CSCI 5832
Natural Language Processing

Jim Martin
Lecture 12

Today 2/26

- Syntax
  - Context-Free Grammars
- Review Quiz
- More grammars

Syntax

- By syntax (or grammar) I mean the kind of implicit knowledge of your native language that you had mastered by the time you were 2 or 3 years old without explicit instruction
- Not the kind of stuff you were later taught in school.

Syntax

- Why should you care?
  - Grammar checkers
  - Question answering
  - Information extraction
  - Machine translation

Context-Free Grammars

- Capture constituency and ordering
  - Ordering is easy

    What are the rules that govern the ordering of words and bigger units in the language
  - What's constituency?

    How words group into units and how the various kinds of units behave wrt one another

CFG Examples

- S -> NP VP
- NP -> Det NOMINAL
- NOMINAL -> Noun
- VP -> Verb
- Det -> *a*
- Noun -> *flight*
- Verb -> *left*

CFGs

- S -> NP VP
  - This says that there are units called S, NP, and VP in this language
  - That an S consists of an NP followed immediately by a VP
  - Doesn't say that that's the only kind of S
  - Nor does it say that this is the only place that NPs and VPs occur

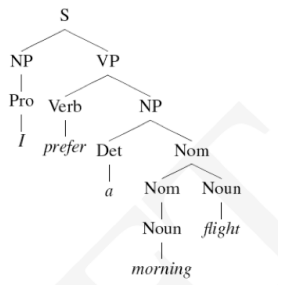
Generativity

- As with FSAs and FSTs you can view these rules as either analysis or synthesis machines
  - Generate strings in the language
  - Reject strings not in the language
  - Impose structures (trees) on strings in the language

Derivations

- A derivation is a sequence of rules applied to a string that accounts for that string
  - Covers all the elements in the string
  - Covers only the elements in the string

Derivations as Trees



Parsing

- Parsing is the process of taking a string and a grammar and returning a (many?) parse tree(s) for that string
- It is completely analogous to running a finite-state transducer with a tape
  - It's just more powerful
    - Remember this means that there are languages we can capture with CFGs that we can't capture with finite-state methods

Other Options

- Regular languages (expressions)
  - Too weak
- Context-sensitive or Turing equiv
  - Too powerful (maybe)

Context?

- The notion of context in CFGs has nothing to do with the ordinary meaning of the word context in language.
- All it really means is that the non-terminal on the left-hand side of a rule is out there all by itself (free of context)

  A -> B C

  Means that
  - I can rewrite an A as a B followed by a C regardless of the context in which A is found
  - Or when I see a B followed by a C I can infer an A regardless of the surrounding context

Key Constituents (English)

- Sentences
- Noun phrases
- Verb phrases
- Prepositional phrases

Sentence-Types

- Declaratives: A plane left

  S -> NP VP
- Imperatives: Leave!

  S -> VP
- Yes-No Questions: Did the plane leave?

  S -> Aux NP VP
- WH Questions: When did the plane leave?

  S -> WH Aux NP VP

Recursion

- We'll have to deal with rules such as the following where the non-terminal on the left also appears somewhere on the right (directly).

  Nominal -> Nominal PP [[flight] [to Boston]]

  VP -> VP PP [[departed Miami] [at noon]]

Recursion

- Of course, this is what makes syntax interesting
  - flights from Denver
  - Flights from Denver to Miami
  - Flights from Denver to Miami in February
  - Flights from Denver to Miami in February on a Friday
  - Flights from Denver to Miami in February on a Friday under $300
  - Flights from Denver to Miami in February on a Friday under $300 with lunch

Recursion

- Of course, this is what makes syntax interesting

  [[flights] [from Denver]]

  [[[Flights] [from Denver]] [to Miami]]

  [[[[Flights] [from Denver]] [to Miami]] [in February]]

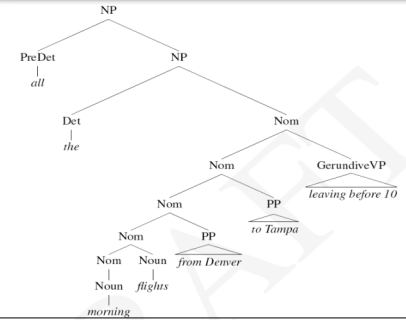
  [[[[[Flights] [from Denver]] [to Miami]] [in February]] [on a Friday]]

  Etc.

The Point

- If you have a rule like
  - VP -> V NP
  - It only cares that the thing after the verb is an NP. It doesn't have to know about the internal affairs of that NP

The Point



Conjunctive Constructions

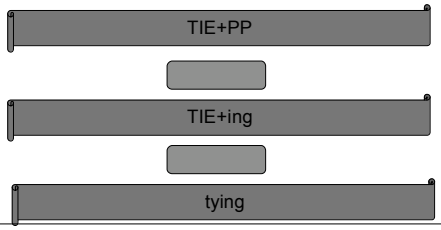
- S -> S and S
  - John went to NY and Mary followed him
- NP -> NP and NP
- VP -> VP and VP
- …
- In fact the right rule for English is
  X -> X and X

Break

- Quiz
  1. 29
  2. slides
  3. True
  4. slides
  5. slides

2...

- Rules...
  - Verb+PresPart -> Verb+ing (lexical)
  - -ie+ing -> -y+ing (surface)



4a: One fish...

One fish two fish red fish blue fish

	One	Two	Red	Blue	Fish
One	0	0	0	0	1
Two	0	0	0	0	1
Red	0	0	0	0	1
Blue	0	0	0	0	1
Fish	0	1	1	1	0

4b: One fish...

One fish two fish red fish blue fish

	One	Two	Red	Blue	Fish
One	1	1	1	1	2
Two	1	1	1	1	2
Red	1	1	1	1	2
Blue	1	1	1	1	2
Fish	1	2	2	2	1

4b

- P(fish|red) = Count(red fish)/Count(red)
  = 2/6 = 1/3
- P(fish|fish) = Count(fish fish)/Count (fish)
  = 1/9

4c

- Would trigrams help?
  - No. Think in terms of the two cases here.
    - There are fish and there are adjs
      - P(fish|ADJ) = 1
      - P(ADJ|fish) = 1
    - A trigram model...
      - P(fish| fish ADJ) = 1
      - P(ADJ | adj fish) = 1
    - But maybe....

5

- Need
  1. Transition table
  2. Observation table
  3. Start table

2/28/08 28

5a

- Transition table

	JJ	NN	ORD
JJ	0	4	0
NN	4	0	1
ORD	0	2	0

2/28/08 29

5a

- Observation table(s)

	One	Fish	Two	Red	Blue	Black	
JJ	0	0	0	1	2	1	
NN	0	6	0	0	0	0	
ORD	1	0	1	0	0	0	

2/28/08 30

5a

- Start table (Pi)

	JJ	NN	ORD
START	0	0	1

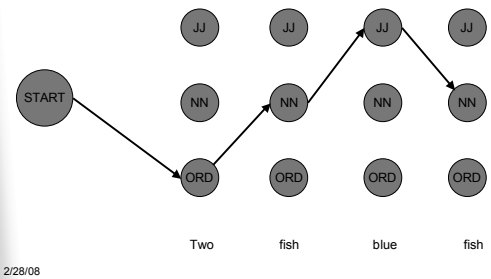
5b

- Two fish blue fish

ORD NN JJ NN

P(ORD|START)*P(NN|ORD)*P(JJ|NN)*P(NN|JJ)*
P(Two|ORD)*P(Fish|NN)*P(Blue|JJ)*P(Fish|NN)

5b



Problems

- Agreement
- Subcategorization
- Movement (for want of a better term)

Agreement

- This dog
- Those dogs
- *This dogs
- *Those dog
- This dog eats
- Those dogs eat
- *This dog eat
- *Those dogs eats

Agreement

- In English,
  - subjects and verbs have to agree in person and number
  - Determiners and nouns have to agree in number
- Many languages have agreement systems that are far more complex than this.

Subcategorization

- Sneeze: John sneezed
- Find: Please find [a flight to NY]$_{NP}$
- Give: Give [me]$_{NP}$[a cheaper fare]$_{NP}$
- Help: Can you help [me]$_{NP}$[with a flight]$_{PP}$
- Prefer: I prefer [to leave earlier]$_{TO\text{-}VP}$
- Told: I was told [United has a flight]$_{S}$
- …

Subcategorization

- *John sneezed the book
- *I prefer United has a flight
- *Give with a flight

- Subcat expresses the constraints that a predicate (verb for now) places on the number and syntactic types of arguments it wants to take (occur with).

So?

- So the various rules for VPs *overgenerate*.
  - They permit the presence of strings containing verbs and arguments that don't go together
  - For example
  - VP -> V NP therefore

    Sneezed the book is a VP since "sneeze" is a verb and "the book" is a valid NP

So What?

- Now *overgeneration* is a problem for a generative approach.
  - The grammar is supposed to account for all and only the strings in a language
- From a practical point of view... Not so clear that there's a problem
  - Why?

Possible CFG Solution

- S -> NP VP
- NP -> Det Nominal
- VP -> V NP
- …

- SgS -> SgNP SgVP
- PlS -> PlNp PlVP
- SgNP -> SgDet SgNom
- PlNP -> PlDet PlNom
- PlVP -> PlV NP
- SgVP ->SgV Np
- …

CFG Solution for Agreement

- It works and stays within the power of CFGs
- But its ugly
- And it doesn't scale all that well

Forward Pointer

- It turns out that verb subcategorization facts will provide a key element for semantic analysis (determining who did what to who in an event).

Movement

- Core (canonical) example
  - My travel agent booked the flight

Movement

- Core example
  - [[My travel agent]$_{NP}$ [booked [the flight]$_{NP}$]$_{VP}$]$_{S}$
- I.e. “book” is a straightforward transitive verb. It expects a single NP arg within the VP as an argument, and a single NP arg as the subject.

Movement

- What about?
  - Which flight do you want me to have the travel agent book?
- The direct object argument to “book” isn’t appearing in the right place. It is in fact a long way from where its supposed to appear.
- And note that its separated from its verb by 2 other verbs.

The Point

- CFGs appear to be just about what we need to account for a lot of basic syntactic structure in English.
- But there are problems
  - That can be dealt with adequately, although not elegantly, by staying within the CFG framework.
- There are simpler, more elegant, solutions that take us out of the CFG framework (beyond its formal power)

Parsing

- Parsing with CFGs refers to the task of assigning correct trees to input strings
- Correct here means a tree that covers all and only the elements of the input and has an S at the top
- It doesn’t actually mean that the system can select the correct tree from among all the possible trees

Parsing

- As with everything of interest, parsing involves a search which involves the making of choices
- We'll start with some basic (meaning bad) methods before moving on to the one or two that you need to know

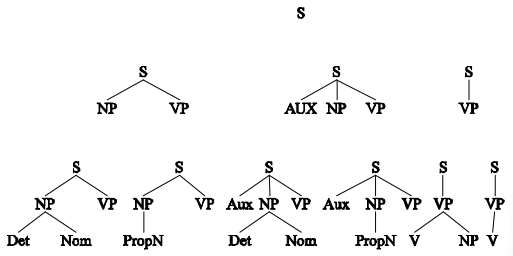
For Now

- Assume…
  - You have all the words already in some buffer
  - The input isn't POS tagged
  - We won't worry about morphological analysis
  - All the words are known

Top-Down Parsing

- Since we're trying to find trees rooted with an S (Sentences) start with the rules that give us an S.
- Then work your way down from there to the words.

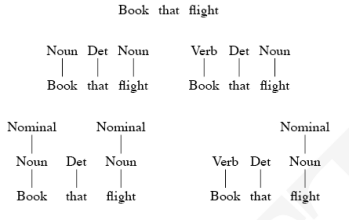
Top Down Space



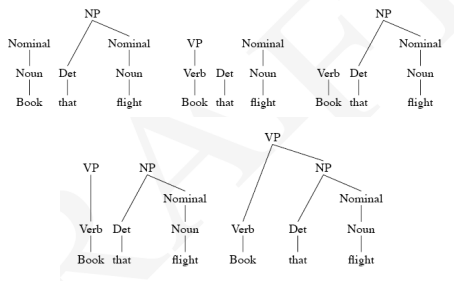
Bottom-Up Parsing

- Of course, we also want trees that cover the input words. So start with trees that link up with the words in the right way.
- Then work your way up from there.

Bottom-Up Space



Bottom Up Space



Control

- Of course, in both cases we left out how to keep track of the search space and how to make choices
  - Which node to try to expand next
  - Which grammar rule to use to expand a node

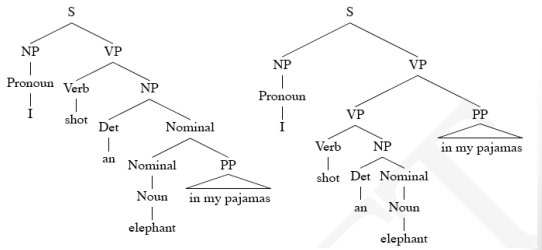
Top-Down and Bottom-Up

- Top-down
  - Only searches for trees that can be answers (i.e. S's)
  - But also suggests trees that are not consistent with any of the words
- Bottom-up
  - Only forms trees consistent with the words
  - But suggest trees that make no sense globally

Problems

- Even with the best filtering, backtracking methods are doomed if they don't address certain problems
  - Ambiguity
  - Shared subproblems

Ambiguity

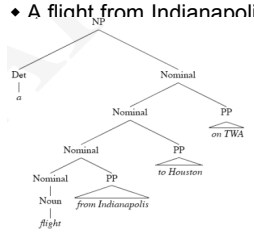


Shared Sub-Problems

- No matter what kind of search (top-down or bottom-up or mixed) that we choose.
  - We don't want to unnecessarily redo work we've already done.

Shared Sub-Problems

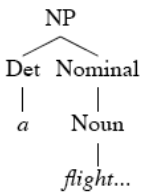
- Consider
  - A flight from Indianapolis to Houston on TWA



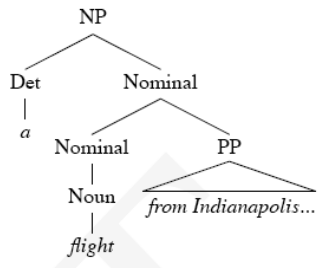
Shared Sub-Problems

- Assume a top-down parse making bad initial choices on the Nominal rule.
- In particular…
  - Nominal -> Nominal Noun
  - Nominal -> Nominal PP

Shared Sub-Problems



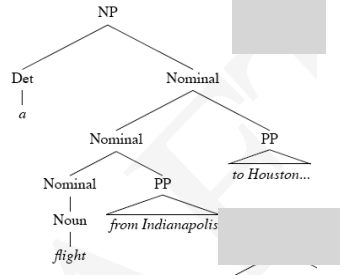
Shared Sub-Problems



2/28/08

64

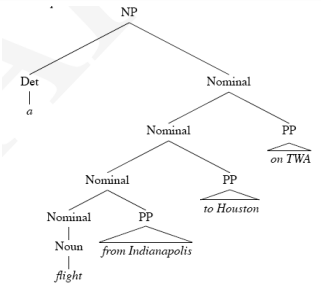
Shared Sub-Problems



2/28/08

65

Shared Sub-Problems



2/28/08

66