Carnegie Mellon University
Language
Technologies
Institute

11-324/11-624/11-724 Human Language for AI

Finite State Phonology

David R. Mortensen

September 26, 2022

Language Technologies Institute
Carnegie Mellon University

Learning Objectives

During this class period, each student will

- Learn what a finite-state transducer is
- Learn how FSTs apply to phonology

Morphophonological Analysis: An Example Problem

Catalan Example I

MASC SG	FEM SG		MASC SG	FEM SG	
əkelʲ	əkelʲə	‘that’	mal	malə	‘bad’
siβil	siβilə	‘civil’	əskerp	əskerpə	‘shy’
ʃop	ʃopə	‘drenched’	sɛk	sɛkə	‘dry’
əspɛs	əspɛsə	‘thick’	gros	grosə	‘large’
baʃ	baʃə	‘short’	koʃ	koʃə	‘lame’
tot	totə	‘all’	brut	brutə	‘dirty’
pɔk	pɔkə	‘little’	prəsis	prəsizə	‘precise’
frənses	frənsezə	‘French’	gris	grizə	‘grey’
kəzat	kəzaðə	‘married’	bwit	bwiðə	‘empty’
rɔʧ	rɔʒə	‘red’	boʧ	boʒə	‘crazy’
orp	orβə	‘blind’	lʲark	lʲarɣə	‘long’
sek	seɣə	‘blind’	fəʃuk	fəʃuɣə	‘heavy’
grok	groɣə	‘yellow’	puruk	puruɣə	‘fearful’
kandit	kandiðə	‘candid’	frɛt	frɛðə	‘cold’

Catalan Example II

MASC SG	FEM SG		MASC SG	FEM SG	
səɣu	səɣurə	‘sure’	du	durə	‘hard’
səɣəðo	səɣəðorə	‘reaper’	kla	klarə	‘clear’
nu	nuə	‘nude’	kru	kruə	‘raw’
flɔɲʤu	flɔɲʤə	‘soft’	dropu	dropə	‘lazy’
əgzaktə	əgzaktə	‘exact’	əlβi	əlβinə	‘albino’
sa	sanə	‘healthy’	pla	planə	‘level’
bo	bonə	‘good’	sərɛ	sərɛnə	‘calm’
suβlim	suβlimə	‘sublime’	al	altə	‘tall’
fɔr	fɔrtə	‘strong’	kur	kurtə	‘short’
sor	sorðə	‘deaf’	bɛr	bɛrðə	‘green’
san	santə	‘saint’	kəlɛn	kəlɛntə	‘hot’
prufun	prufundə	‘deep’	fəkun	fəkundə	‘fertile’
dəsen	dəsentə	‘decent’	dulen	dulentə	‘bad’
əstuðian	əstuðiantə	‘student’	blaŋ	blaŋkə	‘white’

Example Catalan Grammar

Final Devoicing

$$z \rightarrow s/\_\#$$
$$b \rightarrow p/\_\#$$
$$d \rightarrow t/\_\#$$
$$g \rightarrow k/\_\#$$
$$d\widehat{ʒ} \rightarrow t\widehat{ʃ}/\_\#$$

Spirantization

$$b \rightarrow \beta / \left\{ \begin{matrix} V \\ l \\ r \end{matrix} \right\} \_V$$
$$d \rightarrow ð / \left\{ \begin{matrix} V \\ l \\ r \end{matrix} \right\} \_V$$
$$g \rightarrow ɣ / \left\{ \begin{matrix} V \\ l \\ r \end{matrix} \right\} \_V$$
$$d\widehat{ʒ} \rightarrow ʒ$$

Hiatus Resolution

$$V \rightarrow \emptyset \ \_V$$

n-Apocope

$$n \rightarrow \emptyset \ \_\#$$

Plosive Apocope

$$\left\{ \begin{matrix} p \\ t \\ k \end{matrix} \right\} \rightarrow \emptyset \ \_\#$$

D-Deletion

$$D \rightarrow \emptyset$$

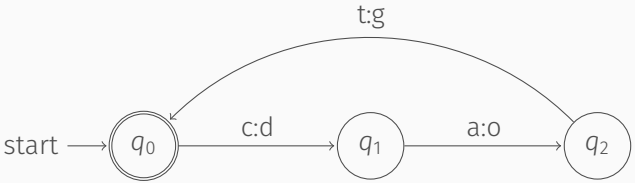
Finite-State Transducers

Formal definition

Formally, an FST is a 6-tuple $(Q, \Sigma, \Gamma, I, F, \delta)$ such that:

- Q is a finite set of states;
- Σ is a finite set of symbols, the *input alphabet*;
- Γ is a finite set of symbols, the *output alphabet*;
- I is a subset of Q, the set of *initial states*;
- F is a subset of Q, the set of *final states* or *acceptor states*;
- $\delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \times Q$ (where ε is the empty string), the *transition relation*.

A Simple FST



Operations

FSTs are closed under:

- Inversion
- Composition
- Concatenation
- Union
- Kleene Closure
- ...

FSTs are not closed under:

- Intersection
- Determination
- ...

Inversion

- Inversion reverses the “direction” of a relation.
- When inverted, a transducer that mapped “kiss^s” to “kisses” would map “kisses” to “kiss^s”
- Inversion is algorithmically simple: you just search through the graph and replace each label $a : b$ with $b : a$
- Inversion is useful because it allows you to write a transducer that maps from and underlying form to a surface form and invert it to get a transducer that maps from surface for to underlying form (no extra work required).
- Inversion can result in transducers that are NON-DETERMINISTIC

Composition

- Composition is the equivalent of joining two transducers together “vertically” so that the output of the first is the input to the second.
- Take two FSTs:
  - One that maps “cat” $\rightarrow$ “dog”
  - One that maps “dog” $\rightarrow$ “pig”
- The composition of the first and the second would be a transducer that maps “cat” $\rightarrow$ “pig”
- The algorithm for composition is quite complicated

Concatenation

- Concatenation is the equivalent of joining two transducers together “horizontally” so that the transducers apply in sequence.
- Take two FSTs:
  - One that maps “cat” $\rightarrow$ “dog”
  - One that maps “dog” $\rightarrow$ “pig”
- The concatenation of these two transducers would be one that maps “catdog” to “dogpig”

Determination

Determination takes a transducer that is non-deterministic (has more than one transition from the same state with the same input label) and yields an equivalent deterministic transducer (one that captures the same relation). This is only possible for some transducers.

FSTs and Phonology

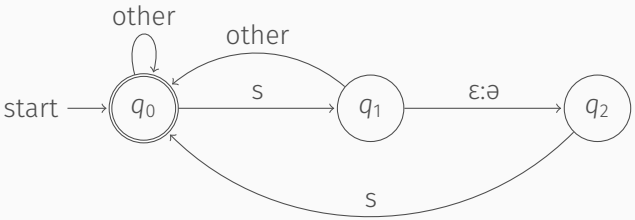
FSTs and phonological rules

In the 1970s, C. Douglas Johnson discovered that most phonological rules that had been proposed to analysis languages since the advent of Generative Phonology (phonology with the kind of string rewrite rules we have learned) were equivalent to a subset of Finite-State Transducers.

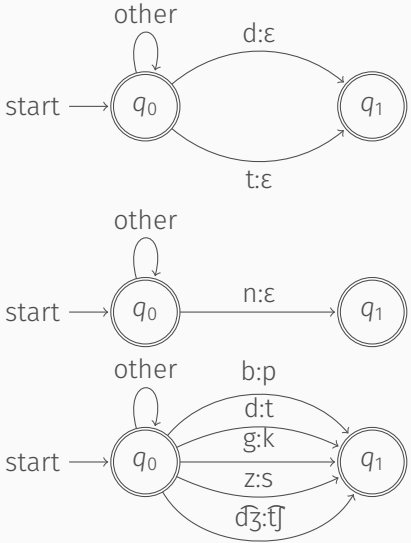
- The exceptions were rules that feed themselves or bleed themselves.
- [+syllabic] $\rightarrow$ [+high] / [+high] C_0 _
- These were used to model “tone spreading” and “vowel harmony”—processes that “spread” a feature out across a word.
- By the late 1970s, phonologists were thinking of other ways to address these phenomena.

Today it is argued, by many computational phonologists, that all of phonology is in fact SUBREGULAR, that is, a principled subset of possible FSTs can account for all of phonology.

Schwa Epenthesis FST

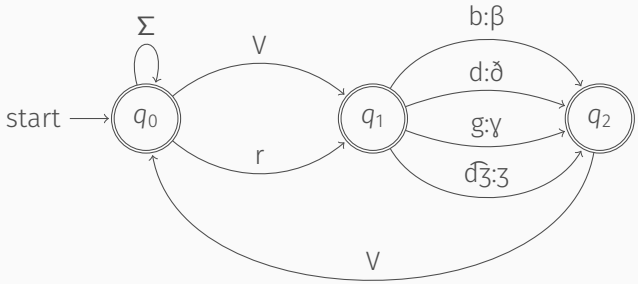


Three FSTs for Catalan



- What do each of these FSTs do?
- To what rules do they correspond?
- Assume, also, an FST that converts voiced plosives and affricates to fricatives in a specific set of contexts (between vowels+/r/ and vowels) as in /orbə/ → [orβə] and /sega/ → [seɣə]. How would you draw it?
- How would you combine these FSTs to model the data from the example?

Spirantization FST



Σ represents the input alphabet. V represents the set of vowels. What problem do you see in this FST?

Questions?