October 9, 2003

Chapter 3.3–3.6

Finite State Morphological Parsing

Review: Mealy machines

- Q: a finite set of states $q_0, q_1, \ldots, q_N$
- Σ: a finite alphabet of complex symbols $i : o$ such that $i \in I$ and $o \in O$. $\Sigma \subseteq I \times O$. I and O may each include ϵ.
- q_0: the start state.
- F: the set of final states, $F \subseteq Q$.
- $\delta(q, i : o)$: the transition matrix.

A spelling rule FST

- FSTs for orthographic rules model context-sensitive rewrite rules, like (3.5):

$$\epsilon \rightarrow / \left\{ \begin{matrix} \text{x} \\ \text{s} \\ \text{z} \end{matrix} \right\} \hat{} \_ \text{ s\#}$$

- They must change the input only when called for (when their environment is satisfied).
- NB: With rule $\rightarrow$ FST compilers, there's no need to write an FST by hand... (but that doesn't mean there's no need to understand them!)

A spelling rule FST

- Note that their inputs have morpheme and word boundary symbols, while their outputs are standard orthography.
- What states does the FST visit in transducing *fox^s#* to *foxes*?
- Find other examples that illustrate each of the five states in the machine.

Building a larger machine

- Figure 3.16 cascades a lexicon FST (T_{lex}, Fig 3.11) with a pile of orthographic rule FSTs (such as $T_{e\text{-}insert}$, Fig 3.14). What does each do?
- How would you use 3.16 to parse a word?
- When would you want to?
- How would you use 3.16 to generate a word?
- When would you want to?
- Does the design allow for orthographic rules which feed each other?

Composition and intersection

- 3.16 cascades one machine that is the result of composing two others, and another machine that is the result of running a whole batch of machines in parallel.
- Intersection allows you to run machines in parallel:
  - Take the Cartesian product of states: $\{q_{ij} \mid q_i \in Q_1,\ q_j \in Q_2\}$
  - For each symbol $a : b$, if that symbol would take machine 1 to q_n and machine 2 to q_m, it takes the combined machine to q_{nm}.

Example of intersection

Machine 1			Machine 2		
	a:a	d:e		a:a	f:g
q_0:	1	-	q_0:	1	-
q_1	-	0	q_1	-	0

- Does the combined machine make use of the full Cartesian product of states?

Ambiguity

- Local v. global ambiguity
- Ambiguity in parsing v. generation
- How could you use an FST to give multiple outputs for one input?

What if you don't have a lexicon?

- Why might you not have a (big enough) lexicon?
- Why might you still want to do morphological parsing?
- The Porter stemmer (Appendix B) is a cascade of rewrite rules sensitive to orthographic properties of words, but without knowledge of any particular lexicon.
- Robust systems combine lexicon-based morphological parsing with techniques for handling unknown words. See in particular Morphological Analyzer ChaSen: http://chasen.aist-nara.ac.jp/

Human morphological parsing

- How much morphological analysis do humans do?
- Stanners et al. (1979) and Marslen-Wilson et al. (1994) find evidence for more analysis of inflectional morphology than derivational morphology. How can they tell?
- Speech errors also indicate morphological analysis. How?
- See also Pinker (1999) *Words and Rules*.