CSCI 5832
Natural Language Processing

Lecture 2
Jim Martin

Today 1/17

- Wrap up last time
- Knowledge of language
- Ambiguity
- Models and algorithms
- Generative paradigm
- Finite-state methods

Course Material

- We'll be intermingling discussions of:
  - Linguistic topics
    - E.g. Morphology, syntax, discourse structure
  - Formal systems
    - E.g. Regular languages, context-free grammars
  - Applications
    - E.g. Machine translation, information extraction

Linguistics Topics

- Word-level processing
- Syntactic processing
- Lexical and compositional semantics
- Discourse processing

Topics: Techniques

- Finite-state methods
- Context-free methods
- Augmented grammars
  - Unification
  - Lambda calculus
- First order logic

- Probability models
- Supervised machine learning methods

Topics: Applications

- Small
  - Spelling correction
  - Hyphenation
- Medium
  - Word-sense disambiguation
  - Named entity recognition
  - Information retrieval
- Large
  - Question answering
  - Conversational agents
  - Machine translation

- Stand-alone
- Enabling applications
- Funding/Business plans

Just English?

- The examples in this class will for the most part be English
  - Only because it happens to be what I know.
- This leads to an (over?)-emphasis on certain topics (syntax) to the detriment of others (morphology) due to the properties of English
- We'll cover other languages primarily in the context of machine translation

Commercial World

- Lot's of exciting stuff going on…

Google Translate

Google Translate

Web Q/A

Summarization

- Current web-based Q/A is limited to returning simple fact-like (factoid) answers (names, dates, places, etc).
- Multi-document summarization can be used to address more complex kinds of questions.

  Circa 2002:

  What's going on with the Hubble?

NewsBlaster Example

The U.S. orbiter Columbia has touched down at the Kennedy Space Center after an 11-day mission to upgrade the Hubble observatory. The astronauts on Columbia gave the space telescope new solar wings, a better central power unit and the most advanced optical camera. The astronauts added an experimental refrigeration system that will revive a disabled infrared camera. "Unbelievable that we got everything we set out to do accomplished," shuttle commander Scott Altman said. Hubble is scheduled for one more servicing mission in 2004.

Weblog Analytics

- Textmining weblogs, discussion forums, message boards, user groups, and other forms of user generated media.
  - Product marketing information
  - Political opinion tracking
  - Social network analysis
  - Buzz analysis (what's hot, what topics are people talking about right now).

Web Analytics

umbria

Nielsen BuzzMetrics

cymfony harnessing influence 2.0™

collective intellect

Categories of Knowledge

- Phonology
- Morphology
- Syntax
- Semantics
- Pragmatics
- Discourse

Each kind of knowledge has associated with it an encapsulated set of processes that make use of it.

Interfaces are defined that allow the various levels to communicate.

This usually leads to a pipeline architecture.

Ambiguity

- *I made her duck*

Ambiguity

- *I made her duck*

- Sources
  - Lexical (syntactic)
    - Part of speech
    - Subcat
  - Lexical (semantic)
  - Syntactic
    - Different parses

Dealing with Ambiguity

- Four possible approaches:
  1. Tightly coupled interaction among processing levels; knowledge from other levels can help decide among choices at ambiguous levels.
  2. Pipeline processing that ignores ambiguity as it occurs and hopes that other levels can eliminate incorrect structures.

Dealing with Ambiguity

  3. Probabilistic approaches based on making the most likely choices
  4. Don't do anything, maybe it won't matter
     - *We'll leave when the duck is ready to eat.*
     - *The duck is ready to eat now.*
     - Does the ambiguity matter?

Models and Algorithms

- By models I mean the formalisms that are used to capture the various kinds of linguistic knowledge we need.
- Algorithms are then used to manipulate the knowledge representations needed to tackle the task at hand.

Models

- State machines
- Rule-based approaches
- Logical formalisms
- Probabilistic models

Algorithms

- Many of the algorithms that we'll study will turn out to be transducers; algorithms that take one kind of structure as input and output another.
- Unfortunately, ambiguity makes this process difficult. This leads us to employ algorithms that are designed to handle ambiguity of various kinds

Paradigms

- In particular..
  - State-space search
    - To manage the problem of making choices during processing when we lack the information needed to make the right choice
  - Dynamic programming
    - To avoid having to redo work during the course of a state-space search
      - CKY, Earley, Minimum Edit Distance, Viterbi, Baum-Welch
  - Classifiers
    - Machine learning based classifiers that are trained to make decisions based on features extracted from the local context

State Space Search

- States represent pairings of partially processed inputs with partially constructed representations.
- Goals are inputs paired with completed representations that satisfy some criteria.
- As with most interesting problems the spaces are normally too large to exhaustively explore.
  - We need heuristics to guide the search
  - Criteria to trim the space

Dynamic Programming

- Don't do the same work over and over.
- Avoid this by building and making use of solutions to sub-problems that must be invariant across all parts of the space.

Administrative Stuff

- Mailing list
  - If you're registered you're on it with your CU account
  - I sent out mail this morning. Check to see if you've received it
- The textbook is now in the bookstore

First Assignment

- Two parts
  1. Answer the following question:
     - *How many words do you know?*
  2. Write a python program that takes a newspaper article (plain text that I will provide) and returns the number of:
     - *Words*
     - *Sentences*
     - *Paragraphs*

First Assignment Details

- For the first part I want…
  - An actual number and a explanation of how you arrived at the answer
  - Hardcopy. Bring to class.
- For the second part, email me your code and your answers to the test text that I will send out shortly before the HW is due.

First Assignment

- In doing this assignment you should think ahead… *having access* to the words, sentences and paragraphs will be useful in future assignments.

Getting Going

- The next two lectures will cover material from Chapters 2 and 3
  - Finite state automata
  - Finite state transducers
  - English morphology

Regular Expressions and Text Searching

- Everybody does it
  - Emacs, vi, perl, grep, etc..
- Regular expressions are a compact textual representation of a set of strings representing a language.

Example

- Find me all instances of the word "the" in a text.
  - `/the/`
  - `/[tT]he/`
  - `/\b[tT]he\b/`

Errors

- The process we just went through was based on two fixing kinds of errors
  - Matching strings that we should not have matched (there, then, other)
    - False positives (Type I)
  - Not matching things that we should have matched (The)
    - False negatives (Type II)

Errors

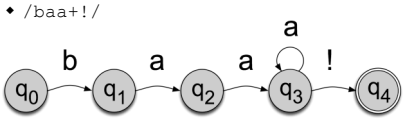
- We'll be telling the same story for many tasks, all semester. Reducing the error rate for an application often involves two antagonistic efforts:
  - Increasing accuracy, or precision, (minimizing false positives)
  - Increasing coverage, or recall, (minimizing false negatives).

Finite State Automata

- Regular expressions can be viewed as a textual way of specifying the structure of finite-state automata.
- FSAs and their probabilistic relatives are at the core of what we'll be doing all semester.
- They also conveniently (?) correspond to exactly what linguists say we need for morphology and parts of syntax.
  - *Coincidence*?

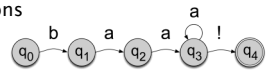
FSAs as Graphs

- Let's start with the sheep language from the text
  - `/baa+!/`



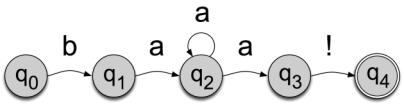
Sheep FSA

- We can say the following things about this machine
  - It has 5 states
  - b, a, and ! are in its alphabet
  - q0 is the start state
  - q4 is an accept state
  - It has 5 transitions



But note

- There are other machines that correspond to this same language



- More on this one later

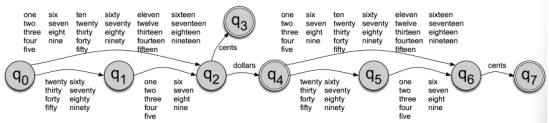
More Formally

- You can specify an FSA by enumerating the following things.
  - The set of states: Q
  - A finite alphabet: Σ
  - A start state
  - A set of accept/final states
  - A transition function that maps QxΣ to Q

About Alphabets

- Don't take that word to narrowly; it just means we need a finite set of symbols in the input.
- These symbols can and will stand for bigger objects that can have internal structure.

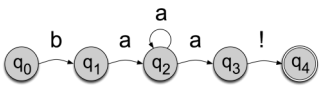
Dollars and Cents



Yet Another View

- The guts of FSAs are ultimately represented as tables

State	b	a	!	e
0	1			
1		2		
2		2,3		
3			4	
4				

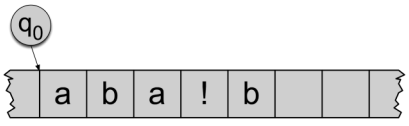


Recognition

- Recognition is the process of determining if a string should be accepted by a machine
- Or… it's the process of determining if a string is in the language we're defining with the machine
- Or… it's the process of determining if a regular expression matches a string
- Those all amount the same thing in the end

Recognition

- Traditionally, (Turing's idea) this process is depicted with a tape.



Recognition

- Simply a process of starting in the start state
- Examining the current input
- Consulting the table
- Going to a new state and updating the tape pointer.
- Until you run out of tape.

D-Recognize

```
function D-RECOGNIZE(tape, machine) returns accept or reject

  index ← Beginning of tape
  current-state ← Initial state of machine
  loop
    if End of input has been reached then
      if current-state is an accept state then
        return accept
      else
        return reject
    elsif transition-table[current-state,tape[index]] is empty then
      return reject
    else
      current-state ← transition-table[current-state,tape[index]]
      index ← index + 1
  end
```


Key Points

- Deterministic means that at each point in processing there is always one unique thing to do (no choices).
- D-recognize is a simple table-driven interpreter
- The algorithm is universal for all unambiguous regular languages.
  - To change the machine, you just change the table.

Key Points

- Crudely therefore… matching strings with regular expressions (ala Perl, grep, etc.) is a matter of
  - translating the regular expression into a machine (a table) and
  - passing the table to an interpreter

Recognition as Search

- You can view this algorithm as a trivial kind of state-space search.
- States are pairings of tape positions and state numbers.
- Operators are compiled into the table
- Goal state is a pairing with the end of tape position and a final accept state
- Its trivial because?

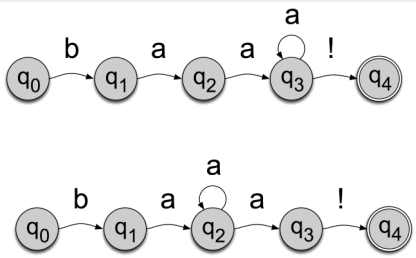
Generative Formalisms

- Formal Languages are sets of strings composed of symbols from a finite set of symbols.
- Finite-state automata define formal languages (without having to enumerate all the strings in the language)
- The term Generative is based on the view that you can run the machine as a generator to get strings from the language.

Generative Formalisms

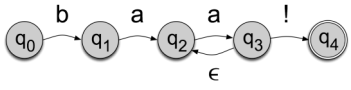
- FSAs can be viewed from two perspectives:
  - Acceptors that can tell you if a string is in the language
  - Generators to produce all and only the strings in the language

Non-Determinism



Non-Determinism cont.

- Yet another technique
  - Epsilon transitions
  - Key point: these transitions do not examine or advance the tape during recognition



Equivalence

- Non-deterministic machines can be converted to deterministic ones with a fairly simple construction
- That means that they have the same power; non-deterministic machines are not more powerful than deterministic ones in terms of the languages they can accept

ND Recognition

- Two basic approaches (used in all major implementations of Regular Expressions)
  1. Either take a ND machine and convert it to a D machine and then do recognition with that.
  2. Or explicitly manage the process of recognition as a state-space search (leaving the machine as is).

Implementations

Program	(Original) Author	Version	Regex Engine
awk	Aho, Weinberger, Kernighan	*generic*	DFA
new awk	Brian Kernighan	*generic*	DFA
GNU *awk*	Arnold Robbins	*recent*	Mostly DFA, some NFA
MKS *awk*	Mortice Kern Systems		POSIX NFA
mawk	Mike Brennan	*all*	POSIX NFA
egrep	Alfred Aho	*generic*	DFA
MKS *egrep*	Mortice Kern Systems		POSIX NFA
GNU Emacs	Richard Stallman	*all*	Trad. NFA (POSIX NFA available)
Expect	Don Libes	*all*	Traditional NFA
expr	Dick Haight	*generic*	Traditional NFA
grep	Ken Thompson	*generic*	Traditional NFA
GNU *grep*	Mike Haertel	Version 2.0	Mostly DFA, but some NFA
GNU *find*	GNU		Traditional NFA
lex	Mike Lesk	*generic*	DFA
flex	Vern Paxson	*all*	DFA
lex	Mortice Kern Systems		POSIX NFA
more	Eric Schienbrood	*generic*	Traditional NFA
less	Mark Nudelman		Variable (usually Trad. NFA)
Perl	Larry Wall	*all*	Traditional NFA
Python	Guido van Rossum	*all*	Traditional NFA
sed	Lee McMahon	*generic*	Traditional NFA
Tcl	John Ousterhout	*all*	Traditional NFA
vi	Bill Joy	*generic*	Traditional NFA

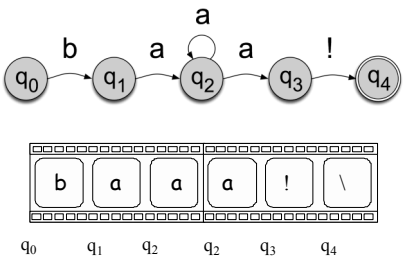
Non-Deterministic Recognition: Search

- In a ND FSA there exists at least one path through the machine for a string that is in the language defined by the machine.
- But not all paths directed through the machine for an accept string lead to an accept state.
- No paths through the machine lead to an accept state for a string not in the language.

Non-Deterministic Recognition

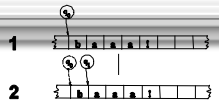
- So success in a non-deterministic recognition occurs when a path is found through the machine that ends in an accept.
- Failure occurs when all of the possible paths lead to failure.

Example

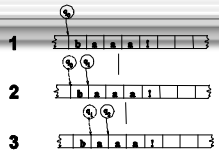


Example

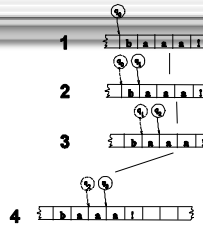
Example



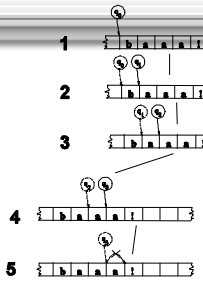
Example



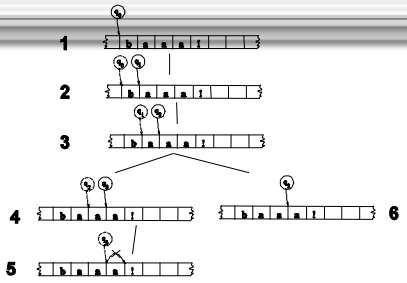
Example



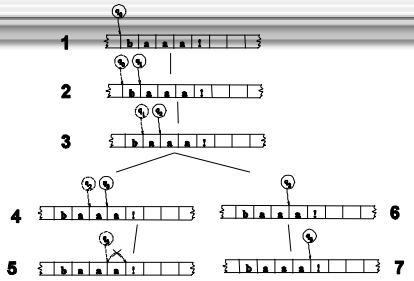
Example



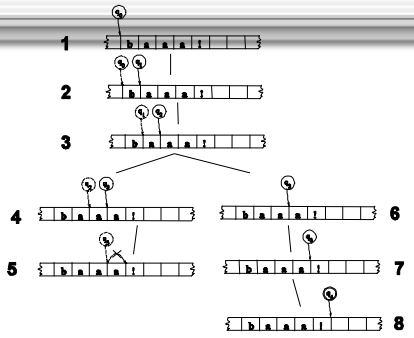
Example



Example



Example



Key Points

- States in the search space are pairings of tape positions and states in the machine.
- By keeping track of as yet unexplored states, a recognizer can systematically explore all the paths through the machine given an input.

Next Time

- Finish reading Chapter 2, start on 3.
  - *Make sure you have the book*
- Make sure you have access to Python