Natural Language Processing

Lecture 5

Today's Outline

Note: Errata (available on book's homepage)

Homework 1

Review FSAs

English Morphology

FSA and Lexicons

Finite State Transducers

Homework 1 (due 9/26)

1. (10 pts) Amelia Bedelia is a "literal-minded but charming housekeeper who never fails to confound the Rogers family" (from the jacket of Amelia Bedelia, by Peggy Parish). For example, when told to "Draw the drapes when the sun comes in," Amelia waits for the right time of day, takes out her pad, and sketches the drapes. What do you think Amelia does when told to "Dress the chicken" after it is delivered by the meat market? Explain why this book has kept children laughing, using the parlance of Chapter 1.

2. Consider recognizing a simple numerical language, consisting of integers and real numbers, where there is an optional preceding plus or minus, and a required digit before the decimal point. Thus 12, 12.34, +12, -12.34, 0.1 are in the language, but .1 is not.

- (10 pts) Design a NFSA using epsilon transitions to recognize this language.
- (10 pts) Design a DFSA that removes the epsilon transitions.
- (10 pts) Write a regular expression using only the primitive operators to recognize this language.

Homework 1 (continued)

3. (60 pts) Implement the ND-Recognize algorithm from Jurafsky and Martin (page 44) using depth first search. Test your algorithm using sheep language NFSA#1 (Fig. 2.18 in Jurafsky and Martin), as well as the NFSA developed for question 2 above. For each test, show several examples where your algorithm accepts an input string, and examples where it does not.

- Your code should be table-driven (i.e., represent the FSA as a table).
- Use whatever programming language you like.
- Don't worry about out of alphabet symbols.
- Your output should include the input and outputs of the algorithm.
- Also, your output should trace the execution of the machine to illustrate the handling of backup.

Review of FSAs

FSAs are formal devices used to recognize strings in a language.

FSAs can be specified using any of a number of formalisms.

FSAs can be combined in various ways to create FSAs for new regular languages.

Morphology

Morphology is the study of the way words are built up from smaller meaningful units called morphemes.

We can usefully divide the world of morphemes into two broad classes:

- stems - the core meaningful units in a lexicon
- affixes - bits and pieces that combine with stems to modify their meanings and grammatical functions

Affixes

Prefixes

- precede the stem
- *unbuckle*

Suffixes

- follow the stem
- *eats*

Infixes

- inserted inside the stem
- *absobl**dylutely*

Circumfixes

- precede and follow the stem
- not in English

Morphological Parsing

Parsing

- producing some structure for an input
- will be relevant for many levels of knowledge

Morphological Parsing (or Stemming)

- taking a surface input and breaking it down into its morphomes
- *foxes* breaks down into the morphemes *fox* (noun stem) and -*es* (plural suffix)
- *rewrites* breaks down into *re*- (prefix) and *write* (stem) and -*s* (suffix)

Why not instead include all words in a dictionary (e.g., all plurals)?

- inefficient because rules apply to many words
- rules apply to new words
- other languages besides English

Applications include IR, MT, spell checkers

Morphology (continued)

And we can also usefully divide morphology into two broad classes:

- inflectional morphology
- derivational morphology

Word Class

By word class we have in mind notions like noun and verb with which you are probably familiar.

We'll go into the gory details later (Chapter 8). For now it is sufficient to know about nouns, verbs, and adjectives.

We're concerned with word classes because the way that stems combine with affixes to form words is to a large degree based on the stem's word class.

Inflectional Morphology

Inflectional morphology concerns the combination of a stem with a grammatical morpheme which more or less results in a word...

- with the same word class as the stem
- which can serve a grammatical role that the stem could not (e.g., agreement)

Nouns and Verbs (in English)

English is simple.

- only nouns, verbs, and sometimes adjectives can be inflected
- small number of possible inflectional affixes

Nouns have a simple inflectional morphology: markers for *plural* and markers for *possessives*.

Verbs are slightly more complex: markers appropriate to the function the verb is being asked to serve.

Regulars and Irregulars

Morphology is complicated by the fact that many words refus to follow the rules.

- mouse/mice, ox/oxen, goose/geese
- go/went, fly/flew

We'll use the terms regular and irregular to refer to words that follow the rules and words that don't.

Nouns

Regular Nouns

- singular (cat, thrush)
- plural (cats, thrushes)
- possessive (cat's, thrushes')

Irregular Nouns

- singular (mouse, ox)
- plural (mice, oxen)

Regulars and Irregular Verbs

See tables on pages 62 and 63 in Jurafsky and Martin

stem: infinitive (I want to walk home), after certain other verbs (I'd rather walk home)

-*s* form: habitual present (She walks every Wednesday)

-*ing* participle: gerund (Walking is fine if you have a side-walk)

-*ed*: past form (I walked home yesterday)

Regular Verbs

If you know a regular verb stem, you can predict the other forms by adding one of the predictable endings and making some regular spelling changes.

Regular verbs cover a large majority of verbs.

The regular class is *productive* (includes new verbs).

Irregular Verbs

Three to eight morphological forms in English.

Small number of verbs, but among the most frequent.

Derivational Morphology

Derivational morphology refers to the stuff that you know but no one ever told you.

Characterized by

- quasi-systematicity
- irregular meaning changes
- changes of word class

English is complex

- derivational is less productive than inflection

Derivational Morphology Examples

See tables on page 64 in Jurafsky and Martin

Finite state Morphological Parsing

Inputs

- cats
- goose

Morphologically parsed output (stem and features)

- cat +N +PL
- (goose +N +SG) or (GOOSE +v) (ambiguous)

Requires

- lexicon (word repository)
- morphotactics (morpheme ordering)
- orthographic rules (spelling changes)

Using FSAs to Represent a Lexicon

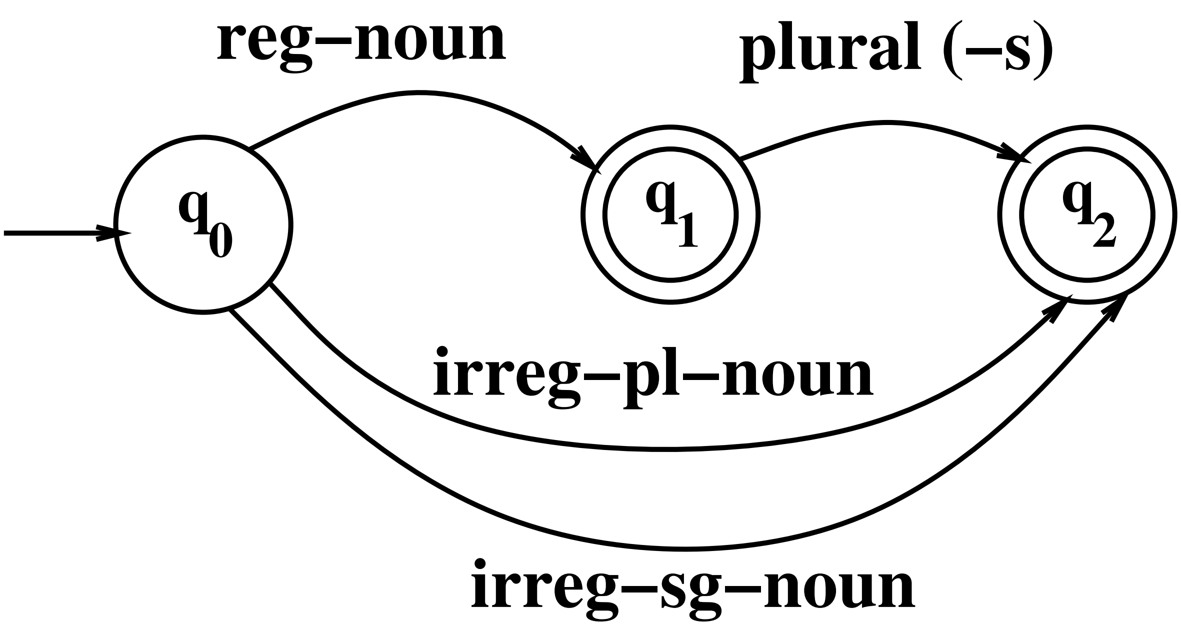
First we'll try to capture the morphotactics.

- the model of morpheme ordering
- e.g., the plural morpheme follows nouns in English

Then we'll layer in the actual words in the lexicon.

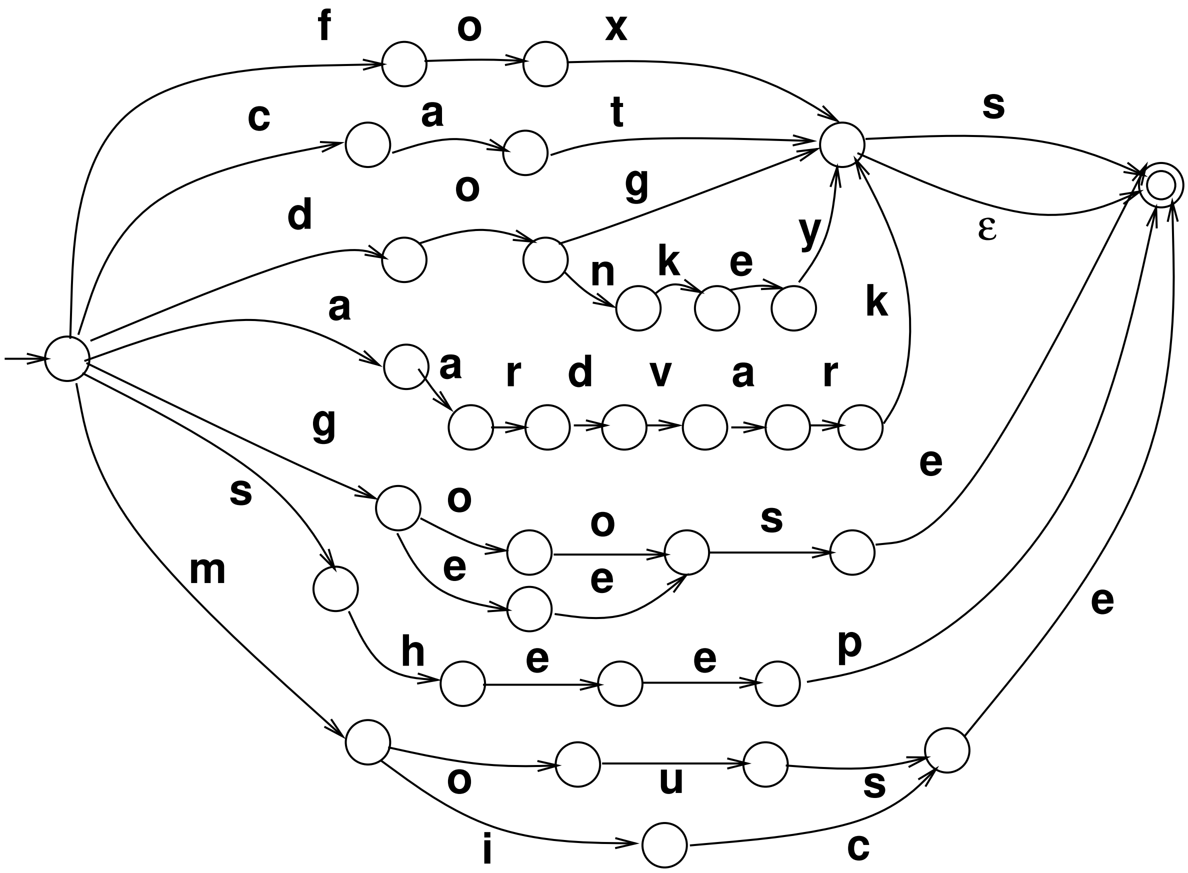
- the list of stems and affixes, together with basic information such as stem class
- an alternative to exhaustive lists, which are often inconvenient or impossible

Capturing Morphotactics using FSA

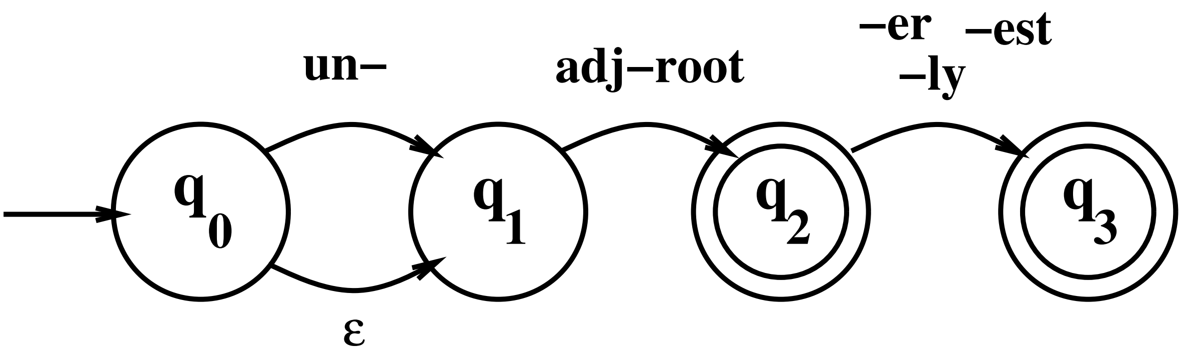


English nominal inflection

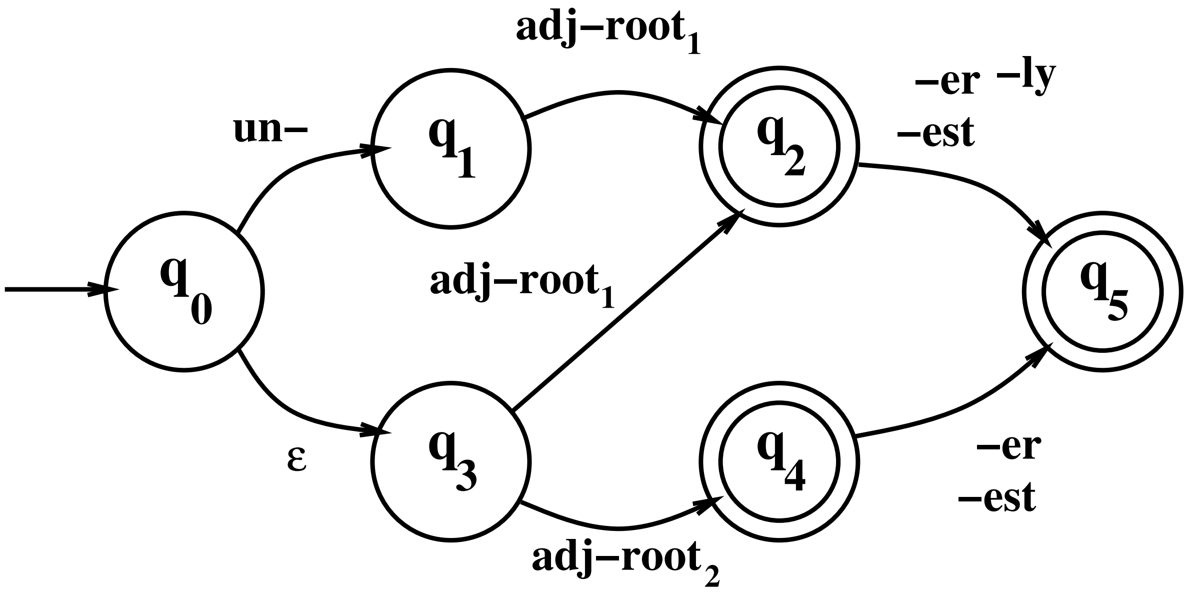
Embedding the Lexicon



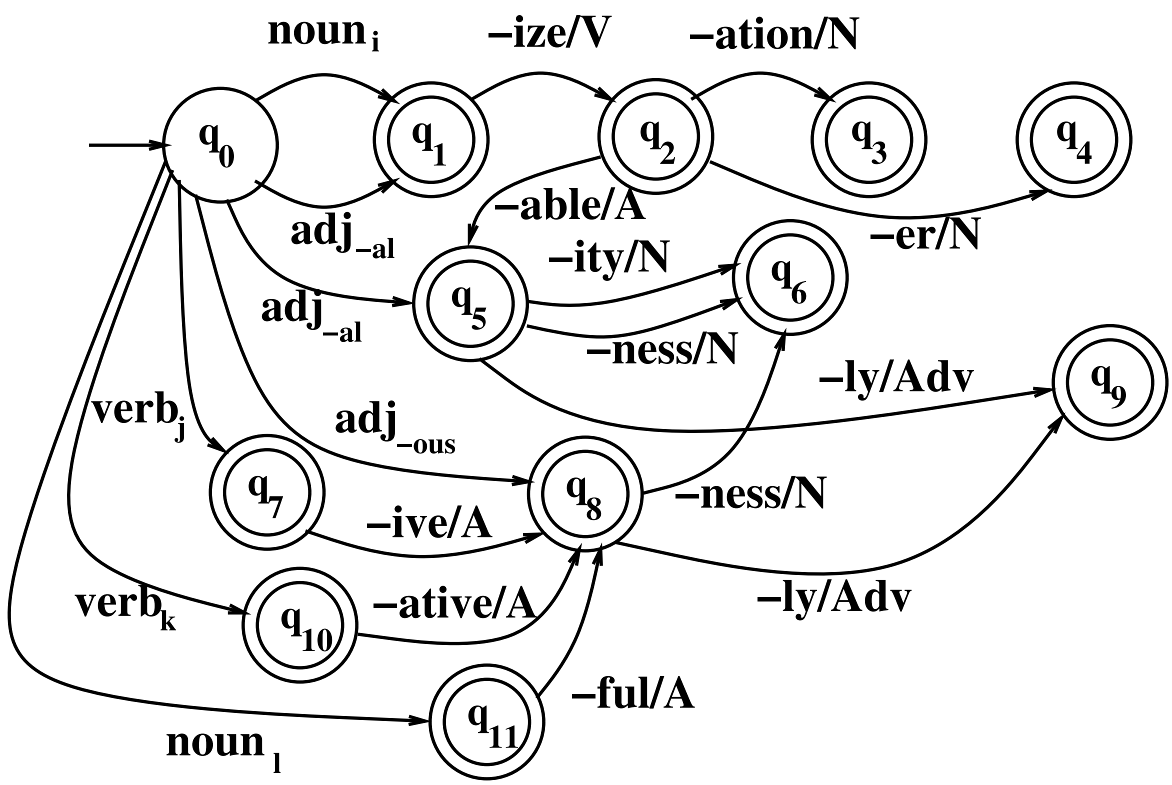
English Adjective Fragment FSA



Second Draft



Modeling Derivational Facts



Examples

- fossilize → fossilization
- equal → equality

Who cares?

Should search engines make use of morphological information?

- how might they make use of it?
- what happens if they do?
- what happens if they don't?

Recognition versus Parsing

Morphological Recognition determines whether an input string of letters makes up a legitimate word by embedding the lexicon in a morphotactic FSA.

Morphological Parsing is the process of taking an input word and creating a structure for it.

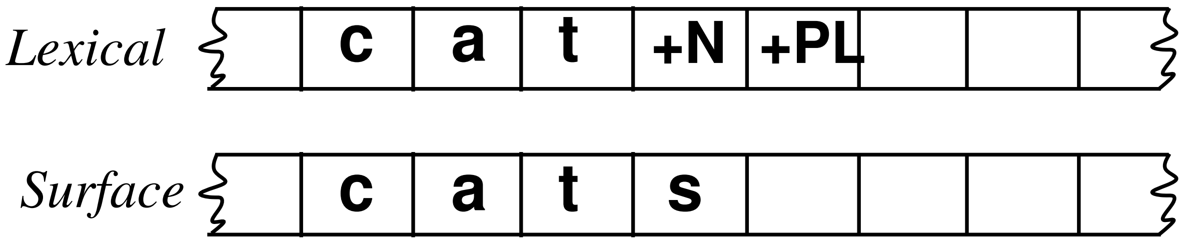
- cats $\rightarrow$ cat +N +PL

Finite State Transducers (FST)

The simple story . . .

- add an extra tape to normal FSA
- add extra symbols to the transitions of a normal FSA
- i.e., one tape has `cats` on it and we write `cat+N+PL` on the other

Tapes and transitions



Two Level Morphology represents words as a correspondence between lexical and surface levels.

Parsing rules maps surface to lexical level.

We usually visualize a FST as a 2-tape FSA which recognizes/generates *pairs* of strings.

Typical Use

In the typical use of FSTs we'll

- read from one tape (just as in FSA recognition), using the second symbol on each transition,
- and write to the second tape, using the first symbol on each transition.

FSTs and Ambiguity

Recall that in ND-FSA recognition, there can be multiple paths through a machine leading to an accept condition. Doesn't matter which is found.

In FSTs, the path taken through the machine actually matters.

Parsing *unionizable*

- `union-ize-able`
- `un-ion-ize-able`
- each represents a valid path through the machine resulting in a different parse.

FSTs and Ambiguity (continued)

There are a number of ways to handle this problem.

- simply accept the first successful structure written on the tape
- run the parse through all possible paths writing out all possible valid structures along the way
- bias that search in some manner so that only one (or a few) valid paths are actually pursued

Much of NLP involves implementing the last option.

FSTs and Ambiguity (continued)

Unionizable represents a problem involving a global ambiguity.

What's the problem with parsing *assess*?

For Next Time

Finish Chapter 3

Acknowledgements

Some of this material was taken from the course web pages of James Martin.