text{otherwise} \end{cases}$$

For any word w, the transduced word w' can be found by:

$$w' = \langle \ltimes_0, s'_1, s'_2, \ldots, s'_{n-2}, \rtimes_{n-1} \rangle$$

α-notation

Aside. The S&C spec can be stated with α-notation (using $\textsc{Out}$ of type $S \times S \to S$); I stick to the basic format here.

English Plural Marker

We can model the English plural marker using two S&C rules, restricting our attention to the [z] $\sim$ [s] alternation. Consider:

- $/\mathrm{z}/ = \{-\textsc{Labial}, +\textsc{Continuant}, +\textsc{Strident}, +\textsc{Voice}\}$
- $/\mathrm{s}/ = \{-\textsc{Labial}, +\textsc{Continuant}, +\textsc{Strident}, -\textsc{Voice}\}$
- $/\mathrm{S}/ = /\mathrm{z}/ \cap /\mathrm{s}/ = \{-\textsc{Labial}, +\textsc{Continuant}, +\textsc{Strident}\}$

Both rules should find the first segment to the left of /S/:

$$\textsc{Inr}_\alpha = [-\textsc{Labial}, +\textsc{Continuant}, +\textsc{Strident}] \quad \textsc{Trm}_\alpha = [\ ] \quad \textsc{Dir} = \mathrm{L}$$

One applies when TRM is +VOICE and the other when it is −VOICE:

$$\textsc{Cnd}_\psi = [+\textsc{Voice}] \qquad \textsc{Cnd}_\phi = [-\textsc{Voice}]$$

One rule feature-fills +VOICE and the other −VOICE:

$$\textsc{Out}_\psi = \lambda s \in S.(s \sqcup \{+\textsc{Voice}\}) \qquad \textsc{Out}_\phi = \lambda s \in S.(s \sqcup \{-\textsc{Voice}\})$$

English Plural Marker

Rules:

1. $\langle \text{INR}_\alpha, \text{TRM}_\alpha, \text{OUT}_\psi, \text{DIR} = \text{L}, \text{CND}_\psi \rangle$ (Feature-fill +VOICE)
2. $\langle \text{INR}_\alpha, \text{TRM}_\alpha, \text{OUT}_\phi, \text{DIR} = \text{L}, \text{CND}_\phi \rangle$ (Feature-fill −VOICE)

Consider the words /dɑg/, /tæb/, /kæt/, and /pæk/:

- $\langle \ltimes, \text{d}, \text{ɑ}, \text{g}, \text{S}, \rtimes \rangle \rightarrow \langle \ltimes, \text{d}, \text{ɑ}, \text{g}, \text{z}, \rtimes \rangle$ (By rule 1, as g $\in$ [+VOICE])
- $\langle \ltimes, \text{t}, \text{æ}, \text{b}, \text{S}, \rtimes \rangle \rightarrow \langle \ltimes, \text{t}, \text{æ}, \text{b}, \text{z}, \rtimes \rangle$ (By rule 1, as b $\in$ [+VOICE])
- $\langle \ltimes, \text{k}, \text{æ}, \text{t}, \text{S}, \rtimes \rangle \rightarrow \langle \ltimes, \text{k}, \text{æ}, \text{t}, \text{s}, \rtimes \rangle$ (By rule 2, as t $\in$ [−VOICE])
- $\langle \ltimes, \text{p}, \text{æ}, \text{k}, \text{S}, \rtimes \rangle \rightarrow \langle \ltimes, \text{p}, \text{æ}, \text{k}, \text{s}, \rtimes \rangle$ (By rule 2, as k $\in$ [−VOICE])

Why restrict to [z] $\sim$ [s] alternation?

S&C operates over a fixed sequence of indexed segments. Insertion creates a new index and deletion removes one, shifting all later indices and disrupting the Θ-based search, which assumes a stable domain of segment indices.

Votic Vowel Harmony

An analysis of vowel harmony in Votic is presented by Leduc et al. (2020). Suffix vowels present the following alternations depending on the backness of the nearest preceding vowel that is not /i/.

[y]∼[u] [e]∼[ə] [æ]∼[ɑ]

Root vowels → fully specified for BACK

/U/, /E/, and /A/ (one per alternation) → underspecified for BACK.

Votic Vowel Inventory Feature Table

	i	y	ɨ	u	e	ø	ə	o	æ	ɑ	U	E	A
HIGH	+	+	+	+	–	–	–	–	–	–	+	–	–
LOW	–	–	–	–	–	–	–	–	+	+	–	–	+
BACK	–	–	+	+	–	–	+	+	–	+			
ROUND	–	+	–	+	–	+	–	+	–	–	+	–	–

Natural Class Challenges

"Any vowel except /i/" does not form a natural class: there is no feature specification that is shared by all vowels except for /i/. To mitigate this issue, Leduc et al. (2020) propose two harmony cases:

- Non-high vowel harmony
- High-round vowel harmony

Each case harmonizes the underspecified suffix vowel with the nearest licensing vowel to its left.

S&C Formulation of Votic Vowel Harmony

All rules initiate leftward searches from vowels:

$$\textsc{Inr}_{syllabic} = [+\textsc{Syllabic}] \quad \textsc{Dir} = \text{L}$$

Two TRM definitions are needed:

- $\textsc{Trm}_{non\text{-}high} = [+\textsc{Syllabic}, -\textsc{High}]$
- $\textsc{Trm}_{high\text{-}round} = [+\textsc{Syllabic}, +\textsc{High}, +\textsc{Round}]$

Two OUT definitions are needed:

- $\textsc{Out}_{\psi} = \lambda s \in S.(s \sqcup \{+\textsc{Back}\})$ Fill $+$BACK
- $\textsc{Out}_{\phi} = \lambda s \in S.(s \sqcup \{-\textsc{Back}\})$ Fill $-$BACK

Two CND definitions are needed:

- $\textsc{Cnd}_{\psi} = [+\textsc{Back}]$
- $\textsc{Cnd}_{\phi} = [-\textsc{Back}]$

Votic Vowel Harmony S&C Ruleset

The S&C ruleset for Votic vowel harmony:

1. $\langle \textsc{Inr}_{syllabic}, \textsc{Trm}_{non\text{-}high}, \textsc{Out}_{\psi}, \text{L}, \textsc{Cnd}_{\psi} \rangle$ — *Non-high harmonize* +BACK
2. $\langle \textsc{Inr}_{syllabic}, \textsc{Trm}_{non\text{-}high}, \textsc{Out}_{\phi}, \text{L}, \textsc{Cnd}_{\phi} \rangle$ — *Non-high harmonize* −BACK
3. $\langle \textsc{Inr}_{syllabic}, \textsc{Trm}_{high\text{-}round}, \textsc{Out}_{\psi}, \text{L}, \textsc{Cnd}_{\psi} \rangle$ — *High-round harmonize* +BACK
4. $\langle \textsc{Inr}_{syllabic}, \textsc{Trm}_{high\text{-}round}, \textsc{Out}_{\phi}, \text{L}, \textsc{Cnd}_{\phi} \rangle$ — *High-round harmonize* −BACK
5. $\langle \textsc{Inr}_{syllabic}, [\,], \textsc{Out}_{\phi}, \text{L}, [\,] \rangle$ — *Default feature-fill* −BACK

Examples:

- /vəttimis-E/ → [vəttimisə] (By rule 1)
- /pehmi-sE/ → [pehmise] (By rule 2)
- /sili-A/ → [siliæ] (By rule 5)
- /tyttærikko-A/ → [tyttærikkoɑ] (By rule 1, bleeds rule 3)

Execution Strategy

Naive Implementation (Nested Loops)

For every segment s_i:

1. Scan the string to find the terminator $\Theta(\text{DIR}, \text{TRM}, i)$.
2. Check licensing and apply change.

Cost: n segments $\times$ $O(n)$ search $=$ $O(n^2)$ **(Quadratic)**

$\Downarrow$

Reverse Scan Method (Two Pointers)

Scan the string in the opposite direction given by DIR, maintaining a boolean flag F. For each segment s_i:

1. If $s_i \in \text{TRM}$, then assign $F \leftarrow s_i \in \text{CND}$.
2. If $s_i \in \text{INR} \wedge F = \text{TRUE}$, then assign $s_i' \leftarrow \text{OUT}(s_i)$, or s_i otherwise.

Cost: n segments visited once $=$ $O(n)$ **(Linear)**

Reverse Scan Algorithm

Input: $w = \langle \ltimes, s_0, \ldots, s_{n-1}, \rtimes \rangle$; $\mathrm{RULE} = \langle \mathrm{INR}, \mathrm{TRM}, \mathrm{OUT}, \mathrm{DIR} = \mathrm{R}, \mathrm{CND} \rangle$
Output: $w' = \langle \ltimes, s'_0, \ldots, s'_{n-1}, \rtimes \rangle$
for $i \leftarrow 0$ **to** $n - 1$ **do**
  $s'_i \leftarrow s_i$
end
$F \leftarrow$ **false**
for $i \leftarrow n - 1$ **to** 0 **do**
  if $s_i \in INR \wedge F$ **then**
    $s'_i \leftarrow \mathrm{OUT}(s_i)$
  end
  if $s_i \in TRM$ **then**
    $F \leftarrow (s_i \in \mathrm{CND})$
  end
end
return w'

One right-to-left pass; each segment is inspected a constant number of times: overall time $\mathcal{O}(n)$.

Compiling S&C rules to FSTs

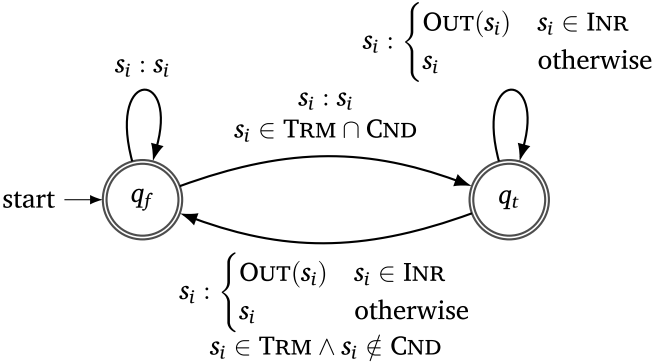
Key Observations:

- Σ is finite. Thus, INR, TRM, and CND (subsets of Σ) are finite.
- OUT only applies to segments in INR.
- DIR is either L or R.

Therefore, we can precompute every possible FST transition.

Linear Algorithm	Finite State Transducer
Current Segment (s_i)	Input Symbol
Algorithm ”Memory” (F)	Machine State (q)
OUT(s_i) calculation	Transition Output ($: y$)

FST Diagram


start
q_f
q_t
$s_i : s_i$
$s_i : \begin{cases} \textsc{Out}(s_i) & s_i \in \textsc{Inr} \\ s_i & \text{otherwise} \end{cases}$
$s_i : s_i$
$s_i \in \textsc{Trm} \cap \textsc{Cnd}$
$s_i : \begin{cases} \textsc{Out}(s_i) & s_i \in \textsc{Inr} \\ s_i & \text{otherwise} \end{cases}$
$s_i \in \textsc{Trm} \wedge s_i \notin \textsc{Cnd}$

Learning Algorithms

S&C-derived FSTs are **sequential** transducers in the sense of Mohri (1997):

1. They are deterministic on their input.
2. They define a function from input strings to output strings.
3. They do not use final output.

Mohri (1997) then defines **subsequential** transducers as those that are sequential and (possibly) generate an additional output string at final states. Thus, since S&C-derived FSTs are **sequential**, they are also **subsequential**.

Subsequential FSTs are "learnable" by algorithms like OSTIA(Heinz, 2018; Oncina et al., 1993).

Takeaway

Any phonological process expressible using S&C:

- Compiles into an **efficient**, deterministic finite-state transducer
- Lies within the **subsequential** class—which is *identifiable from positive data* (e.g. by OSTIA).

References:

Bale, A., Papillon, M., and Reiss, C. (2014). Bale, Alan, Maxime Papillon, and Charles Reiss. 2014. *Targeting underspecified segments: A formal analysis of feature-changing and feature-filling rules. Lingua* 148: 240–253.

Bale, A. and Reiss, C. (2015). *Phonology: A formal introduction*. The MIT Press.

Dabbous, R., Gorman, K., and Reiss, C. (2025). Tutorial on substance-free logical phonology. LingBuzz 008746. Handout from an invited tutorial at the Linguistic Society of America Annual Meeting, Philadelphia, PA.

Dabbous, R., Leduc, M., Mousavi, F., Reiss, C., and Shen, D. T.-C. (2021). Satisfying long-distance relationships (without tiers): A strictly anti-local approach to phonology. *LingBuzz*. lingbuzz/006329.

Gorman, K. and Reiss, C. (2025). Metaphony in substance-free logical phonology. *To appear in a special issue of Phonology*. Reference: lingbuzz/008634.

Heinz, J. (2018). The computational nature of phonological generalizations. In Hyman, L. and Plank, F., editors, *Phonological Typology*, Phonetics and Phonology, chapter 5, pages 126–195. De Gruyter Mouton.

Leduc, M., Reiss, C., and Volenec, V. (2020). Votic vowel harmony in substance free logical phonology. In *The Oxford Handbook of Vowel Harmony*. Oxford University Press. Reference: lingbuzz/005339.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

Oncina, J., García, P., and Vidal, E. (1993). Learning subsequential transducers for pattern recognition interpretation tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15:448 – 458.

α-notation

Worked examples like Votic require multiple nearly identical rules (e.g. one for $+$Back, one for $-$Back, etc.). These can be collapsed using **α-notation**.

Idea: Let Out see both s_i and the segment that licensed the change.

$$s_i'(\text{RULE}) = \begin{cases} \textsc{Out}(s_i,\, s_{\Theta(\textsc{Dir},\textsc{Trm},i)}) & \text{if } s_i \in \textsc{Inr} \wedge \Theta(\textsc{Dir}, \textsc{Trm}, i) \downarrow \wedge s_{\Theta(\textsc{Dir},\textsc{Trm},i)} \in \textsc{Cnd}, \\ s_i & \text{otherwise.} \end{cases}$$

Execution

Instead of a boolean flag, store the index of the last segment that raised the flag and reference it when passing s_i to Out. When the flag would be lowered, set the index to -1.

Result: still $\mathcal{O}(n)$, gives Out reference to $s_{\Theta(\textsc{Dir},\textsc{Trm},i)}$.

α-notation FST