

Welcome to Ling 473

- Dates of Instruction
- Instructor
- Prerequisites
- UW Degree and Certificate Programs in Comp. Ling.
- Program Faculty
- Getting Set Up
- Class Tools
- Software
- Programming Languages
- Assignments
- Grading

Calendar

Class Meetings: Tues., Thurs. 4:30 – 6:20 pm
 July 21 – September 8, 2016

Three lecture attendance options:

- In person
- Live broadcast via AdobeConnect
- Recordings posted later after each class



Since viewing class recordings does not allow for interactive participation, this attendance option should be used sparingly

Instructor

Glenn Slayden

gslayden@uw.edu

B.S. Econ. UPenn/Wharton

B.Mus Cornish College of the Arts

UW CLMS “Array TFS Storage”

Ph.C., UW Comp. Ling. PhD track

Experience:

7 years SDE Microsoft

Microsoft Research – Bing Translator ([Thai integration](#))

Research Interests:

Thai language: precision grammar, lexicography, and online language learning; analytical MT: parsing, generation, and semantic transfer

Website:

<http://www.thai-language.com>



Ling 473 Prerequisites

CSE 326 or 373: Data Structures

Abstract data types and their implementations as data structures. Efficient use of algorithms employing these data structures; asymptotic analyses. Dictionaries: balanced search trees, hashing. Priority queues: heaps. Disjoint sets with union, find. Graph algorithms: shortest path, minimum spanning tree, topological sort, search. Sorting.

STAT 390 or 391: Probability and Statistics for Computer Science


Fundamentals of probability and statistics from the perspective of the computer scientist. Random variables, distributions and densities, conditional probability, independence. Maximum likelihood, density estimation, Markov chains, classification. Applications in computer science.

(or equivalents)

 Ling 473 is required for the NLT Certificate Program

 CLMS Program: determination by placement test

UW NLT: Certificate in Natural Language Technology

- <http://www.pce.uw.edu/certificates/natural-language-technology.html>
- 11 credits
- Ling 473  *you are here*
- 570 Shallow processing

Techniques and algorithms for associating relatively surface-level structures and information with natural language corpora, including POS tagging, morphological analysis, preprocessing/segmentation named-entity recognition, chunk parsing, and word-sense disambiguation. Examines linguistic resources that can be leveraged for these tasks (e.g., WordNet).
- 571 Deep processing

Algorithms for associating deep or elaborated linguistic structures with naturally occurring linguistic data (parsing/semantics/discourse), and to produce natural language strings from input semantic representations (generation).
- Consider GNM status if you think you might convert to CLMA
<http://www.grad.washington.edu/admissions/gnm.shtml>

CLMS: Professional Ma.Sci. in Computational Linguistics

- <http://www.compling.washington.edu/compling/>
- 43 credits
- Ling 450 Phonetics
 - Be sure to take 550 instead if you think you might convert to UW Ph.D.
- Ling 566 Syntax for computational linguistics
 - Introduction to syntactic analysis and concepts with emphasis on the formally precise encoding in linguistic hypotheses and the design of grammars that can be scaled to practical applications. Coursework progressively builds up a consistent grammar for a fragment of English, while also considering data and phenomena from other languages
- Ling 570, 571, 572, 573 : core sequence
- 3 electives (1 comp. ling., 1 ling., and 1 related area)
- Thesis or internship/report option (10 credits)

UW Ph.D in Computational Linguistics

- A newly designed Ph.D track
- <https://linguistics.washington.edu/phd-linguistics#compling>
- This is a full Ph.D in linguistics from the UW

UW Computational Linguistics Faculty

- Emily Bender, Faculty Director
 - computational syntax, computational semantics, linguistic typology
- Fei Xia
 - statistical and hybrid methods, grammar, machine learning, automatic document understanding
- Gina-Anne Levow
 - speech and intonation processing, prosody, human-computer dialogue

UW Linguistics

- <https://linguistics.washington.edu/>
- Richard Wright, Department Chair
- Michael Furr, Administrator
- Joyce Parvi, Office Manager
 phoneme@uw.edu
 Padelford Hall, A210
- T.B.D., Systems Administrator
 brodbd@uw.edu

Getting Set Up

- UW NetID
- Husky Card
- Computational Linguistics Cluster Account
- Keypad access to “Treehouse” lab in Guggenheim
- Recommended textbooks

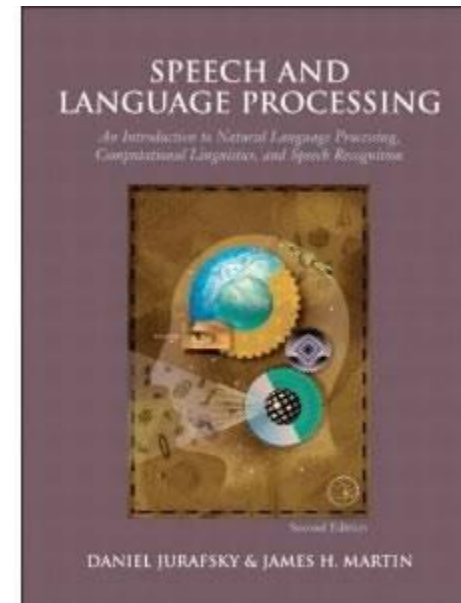
Daniel Jurafsky and James H. Martin. (2008) *Speech and Language Processing (2nd edition)*. New Jersey: Prentice-Hall.

Christopher D. Manning and Hinrich Schütze. (1999) *Foundations of Statistical Natural Language Processing*. Cambridge: MIT Press.



Both of these textbooks are required for CLMS and NLT students when taking Ling 570/571. If doing so in the fall, purchase now.

Daniel Jurafsky and James H. Martin. (2008) *Speech and Language Processing (2nd edition)*. New Jersey: Prentice-Hall.



Class Tools

- Catalyst Tools
 - GoPost
 - <https://catalyst.uw.edu/gopost/board/gslayden/42819>
 - CollectIt
 - <https://catalyst.uw.edu/collectit/dropbox/gslayden/38629>
 - GradeBook
 - <https://catalyst.uw.edu/gradebook/gslayden/98769>
- Treehouse wiki
 - <http://depts.washington.edu/uwcl/twiki/bin/view.cgi/Main/WebHome>
- CLMS Survival Guide:
 - <http://depts.washington.edu/uwcl/twiki/bin/view.cgi/Main/CLMASurvivalGuide>
- Computational Linguistics Linux cluster
 - Interactive head nodes: patas.ling.washington.edu; dryas.ling.washington.edu
 - About 50 processors on ~20 machines for batch jobs
 - Files for certain assignments may be found in /opt/dropbox
 - Licensed corpora database: <https://vervet.ling.washington.edu/db/index.php>
 - Condor batch submission system; Network File System

Software

- L^AT_EX
 - Consider if you're continuing in MA or Ph.D
 - Can create PS or PDF files
 - On Windows: Install MiKTeX and Texmaker
- Ability to create PDF files
 - Adobe Acrobat (academic pricing at UW Bookstore)
 - Open Office?
 - PDF creator
 - <http://depts.washington.edu/uwcl/twiki/bin/view.cgi/Main/WordToPDF>
- SSH Terminal Programs available from UWICK
 - <http://www.washington.edu/uware/uwick/>
 - WinSCP
 - SSH Tectia
 - Tera Term
 - Putty

Programming Languages

- You may use any programming language that's available on the cluster:

C

C++

C# (mono)

Java

Python

Perl



Of these, I am most familiar with C#, C, and C++. Other languages are fine to use, but the feedback I provide on your code may be less detailed. Assignment solutions will be provided in C#.

- This is not a basic programming class. We will not cover how to create, edit, compile, and run programs.

Written Assignments

- Probability and statistics problems
- Preferred submission format: PDF
- Please do not write homework by hand
 - Now is the time to learn LaTeX or an equation layout tool

Programming Projects

- Code must run on UW Comp. Ling. Cluster
 - or you will receive no credit. This is because some projects may reference licensed corpora which you may not copy
 - I won't spend time figuring out why your code doesn't run.
 - You can develop on a home machine, but make sure you test thoroughly on the cluster
- Please follow instructions
- TAR and submit to CollectIt:
 - all required source code and files (except public files)
 - point to public files where possible
 - output file captured from stdout
 - prose description (write-up) of your work

Grading

- 5 Written Assignments: 30%
- 7 Programming Projects: 65%
- Class and GoPost Participation: 5%

- Policies:

<http://courses.washington.edu/ling473/student-resources.html>

Computational Linguistics

- A Quick Survey of the field
- Why is language hard?
- A resource: The Penn Treebank (PTB)
- Finding patterns in text: Regular Expressions
- Homework 1:
- Project 1: PTB and RegEx

What is Computational Linguistics?

Applying quantitative techniques to the analysis and processing of human (“natural”) languages.

- Naturally, computers are well-suited to this sort of Natural Language Processing (NLP)
- Cross-disciplinary
 - Linguistics
 - Computer science
 - Mathematics
 - Electrical Engineering
- Computational linguists come from all of these backgrounds

Linguistics

- Nearly all research areas in linguistics can benefit from computational techniques:
 - Phonetics
 - Phonology
 - Morphology
 - Syntax
 - Semantics
 - Pragmatics
 - Discourse Analysis
 - Information Structure
 - Typology

Computer Science

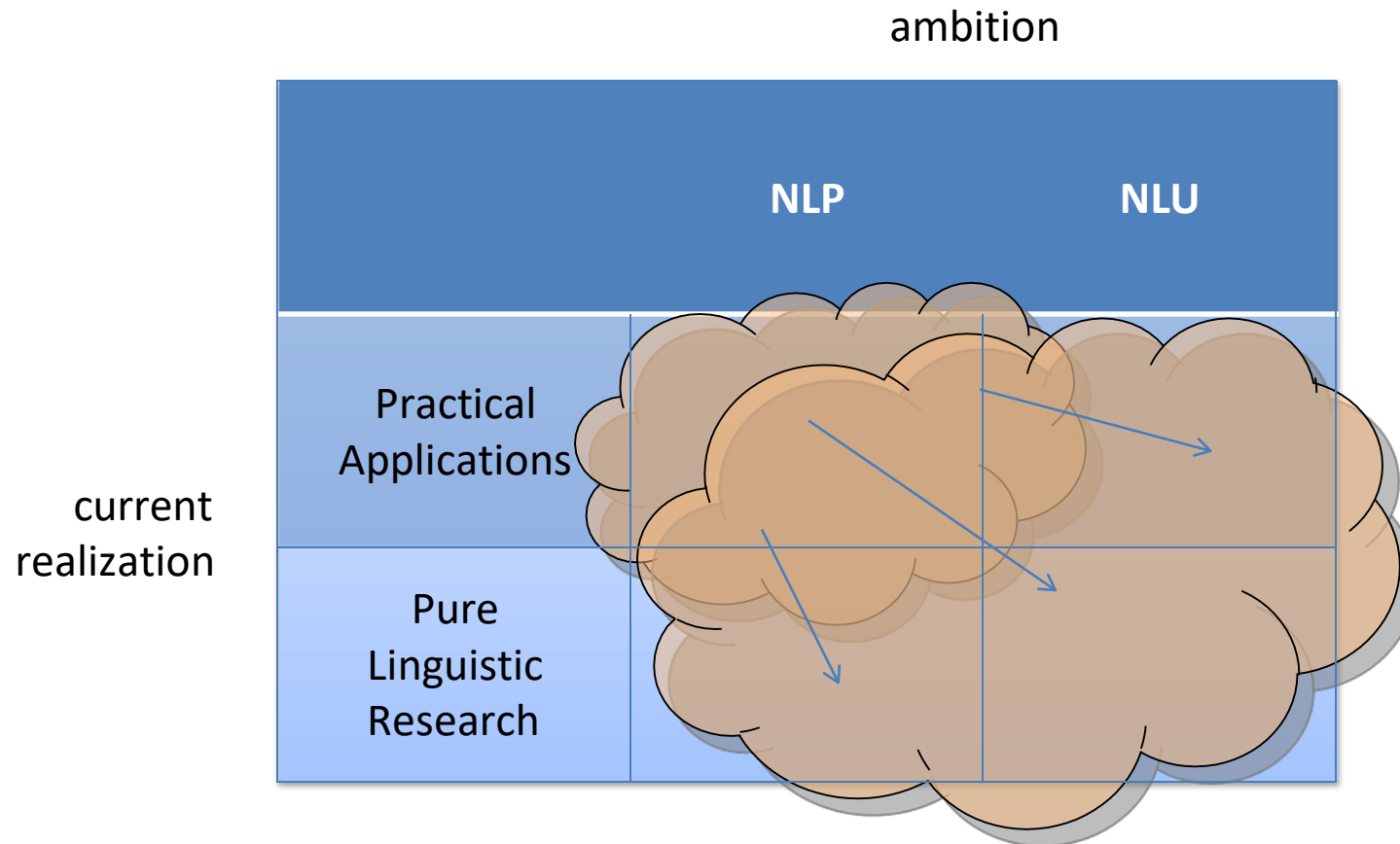
- All fundamental CS data structures and principles apply: Landau (“Big-O”) notation
- Large scale text processing; RegEx; strings; character encoding; data conversion
- Databases; high-throughput computing
- Specialized data structures and techniques
 - String hashes
 - Trie
 - Dynamic programming

Mathematics

- Probability
 - Modeling the occurrence(s) of events
- Statistics
 - Application of probabilistic models
- Set Theory
 - Intersection, Union, Exclusion
- Logic
 - Boolean logic
 - First order logic (predicate calculus)
 - Markov (probabilistic) logic, entailment
- Information Theory
 - Entropy
 - Shannon channels
- Advanced Math
 - Kernel functions (for Support Vector Machines...)
 - Matrix decomposition (i.e. Singular Value Decomposition...)

- Two ambitions are at different stages of realization
 - Natural Language Processing (NLP)
 - Natural Language Understanding (NLU)

State of the Art



Natural Language Understanding

- Still largely in the research stage
- Hybrid of linguistically motivated analytical (“rule-based”) systems with sensible application of stochastic methods seems necessary
- Deep processing applications continue to mature
 - ERG: English Resource Grammar (Flickinger 2002)
 - <http://www.delph-in.net/erg/>
 - World knowledge, semantics, intricate grammatical knowledge

NLP / NLU Subfields

- IE: Information Extraction
- IR: Information Retrieval
- MT: Machine Translation
- Automatic Document Summarization
- QA: Question Answering
- ASR: Automatic Speech Recognition
- NLG: Natural Language Generation
- Speech Synthesis
- CALL: Computer-Assisted Language Learning
- Alternative Input Methods

NLP / NLU Subtasks

- Tokenization (word breaking)
- Sentence Breaking
- Morphological Analysis
 - POS tagging
 - Stemming
- NER: Named Entity Recognition
- WSD: Word Sense Disambiguation
- Anaphora and reference resolution
- Parsing and generation
- Dialogue management and discourse analysis
- Clustering
- Classification
- Treebanking and Corpora curatorship
- ...

NLP/NLU Approaches

- Analytical “rule-based”
 - Intuit a set of rules
 - Implement them
 - Evaluate
- Statistical
 - Train a statistical model on a large set of observations
 - Use the model to make predictions about unseen inputs

Analytical (Rule-Based) Techniques

Intuition: In human brains, language operates by a set of rules. Let's infer these and implement them in a computer.

Q: Any problems with this?

1. We don't know how language operates in human brains
2. How will we infer these rules? What form will they have?
3. Will these putative rules be amenable to procedural implementation?
4. How will we test and evaluate our progress?

Analytical MT: a case study

Machine Translation (MT) was the first goal: naïve ebullience!

When I look at an article in Russian, I say: “This is really written in English, but it has been coded in some strange symbols.”

Weaver 1947

Analytical MT: a case study

Another 19 Years Later: wet-blanket pessimism

There has been no machine translation of general scientific text, and none is in immediate prospect... After 8 years of work, the project] had to resort to post-editing [which] took slightly longer to do and was more expensive than conventional human translation.”

ALPAC Report 1966



Recommendation of the ALPAC Report: suspend government funding for Machine Translation

Analytical MT: a case study

41 Years Later: realistic progress

Progress on combining rule-based and data-driven approaches to MT will depend on a sustained stream of state-of-the-art, MT-oriented linguistics research... Despite frequent cycles of overly high hopes and subsequent disillusionment, [MT] is the type of application that may demand knowledge-heavy, 'deep' approaches to NLP for its ultimate, long-term success.

Oepen et al. 2007

Why is language so hard?

- As a simplification, let's just consider text
(speech processing has a separate set of problems)
- Text representation of language

I saw a man with a telescope.

- This is a sequence of symbols
- This is a “string” of characters or Unicode code points
- This is an ordered collection of “words”
- This is a “sentence”
- This is a “surface” representation of a speech act
- This is a surface representation of a semantic proposition

Words

Isa wa manwi thate lesco pe.

Observation: it appears that, *in this language*, one role of the space character is to partition symbols (letters) into units called “words.”



Note: we always try to be aware of, and explicitly state, any assumptions that may not be cross-linguistically valid

Words

It is not always the case that identifying “words” is straightforward:

ผมเห็นผู้ชาย กับ โทรทรรศน์				
ผม	เห็น	ผู้ชาย	กับ	โทรทรรศน์
p ^h ǒm	he̯n	p ^h ûi: tɕ ^h a:j	kàp	t ^h o: rá t ^h át
1-sg	see	man	with	telescope
“I saw (a) man (who was) with a telescope.”				



The International Phonetic Alphabet (IPA), is the standard form of phonetic transcription.



“person-male” : 1 word or 2?



This type of formatting for linguistic examples is called “Interlinear Glossed Text,” or IGT

Word Classes

- Can we identify a closed set of word classes to generalize about them?
- These are called Parts of Speech.
- In computational linguistics: POS tags
- Closed and open classes

Closed Word Classes

- A closed class
 - Has a limited number of members
 - Is generally not open to new member production
- Closed word classes
 - Conjunction
 - Determiner
 - Pronoun
 - Auxiliary verb
 - ...and others

Open Word Classes

- An open class:
 - Has a large number of members
 - May accept new members over time
 - May allow new members to be heuristically generated
- Open word classes: may allow compounding, inflecting, or productive morphology
 - Noun
 - Verb
 - Adjective
 - Adverb
 - ...and others

Sentences

**the carnival. I saw a man with a*



Observation: it appears that, in this language, one role of the period is to partition words into sentences.

Ok, let's dismiss those issues too. We'll assume we're given the text of one sentence, unambiguously composed of words. Good enough?

Convention for linguistic examples

- Sentences marked ungrammatical are marked with an asterisk
**Sentence is ungrammatical this.*
- Sentences judged marginal are marked with a question mark
?We gave candies to the children because they were sweet.
- Grammatical sentences can be semantic nonsense
Colorless green ideas sleep furiously.
- Grammatical sentences which don't convey the intended meaning are marked with a hash
#It's raining outside but I don't believe that it's raining.

Test the model

**Man a with saw telescope l a.*

Observation: the ordering of words appears to be important in this language.

English is generally Subject-Verb-Object (SVO), which is less common among world languages than SOV. In Russian, word order is more restricted in transitive clauses. Some languages, such as Datooga in Tanzania have free word order. Hixkaryana and Tamil are a few of examples of the rare OVS languages.

More data

With a telescope I saw a man.

#With a man I saw a telescope.

#I saw a telescope with a man.

*A man saw a telescope with I.

I, with a telescope saw a man.

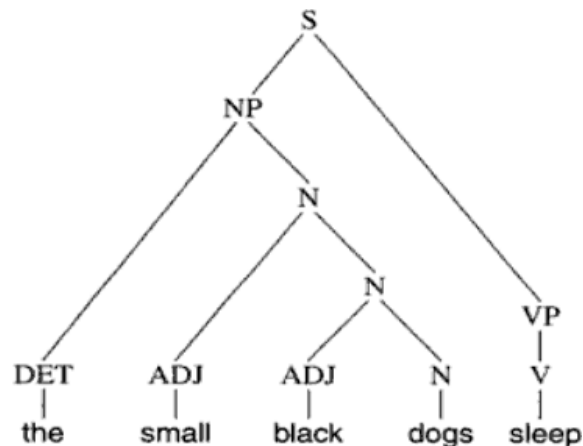
Observation: some orderings have more felicity than others



Can we find consistent generalizations to capture this? If so, what would we call such a set?

Constituents

- Can sentences be analyzed as containing sub-units which consist of one or more words?



(Hausser 1998)

This is a phrase structure tree.

Hypothesis: The class of grammatical constituents is closed.

Constituent Types

- Constituents are characterized by the part-of-speech of their main word, the “head.”
- Thus, we notice that for many languages, a sentence comprises a:
 - subject (a Noun Phrase or NP)
 - predicate (a Verb Phrase or VP)
- These may be composed of other constituents
 - Prepositional phrase (PP)
 - Determiner phrase (DP)

Constituent Construction

- Noun phrases (NPs)

(DET NN) *The ostrich*

(NNP) *Kim*

(NN NN) *container ship*

(DET JJ NN) *A purple lawnmower*

(DET JJ NN) *That darn cat*

- Verb phrases (VPs)

(VB) tango

(VBD NP NP) gave the dog a bone

(VBD NP PP) gave a bone to the dog

Syntax

The set of rules governing permissible constructions in a language.

- Syntax constrains the ways in which words may be combined to form constituents and sentences.
- Syntax forms one part of the description, or *grammar*, of a language.



Prescriptive v. descriptive grammar

- Prescriptive
 - Rules against certain usages. Few if any rules for what *is* allowed.
 - Prepositions are not for ending sentences with.
- Descriptive
 - Rules characterizing what people *do* say.
 - Goal is to characterize all and only what speakers find acceptable.
 - Based on the scientific method

Slide: Emily Bender

Artificiality of prescriptive rules

- Fill in the blanks: *he/his, they/their, or something else?*

Everyone insisted that ___ record was unblemished.

Everyone drives ___ own car to work.

Everyone was happy because ___ passed the test.

Everyone left the room, didn't ___?

Everyone left early. ___ seemed happy to get home.

Slide: Bender, Sag, Wasow 2003

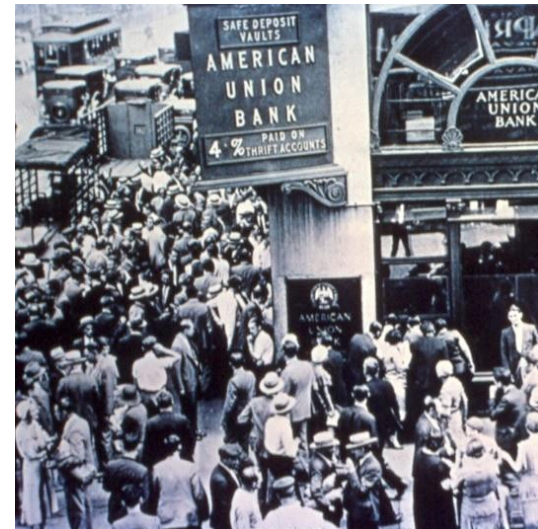
Two kinds of ambiguity

1. Lexical ambiguity

The bank is crumbling.



?



Two kinds of ambiguity

2. Structural ambiguity

I saw a man with a telescope.



?



?



Is that all?



I (often) saw a man with a telescope.

Ambiguity

Q: What kind of ambiguity does the following sentence illustrate?

Have that report on my desk by Friday.

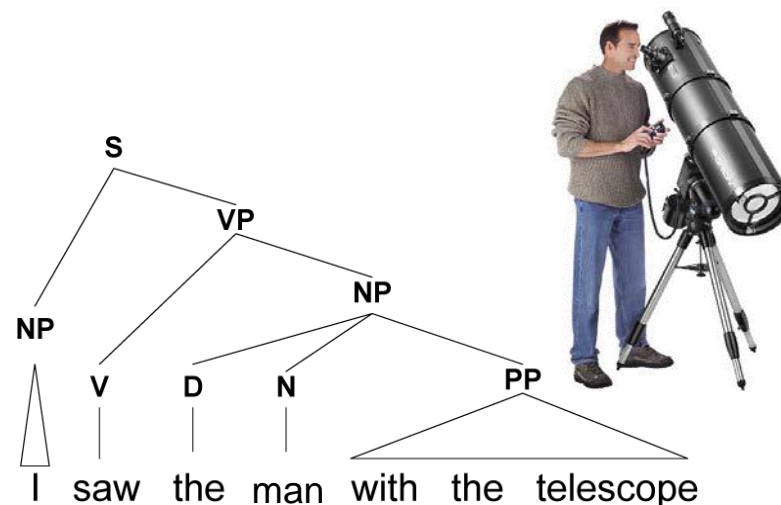
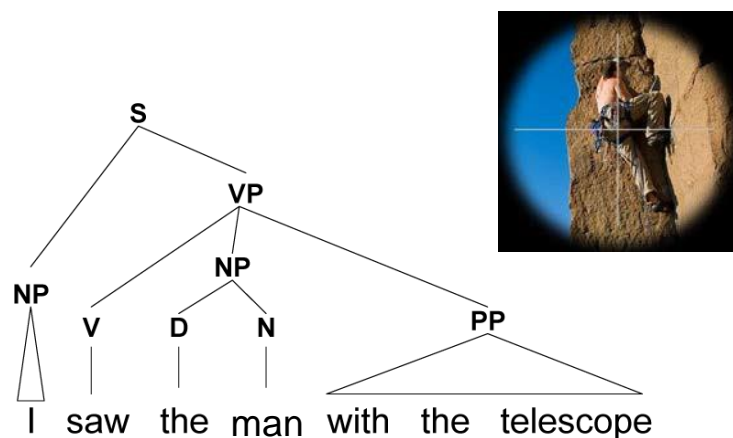
A: Both structural and lexical ambiguity.



In English, prosody in speech provides disambiguation by, for example, marking topic and focus.

Constituents

- Constituents help us understand and classify syntactic structure
- Here, phrase structure trees show us how constituent structure helps us characterize ambiguity



Trees: Dan Jinguji



Garden Path Sentences

These curiosities are another kind of ambiguity which are more relevant to psycholinguistics.

The old man the boat.

The generic NP “the old” and the verb “to man” are much more rarely used than the NP “the old man.”

The horse raced past the barn fell.

The relative clause “raced past the barn” is not introduced by “which.”

Analytical NLP: Summary

- Language has significant complexity
- Ambiguity is an inherent feature of language
- Inferring the syntactic rules of a language is difficult
- It appears that syntactic rules *are* amenable to computational approximation, but there are a great number of them, and they are subtle
- See next slide for the obligatory optimism...

Latest Work: Hybrid analytical/statistical systems

- Examples:

- Stephan Oepen, Erik Velldal, Jan Tore Lønning, Paul Meurer, Victoria Rosén, and Dan Flickinger. 2007. “Towards hybrid quality-oriented machine translation”

<http://www.mt-archive.info/TMI-2007-Oepen.pdf>

- Parse ranking in the English Resource Grammar
- Unsupervised learning of rules
 - Poon and Domingos 2009 “Unsupervised Semantic Parsing” (UW CSE)

<http://www.aclweb.org/anthology-new/D/D09/D09-1001.pdf>

Statistical Natural Language Processing

- Large gains in practical application of stochastic methods in the past 15 years
- Example: MT
 - Microsoft Bing Translator
 - Google translate
 - GIZA++/Moses SMT toolkit
- Insight: clever math—with limited linguistic motivation—can work surprisingly well



These improvements are not just due to Moore's law

The mid-1990s

- Advances in computing power
- Perceived lack of progress in analytical NLP
- What if the “rules” of linguistics as we intuitively imagine them are too hard (or too incorrect, or too biased...) to capture?
- Let’s forget any preconceived notion of what the rules *should* be and use math to characterize what the rules *appear* to be in practice.
- German *Verbmobil* project had success with statistical machine translation (This large project also developed rule-based systems)

Source: Koehn 2010 “Statistical Machine Translation”

Corpus Linguistics

The study of language as expressed in samples (corpora) or “real world” text.

- Wikipedia

This can be thought of as a variant of analytical NLP where the form, substance, and quantity of “rules” are generated automatically according to the maximization of an empirical objective function.

Treebank

- A collection of text with syntactic structure annotation for each sentence
 - Human-annotated
 - Machine generated by precision grammar
- Penn Treebank (PTB) (Marcus et al. 1994)
<http://www.cis.upenn.edu/~treebank/>
- Linguistic Data Consortium (LDC)
<http://www ldc.upenn.edu/>



PTB Example

```
( (S (NP (NNP John) )  
    (VP (VBZ loves)  
        (NP (NNP Mary) ) )  
    (. .) ) )
```

PTB tag set

CC	Coordinating conjunction	PRP	Possessive pronoun
CD	Cardinal number	RB	Adverb
DT	Determiner	RBR	Adverb, comparative
EX	Existential there	RBS	Adverb, superlative
FW	Foreign word	RP	Particle
IN	Preposition or subordinating conjunction	SYM	Symbol
JJ	Adjective	TO	to
JJR	Adjective, comparative	UH	Interjection
JJS	Adjective, superlative	VB	Verb, base form
LS	List item marker	VBD	Verb, past tense
MD	Modal	VBG	Verb, gerund or present participle
NN	Noun, singular or mass	VCN	Verb, past participle
NNS	Noun, plural	VBP	Verb, non 3rd person singular present
NNP	Proper noun, singular	VBZ	Verb, 3rd person singular present
NNPS	Proper noun, plural	WDT	Wh-determiner
PDT	Predeterminer	WP	Wh-pronoun
POS	Possessive ending	WP	Possessive wh-pronoun
PRP	Personal pronoun	WRB	Wh-adverb

Counting

- Corpus linguistics is essentially about counting and arranging collections and sequences of elements (words, characters).
- Elements are collected together into sets or bags
 - This can be done in a number of ways as we'll see
- It's important to keep track of what we are counting. We can count:
 - The number of elements in a collection
 - This is fairly straightforward. This is called the “cardinality” of the collection
 - The number of occurrences of a particular element in a collection
 - We can distinguish this by calling it a “tally.”
 - The number of collections of a certain type that are found in a corpus
 - The number of collections of a certain type that could possibly be formed from another collection of elements

Collections

- Consider a collection of n elements. It can be ordered or unordered, and distinct (all elements are unique), or with repeated elements.

	Ordered	Unordered
Distinct	(i.e. MRU cache)	vocabulary, alphabet, set
Not-Distinct	sentence, string, vector, sequence, word	bag, multiset



We distinguish ordering from sorting. Ordering refers to whether the order matters in the modeling context, not whether the elements happen to be in some sorted order for practical (programming) purposes

Distinctness

- Distinct, i.e. no repetition (“set,” “vocabulary”)

```
{ apple, pear, banana, orange, pomelo }
```

we can count:

- The number of distinct elements (“cardinality,” “vocabulary size”)

- With repetition (“bag” or “multiset”)

```
{ apple, apple, banana, apple }
```

we can count:

- The number of distinct elements
- The number of times each element appears (“tally” or “multiplicity”)

- In mathematics, the word ‘set’ generally means unordered and distinct unless otherwise specified
- Parentheses are usually used when order matters (n-tuples), braces are used when order doesn’t matter { sets }.

Ordering

Ordered, (“string,” “vector,” “sequence,” or “n-tuple”)

```
( a, man, gave, sandy, a, pomelo )  
                               ≠  
( a, a, gave, kim, pomelo, sandy )
```

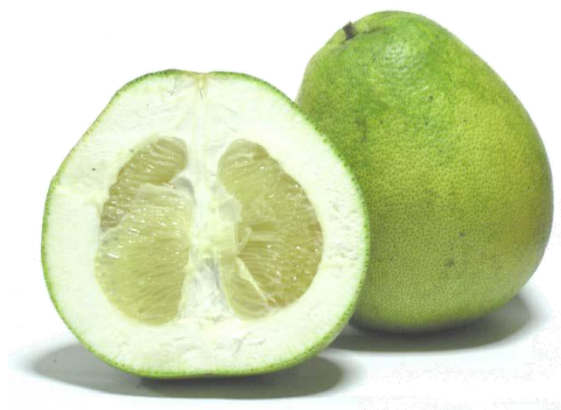
Unordered, Not-distinct (“bag”)

```
{ a, man, gave, sandy, a, pomelo }  
                               =  
{ sandy, pomelo, man, gave, a, a }
```

Unordered, Distinct (“vocabulary”)

```
{ a, man, gave, sandy, pomelo }
```

- Distinct collections where the order matters are less common. An example is the set of (distinct) words used in a document, by order of appearance. This is a form of Most Recently Used (MRU) cache.



Vocabulary

The distinct set of elements which appear in some other set

- Examples:
 - The words used in a document or sentence
 - All the words of a language
 - The characters used in a word
 - The English alphabet is the ‘vocabulary’ of symbols used in writing the language

Tallying

Tallies:

The count (number of occurrences) for a each distinct element of a non-distinct set

- Each element becomes associated with its count
- Tallying creates a distinct set of tuples (a language model) from a non-distinct set (a corpus)
- Tallying is probably the most commonly used programming pattern in corpus linguistics

Tally Example

"a man gave sandy a pomelo"

(a, 2)

(man, 1)

(gave, 1)

(sandy, 1)

(pomelo, 1)

A tally is a count. We can divide each tally by the vocabulary size to obtain *frequencies*.

(a, .33)

(man, .16)

(gave, .16)

(sandy, .16)

(pomelo, .16)

Combining Counts

- Fundamental counting principle:

The cross product of the sets

$\{m_0 \dots m_{cc_{mm}}\}$ and $\{n_0 \dots n_{cc_{nn}}\}$

has $cc_{mm} \times cc_{nn}$ members

- That is, there are $m \times n$ ways to combine exactly one element each from sets of size m and n .
- Combine independent counts by multiplying

Combining Counts

- It follows that if we choose m times from the same set of n elements without depletion (or with replacement), there are n^m possible results. (This includes all possible orderings)

Example:

Simplistically, how many four-letter words can be formed with the English alphabet?

$$26^4 = 456,976$$

Counting Sets: Combinatorics

- The basics of counting and arranging sequences:

Permutations:

all possible orderings of n elements

Combinations:

all (unordered) sets of k elements that can be selected from n elements

Variations:

all orderings of k elements that can be selected from n elements.

Permutations

{ o, r, a, n, g, e }

6 choices	a						{ o, r, n, g, e }
5 choices	a	e					{ o, r, n, g }
4 choices	a	e	g				{ o, r, n }
3 choices	a	e	g	n			{ o, r }
2 choices	a	e	g	n	o		{ r }
1 choice	a	e	g	n	o	r	{ }

$$6 \times 5 \times 4 \times 3 \times 2 \times 1 =$$
$$6! = 720$$



This is the factorial function,
denoted by $n!$

Permutations

- The previous example, ‘orange’ had no repeated letters in the input.
- Permutation only rearranges elements, so it can never “generate” repetition (like we will see with combinations).
- But if the input contains repetitions of some value(s), we may want to eliminate duplicate outputs.
- To do this, we simply divide $n!$ by the number of those combinations that are indistinguishable from each other
- Applying the fundamental counting principle to combine each set of repeated values $\{r_{i,0}, r_{i,1}, \dots, r_{i,m}\}$ in the input, this denominator is given by:

$$\prod_i m_{ii}!$$

Permutations with Repeated Input Values

Example:

mississippi

11 letters

repeated group { i, i, i, i } : cardinality 4

repeated group { s, s, s, s } : cardinality 4

repeated group { p, p } : cardinality 2

$$\frac{11!}{4! \times 4! \times 2!} = 34,650$$



Some sources consider counting the permutations of a subset of a set's elements to be a type of permutation, but I treat this separately—see 'Variations'

Combinations

- Also called k -combinations
- How many unordered sets of size k can be formed from a set of cardinality n ?
- We've already considered the case of not depleting (i.e., replacing) when ordering of the output matters:

n^k (a.k.a. variations, with repetition)



And as for unordered sets of k from n with replacement, see Assignment 1 - extra credit

- If depleting, we can derive this answer from the fundamental counting principle

Combinations

Example: We have seven documents and we need to find the group of three documents that is most divergent from the corpus. How many groups of 3 will we have to test?

1st choice: 7 documents to choose from

2nd choice: 6 remaining documents

3rd choice: 5 remaining documents.

$$7 \times 6 \times 5 = 210 \text{ sets.}$$

But ordering doesn't matter. Since each of these sets has 3! orderings, we divide by $3! = 6$, thus counting each set just once.

$$210 \div 6 = 35$$

Combinations

- How can we generalize $n(n-1) \dots (n-k+1)$

Let's use factorial:

$$\frac{n(n-1) \dots (n-k+1)(n-k) \dots k \dots (2)(1)}{(n-k) \dots k \dots (2)(1)} = \frac{n!}{(n-k)!}$$

Don't forget to cancel out the permutations, since ordering doesn't matter:

$$\binom{n}{k} = \frac{n!}{(n-k)! k!}$$



This is called the binomial coefficient, or choose function.

Variations

- Variations are permutations of combinations: ordered subsets of size k chosen from a set of n .
- With repetition in the output:

$$n^k$$

- *Without repetition in the output:*

Same as the way we derived the choose function, but we don't need to cancel out the repeated outputs

$$\frac{n!}{(n - k)!}$$

Next time

- Linux, Cluster Computing, Regular Expressions