CS114: Finite State Automata, Words, Transducers

January 22, 2014

Prof. Marie Meteer

Brandeis University

Additional slides courtesy of Jurafsky & Martin, James Pustejovsky and , Ray Mooney

Assignment 1: Sentence pivots

- Background
  - The theory of “given” and “new” says that the first part of a sentence grounds it in the context (the “given” part) and the second provides information (the “new” part)
  - One study looked at how to find the “pivot” between given and new based on the syntactic structure of the sentence
    - “Modeling Conversational Speech for Speech Recognition” Meteer & Iyer, 1997
  - The goal was to see if the vocabulary and language model for these two parts was different
- Task (part 1)
  - Write a program that uses lexical and part of speech information to split a sentence into its given and new parts
  - Base the split on finding the “first strong verb”

Programming goals

- Get used to Python and NLTK data
- Write a modularized program that separates the declarative rules from the control structure
- Write a program that is meant to be one component in a larger sequence
  - Use internal data structures that can be further modified
  - Separate "read" and "write" functions from the core program since you may not always be writing out the result
  - Put all content specific information in declarative rules so they can be changed for different types of input

Pivot point: After the first strong verb

- Before the pivot, after the pivot, no pivot
  - A.1: Uh/UH ,/, do/VBP you/PRP have/VB a/DT pet/NN Randy/NNP ?/.
  - B.2: Uh/UH ,/, yeah/UH ,/, currently/RB we/PRP have/VBP a/DT poodle/NN ./.
  - A.3: A/DT poodle/NN ,/, miniature/JJ or/CC ,/, uh/UH ,/, full/JJ size/NN ?/.
  - B.8: Well/UH ,/, um/UH ,/, I/PRP would/MD n't/RB ,/, uh/UH ,/, I/PRP definitely/RB would/MD n't/RB dispute/VB that/IN
  - B.22: And/CC I/PRP think/VBP ,/, uh/UH ,/, having/VBG listened/VBN to/IN you/PRP relative/JJ to/IN the/DT economy/NN thing/NN

Guidance

- Don't worry about the theory. Just find the first strong verb
- Follow the programming guidelines
- Keep your rules out of the control structure—you'll be looking at other kinds of data going forward on the same task

Words

- Finite-state methods are particularly useful in dealing with a lexicon
- Many devices, most with limited memory, need access to large lists of words
- And they need to perform fairly sophisticated tasks with those lists
- So we'll first talk 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 further 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 (PoS) 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 in 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

- 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

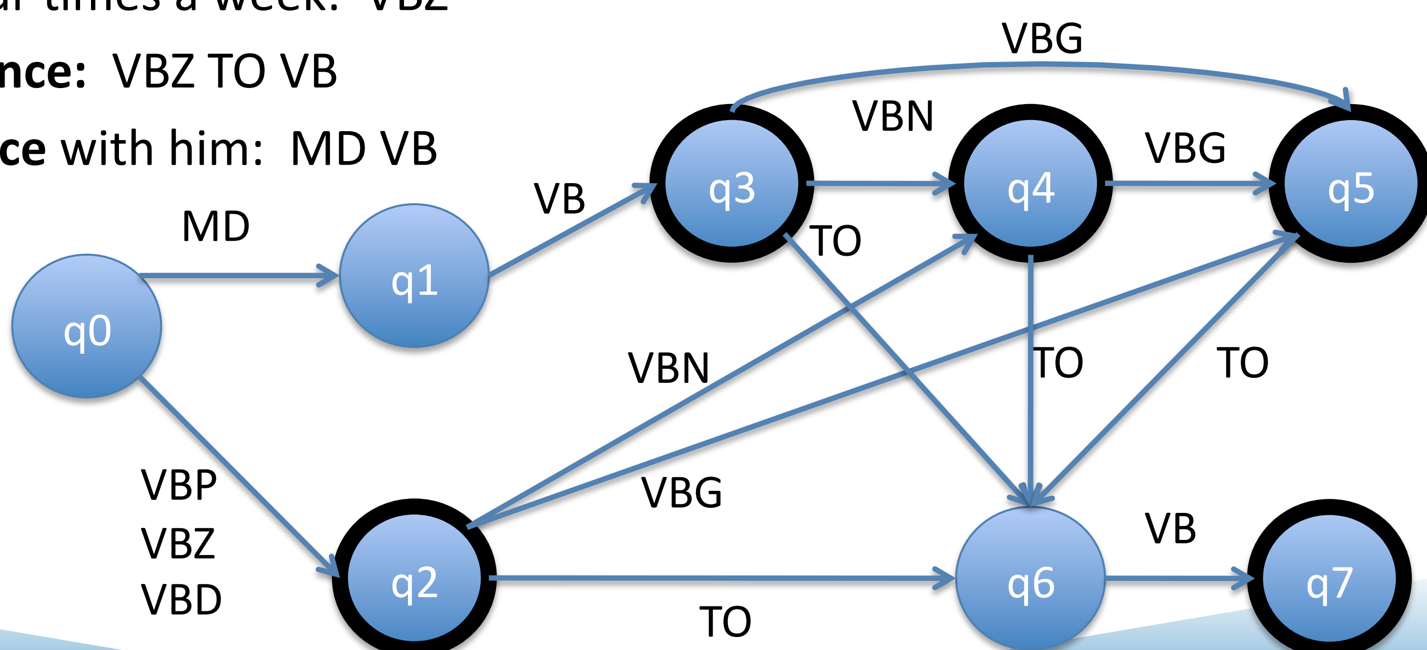
- Regulars…
  - Walk, walks, walking, walked, walked
- Irregulars
  - Eat, eats, eating, ate, eaten
  - Catch, catches, catching, caught, caught
  - Cut, cuts, cutting, cut, cut

Verb forms: Not just affixes

- Progressive: be ---ing
- Perfect: have ---ed
- Modality expressed as a word
  - Should, would, could
- Tense affects the first element in the verb group (unless it's a modal)

FSA for Verb Group Parts of Speech

- I **could have danced** all night: MD VB VBN
- I **was dancing** when the lights went out: VBD VBG
- We **danced** the night away: VBD
- I **would have been dancing**, but ...: MD VB VBN VBG
- He **has danced** his whole life: VBZ VBN
- She **dances** four times a week: VBZ
- He **loves to dance:** VBZ TO VB
- She **might dance** with him: MD VB



Inflectional Morphology

- So inflectional morphology in English is fairly straightforward
- Except that it is highly ambiguous
  - Same endings used for multiple things
    - Plural nouns, present tense 3rd person verbs, possessive
    - Past, perfect, passive
- And complicated by the fact that are irregularities
  - Too many conquerors

Derivational Morphology

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

Derivational Examples

- Verbs and Adjectives to Nouns

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

Nouns and Verbs to Adjectives

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

Example: *Compute*

- Many paths are possible…
- Start with compute
  - Computer -> computerize -> computerization
  - Computer -> computerize -> computerizable
- But not all paths/operations are equally good (allowable?)
  - Computer -> *Computeree ?? *Computerness??
  - Clue
    - Clue -> *clueable
    - Clueless, Clueful?
    - Unkempt, kempt?, kemptify (meaning to comb one's hair)

Why care about morphology?

- 'Stemming' in information retrieval
  - Might want to search for "going home" and find pages with both "went home" and "will go home"
- Morphology in machine translation
  - Need to know that the Spanish words quiero and quieres are both related to querer 'want'
- Morphology in spell checking
  - Need to know that misclaim and antiundoggingly are not words despite being made up of word parts

Can’t just list all words

- Turkish
- Uygarlastiramadiklarimizdanmissinizcasina
  - (behaving) as if you are among those whom we could not civilize
- ’ Uygar `civilized’ + las `become’ + tir `cause’ + ama `not able’ + dik `past’ + lar ‘plural’+ imiz ‘p1pl’ + dan ‘abl’ + mis ‘past’ + siniz ‘2pl’ + casina ‘as if’

What we want

- Something to automatically do the following kinds of mappings:
- Cats   cat +N +PL
- Cat   cat +N +SG
- Cities   city +N +PL
- Merging   merge +V +Present-participle
- Caught   catch +V +past-participle

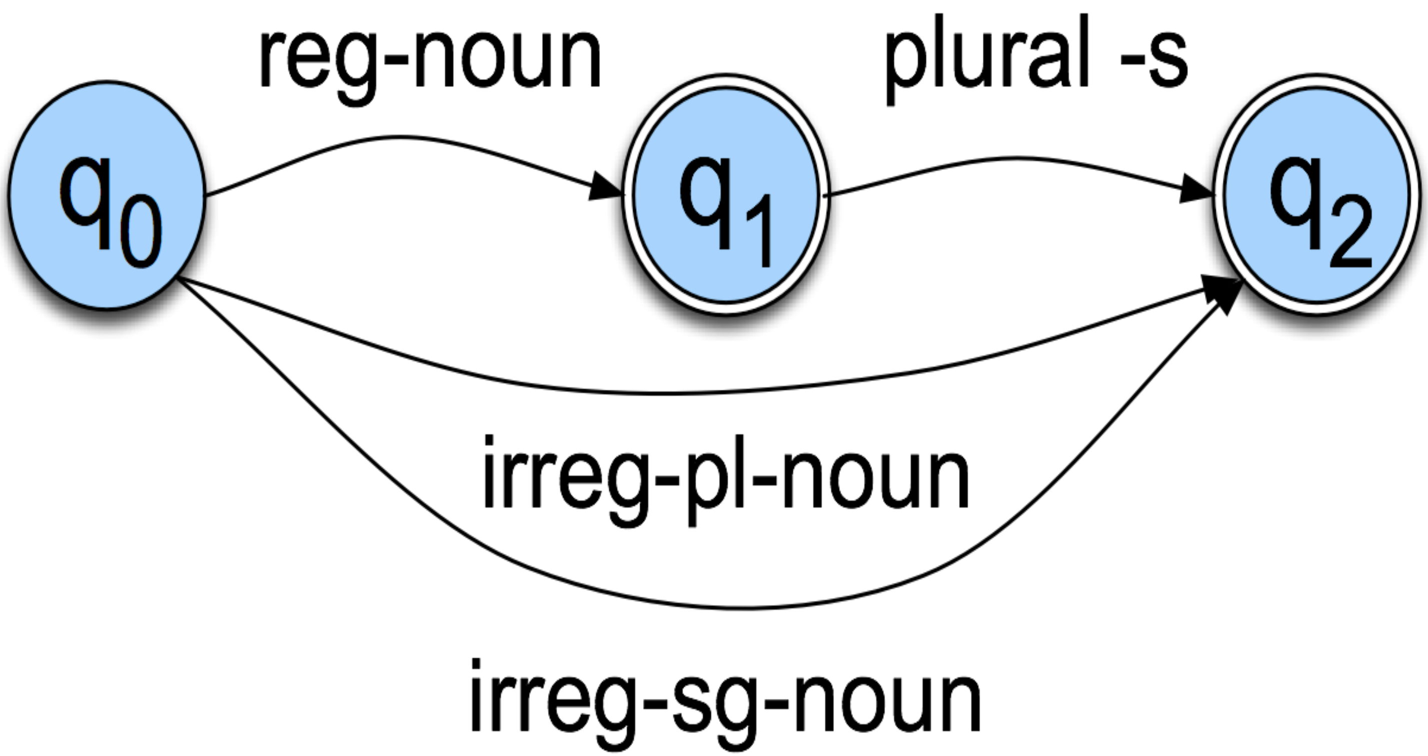
Morpholgy and FSAs

- 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
  - And do so in a way that doesn't require us to in effect list all the words in the language

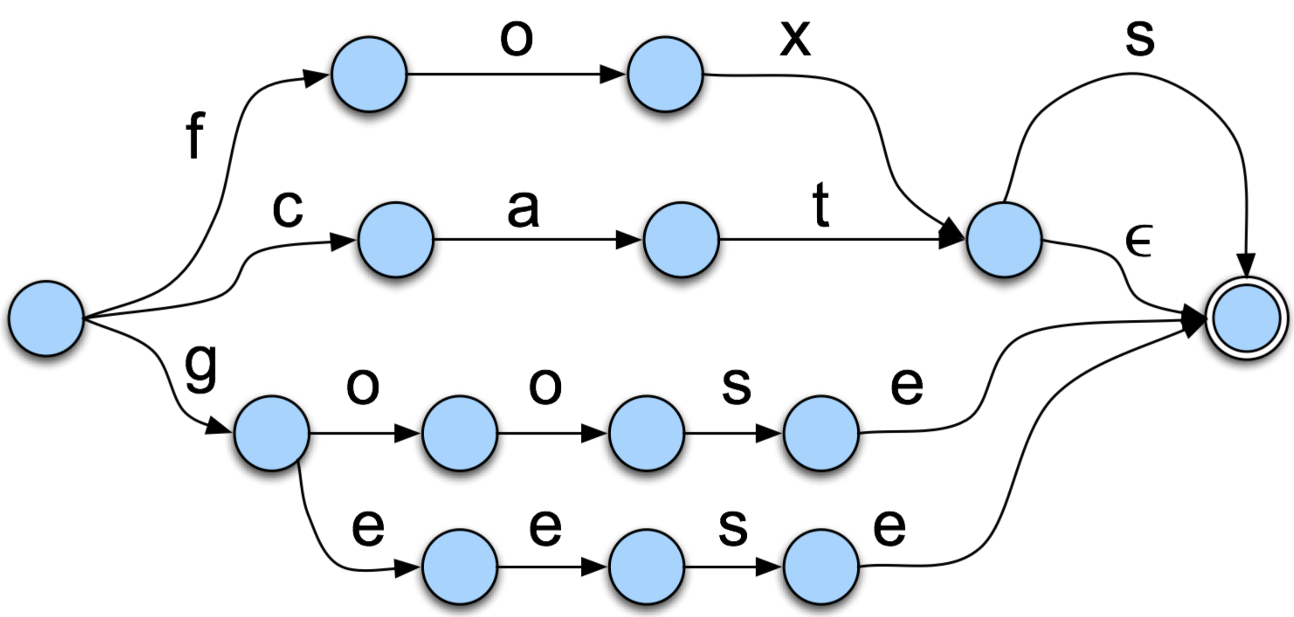
Start Simple

- Regular singular nouns are ok
- Regular plural nouns have an -s on the end
  - Note in speech there are three variants
    - –s, -z, or –ix-z
    - Cats, dogs, bushes
- Irregulars are ok as is

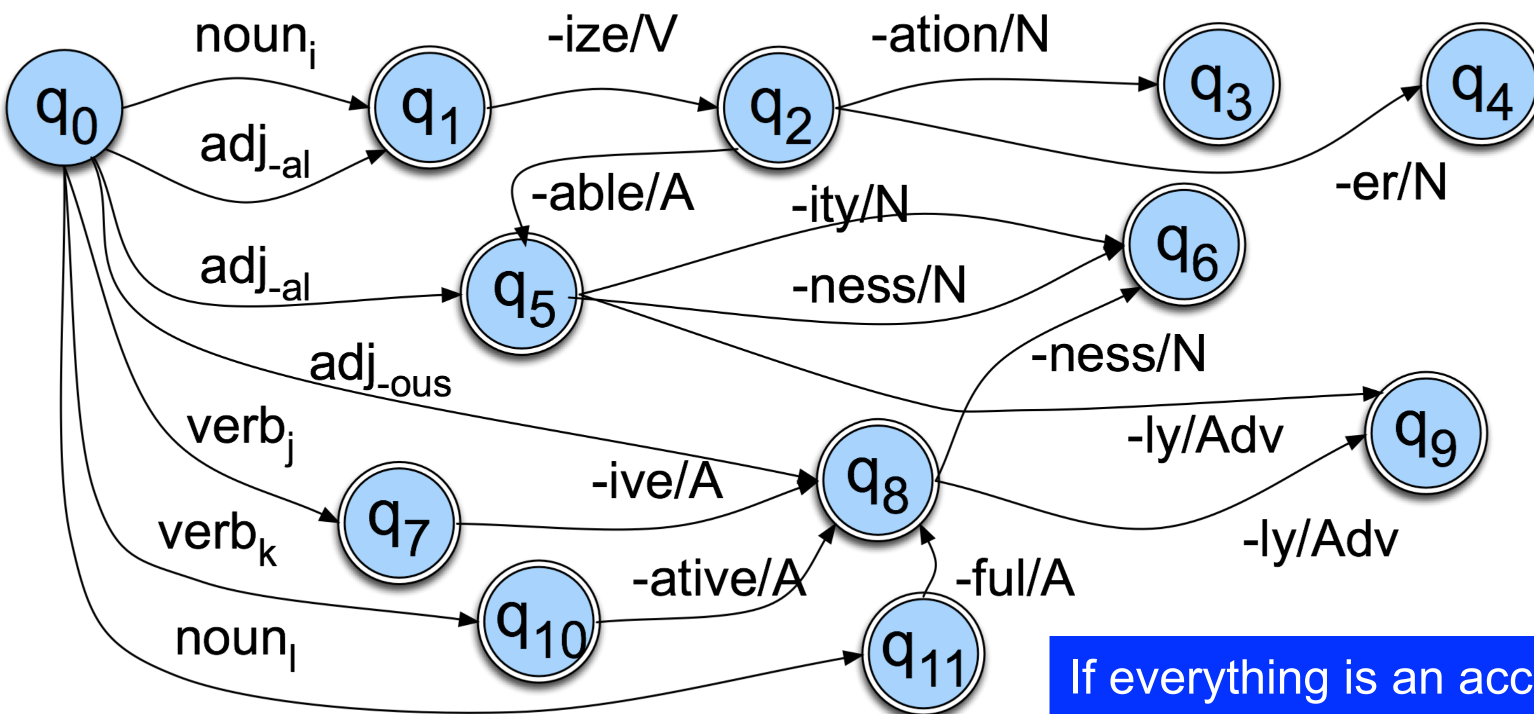
Simple Rules



Now Plug in the Words



Derivational Rules



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

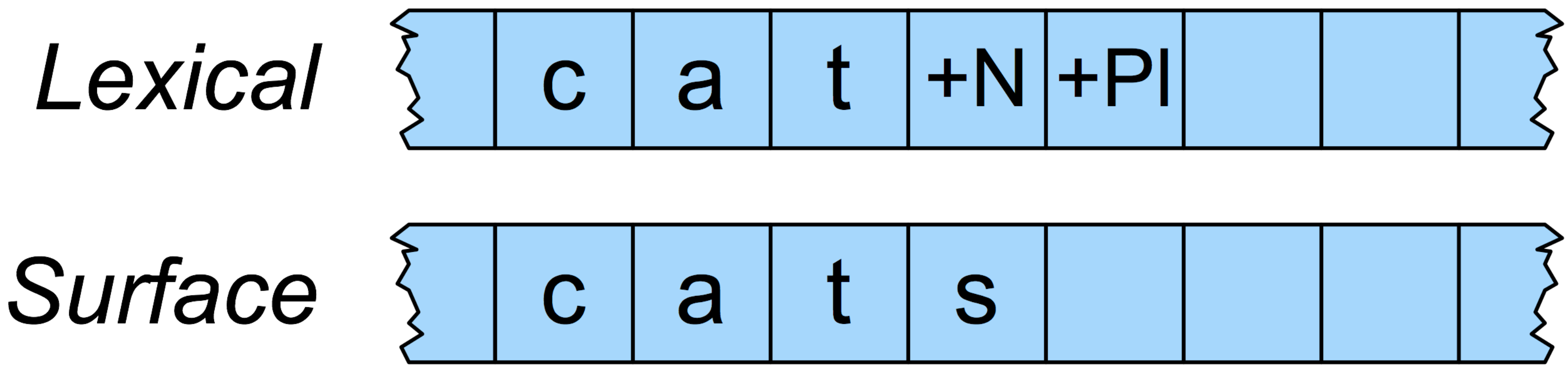
Parsing/Generation vs. Recognition

- We can now run strings through these machines to recognize strings in the language
- But recognition is usually not quite what we need
  - Often if we find some string in the language we might like to assign a structure to it (parsing)
  - Or we might have some structure and we want to produce a surface form for it (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"

FSTs



Applications

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

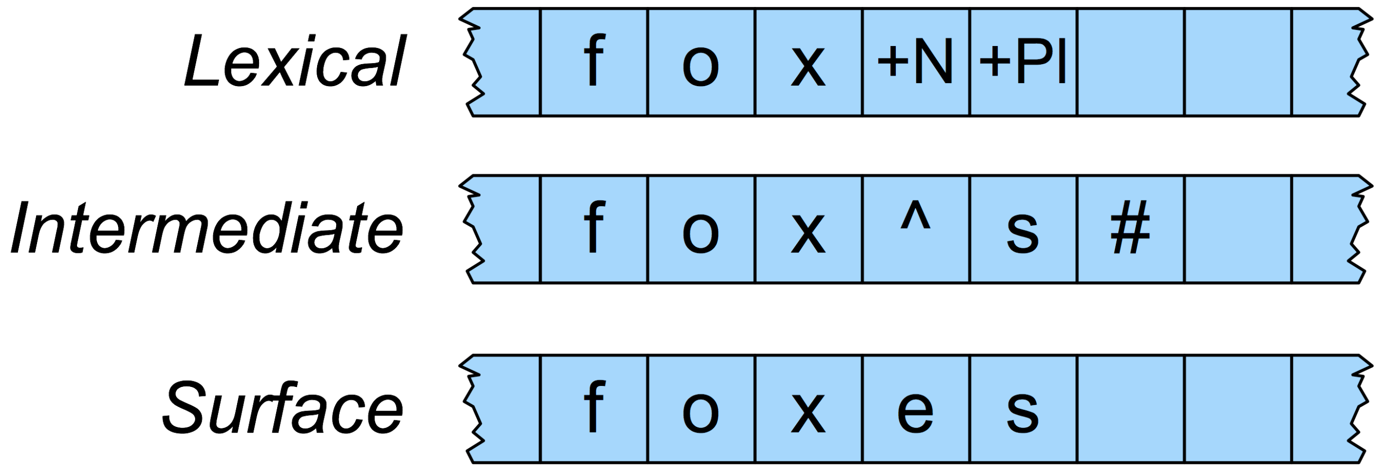
The Details

- Of course, its 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

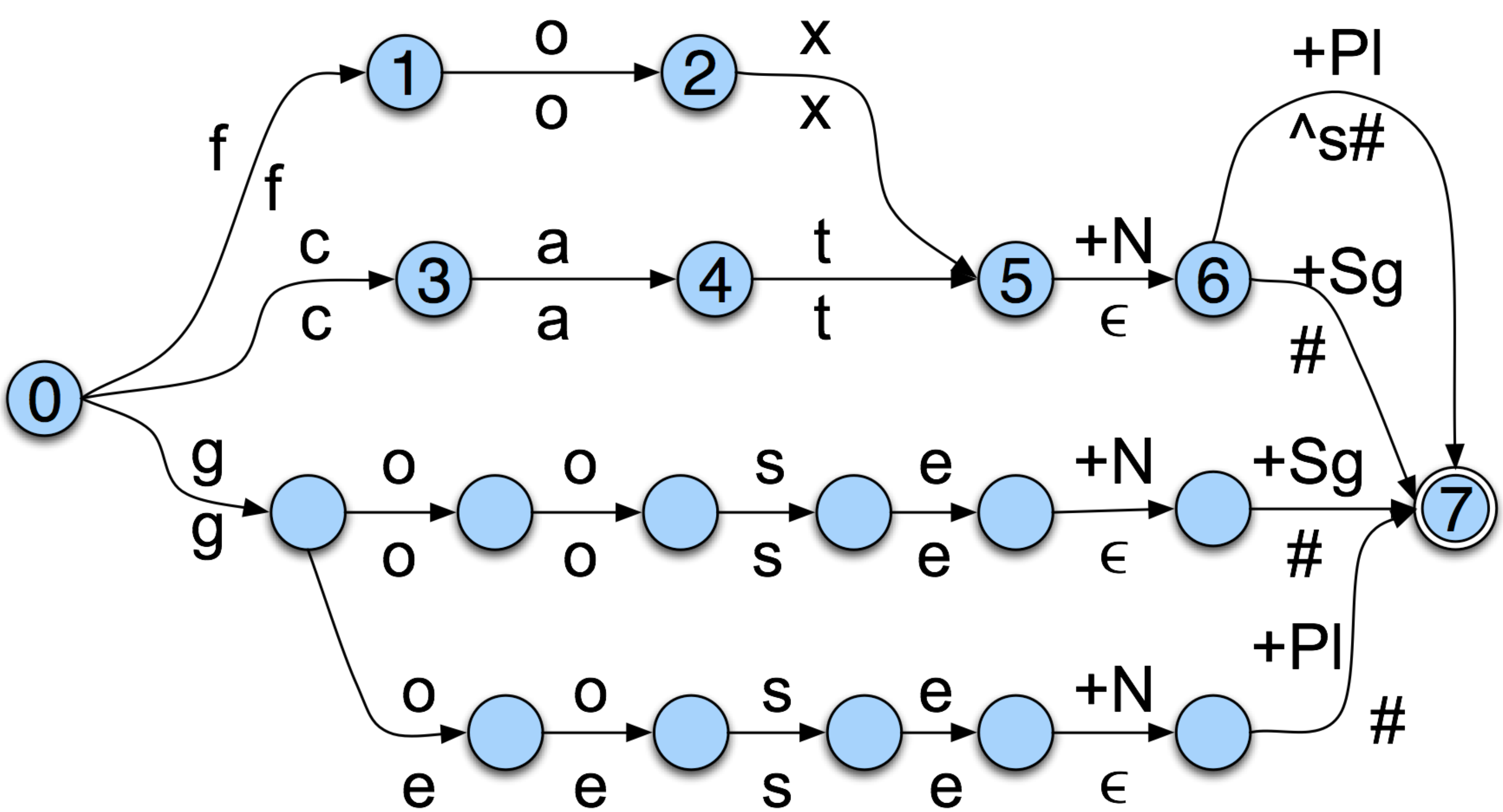
- To deal with these complications, we will add more tapes and use the output of one tape machine as the input to the next
- So to handle irregular spelling changes we'll add intermediate tapes with intermediate symbols

Multi-Level Tape Machines



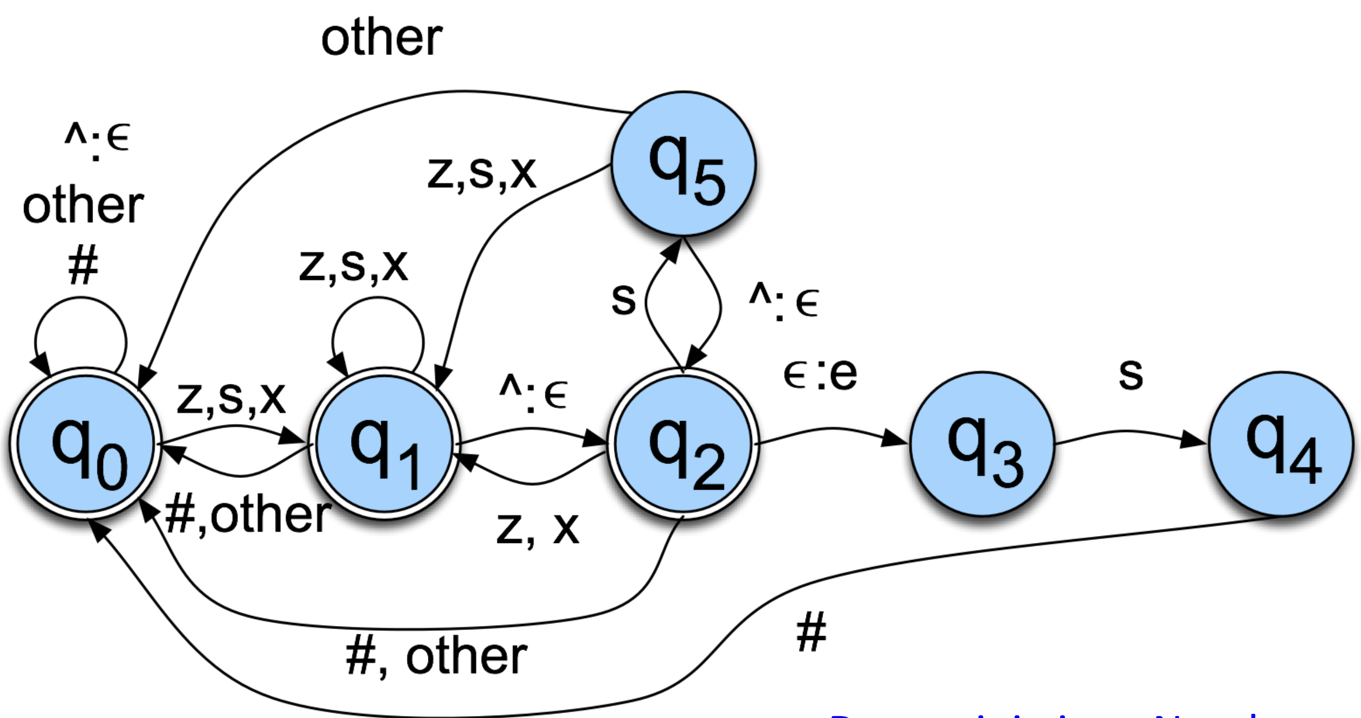
- We use one machine to transduce between the lexical and the intermediate level, and another to handle the spelling changes to the surface tape

Lexical to Intermediate Level



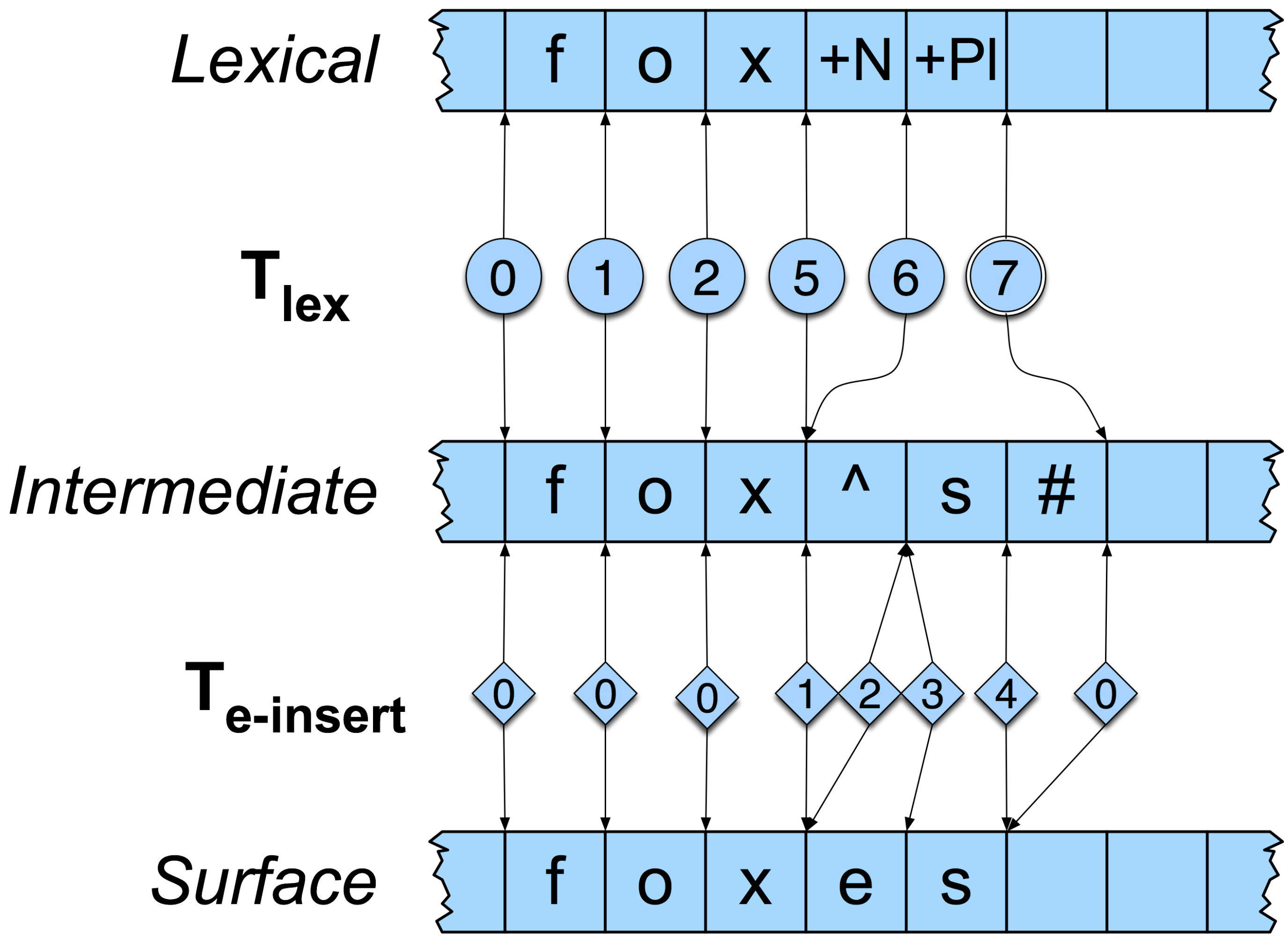
Intermediate to Surface

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



Deterministic or Nondeterministic?

Foxes



Lexical
f
o
x
+N
+Pl
T lex
0
1
2
5
6
7
Intermediate
f
o
x
^
s
#
T e-insert
0
0
0
1
2
3
4
0
Surface
f
o
x
e
s

Cascades

- This is an architecture 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

More about Words

- Tokenization: Can't just take words for granted
  - Finding the words
  - Sentence segmentation
  - Word segmentation
- Spell check and Edit Distance

Tokenization

- Segmenting words and sentences in running text
- Why not just periods and 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.''
- **Words like: cents. said, positive." Crazy?**

One can't segment on punctuation alone

- Word-internal punctuation
  - m.p.h
  - Ph.D.
  - AT&T
  - 01/02/06
  - Google.com
  - 555,500.50
- Expanding clitics
  - What're -> what are
  - I'm -> I am
- Multi-token words
  - New York
  - Rock 'n' roll

Sentence Segmentation

- !, ? 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, or machine-learning

Word Segmentation in Chinese

- Some languages don't have spaces
  - Chinese, Japanese, Thai, Khmer
- Chinese:
  - Words composed of characters
  - Characters are generally 1 syllable and 1 morpheme.
  - Average word is 2.4 characters long.
  - Standard segmentation algorithm:
    - Maximum Matching (also called Greedy)

Maximum Matching Word Segmentation

- Given a wordlist of Chinese, and a string.
  - 1) Start a pointer at the beginning of the string
  - 2) Find the longest word in dictionary that matches the string starting at pointer
  - 3) Move the pointer over the word in string
  - 4) Go to 2
- How about speech recognition?

English example (Palmer 00)

  - the table down there
  - thetabledownthere Theta bled own there
- Works astonishingly well in Chinese
- Far better than this English example suggests
- Modern algorithms better still: probabilistic segmentation

Spell-checking and Edit Distance

- Non-word error detection:
  - detecting “graffe”
- Non-word error correction:
  - figuring out that “graffe” should be “giraffe”
- Context-dependent error detection and correction:
  - Figuring out that “war and piece” should be peace

Non-word error detection

- Any word not in a dictionary
- Assume it's a spelling error
- Need a big dictionary!
- What to use?
  - FST dictionary!!
    - But what issues did we raise with earlier?
    - Can we use it for all kinds of morphology?

Isolated word error correction

- How do I fix "graffe"?
  - Search through all words:
    - graf
    - craft
    - grail
    - giraffe
  - Pick the one that's closest to graffe
  - What does "closest" mean?
  - We need a distance metric.
  - The simplest one: edit distance.
    - (More sophisticated probabilistic ones: noisy channel)

Edit Distance

- The minimum edit distance between two strings
- Is the minimum number of editing operations
  - Insertion
  - Deletion
  - Substitution
- Needed to transform one into the other

Minimum Edit Distance

```
I N T E * N T I O N

* E X E C U T I O N
d s s   i s
```

- If each operation has cost of 1
- Distance between these is 5
- If substitutions cost 2 (Levenshtein)
- Distance between these is 8

How to come up with the minimum?

- Try all possibilities

```
I N T E N T I O N

                    E X E C U T I O N
d d d d d d d d d   i i i i i i i i i = 18


       I N T E N T I O N

       E X E C U T I O N
       s s s s s           = 10
```

Distance Matrix Computation

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Insertion: Add 1

Substitution: Add 0 if same, 2 if diff

Deletion: Add 1

Distance Matrix

N	9										
O	8										
I	7										
T	6										
N	5										
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Min of 4,6,6

Min of 5,3,5

Min of 2,2,2

Min of 8,6,8

Distance Matrix

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Distance Matrix with shortest path

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Another example

R	I	G	H	T				
					R	I	T	E
D	D	D	D	D	I	I	I	I
1	1	1	1	1	1	1	1	1

Edit Distance

9

R	I	G	H	T
R	I	T	E	
		S	S	D
0	0	2	2	1

5

R	I	G	H	T	
R	I			T	E
		D	D		I
0	0	1	1	0	1

3

Minimum Edit Distance Algorithm

- Create Matrix
- Initialize 1 – length in LH column and bottom row
- For each cell
  - Take the minimum of:
    - Deletion: +1 from left cell
    - Insertion: +1 from cell below
    - Substitution: Diagonal +0 if same +2 if different
  - Keep track of where you came from

Example

- Minimum of:
  - 1+1 (left right)
  - 1+1 (bottom up)
  - 0+0 (diagonal)
- Minimum of:
  - 0+1 (left right)
  - 2+1 (bottom up)
  - 1+2 (diagonal)

T	5				
H	4				
G	3				
I	2				
R	1				
#	0	1	2	3	4
	#	R	I	T	E

Answer to Right-Rite

T	5				
H	4				
G	3				
I	2				
R	1	2, 0, 2			
#	0	1	2	3	4
	#	R	I	T	E

In each box X, Y, Z values are

- X: From left: Insert-add one from left box
- Y: Diagonal, Compare-0 if same, 2 if different
- Z: From below: Delete-add one from lower box

Minimum is highlighted in red with arrow to source

NOTE: All boxes will have arrows. I didn't show them all. Only one path back to root.

Answer to Right-Rite

T	5				
H	4				
G	3				
I	2	3, 3, 1	2, 0, 2		
R	1	2, 0, 2	1, 3, 3		
#	0	1	2	3	4
	#	R	I	T	E

In each box X, Y, Z values are

- X: From left: Insert-add one from left box
- Y: Diagonal, Compare-0 if same, 2 if different
- Z: From below: Delete-add one from lower box

Minimum is highlighted in red with arrow to source

NOTE: All boxes will have arrows. I didn't show them all. Only one path back to root.

Answer to Right-Rite

T	5	6, 6, 4	5, 5, 5	6, 2, 4	3, 5, 5
H	4	5, 5, 3	4, 4, 2	3, 3, 3	4, 4, 4
G	3	4, 4, 2	3, 3, 1	2, 2, 2	3, 3, 3
I	2	3, 3, 1	2, 0, 2	1, 3, 3	2, 4, 4
R	1	2, 0, 2	1, 3, 3	2, 4, 4	3, 5, 5
#	0	1	2	3	4
	#	R	I	T	E

In each box X, Y, Z values are

- X: From left: Insert-add one from left box
- Y: Diagonal, Compare-0 if same, 2 if different
- Z: From below: Delete-add one from lower box

Minimum is highlighted in red with arrow to source
NOTE: All boxes will have arrows.
I didn't show them all.
Only one path back to root.

Answer to Right-Rite

T	5	6, 6, 4	5, 5, 5	6, 2, 4	3, 5, 5
H	4	5, 5, 3	4, 4, 2	3, 3, 3	4, 4, 4
G	3	4, 4, 2	3, 3, 1	2, 2, 2	3, 3, 3
I	2	3, 3, 1	2, 0, 2	1, 3, 3	2, 4, 4
R	1	2, 0, 2	1, 3, 3	2, 4, 4	3, 5, 5
#	0	1	2	3	4
	#	R	I	T	E

In each box X, Y, Z values are

- X: From left: Insert-add one from left box
- Y: Diagonal, Compare-0 if same, 2 if different
- Z: From below: Delete-add one from lower box

Minimum is highlighted in red with arrow to source
NOTE: All boxes will have arrows.
I didn't show them all.
Only one path back to root.

Summary

- Minimum Edit Distance
- A "dynamic programming" algorithm
- We will see a probabilistic version of this called "Viterbi"