Algorithms for FSMs
(finite-state machines)

- Recognition of a string in a regular language: is a string accepted by an NFA?
- Conversion of regular expressions to NFAs
- Determinization: converting NFA to DFA
- Converting an NFA into a regular expression
- Other useful closure properties: union, concatenation, Kleene closure, intersection
The Lexicon

- Assume we process natural language input in the form of sequences of words
- The sequence is processed to compute some meaningful response or translation
- The role of the lexicon is to associate linguistic information with words of the language
- Many words are ambiguous: with more than one entry in the lexicon
- Information associated with a word in a lexicon is called a **lexical entry**

Lexicons stored as FSMs

- done, 4:- do:V+PP (*the turkey was done*); done:A (*a done deal*)
- doe, 3:- doe: N (*a deer, a female deer; U.S. Dept. of Energy*)
Lexicons stored as FSMs

<table>
<thead>
<tr>
<th>DICTIONARIES</th>
<th>Name</th>
<th>FDELAF French V.7</th>
<th>FDELACF Compound</th>
<th>GDELAF General</th>
<th>EDELAF English</th>
<th>IDELAF Italian</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nb of lines</td>
<td>672,000</td>
<td>156,000</td>
<td>828,000</td>
<td>145,000</td>
<td>612,000</td>
<td></td>
</tr>
<tr>
<td>Initial size</td>
<td>21.2 Mb</td>
<td>5.6 Mb</td>
<td>27.9 Mb</td>
<td>3.6 Mb</td>
<td>20 Mb</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AUTOMATA</th>
<th>Nb of states</th>
<th>84,600</th>
<th>322,800</th>
<th>389,050</th>
<th>48,770</th>
<th>78,320</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nb of transitions</td>
<td>181,010</td>
<td>466,570</td>
<td>633,320</td>
<td>101,970</td>
<td>177,350</td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>2.4 Mb</td>
<td>7.6 Mb</td>
<td>10 Mb</td>
<td>1.2 Mb</td>
<td>2.3 Mb</td>
<td></td>
</tr>
<tr>
<td>Compacted size</td>
<td>8.15 Kb</td>
<td>1.88 Mb</td>
<td>2.70 Mb</td>
<td>447 Kb</td>
<td>506 Kb</td>
<td></td>
</tr>
<tr>
<td>Nb of codes</td>
<td>11,200</td>
<td>1,750</td>
<td>14,950</td>
<td>1,590</td>
<td>11,190</td>
<td></td>
</tr>
<tr>
<td>Size of codes</td>
<td>358 Kb</td>
<td>445 Kb</td>
<td>403 Kb</td>
<td>25 Kb</td>
<td>257 Kb</td>
<td></td>
</tr>
<tr>
<td>Total final size</td>
<td>1.2 Mb</td>
<td>1.9 Mb</td>
<td>3.1 Mb</td>
<td>470 Kb</td>
<td>1.1 Mb</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TIME SPENT</th>
<th>Constr. (CRAY)</th>
<th>-</th>
<th>12h40</th>
<th>18h53</th>
<th>-</th>
<th>-</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constr. (HP)</td>
<td>12h30</td>
<td>-</td>
<td>-</td>
<td>4'55&quot;</td>
<td>12'30</td>
<td></td>
</tr>
<tr>
<td>Constr. (NLX4)</td>
<td>1h18</td>
<td>-</td>
<td>-</td>
<td>17</td>
<td>1h20</td>
<td></td>
</tr>
<tr>
<td>Look-up (HP)</td>
<td>90 w/ms</td>
<td>90 w/ms</td>
<td>90 w/ms</td>
<td>90 w/ms</td>
<td>90 w/ms</td>
<td></td>
</tr>
</tbody>
</table>

---

from (Mohri, 1995)

Simple NL grammars in FSMs

Numbers 1-99 in English
Morphology

• Words of a language have similarities between them and regularities within their structure
• A single word can be viewed as made up of smaller units, called morphemes
• free morphemes: act as words in isolation (e.g. think, permanent, local)
• bound morphemes: operate only as parts of other words (e.g. ize, ing, re)
• In English, bound morphemes are usually affixes; suffixes at the end of the word; prefixes at the front

Morphology

• The morpheme that forms the central meaning unit in a word is called the stem
• A word is made up of one or more morphemes:
  – think stem think
  – rethink prefix re, stem think
  – localize stem local, suffix ize
  – delexicalize prefix de, stem lexical, suffix ize
Inflectional Morphology

- Assume we want to represent the fact that English nouns form plurals (to avoid duplication in dictionaries)
- Very simple type of grammar development
- The plural:
  
  \[
  \text{cat+N+PL := cats} \quad \text{dog+N+PL := dogs, etc.}
  \]
- Exceptions:
  
  \[
  \text{fox+N+PL := foxes} \quad \text{mouse+N+PL := mice}
  \]
- Exercise: draw an FSM to represent the 4 words above and their plurals. Condition: only use a single transition for the plural suffix +s
FSM for simple morphology

- Problem: represents *foxes* as *foxs*
- Not a pure irregular case like *mouse/mice*
- e.g. *pass/passes, fax/faxes, suffix/suffixes*
- We will revisit this problem with a model more powerful than FSMs

FSM for simple morphology

- Lessons from grammar development:
  - Many regular cases (captured by a simple rule)
  - Some exceptional cases (*mice*) that are irregular and have no simple generative rule
  - Some cases that appear irregular but are rules that apply in the right context
- These observations are pervasive in grammar development (for other modules of natural language)
Derivational morphology

- Some suffixes can change the part of speech for a word, e.g. demon/N demonize/V
- We will write demonize/V as demon+ize/V to emphasize the suffix addition
- Notation, N: noun  V: verb  A: adjective
- Draw an NFA for the following data (transitions on morphemes):
  - demon/N  demon+ize/V  demon+ize+ation/N
  - demon+ize+able/A  demon+ize+er/N
  - formal/A  formal+ity/N  formal+ness/N
Derivational morphology

• Now modify the original NFA to accept the following additional strings:
  formal+ize/V  formal+ize+ation/N
  formal+ize+able/A  formal+ize+er/N
  demon+ize+able+ity/N
• Note that we assume that demon+ize+able+ity is mapped to demonizability
Morphology and lexicons

• Constructing a lexicon can benefit from the morphological analysis of words
• Especially in languages with productive morphology. For example, in Turkish:
uygarlaştıramadıklarımızdanmışsınızcasına
uygar + laş + tır + ama + dık + lar
civilized + “become” + “to cause” + “not able” + ppart + pl
+ ımiş + dan + mış + sıınız + casına
+ 1stpers possessive + “from/among” + past + 2nd person pl + adverbial

(behaving) as if you are among those whom we could not cause to become civilized

Morphology and lexicons

• Morphology is useful for lexicons:
  – Smaller size for the lexicons: various forms are computed by applying a rule rather than storing all possible forms
  – Ease of entering data: only the stem and category needs to be added and all possible forms are computed
  – New word forms (perhaps previously unknown to the lexicon builder) are produced. e.g. demonizability
  – Simpler and faster lookup (when using finite-state methods)
Summary

• Some aspects of morphology can be captured in a FSM, e.g. cat+N+PL := cats
• Other aspects are not elegantly captured. We have to create a sub-class for words like fox+N+PL := foxes and treat them differently
• Algorithms for FSMs can be used to recognize/generate/store these simple linguistic forms