CZ4045 Natural Language Processing

Words and Transducers
(Chapter 3)

Outline

- English Morphology
- Stemming: Normalizing the words
- Tokenizing: Getting the words (or word-like elements)
- Segmentation: Getting sentences

English Morphology

- Morphology is the study of the ways that words are built up from smaller meaningful units called morphemes

- We can usefully divide morphemes into two categories
  - Stems: The core meaning-bearing units
  - Affixes: Bits and pieces that adhere to stems to **change their meanings** and **grammatical functions**

  - Example
    - cat → cats
    - regular → irregular

English Morphology

- We can further divide morphology up into two broad classes
  - **Inflectional:** has the same word class as the original, cat -> cats
  - **Derivational:** Changes of word class, care -> careless

- Word Classes
  - By word class, we have in mind familiar notions like noun and verb
    - We'll go into the details in POS tagging (Chapter 5)
  - Right now we're concerned with word classes because **the way that stems and affixes combine** is based to a large degree on the word class of the stem

Inflectional Morphology

- Inflectional morphology concerns the combination of stems and affixes where the resulting word:
  - Has the same word class as the original
  - Nouns are simple
    - Markers for plural and possessive
    - E.g. table, tables
  - Verbs are only slightly more complex
    - Markers appropriate to the tense of the verb
    - E.g. Walk, walks, walking

Regulars and Irregulars

- It is a little complicated by the fact that some words misbehave (refuse to follow the rules)
  - Mouse/mice, goose/geese, ox/oxen
  - Go/went, fly/flew

- The terms regular and irregular are used to refer to words that follow the rules and those that don't

Regular and Irregular Verbs

- So inflectional morphology in English is fairly straightforward
- But is complicated by the fact that there are irregularities

- Regulars…
  - Walk, walks, walking, walked, walked
- Irregulars
  - Catch, catches, catching, caught, caught
  - Cut, cuts, cutting, cut, cut

Derivational Morphology

- Derivational morphology
  - More complicated.
  - Many paths are possible…
  - Start with compute
    - Computer -> computerize -> computerization
    - Computer -> computerize -> computerizable
  - Meaning change
    - E.g., care -- careless
  - Changes of word class

Derivational Examples

- Nouns and Verbs → Adjectives

-al	computation	computational
-able	embrace	embraceable
-less	clue	clueless

- Verbs and Adjectives → Nouns

-ation	computerize	computerization
-ee	appoint	appointee
-er	kill	killer
-ness	fuzzy	fuzziness

Morphology and FSAs

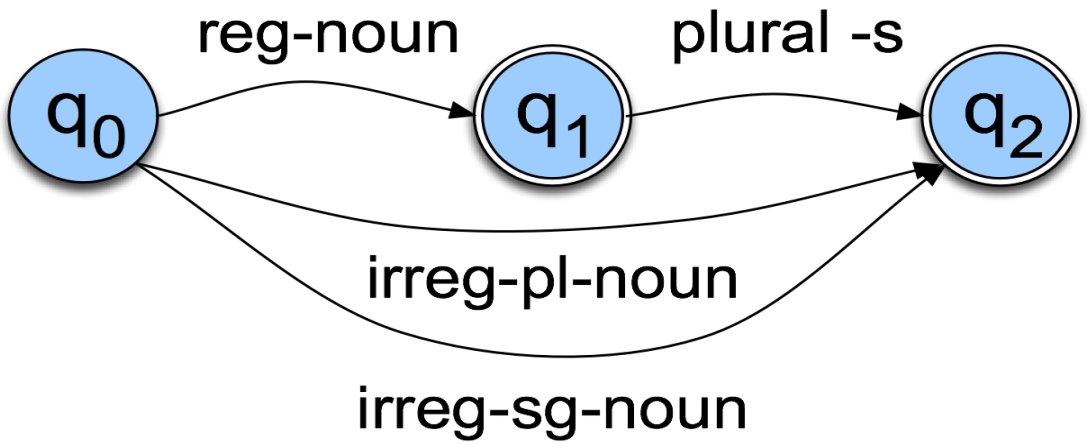
- We'd like to use the machinery provided by FSAs to capture these facts about morphology
  - Accept strings that are in the language
  - Reject strings that are not
  - Determine whether an input string of letters make up a legitimate English words

- So that we do not have to list all the words in the language

Start Simple

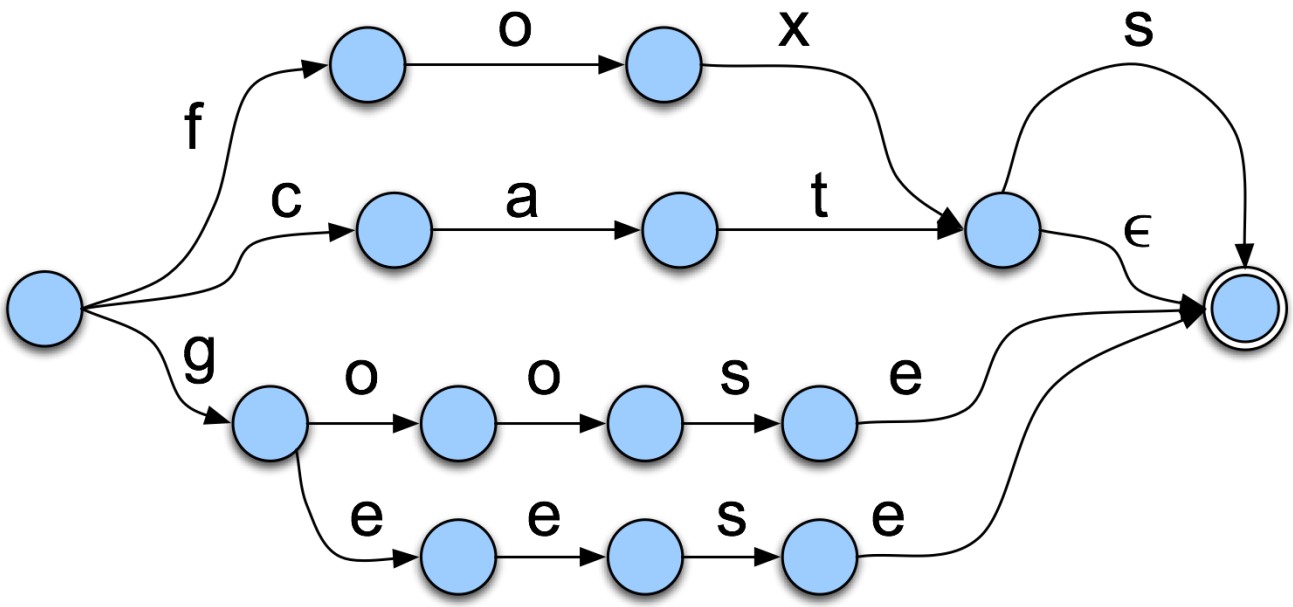
- Regular singular nouns are ok
  - Regular plural nouns have an -s on the end
- Irregulars are ok as is

- Simple Rules



Now, Plug in the Words

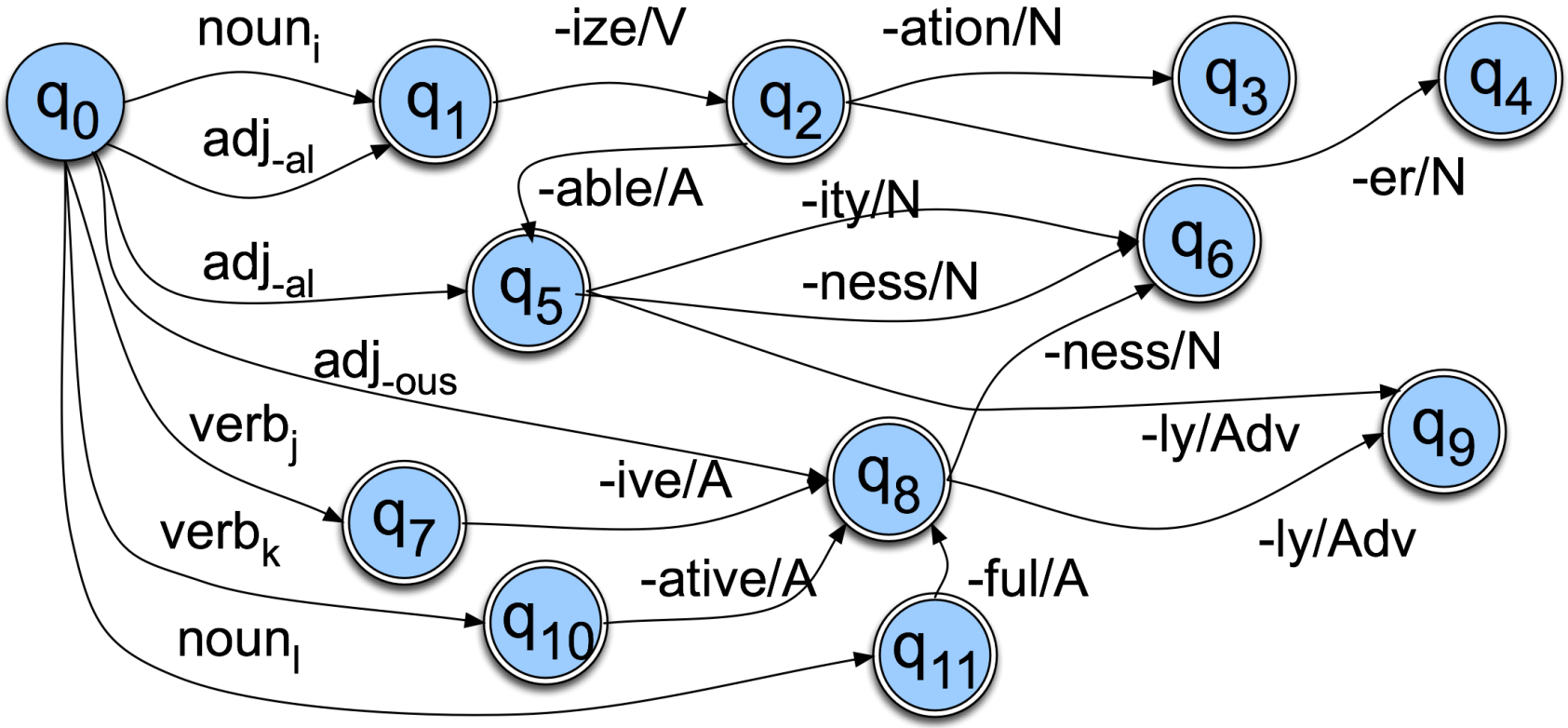
- Replace the class names like “reg-noun” with FSAs that recognize all the words in that class.



- Recognize strings, e.g. geese, goat, foxs
  - → more complicated solutions are needed

Derivational Rules

If everything is an accept state, how do things ever get rejected?



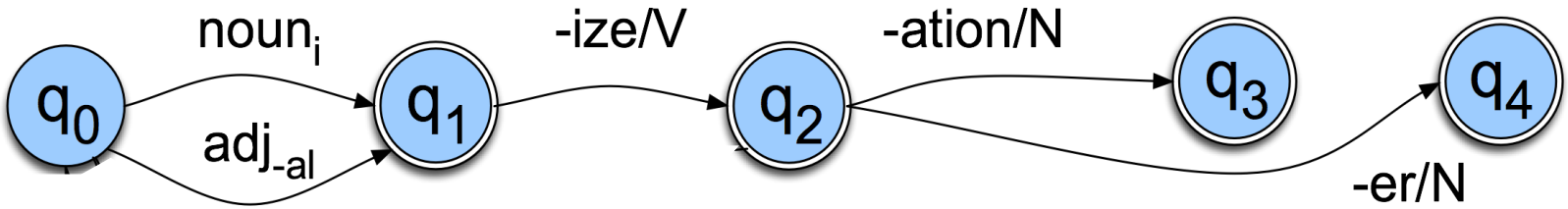
Notations:

$noun_i$: A subset of nouns that can accept *–ize*.

adj_{al} : Adjectives ending with *–al*

Exercise: Write a regular expression for the FSA

- A|B – A or B
- (ABC) – ABC as a component
- A? – A is optional



Parsing

- We can now run strings through these machines to recognize strings in the language
  - Spelling checking

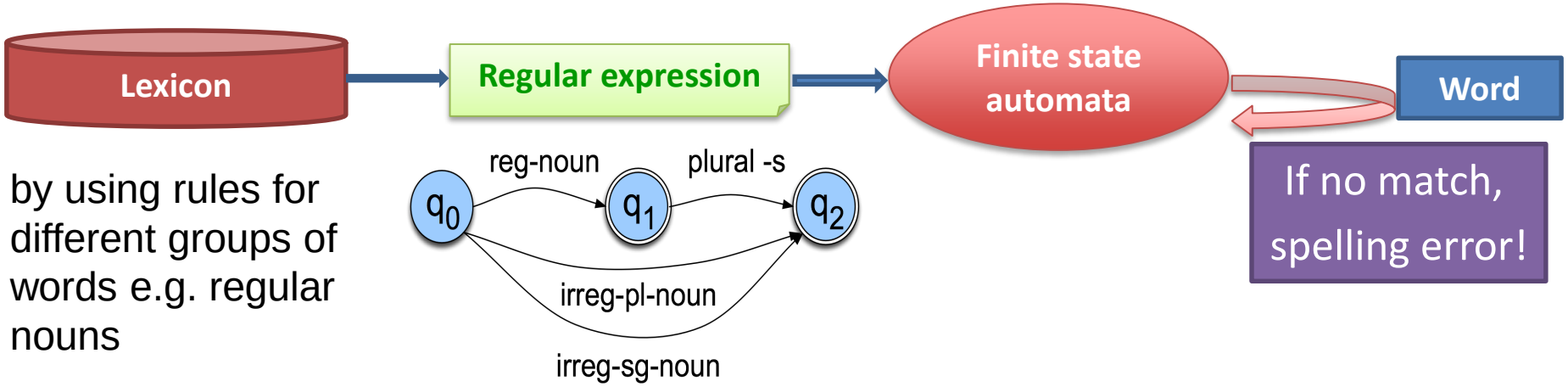
- Often if we find some string in the language we might like to assign a structure to it (parsing)
  - From “cats” to “cat +N +PL”
  - From “caught” to “catch+V+past”

- The kind of parsing we’re talking about is normally called morphological analysis

Applications

- Application 1: An important stand-alone component of many applications (spelling correction, information retrieval)

- Application 2: Simply a link in a chain of further linguistic analysis (e.g. parsing)



by using rules for different groups of words e.g. regular nouns

Light-Weight Morphology

- Sometimes you just need to know the stem of a word and you don't care about the structure.
  - E.g. camera, cameras
- In fact you may not even care if you get the right stem, as long as you get a consistent string—Stemming
- Stemming for Information Retrieval
  - Run a stemmer on the documents to be indexed
  - Run a stemmer on users' queries
  - Match to the index

Porter's stemming

- No lexicon needed

- Basically a set of staged sets of rewrite rules that strip suffixes
  - ING → ε (e.g., monitoring → monitor)
  - SSES→ SS (e.g., grasses → grass)

- More Example (recursive)
  - Computerization
    - ization -> -ize computerize
    - ize -> ε computer

Porter's stemming

- Handles both inflectional and derivational suffixes
- Doesn't guarantee that the resulting stem is really a stem
  - Lack of guarantee doesn't matter for IR
- Code:
  - http://tartarus.org/martin/PorterStemmer/
  - Implementations in C, Java, Perl, Python, C#, Lisp, Ruby, VB, javascript, php, Prolog, Haskell, matlab, tcl, D, and erlang

Stemming used in search

- Recall: reduce false negative? (Not matching things that we should have matched)
  - Query: “dog”
  - Doc 1: I love my dog
  - Doc 2: I do not like dogs

Works in this case

- Precision: increase false positives? (Matching strings that we should not have matched)
  - Query: “policy”
  - Doc 3: Singapore policy on gum
  - Doc 4: Singapore police cool

Wrong results here

policy—policies
police—policing

Tokenizing

- Identifying the tokens (words and symbols) in a text that we may want to deal with
- Called word segmentation, word tokenization
  - tokenizer
- Pretty much a prerequisite to doing anything interesting

Tokenizing

- For English, why not just use white-space?
  - **Mr. Sherwood said reaction to Sea Containers' proposal has been "very positive." In New York Stock Exchange composite trading yesterday, Sea Containers closed at $62.625, up 62.5 cents.**
  - **"I said, 'what're you? Crazy?' " said Sadowsky. "I can't afford to do that.''**

- Using white-space gives you words like:
  - **cents.**
  - **said,**
  - **positive."**
  - **Crazy?'**

Punctuation Issues

- Word-internal punctuation
  - M.P.H.
  - Ph.D.
  - AT&T
  - 01/02/06
  - Google.com
  - Yahoo!
  - 555,500.50
- Clitics
  - What’re
  - I’m
- Multi-token words (named entity detection)
  - New York
  - Rock ‘n’ roll

Language Issues

- Chinese and Japanese have no spaces between words
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃　现在　居住 在　美国　　东南部　　的　佛罗里达
  - Sharapova now　lives in　　US　southeastern　　Florida

- Example of other languages? Thai

- Further complicated in Japanese, with multiple alphabets intermingled
    - フォーチュン500社は情報不足のため時間あた$500K(約6,000万円)

Segmentation in Chinese

- Words composed of characters
- Average word is 2.4 characters long.
- Standard segmentation algorithm:
  - Maximum Matching or Maxmatch (also called greedy algorithm)

Maximum Matching Word Segmentation

- Given a lexicon of Chinese, and a string

- Start a pointer at the beginning of the string
- Find the longest word in dictionary that matches the string starting at pointer
- Move the pointer over the word in string
- Go to 2

Lexicon (Dictionary)
the
table
down
there
…

thetabledownthere

the table down there

Another Example

thetabledownthere

theta bled own there

Lexicon (Dictionary)
the
theta
table
down
there
bled
own

- But works pretty well in Chinese
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
  - Sharapova now lives in US southeastern Florida

- State-of-the-art solutions are mostly based on probabilistic

Practical Examples

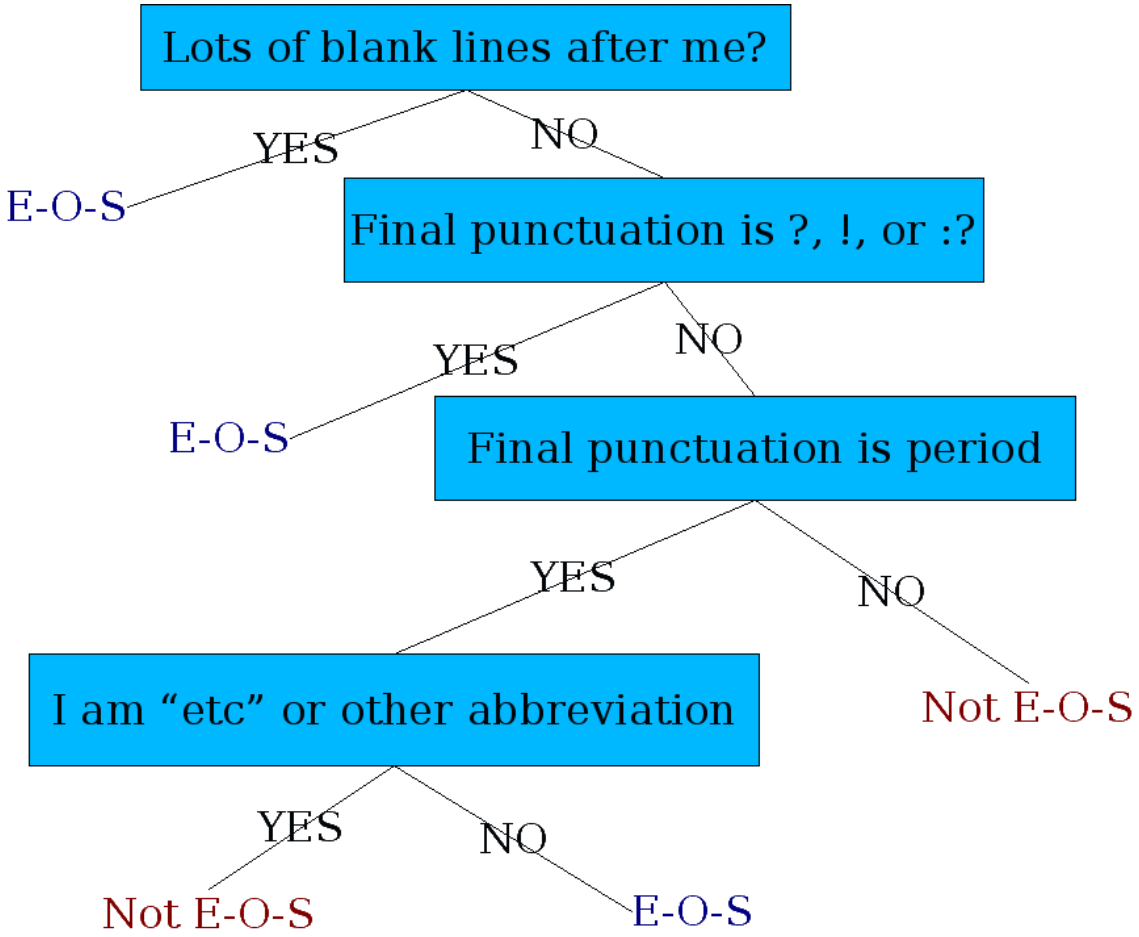
- URL segmentation
  - www.dietsthatwork.com
  - www.choosespain.com

- Hashtag segmentation
  - #unitedbrokemyguitar
  - #manchesterunited
  - allows Twitter users to track what many people (especially people whom you aren't already following) are reporting or thinking about a particular topic or event.

Sentence Segmentation

- !, ? are relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
- General idea:
  - Build a binary classifier:
    - Looks at a "."
    - Decides EndOfSentence/NotEOS
    - Could be hand-written rules, sequences of regular expressions, or machine-learning

Decision Tree Version (An example solution)



Summary

- English Morphology
- Morphological analysis: Identifying morphological structure of words
- Stemming: Normalizing the words
- Tokenizing: Getting the words (or word-like elements)
- Segmentation: Getting sentences