Morphology: Word Formation, FSAs and FSTs

Instructor: Jackie CK Cheung

COMP-550

Fall 2018

J&M Ch 2.2, Ch 3

To Do Now

Install Python 2.7 if you don't already have it

Install NLTK and scikit-learn packages

- Course outline has more links

Review

Match the following terms to their object of study

Semantics	Sound patterns
Pragmatics	Passage structure
Discourse	Word structure
Syntax	Implied meaning
Phonology	Speech sounds
Phonetics	Sentence structure
Morphology	Literal meaning

Today's Topics

English morphology

- Inflectional and derivational morphology

Lemmatization vs. stemming

Formalization as FSA, FST

Starting Small

We begin by starting from the smallest level of grammatical unit in language, the **morpheme**.

anti- dis- establish -ment -arian -ism

Six morphemes in one word

cat -s

Two morphemes in one word

of

One morpheme in one word

Types of Morphemes

Free morphemes

May occur on their own as words (*happy, the, robot*)

Bound morphemes

Must occur with other morphemes as parts of words

Most bound morphemes are **affixes**, which attach to other morphemes to form new words.

Prefixes come before the **stem**: *un-*, as in *unhappy*

Suffixes come after the stem: *-s*, as in *robots*

Infixes go inside: *-f**king-*, as in *abso-f**king-lutely*

(Not really an infix, but as close as we get in English)

-ma-, in *edu-ma-cation*

Circumfixes go around: *em- -en*, as in *embolden*

Derivation vs. Inflection

Inflectional morphology is used to express some kind of grammatical function required by the language

go -> goes *think -> thought*

Derivational morphology is used to derive a new word, possibly of a different part of speech

happy -> happily *establish -> establishment*

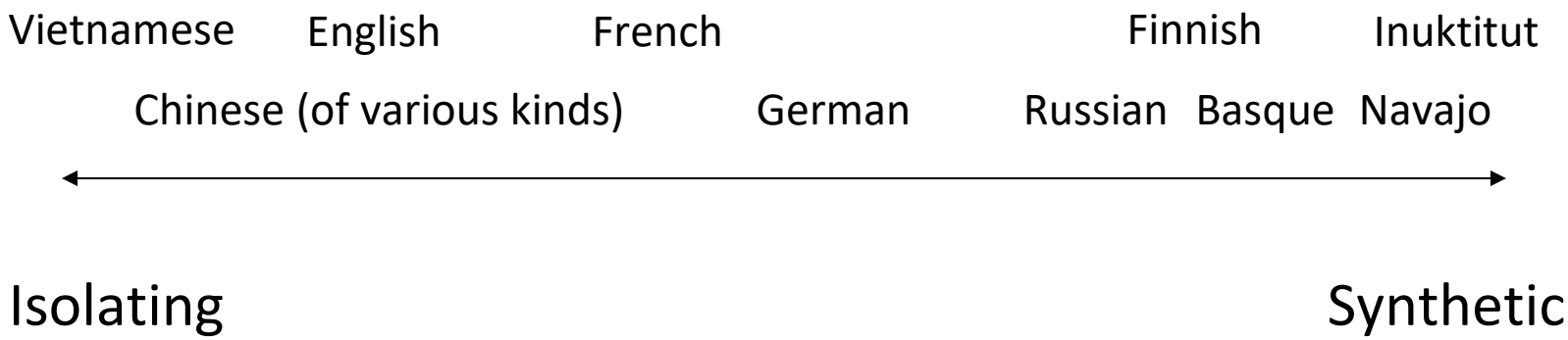
Can we come up with some prefixes and suffixes in English? Are they derivational or inflectional?

Cross-linguistic Variation

Languages across the world vary by their morpheme-to-word ratio.

Isolating languages: low morpheme-to-word ratio

Synthetic languages: high morpheme-to-word ratio



Note: this chart is only a rough guide, not a precise ranking!

Compare

English

I ca-n’t hear very well. 6 morphemes/5 (or 6) words = 1.2

Cantonese

我 聽 得 唔 係 好 好 7 / 7 = 1.0

ngo5 teng1 dak1 m4 hai6 hou2 hou2

I hear able NEG be very good

French

Je ne peux pas entend-re très bien. 9 / 7 = 1.29

I NEG can-1SG NEG hear-INF very well

Inuktitut

ᑐᓴᕐᔪᐊᑭᓇᓐᖏᑦᑐᕕᒃᓱᒪ 8 / 1 = 8.0

tusaa-tsia-runna-nngit-tu-alu-u-junga

hear-able-NEG-NOM-very-be-1SG

In Other Words...

English morphology is relatively boring!

(And this is why we're only spending one class on it.)

But It Still Matters!

Recognize whether a word is actually English

foxes vs. **foxs*

Abstract away details that don't matter for an application

The campers saw a bear.

The campers see a bear.

The camper saw a bear.

- In all cases, a bear was seen!

Generate the correct form of a word

see +PRESENT +3SG -> sees

see +PAST +2PL -> saw

Computational Tasks

1. **Morphological recognition**

   Is this a well formed word?

2. **Stemming**

   Cut affixes off to find the stem

   - *airliner -> airlin*

3. **Morphological analysis**

   Lemmatization – remove inflectional morphology and recover lemma (the form you'd look up in a dictionary)

   - *foxes -> fox*

   Full morphological analysis – recover full structure

   - *foxes -> fox* +N +PL

Task 1: Morphological Recognition

Are these valid English words?

friendship, unship, defriender, friendes

Relevant issues:

What prefixes and suffixes can go with a word?

- *-ship, un-, de-, -s*

Different forms of an affix

- *fox -> foxes*
- *friend -> friends*
- *fly -> flies*

Exceptions

- *goose -> geese*

Components

What we need to build an English morphological recognizer:

Lexicon

A list of words in English, along some of their properties

Morphotactics

Rules for how morphemes can combine in English

Orthographic rules

Rules to deal with regular changes in spelling as morphemes combine

Lexicon

A list of all words, affixes and their behaviours. Entries are often called **lexical items** (a.k.a., lexical entries, lexical units).

- e.g., Noun declensions (Jurafksy and Martin, p. 54)

reg-noun	irreg-pl-noun	irreg-sg-noun	plural
fox	*geese*	*goose*	*-s*
cat	*sheep*	*sheep*	
aardvark	*mice*	*mouse*	

Morphotactics

Tells us the sequence in which morphemes may be combined.

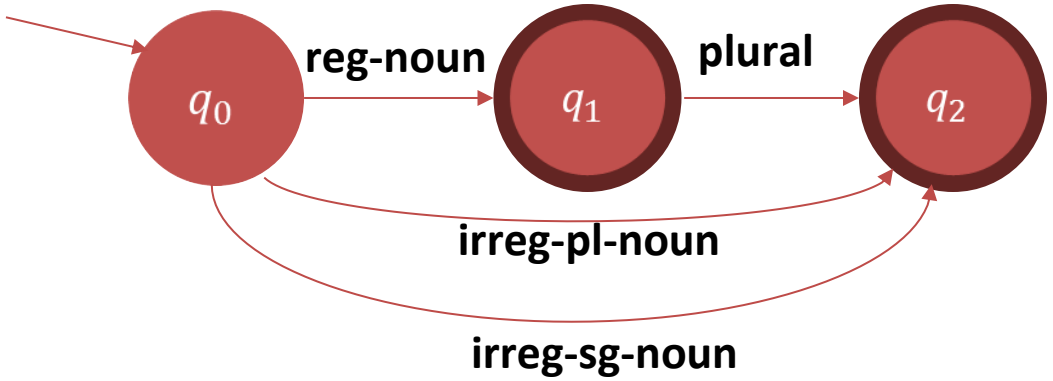
- e.g., English nouns:

  reg-noun

  reg-noun + -s

  irreg-sg-noun

  irreg-pl-noun



Nodes with outline represent an **accepting state**. (This is a word)

This is a representation of a **finite state automaton** (**FSA**).

Finite State Automata

A model of computation that takes in some input string, processes them one symbol at a time, and either **accepts** or **rejects** the string.

- e.g., we write a FSA to accept only valid English words.

A particular FSA defines a **language** (a set of strings that it would accept).

- e.g., the language in the FSA we are writing is the set of strings that are valid English words.

Regular languages are languages that can be described by an FSA (i.e., the FSA accepts exactly those strings that are in the language)

Definition of FSA

A FSA consists of:

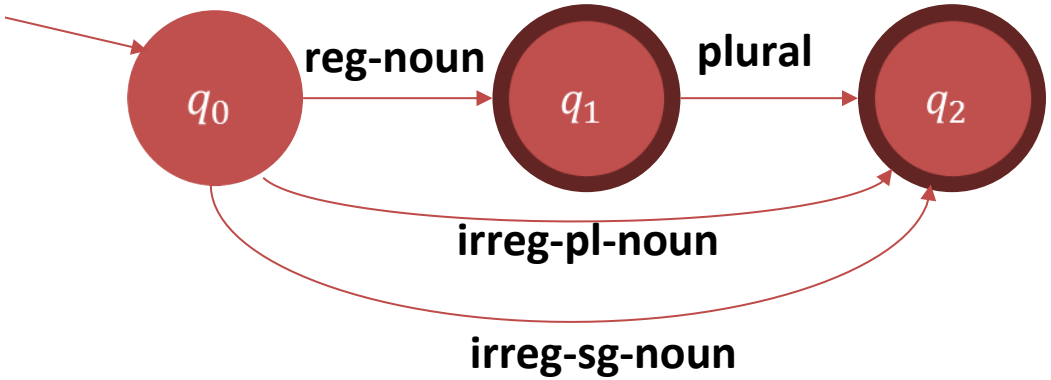
- Q finite set of states
- $\sum$ set of input symbols
- $q_0 \in Q$ starting state
- $\delta : Q \times \Sigma \rightarrow P(Q)$

transition function from current state and input symbol to possible next states

  - $P(Q)$ is the power set of Q.

- $F \subseteq Q$ set of accepting, final states

Example: Components of a FSA



States: q_0, q_1, q_2

Input symbols: reg-noun, plural, irreg-pl-noun, irreg-sg-noun

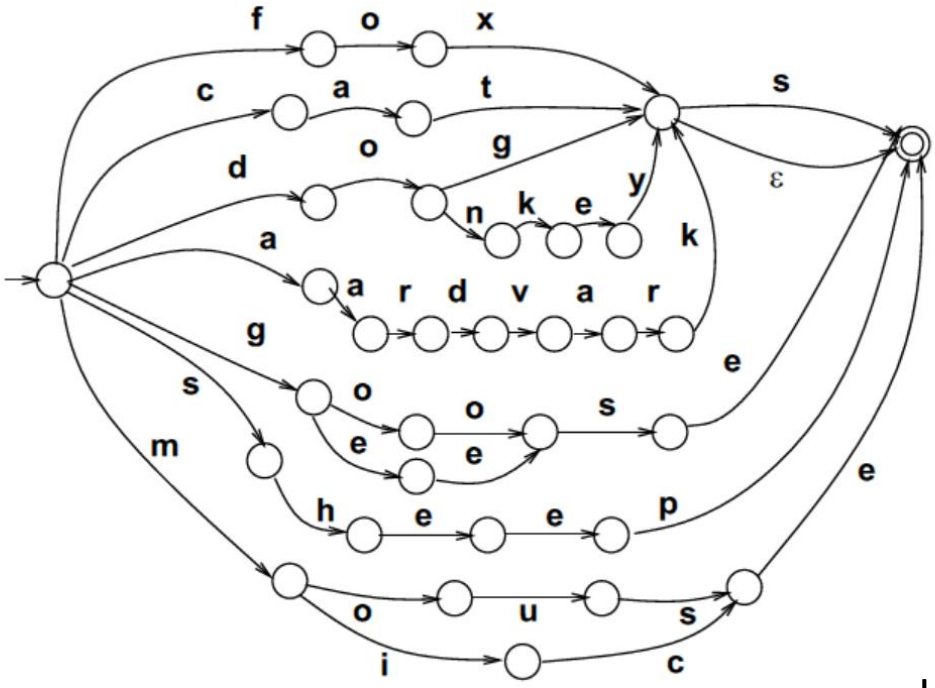
Starting state: q_0

Transition function: (q_0, reg-noun $\rightarrow$ q_1), … (as described by graph!)

Accepting state: q_1, q_2

First Attempt

Expand the morphotactic FSA by replacing the classes with the words from that class in the lexicon



J&M

Problem: *foxs* is accepted, not *foxes*!

Orthographic Rules

Regular orthographic variations of the plural suffix

- Ends with consonant + *y*: replace *y* with *ies*
  - e.g.*, pony, sky* but not *boy*
- Ends with *-s*, *-z*, *-x*, *-ch*, *-sh* -> add *-es*
  - e.g., *kiss*, *dish*, *witch*

Exercise: Extend the FSA in the previous slide to account for these variations

- Check your FSA with test cases: it should accept English words that you model and reject those that are not
- Is there a better way to handle this?

Task 2: Stemming – Porter Stemmer

An ordered list of rewrite rules to approximately recover the stem of a word (Porter, 1980)

- Basic idea: chop stuff off and glue some endings back on
- Not perfect, but sometimes results in a slight improvement in downstream tasks
- Advantage: no need for lexicon

Examples of Porter Stemmer Rules

ies -> i

- *ponies -> poni*

ational -> ate

- *relational -> relate*

If word is long enough (# of syllables, roughly speaking),

al -> ε

- *revival -> reviv*

Task 3: Morphological Parsing

Recover an analysis of the word structure

- *foxes -> fox +N +PL*
- *foxes -> fox +V +3SG.PR*

Add an intermediate layer to handle regular orthographic variations

Surface	*foxes*	*cats*
Intermediate	*fox^s#*	*cat^s#*
Underlying	*fox +N +PL*	*cat +N + PL*

- Lets us not have to deal with intricacies in the orthographic rules at the same time as the rest
- Irregular nouns handled by intermediate->underlying step

Expanded Lexicon

Include mapping for irregular nouns

reg-noun	irreg-pl-noun	irreg-sg-noun	plural
fox	*g e:o e:o se*	*goose*	*s:^s*
cat	*sheep*	*sheep*	
aardvark	*m i:o ɛ:u c:s e*	*mouse*	

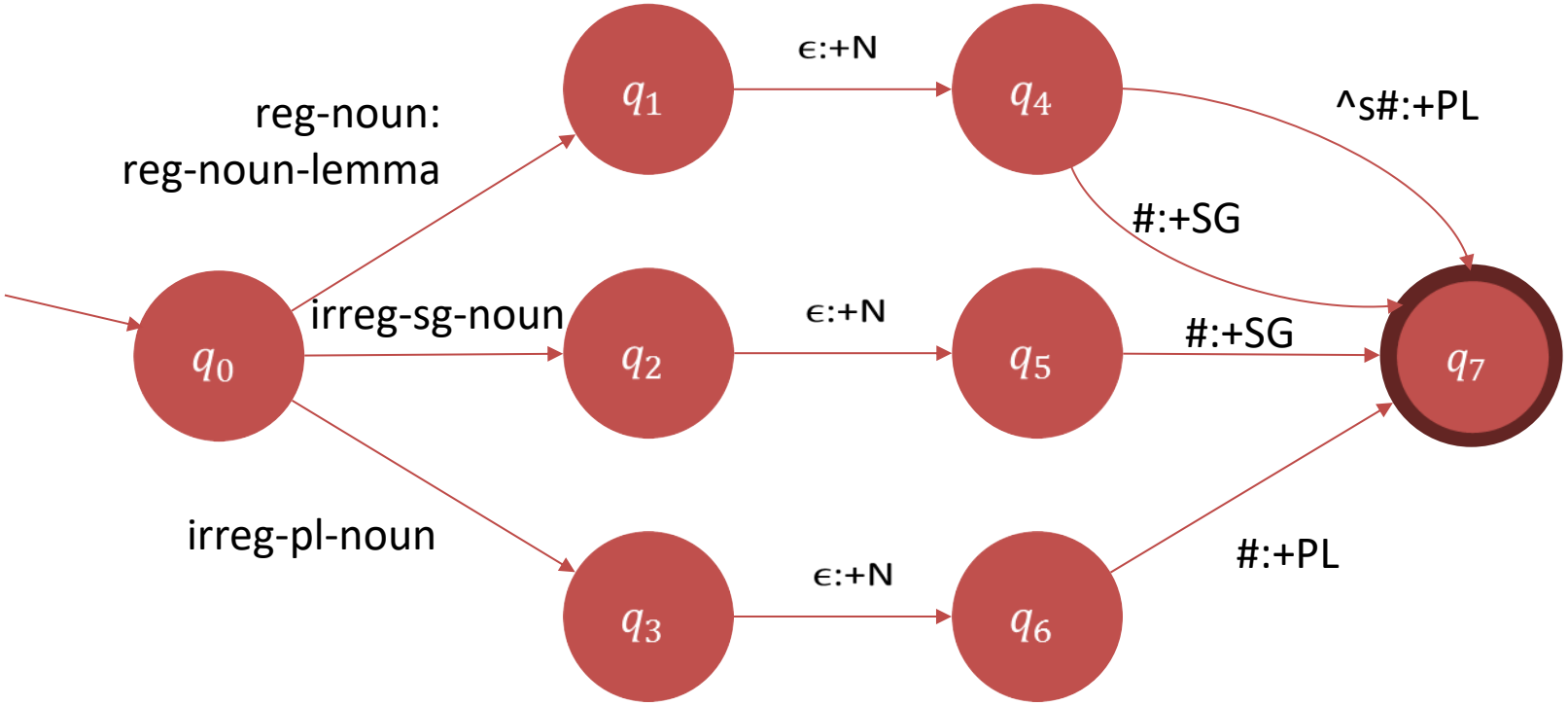
Notation:

- Single letter: map a letter to itself
- Implicit ɛ:# transition at ends of words
- Note: In J&M p62, the table is for generation, so the letters are flipped

Finite State Transducers

Next step: Intermediate to underlying level

- Need to expand parts like “reg-noun” below with lexicon



Question: what happens with *sheep*?

Example: surface-to-intermediate FST

Compared to previous FSA, need to:

- add outputs
- add states for the necessary adjustments as necessary
- i.e., this should map from the surface to the intermediate level

Let's sketch out this FST!

Now that we have added outputs, the machine is no longer a FSA. It is a **Finite State Transducer**.

Definition of FST

A FST consists of:

- Q finite set of states
- Σ, Δ sets of input symbols, output symbols
- $q_0 \in Q$ starting state
- $\delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times (\Delta \cup \{\varepsilon\}) \times Q$

  transition relation that specifies, for current state and input symbol, the possible pairs of output symbol and next state
- $F \subseteq Q$ set of accepting, final states

Identify the above components in the previous FST

Composing FSTs

We have two FSTs:

1. Surface to intermediate FST1 *fox -> fox#*
2. Intermediate to underlying FST2 *fox# -> fox +N +SG*

Compose them to make full morphological parser

- surface -> FST1 -> intermediate -> FST2 -> underlying

The composed machine is also a FST.

Inverting FSTs

We now have a FST for morphological parsing. What about morphological generation?

- Simply flip input and output symbol!

  ^s#:+PL becomes +PL:^s#

- underlying -> $FST^{-1}2$ -> intermediate -> $FST^{-1}1$ -> surface

Overall Picture

Build a lexicon of the words you care about

- Handle regular orthographic variations using this intermediate representation – write some rules or FSTs to describe them
- Handle irregular words by building exception lists

The multiple FSTs that you write will be combined in various ways to produce the final morphological analyzer/generator:

- Composition, intersection, …

In general, FSAs and FSTs are very useful models that pop up in many areas of NLP!

Exercise

Write a (surface to intermediate) transducer to segment monosyllabic verbs. Deal with the *-ed* and *-ing* suffixes.

- e.g., *jump -> jump#* *jumped -> jump^ed#*
  jumping -> jump^ing#
- Include some irregular verbs (e.g., *see*, *hit*)
- Deal with verbs that end in *e* (e.g., *hope*, *hate*)
- Deal with consonant doubling – when in a CVC pattern
  - e.g., *chat -> chatted, chatting* *bat -> batted, batting*
    [just handle the case of *t*]

Then, write the template of the intermediate to underlying transducer as in slide 26.