
MORPHOLOGY AND FINITE- STATE METHODS

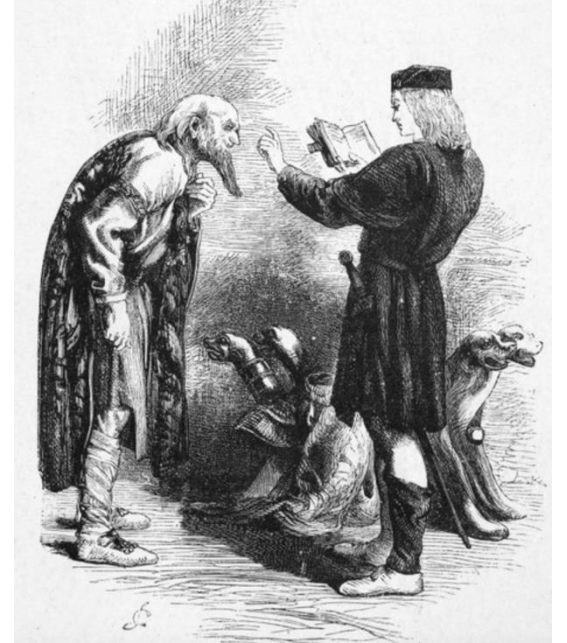
Mehmet Can Yavuz, PhD.

Adapted from Julia Hockenmaier, NLP S2023 - course material
<https://courses.grainger.illinois.edu/cs447/sp2023/>



KEY CONCEPTS

- Words
- Morphology
- Automata



Polonius: What do you read, my lord?
Hamlet: Words, words, words.
Act 2, scene 2, Hamlet

HOW MANY DIFFERENT WORDS ARE THERE IN ENGLISH?

How large is the **vocabulary** of English (or any other language)?

Vocabulary size = the number of distinct word types Google N-gram corpus: 1 trillion tokens,

13-million-word types that appear 40+ times

[here, we're treating inflected forms (took, taking) as distinct]

You may have heard statements such as

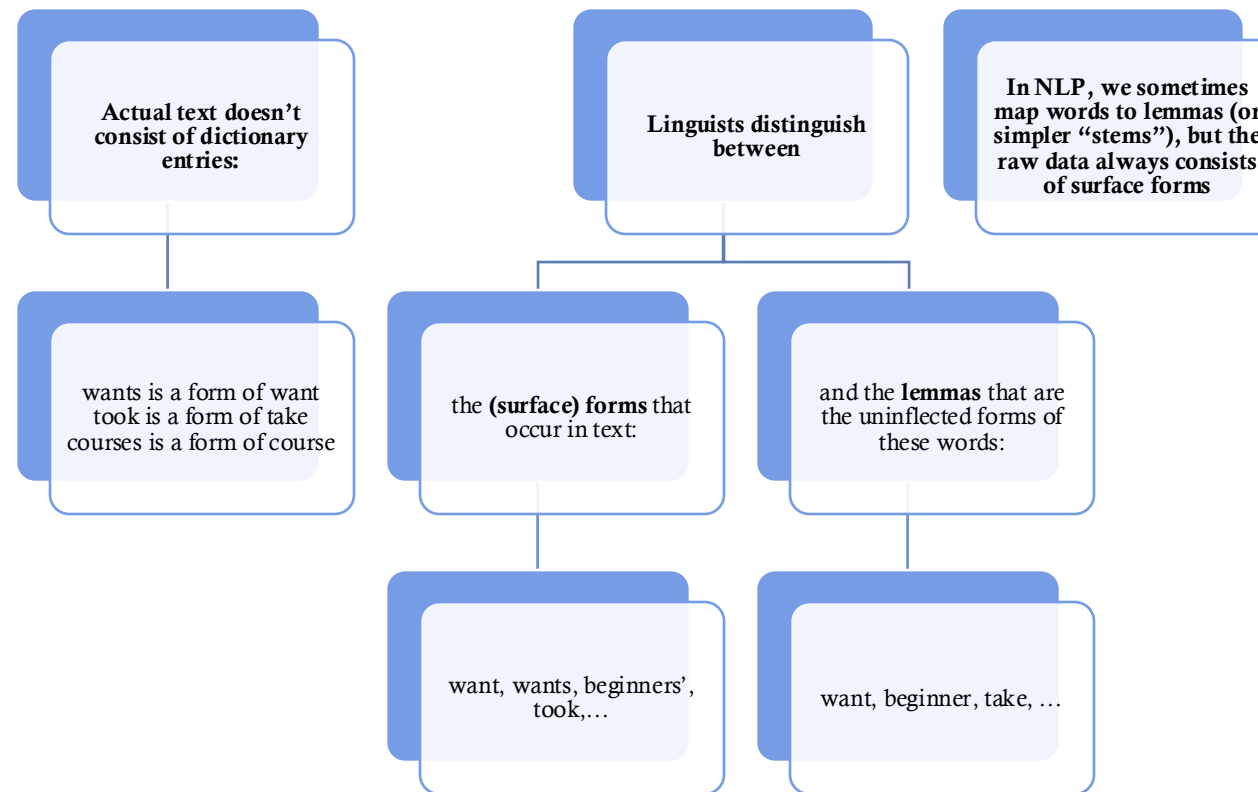
“adults know about 30,000 words”

“you need to know at least 5,000 words to be fluent”

Such statements do not refer to inflected word forms (take/takes/taking/take/takes/took) but to lemmas or dictionary forms (take) and assume if you know

a lemma, you know all its inflected forms too.

WHICH WORDS APPEAR IN THIS TEXT?



A TURKISH WORD

uygarlaştıramadıklarımızdanmışsınızcasına

uygar_laş_tır_ama_dık_lar_ımız_dan_mış_sınız_casına

“as if you are among those whom we were not able to civilize

(=cause to become civilized)” uygar: *civilized*

_laş: *become*

_tır: *cause somebody to do something*

_ama: *not able*

_dık: *past participle*

_lar: *plural*

_ımız: *1st person plural possessive (our)*

_dan: *among* (ablative case)

_mış: *past*

_sınız: *2nd person plural (you)*

_casına: *as if* (forms an adverb from a verb)

INFLECTIONAL MORPHOLOGY IN ENGLISH

Verbs:

- Infinitive/present tense: walk, go
- 3rd person singular present tense (s-form): walks, goes

Simple past: walked, went

- Past participle (ed-form): walked, gone
- Present participle (ing-form): walking, going

Nouns:

- Common nouns inflect for number:
 - singular (book) vs. plural (books)
- Personal pronouns inflect for person, number, gender, case:
 - I saw him; he saw me; you saw her; we saw them; they saw us.

DERIVATIONAL MORPHOLOGY IN ENGLISH

Nominalization:

V + -ation: computerization

V+ -er: killer

Adj + -ness: fuzziness

Negation:

un-: undo, unseen, ...

mis-: mistake,...

Adjectivization: V+ -able: doable
N + -al: national

MORPHEMES: STEMS, AFFIXES

dis-grace-ful-ly

- prefix-stem-suffix-suffix

Many word forms consist of a *stem* plus a number of *affixes* (*prefixes* or *suffixes*)

- Exceptions: *Infixes* are inserted inside the stem
 - *Circumfixes* (German *gesehen*) surround the stem
 - Morphemes: the smallest (meaningful/grammatical) parts of words.
 - *Stems* (grace) are often free morphemes.
 - Free morphemes can occur by themselves as words.
 - *Affixes* (dis-, -ful, -ly) are usually bound morphemes.
 - Bound morphemes *have* to combine with others to form words.
-

MORPHEMES AND MORPHS

The same information (plural, past tense, ...) is often expressed in different ways in the same language.

One way may be more common than others, and exceptions may depend on specific words:

Most plural nouns: add **-s** to singular:
book-books,

but: box-boxes, fly-flies,
child-children

Most past tense verbs add **-ed** to
infinitive: walk-walked,

but: like-liked, leap-leapt

Such exceptions are called *irregular*
word forms

Linguists say that there is one
underlying morpheme

(e.g. for plural nouns) that is
“realized” as different “surface”
forms (morphs) (e.g. -s/-es/-ren)

Allomorphs: two different realizations
(-s/-es/-ren) of the same underlying
morpheme (plural)

SIDE NOTE: “SURFACE”?

This terminology comes from Chomskyan Transformational Grammar.

- - Dominant early approach in theoretical linguistics, superseded by other approaches (“minimalism”).
- - Not computational, but has some historical influence on computational linguistics (e.g. Penn Treebank)

“Surface” = standard English (Chinese, Hindi, etc.).

“Surface string” = a written sequence of characters or words

vs. “Deep”/“Underlying”
structure/representation:

- A more abstract representation.
- Might be the same for different sentences/words with the same meaning.

FINITE-STATE

AUTOMATA AND REGULAR

LANGUAGES

FORMAL LANGUAGES

An alphabet Σ is a set of symbols:

- e.g. $\Sigma = \{a, b, c\}$

A string ω is a sequence of symbols, e.g. $\omega = abcb$.

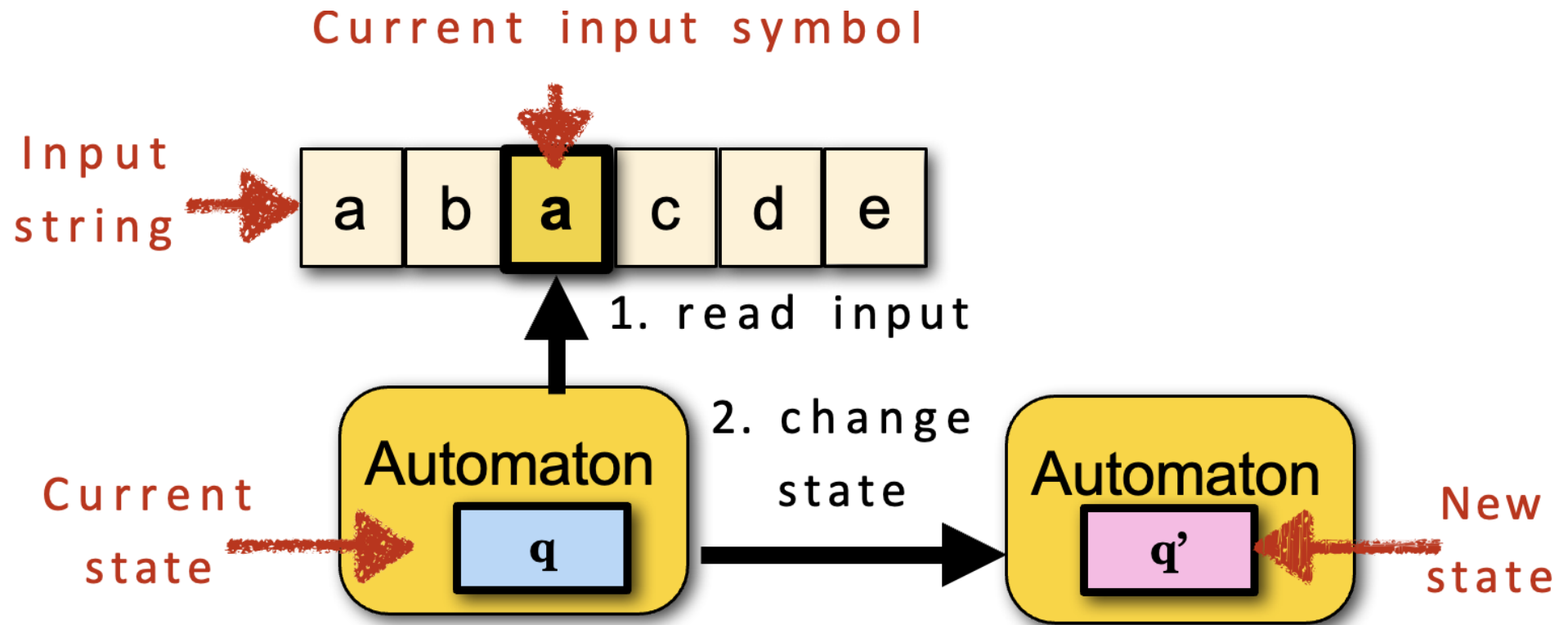
- The empty string ε consists of zero symbols.

The Kleene closure Σ^* ('sigma star') is the (infinite) set of all strings that can be formed from Σ :

- $\Sigma^* = \{\varepsilon, a, b, c, aa, ab, ba, aaa, \dots\}$

A language $L \subseteq \Sigma^*$ over Σ is also a set of strings.

- Typically, we only care about proper subsets of Σ^* ($L \subsetneq \Sigma^*$).
-

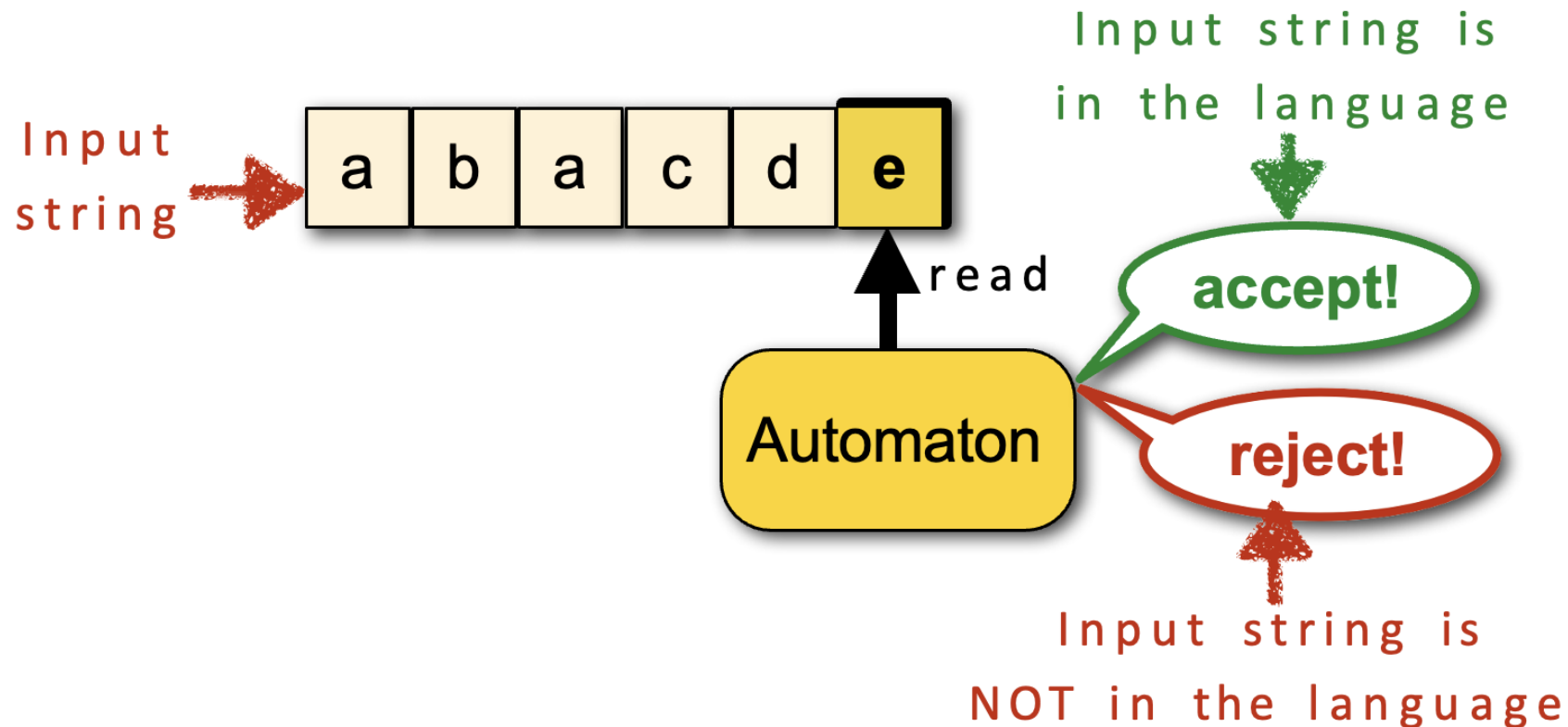


AUTOMATA AND LANGUAGES

- An automaton is an abstract model of a computer. It *reads* an input string symbol by symbol.
- It *changes* its internal state depending on the current input symbol and its current internal state.

AUTOMATA AND LANGUAGES

- The automaton either *accepts* or *rejects* the input string.
- Every automaton defines a language (= the set of strings it accepts).

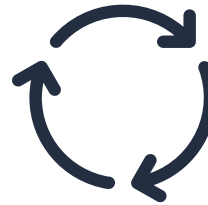


AUTOMATA AND LANGUAGES

Different types of automata define different language classes:



—**Finite-state** automata define **regular** languages

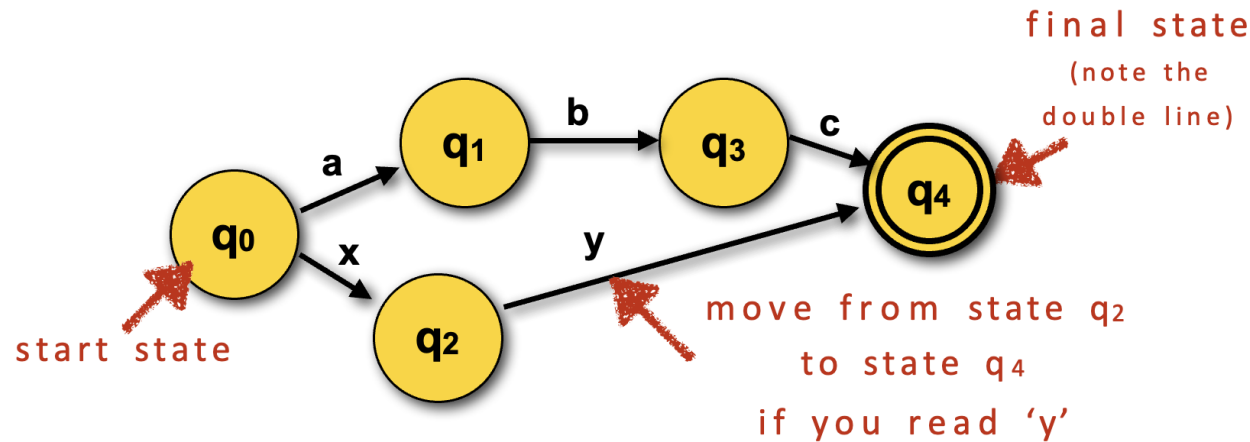


—**Pushdown** automata define **context-free** languages



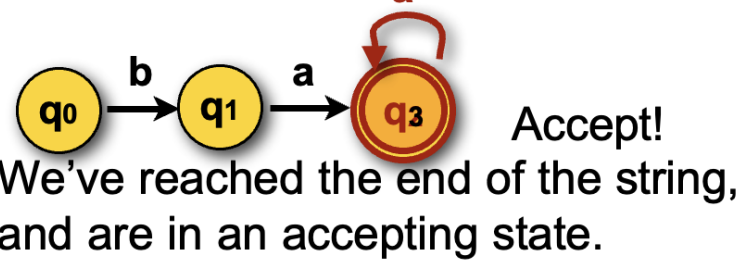
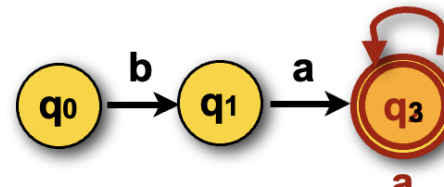
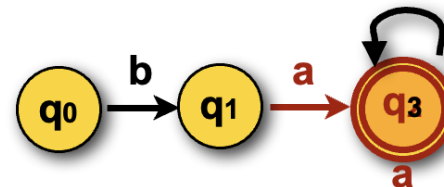
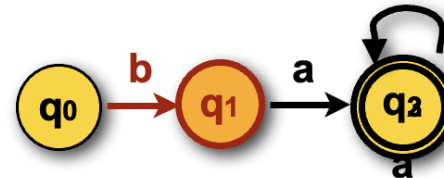
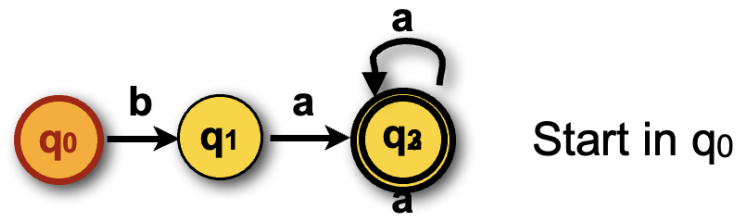
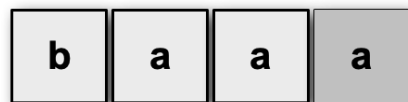
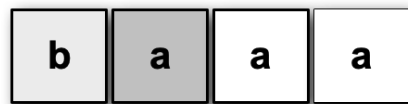
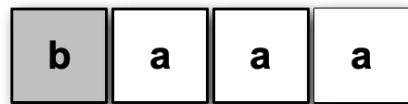
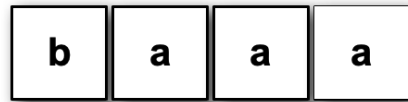
—**Turing machines** define recursively enumerable language

FINITE-STATE AUTOMATA



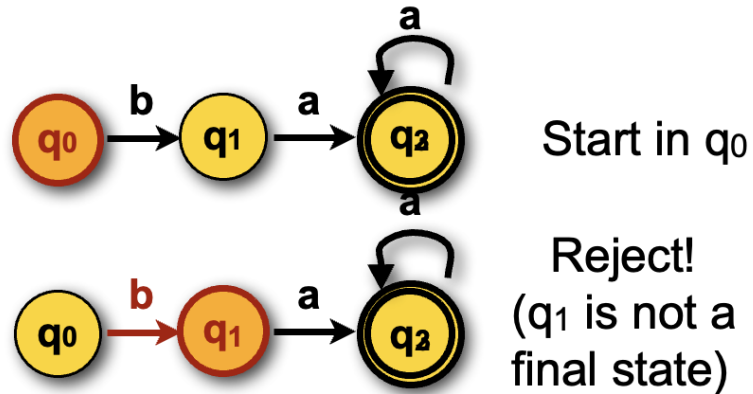
A (deterministic) finite-state automaton (FSA) consists of:

- a **finite set of states** $Q = \{q_0, \dots, q_N\}$, including a **start state** q_0
 - and one (or more) **final (=accepting) states** (say, q_N)
 - a (**deterministic**) transition function
 - $\delta(q, w) = q'$ for $q, q' \in Q, w \in \Sigma$
-

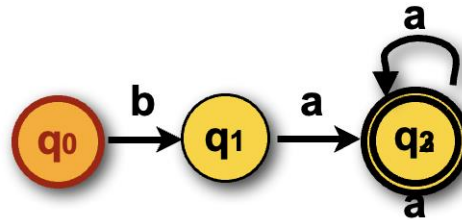
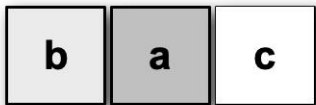


b

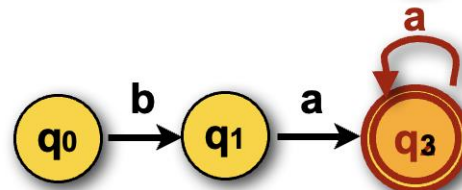
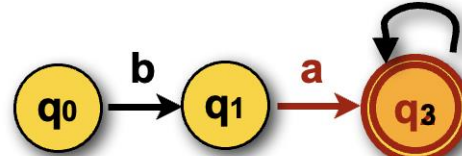
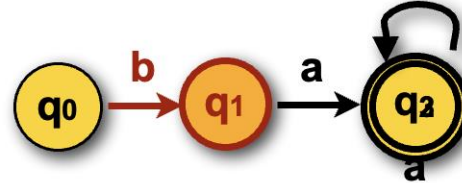
b



REJECTION:
AUTOMATON
DOES NOT
END UP IN
ACCEPTING
STATE

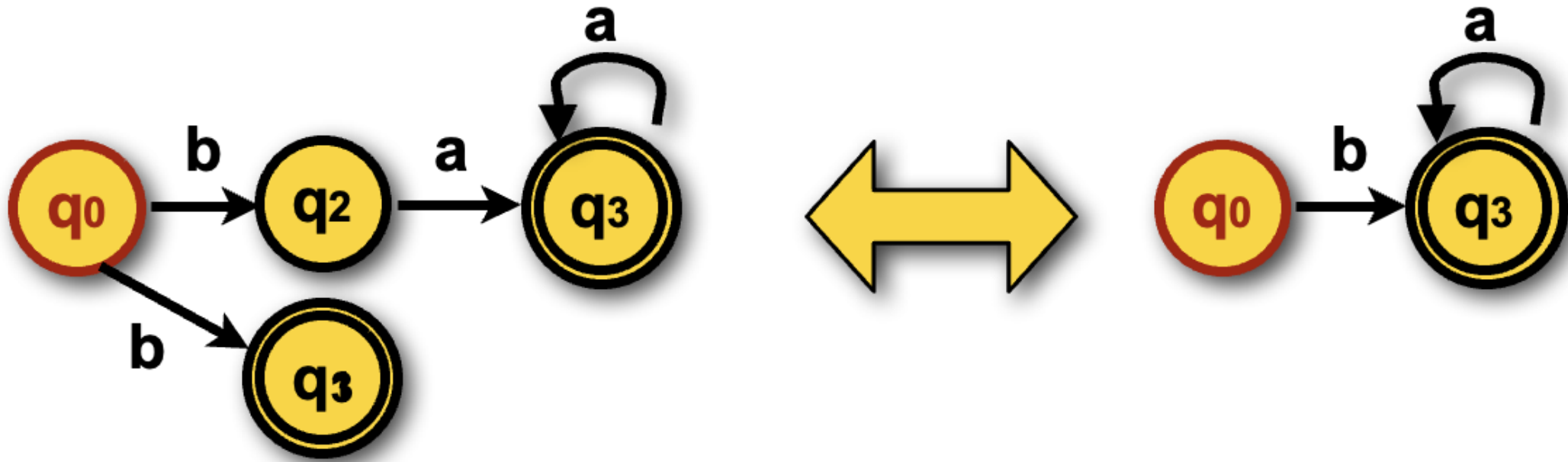


Start in q_0



Reject!
(There is no
transition
labeled 'c')

REJECTION: TRANSITION NOT DEFINED

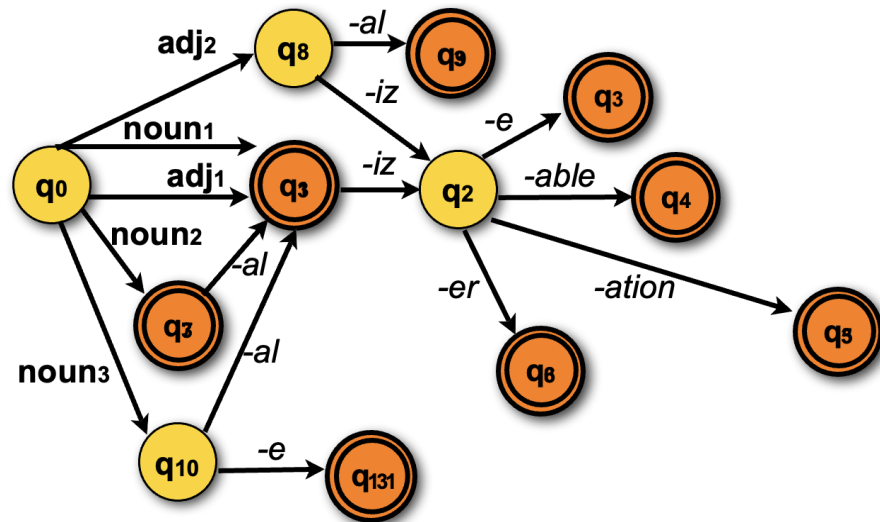


FINITE STATE AUTOMATA (FSAS)

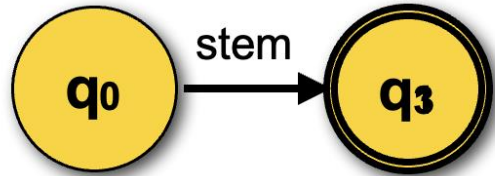
- Every NFA can be transformed into an equivalent DFA:
- Recognition of a string w with a DFA is linear in the length of w
- Finite-state automata define the class of regular languages
- $L_1 = \{ a^n b^m \} = \{ ab, aab, abb, aaab, abb, \dots \}$ is a regular language,
- $L_2 = \{ a^n b^n \} = \{ ab, aabb, aaabbb, \dots \}$ is not (it's context-free).
- You cannot construct an FSA that accepts all the strings in L_2 and nothing else.

FSAS FOR DERIVATIONAL MORPHOLOGY

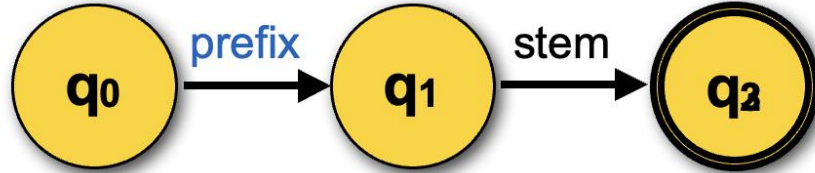
- **noun₁** = {*fossil, mineral, ...*}
- **adj₁** = {*equal, neutral*}
- **adj₂** = {*minim, maxim*}
- **noun₂** = {*nation, form, ...*}
- **noun₃** = {*natur, structur, ...*}



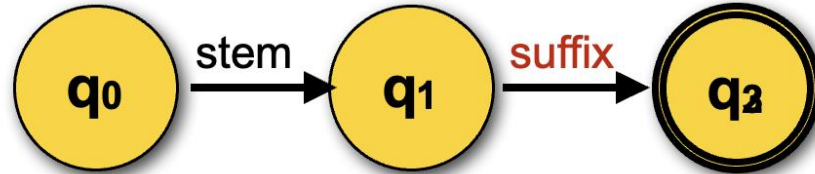
Zoom
grace:



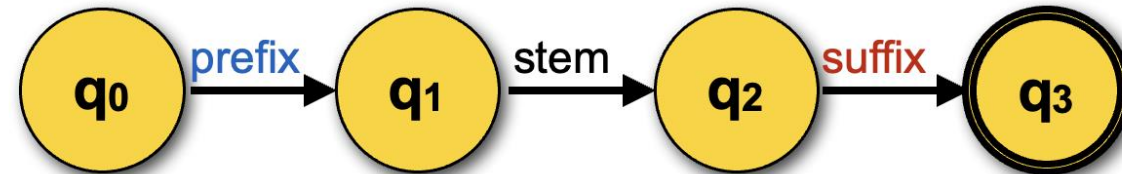
dis-grace:



grace-ful:

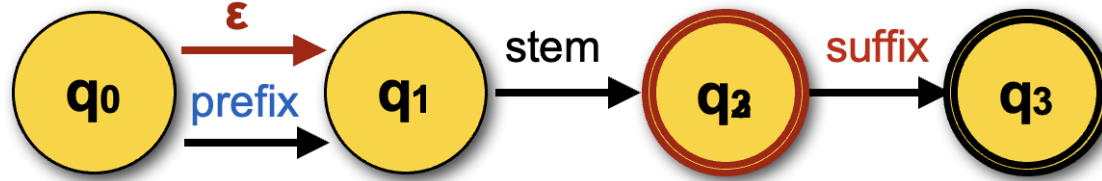


dis-grace-ful:



FINITE STATE AUTOMATA FOR MORPHOLOGY

grace,
dis-grace,
grace-ful,
dis-grace-ful



UNION:
MERGING
AUTOMATA

REGULAR EXPRESSIONS

Regular expressions (regexes) can also be used to define a regular language.

Simple patterns:

- - **Standard characters** match themselves: 'a', '1'
- - **Character classes:** '[abc]', '[0-9]', **negation:** '[^aeiou]'
 - (Predefined: \s (whitespace), \w (alphanumeric), etc.)
- - **Any character** (except newline) is matched by '.'

Complex patterns: (e.g. `^[A-Z]([a-z])+\s`)

- - **Group:** '(...)'
- - **Repetition:** 0 or more times: '*', 1 or more times: '+'
- - **Disjunction:** '... | ...'
- - **Beginning of line** '^' and **end of line** '\$'

FINITE-STATE TRANSDUCERS FOR MORPHOLOGY

RECOGNITION VS. ANALYSIS

Input
(Surface form)

c	a	t	s	
---	---	---	---	--



Output
(Lexical form)

c	a	t	+N	+pl
---	---	---	----	-----

- FSAs can recognize (accept) a string, but they don't tell us its internal structure.
- We need is a machine that maps (transduces) the input string into an output string that encodes its structure:

disgracefully

dis	grace	ful	ly
<i>prefix</i>	<i>stem</i>	<i>suffix</i>	<i>suffix</i>
<i>NEG</i>	grace+N	+ADJ	+ADV

MORPHOLOGICAL PARSING

MORPHOLOGICAL GENERATION



We cannot enumerate all possible English words, but we would like to capture the rules that define whether a string *could* be an English word or not.



That is, we want a procedure that can generate (or accept) *possible* English words...

grace, graceful, gracefully disgrace, disgraceful, disgracefully, ungraceful, ungracefully, undisgraceful, undisgracefully,...



without generating/accepting impossible English words

*gracelyful, *gracefuly, *disungracefully, ...
• NB: * is linguists' shorthand for "this is ungrammatical"

FINITE STATE AUTOMATA (FSAS)

A finite-
state
automaton
 $M = \langle Q, \Sigma, q_0, F, \delta \rangle$
consists of:

A finite set of **states** $Q = \{q_0, q_1, \dots, q_n\}$

A finite **alphabet** Σ of input symbols (e.g. $\Sigma = \{a, b, c, \dots\}$)

A designated **start state** $q_0 \in Q$

A set of **final states** $F \subseteq Q$

A **transition function** δ :

For a **deterministic (D)FSA**: $Q \times \Sigma \rightarrow Q$ $\delta(q, w) = q'$ for $q, q' \in Q, w \in \Sigma$

If the current state is q and the current input is w , go to q'

For a **nondeterministic (N)FSA**: $Q \times \Sigma \rightarrow 2^Q$ $\delta(q, w) = Q'$ for $q \in Q, Q' \subseteq Q, w \in \Sigma$

If the current state is q and the current input is w , go to any $q' \in Q'$

A finite-state transducer $T = \langle Q, \Sigma, \Delta, q_0, F, \delta, \sigma \rangle$ consists of:

- A finite **set of states** $Q = \{q_0, q_1, \dots, q_n\}$
- A finite alphabet Σ of **input symbols** (e.g. $\Sigma = \{a, b, c, \dots\}$)
- A finite alphabet Δ of **output symbols** (e.g. $\Delta = \{+N, +p1, \dots\}$)
- A designated **start state** $q_0 \in Q$
- A set of **final states** $F \subseteq Q$
- A **transition function** $\delta: Q \times \Sigma \rightarrow 2^Q$

$$\delta(q, w) = Q' \quad \text{for } q \in Q, Q' \subseteq Q, w \in \Sigma^*$$

- An **output function** $\sigma: Q \times \Sigma^* \rightarrow \Delta^*$
 - $\sigma(q, w) = \omega$ for $q \in Q, w \in \Sigma^*, \omega \in \Delta^*$
 - If the current state is q and the current input is w , write ω .

(NB: Jurafsky&Martin (2nd ed.) define $\sigma: Q \times \Sigma^* \rightarrow \Delta^*$. Why is this equivalent?)

FINITE-STATE TRANSDUCERS

An FST $T = L_{in} \times L_{out}$ defines a **relation**
between **two regular languages** L_{in} and L_{out} :

$L_{in} = \{\mathbf{cat}, \mathbf{cats}, \mathbf{fox}, \mathbf{foxes}, \dots\}$

$L_{out} = \{cat+N+sg, cat+N+pl, fox+N+sg, fox+N+pl \dots\}$

$T = \{ \langle \mathbf{cat}, cat+N+sg \rangle,$

$\langle \mathbf{cats}, cat+N+pl \rangle,$

$\langle \mathbf{fox}, fox+N+sg \rangle,$

$\langle \mathbf{foxes}, fox+N+pl \rangle \}$

FINITE-STATE TRANSDUCERS

Inversion T^{-1} :

- The inversion (T^{-1}) of a transducer switches input and output labels.
- *This can be used to switch from parsing words to generating words.*

Composition ($T \circ T'$): (Cascade)

- Two transducers $T = L_1 \times L_2$ and $T' = L_2 \times L_3$ can be composed into a third transducer $T'' = L_1 \times L_3$.
- *Sometimes intermediate representations are useful*

SOME FST OPERATIONS

Peculiarities of English spelling (orthography)

The same underlying morpheme (e.g. *plural-s*)

can have different orthographic “surface realizations” (-s, -es)

This leads to spelling changes at morpheme boundaries:

- E-insertion: fox +s = fox**e**s
- E-deletion: make**e** +ing = making

ENGLISH SPELLING RULES

INTERMEDIATE REPRESENTATIONS

- English plural -s: cat \Rightarrow cats

dog \Rightarrow dogs

- but: fox \Rightarrow foxes,
buses

bus \Rightarrow
buzz \Rightarrow buzzes

We define an intermediate representation to capture morpheme boundaries (^) and word boundaries (#):

- Intermediate-to-Surface Spelling Rule:
- If plural 's' follows a morpheme ending in 'x', 'z' or 's', insert 'e'.

Lexicon:

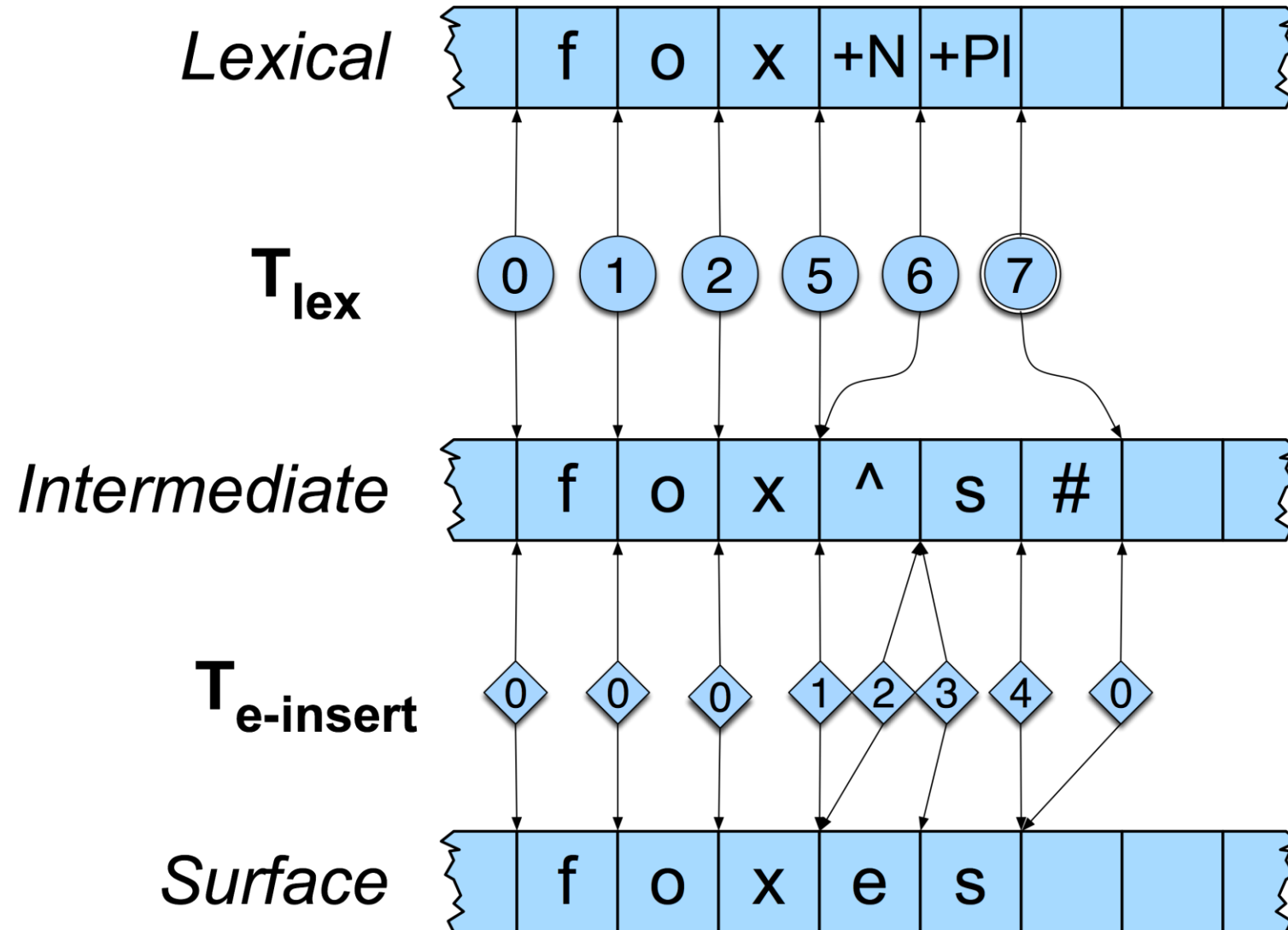
\Rightarrow *Intermediate representation:*

\Rightarrow *Surface string:*

cat+N+PL **fox+N+PL**

cat^s# **fox^s#**

cats **foxes**



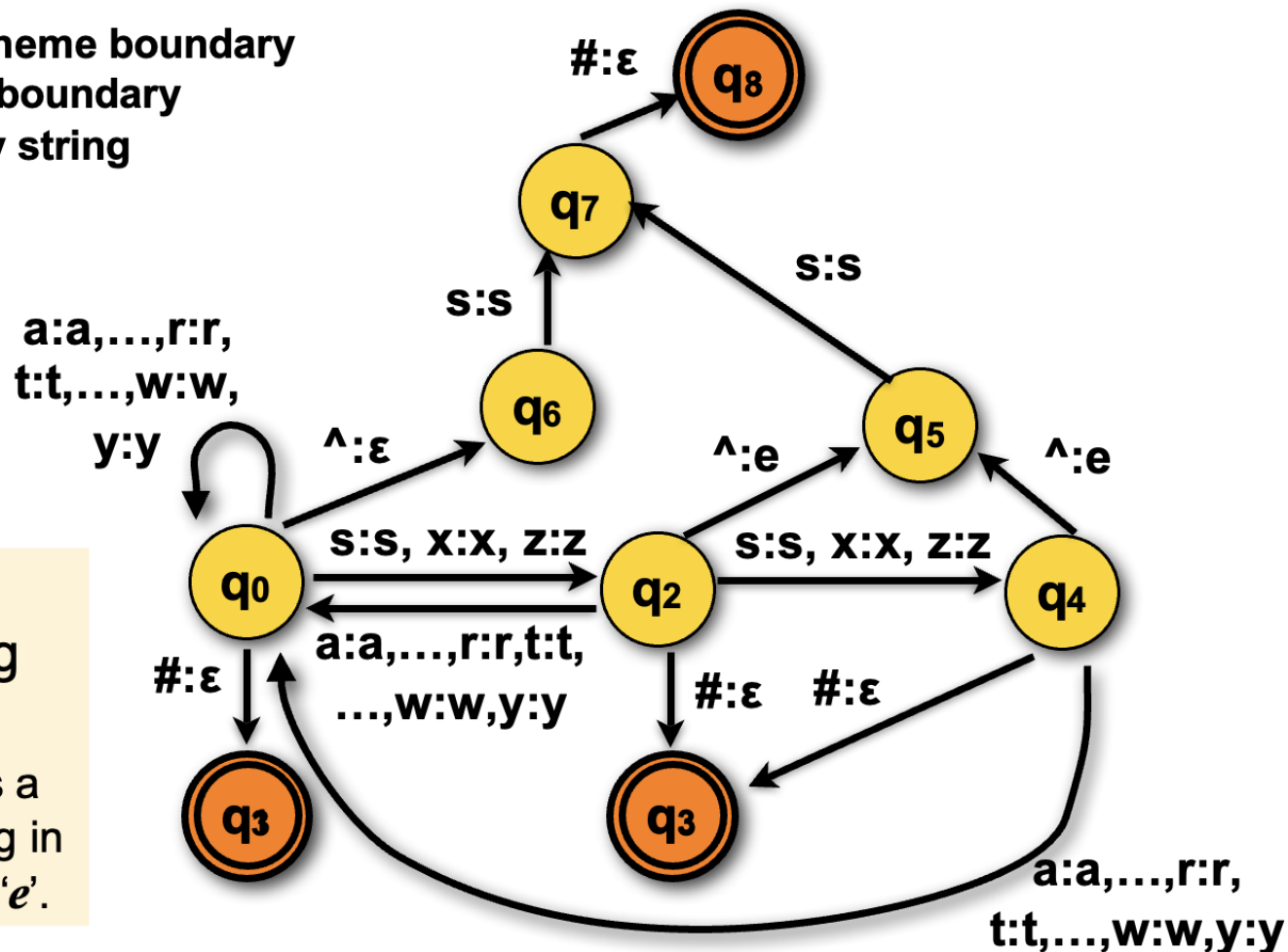
FST COMPOSITION/CASCADE:



\wedge = morpheme boundary
 $\#$ = word boundary
 ϵ = empty string

Intermediate-to-Surface Spelling Rule:

If plural 's' follows a morpheme ending in 'x', 'z' or 's', insert 'e'.



TE-INSERT: INTERMEDIATE TO SURFACE
LEVEL



DEALING WITH AMBIGUITY

- *book*: *book +N +sg or book +V?*
 - Generating words is generally unambiguous,
 - but analyzing words often requires disambiguation.
-
- We need a nondeterministic FST.
 - Efficiency problem: Not every nondeterministic FST can be translated into a deterministic one!
-
- We also need a scoring function to identify which analysis is more likely.
 - We may need to know the context in which the word appears: (I read a **book** vs. I **book** flights)
-

WHAT ABOUT COMPOUNDS?

