Foundations of Natural Language Processing
Lecture 11b
Morphology: Finite State Transducers

Alex Lascarides
(slides based on those of Shay Cohen)

School of informatics

Recap: The Problem of Morphological Parsing

- English has concatenative morphology. Words can be made up of a main stem plus one or more affixes carrying grammatical information.

Surface form:	cats	walking	smoothest
Lexical form:	cat+N+PL	walk+V+PresPart	smooth+Adj+Sup

- Morphological parsing is the problem of extracting the lexical form from the surface form. (For ASR, this includes identifying the word boundaries.)

- We should take account of:
  - Irregular forms (e.g. goose $\rightarrow$ geese)
  - Systematic rules (e.g. 'e' inserted before suffix 's' after s,x,z,ch,sh: fox $\rightarrow$ foxes, watch $\rightarrow$ watches)
  - Things that look like affixes but aren't (*proactive* vs. *protect*)

Today: Introduction to Finite State Transducers.

Why bother?

- Any NLP tasks involving grammatical parsing will typically involve morphology parsing as a prerequisite.

- Search engines: e.g. a search for 'fox' should return documents containing 'foxes', and vice versa.

- Even a humble task like spell checking can benefit
  e.g. *sleeped* $\rightarrow$ *slept*

Why an exhaustive word list isn't adequate

But why not just list all derived forms separately in our wordlist (e.g. *walk, walks, walked, walking*)?

- Might be OK for English, but not for a morphologically rich language — e.g. in Turkish, can pile up to 10 suffixes on a verb stem, leading to 40,000 possible forms for some verbs!

- Similarly for noun compounding in German, which is highly productive.

- Even for English, morphological parsing makes adding/learning new words easier (and it makes POS tagging easier).

- In speech processing, word breaks aren't known in advance.

How expressive is morphology?

- Morphemes are tacked together in a rather 'regular' way.
- But (in contrast to sentential syntax)
  there are no long range dependencies.
- So finite-state machines are a good way to model morphology.
  - No need for "unbounded memory".

Parsing and generation

Parsing here means going from the surface to the lexical form.

Eg. foxes $\rightarrow$ fox +N +PL.

Generation is the opposite process: fox +N +PL $\rightarrow$ foxes.
It's helpful to consider these two processes together.

Either way, it's often useful to proceed via an intermediate form, corresponding to an analysis in terms of morphemes (= minimal meaningful units) before orthographic rules are applied.

Surface form:	foxes
Intermediate form:	fox ^ s #
Lexical form:	fox +N +PL

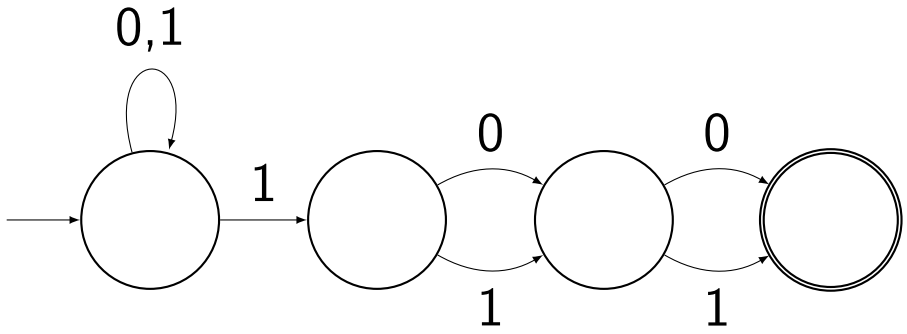
(^ means morpheme boundary, # means word boundary.)

NB. The translation between surface and intermediate form is exactly the same if 'foxes' is a 3rd person singular verb!

Nondeterministic Finite State Automatons (NFAs)

- A nondeterministic finite state automaton (NFA) is a finite state machine where a state can have more than one outgoing arc.

E.g., $(0|1)^*1(0|1)^2$ is captured by the following:

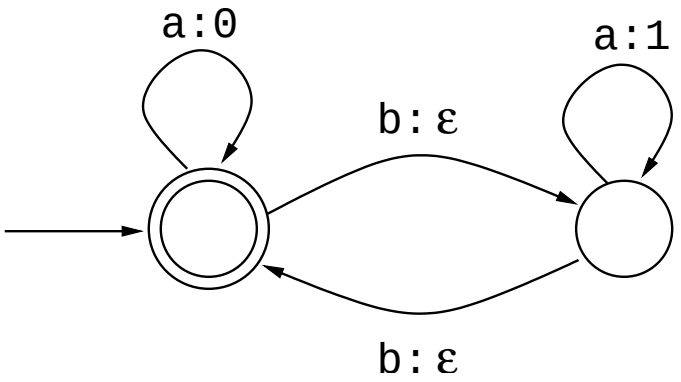


- An ϵ-NFA allows an input (in parsing) or output (in generation) defined by an arc to be the empty string.

Finite-state Transducers

- ϵ-NFAs over an input alphabet Σ can capture transitions that (optionally) produce *output* symbols (over a possibly different alphabet Π).

input alphabet $\Sigma = \{a, b\}$
output alphabet $\Pi = \{0, 1\}$:



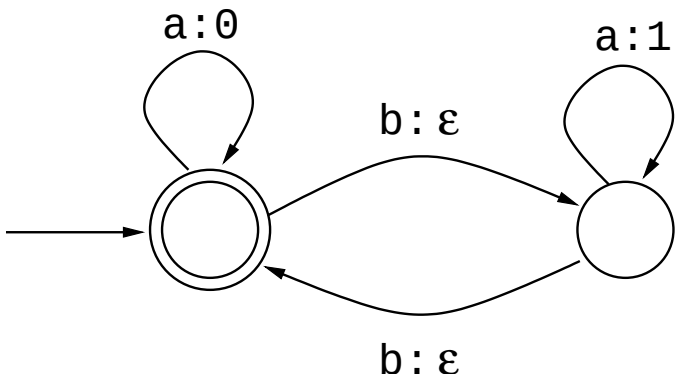
- Such a thing is called a finite state transducer.

- In effect, it specifies a (possibly multi-valued) translation from one regular language to another:

  – $abba \mapsto 00$, $aababa \mapsto 0010$ . . .

- More than one output can be possible (e.g., an arc labelled $a : 0, 1$)

Quick exercise



What output will this produce, given the input $aabaaabbbab$?

1. 001110
2. 001111
3. 0011101
4. More than one output is possible.

Summary

- Morphology parsing: surface form, intermediate form, lexical form.
- There is irregularity and ambiguity.
- But the structure of word depends only on that word!
- FSTs are therefore a useful way of capturing that structure:
  - They capture valid mappings from strings to strings.
  - **Next time:**
    FST for surface form$\mapsto$ intermediate form
    FST for intermediate form $\mapsto$ lexical form