September 30, 2003

Chapters 1, 2

Introduction, Regular Expressions

What is Computational Linguistics?

- Processing language on computers for HCI/other applications
  - Voice recognition
  - Speech synthesis
  - Natural language understanding
  - Information retrieval
  - Machine translation
  - …

What is Computational Linguistics?

- Linguistic research using computers
  - Searching large corpora for patterns/examples
  - Modeling human linguistic competence and performance
    - Computational psycholinguistics
    - Grammar engineering
  - …

Statistical v. symbolic methods

- Statistical methods involve *training* a system on a body of data so that it can predict the most probable result for any further new piece of data.
- Symbolic methods involve hand-coding linguistic knowledge and then applying that knowledge to tasks.
- Which methods are better depends on the application.
- Statistical methods provide *robustness*.
- Symbolic methods provide *precision*.
- Statistical and symbolic methods can be combined.

Administrivia

- Who is here? Please email ebender@u:

  name, major, year, linguistics background, CS/programming background, languages spoken, what you want to get out of this class

- Web page: `http://courses.washington.edu/ling472`

- Syllabus...

Letting the computer know who's boss

- Computer 'literacy' is really a combination of experience and attitude.
- Experience gives you the answers to many questions and a sense of what the possible space of answers is.
- The important attitude boils down to confidence in one's ability to find the answer to a new question.
- There are always new questions because:
  - the technology is always developing
  - there is too much out there for any one person to know it all

Letting the computer know who's boss

- Keep in mind:
  - It's always obvious once you know the answer.
  - All pieces of software were designed by some person or people with some functionality in mind.
  - Places to look for answers:
    - on-line documentation (UNIX: man, info, Windows: help)
    - product web sites (esp. discussion forums)
    - Google: websites, and especially newsgroups
    - off-line documentation (i.e., books)

Regular expressions and regular languages

- A language is a set of strings.
- A regular language is the set of strings represented by some regular expression.
- Equivalently, a regular language is the set of strings recognized (or generated) by a finite state automaton.

Formal definition of regular languages

- ϵ is the *empty string*
- ϕ is the *empty set*
- Σ is an alphabet (set of symbols)

Formal definition of regular languages

- The class of regular languages over is formally defined as:
  - ϕ is a regular language
  - $\forall a \in \Sigma \cup \epsilon$, $\{a\}$ is a regular language
  - If L_1 and L_2 are regular languages, then so are:

    (a) $L_1 \cdot L_2 = \{xy \mid x \in L_1, y \in L_2\}$ (concatenation)

    (b) $L_1 \cup L_2$ (union or disjunction)

    (c) L_1^* (Kleene closure)

(Jurafsky & Martin 2000:49)

Regular languages continued

- Not all sets of strings are regular languages.
- Other classes of languages are more or less inclusive than regular languages (e.g., context-free languages, context-sensitive languages).
- This is a taste of *formal language theory*, pioneered by Chomsky (1956).
- For more: CSE 531

Regular expressions are useful for search

- Return all lines/documents containing a string of the specified language.
- Allows people to search for something that is conceptually one thing but linguistically or orthographically varied, e.g.: [Uu]niversit(y—ies)
- (Also allows searches for two things at the same time, e.g.: cat—dog.)

Regular expression syntax

- concatenation, disjunction, Kleene * (see formal def)
- plus some syntactic sugar, i.e., other operations definable in terms of the above
- similar across Perl, Unix (including grep), and MS products
- we'll be using Perl

Perl regular expression syntax

- concatenation = concatenation

  `/abc/`, `/ab c/`

- disjunction: |, []

  `/[Aa][Bb][Cc]/`, `/cat|dog/`, `/kitt(y|ies)/`

- Kleene * (and + and ? and ranges)

  `/aa*/`, `/aa?a*/`, `/a+/`, `/a{1,}/`, `/a{1,10}/`

- Ranges, negation and wildcard:

  `/[A-Z][0-9][a-z][d-l][^jk][^t-z]./`, `/\d/`

Perl regular expression syntax

- Escape character:

  `/\[\.\|\{^/`

- Anchors: beginning and end of lines

  `/^Once upon a time/`, `/The end\.$/`

- Operator precedence hierarchy:

  parenthesis $\rangle\rangle$ counters $\rangle\rangle$ sequences and anchors $\rangle\rangle$ disjunction