Today 1/24

- English Morphology
- FSAs and Morphology
- Break
- FSTs

Transition

- Finite-state methods are particularly useful in dealing with a lexicon
- Lots of devices, some with limited memory, need access to big lists of words
- And they need to perform fairly sophisticated tasks with those lists
- So we'll switch to talking about some facts about words and then come back to computational methods
**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 classes
  - Stems: The core meaning-bearing units
  - Affixes: Bits and pieces that adhere to stems to change their meanings and grammatical functions

**English Morphology**

- We can also divide morphology up into two broad classes
  - Inflectional
  - Derivational

**Word Classes**

- By word class, we have in mind familiar notions like noun and verb
- We’ll go into the gory details in 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
  - Serves a grammatical/semantic purpose that is
    - Different from the original
    - But is nevertheless transparently related to the original

**Nouns and Verbs (English)**

- Nouns are simple
  - Markers for plural and possessive
- Verbs are only slightly more complex
  - Markers appropriate to the tense of the verb

**Regulars and Irregulars**

- Ok, so it gets 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

- Regulars…
  - Walk, walks, walking, walked
- Irregulars
  - Eat, eats, eating, ate, eaten
  - Catch, catches, catching, caught, caught
  - Cut, cuts, cutting, cut, cut

Inflectional Morphology

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

Derivational Morphology

- Derivational morphology is the messy stuff that no one ever taught you.
  - Quasi-systematicity
  - Irregular meaning change
  - Changes of word class
Derivational Examples

- Converting verbs and adjectives to nouns

<table>
<thead>
<tr>
<th>-tion</th>
<th>computerize</th>
<th>computerization</th>
</tr>
</thead>
<tbody>
<tr>
<td>-ee</td>
<td>appoint</td>
<td>appointee</td>
</tr>
<tr>
<td>-er</td>
<td>kill</td>
<td>killer</td>
</tr>
<tr>
<td>-ness</td>
<td>fuzzy</td>
<td>fuzziness</td>
</tr>
</tbody>
</table>

Derivational Examples

- Nouns and verbs to adjectives

<table>
<thead>
<tr>
<th>-all</th>
<th>computation</th>
<th>computational</th>
</tr>
</thead>
<tbody>
<tr>
<td>-able</td>
<td>embrace</td>
<td>embraceable</td>
</tr>
<tr>
<td>-less</td>
<td>clue</td>
<td>clueless</td>
</tr>
</tbody>
</table>

Compute

- Many paths are possible...
- Start with compute
  - Computer -> computerize -> computerization
  - Computer -> computerize -> computerizable
- But not all paths/operations are equally good (or even allowable)
  - Clue -> clueable
Morphology and FSAs

- We’d like to use the machinery provided by FSAs to capture facts about morphology
- I.e. Accept strings that are in the language
- And reject strings that are not
- And do it in a way that doesn’t require us to in effect 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

\[ q_0 \xrightarrow{\text{reg-noun}} q_1 \xrightarrow{\text{plural -s}} q_2 \xrightarrow{\text{irreg-pl-noun}} \]
\[ \xrightarrow{\text{irreg-sg-noun}} \]
Now Add in the Words

Derivational Rules

Homework

• How big is your vocabulary?
Homework

• Strings are an easy and not very good way to represent texts
• Normally, we want lists of sentences that consist of lists of tokens, that ultimately may point to strings representing words (lexemes)
• Lists are central to Python and will make your life easy if you let them

Parsing/Generation vs. Recognition

• We can now run strings through these machines to recognize strings in the language
  • Accept words that are ok
  • Reject words that are not
• But recognition is usually not quite what we need
  • Often if we find some string in the language we might like to find the structure in it (parsing)
  • Or we have some structure and we want to produce a surface form (production/generation)
• Example
  • From "cats" to "cat +N +PL"

Finite State Transducers

• The simple story
  • Add another tape
  • Add extra symbols to the transitions
  • On one tape we read "cats", on the other we write "cat +N +PL"
Applications

- The kind of parsing we’re talking about is normally called morphological analysis
- It can either be
  - An important stand-alone component of an application (spelling correction, information retrieval)
  - Or simply a link in a chain of processing

FSTs

<table>
<thead>
<tr>
<th>Lexical</th>
<th>c a t</th>
<th>+N</th>
<th>+Pl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surface</td>
<td>c a t s</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Transitions

- c:c means read a c on one tape and write a c on the other
- +N: means read a +N symbol on one tape and write nothing on the other
- +PL:s means read +PL and write an s
Typical Uses

• Typically, we’ll read from one tape using the first symbol on the machine transitions (just as in a simple FSA).
• And we’ll write to the second tape using the other symbols on the transitions.

Ambiguity

• Recall that in non-deterministic recognition multiple paths through a machine may lead to an accept state.
• Didn’t matter which path was actually traversed
• In FSTs the path to an accept state does matter since different paths represent different parses and different outputs will result

Ambiguity

• What’s the right parse (segmentation) for
  • Unionizable
  • Union-ize-able
  • Un-ion-ize-able
• Each represents a valid path through the derivational morphology machine.
Ambiguity

There are a number of ways to deal with this problem:
- Simply take the first output found
- Find all the possible outputs (all paths) and return them all (without choosing)
- Bias the search so that only one or a few likely paths are explored

The Gory Details

Of course, it's not as easy as
- "cat +N +PL" <-> "cats"
- As we saw earlier there are geese, mice and oxen
- But there are also a whole host of spelling/pronunciation changes that go along with inflectional changes
  - Cats vs Dogs
  - Fox and Foxes

Multi-Tape Machines

To deal with this we can simply add more tapes and use the output of one tape machine as the input to the next
- So to handle irregular spelling changes we'll add intermediate tapes with intermediate symbols
Generativity

- Nothing really privileged about the directions.
- We can write from one and read from the other or vice-versa.
- One way is generation, the other way is analysis.

Multi-Level Tape Machines

- We use one machine to transduce between the lexical and the intermediate level, and another to handle the spelling changes to the surface tape.

Lexical to Intermediate Level

Diagram showing the transition between lexical, intermediate, and surface levels.
Intermediate to Surface

- The add an "e" rule as in fox^s# <-> foxes#

Foxes

Note

- A key feature of this machine is that it doesn't do anything to inputs to which it doesn't apply.
- Meaning that they are written out unchanged to the output tape.
- Turns out the multiple tapes aren't really needed; they can be compiled away.
Overall Scheme

- We now have one FST that has explicit information about the lexicon (actual words, their spelling, facts about word classes and regularity).
  - Lexical level to intermediate forms
- We have a larger set of machines that capture orthographic/spelling rules.
  - Intermediate forms to surface forms

Cascades

- This is a scheme that we'll see again and again.
  - Overall processing is divided up into distinct rewrite steps
  - The output of one layer serves as the input to the next
  - The intermediate tapes may or may not wind up being useful in their own right
Next Time

- Finish Chapter 3 start on 4