

CSE 842

Natural Language Processing

Lecture 8: Context Free Grammar

2/9/2009

CSE842, Spring 2009, MSU

1

Reminder

- Homework1 written part due today
- Homework 1 programming assignment
 - Due 2/11, 11:59pm
 - Electronically hand in your homework:
<http://www.cse.msu.edu/handin>
 - If you do not have a CSE user/password, please contact manager@cse.msu.edu to request one for this course.
- Homework 2 assignment will be posted after the class

2/9/2009

CSE842, Spring 2009, MSU

2

What is Grammar

- A vogue in the 19th century to describe the structures of an area of knowledge
(e.g., Busby's "A Grammar of Music"
Field's "A Grammar of Colouring")
- The meaning of grammar:
 - Used to be: "A listing of principles or structures"
 - Now: "principles or structures as a field of inquiry"
 - The grammar of syntax

2/9/2009

CSE842, Spring 2009, MSU

3

Syntax

- Covered: POS categories
- To be covered:
 - Constituency
 - Grammatical relations
 - Subcategorization and dependencies

2/9/2009

CSE842, Spring 2009, MSU

4

What is a constituent?

- The idea: groups of words may behave as a single unit or phrase -> constituent
- Example: noun phrase act like a unit
 - "she"
 - "Michael"
 - "well weathered three story structure"
- How can we model the constituency?
 - With context free grammars

2/9/2009

CSE842, Spring 2009, MSU

5

Context-Free Grammars

- Chomsky (1956) Backus (1959)
- Set of rules (productions) – expressing the way symbols of the language can be grouped together
 - NP -> Det Nominal
 - NP -> Proper Noun
 - Nominal -> Noun | Noun Nominal
- Lexicon
 - Det -> a
 - Det -> the
 - Noun -> flight

2/9/2009

CSE842, Spring 2009, MSU

6

Context Free Grammars

- Capture constituents and ordering
 - Need something else for grammatical relations and dependency relations
- Consists of
 - Set of terminals (words)
 - Set of non-terminal (the constituent of language)
 - Sets of rules of the form $A \rightarrow \alpha$, where α is a string of zero or more terminals and non-terminals
- They are equivalent to Backus Normal form grammars

2/9/2009

CSE842, Spring 2009, MSU

7

Formal Definition of CFG

- Set of non-terminal symbols N
- Set of terminals Σ
- Set of productions $A \rightarrow \alpha$
 $A \in N, \alpha \text{ string } (\Sigma \cup N)^*$
- A designated start symbol

2/9/2009

CSE842, Spring 2009, MSU

8

Some NP Rules

- Here are some rules for our noun phrases

$NP \rightarrow Det \text{ Nominal}$
 $NP \rightarrow ProperNoun$
 $Nominal \rightarrow Noun \mid Nominal \text{ Noun}$

- Together, these describe two kinds of NPs.
 - One that consists of a determiner followed by a nominal
 - And another that says that proper names are NPs.
 - The third rule illustrates two things
 - An explicit disjunction
 - Two kinds of nominals
 - A recursive definition
 - Same non-terminal on the right and left-side of the rule

2/9/2009

CSE842, Spring 2009, MSU

9

L0 Grammar

Grammar Rules	Examples
$S \rightarrow NP VP$	I + want a morning flight
$NP \rightarrow Pronoun$	I
$Proper-Noun$	Los Angeles
$Det \text{ Nominal}$	a + flight
$Nominal \rightarrow Nominal \text{ Noun}$	morning + flight
$Noun$	flights
$VP \rightarrow Verb$	do
$Verb \text{ NP}$	want + a flight
$Verb \text{ NP PP}$	leave + Boston + in the morning
$Verb \text{ PP}$	leaving + on Thursday
$PP \rightarrow Preposition \text{ NP}$	from + Los Angeles

2/9/2009

CSE842, Spring 2009, MSU

10

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

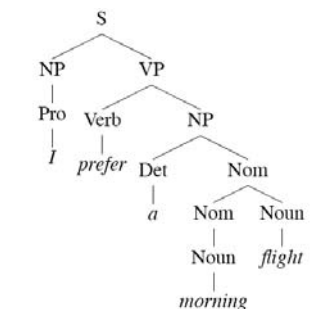
2/9/2009

CSE842, Spring 2009, MSU

11

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



2/9/2009

CSE842, Spring 2009, MSU

12

Key Constituents

- Sentences
- Noun phrases
- Verb phrases
- Prepositional phrases

2/9/2009

CSE842, Spring 2009, MSU

13

Sentence Types

- Declaratives
S → NP VP (e.g., John left)
- Imperatives
S → VP (e.g., Leave!)
- Yes No Questions
S → Aux NP VP (e.g., Did John leave?)
- WH Questions
S → Wh-word VP (e.g., who left?)
S → Wh-word Aux NP VP (e.g., When did John leave?)

2/9/2009

CSE842, Spring 2009, MSU

14

Noun Phrases

- Let's consider the following rule in more detail...

$NP \rightarrow Det\ Nominal$

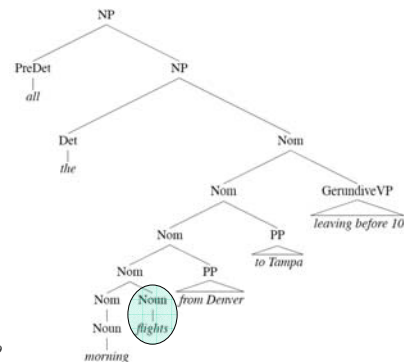
- Most of the complexity of English noun phrases is hidden in this rule.
- Consider the derivation for the following example
– All the morning flights from Denver to Tampa leaving before 10

2/9/2009

CSE842, Spring 2009, MSU

15

Noun Phrases



2/9/2009

16

NP Structure

- Clearly this NP is really about *flights*. That's the central critical noun in this NP. Let's call that the *head*.
- We can dissect this kind of NP into the stuff that can come before the head, and the stuff that can come after it.

2/9/2009

CSE842, Spring 2009, MSU

17

Determiners

- Noun phrases can start with determiners...
- Determiners can be
 - Simple lexical items: *the, this, a, an, etc.*
 - A car
 - Or simple possessives
 - John's car
 - Or complex recursive versions of that
 - John's sister's husband's son's car

2/9/2009

CSE842, Spring 2009, MSU

18

Nominals

- Contains the head and any pre- and post-modifiers of the head.
 - Pre
 - Quantifiers, cardinals, ordinals...
 - Three cars
 - Adjectives and Aps
 - large cars
 - Ordering constraints
 - Three large cars
 - ?large three cars

2/9/2009

CSE842, Spring 2009, MSU

19

Postmodifiers

- Three kinds
 - Prepositional phrases
 - From Seattle
 - Non-finite clauses
 - Arriving before noon
 - Relative clauses
 - That serve breakfast
- Same general (recursive) rule to handle these
 - *Nominal* → *Nominal PP*
 - *Nominal* → *Nominal GerundVP*
 - *Nominal* → *Nominal RelClause*

2/9/2009

CSE842, Spring 2009, MSU

20

Gerunds

- any flights [arriving after ten p.m]
- *Nominal* → *Nominal GerundVP*
- *GerundVP* → *GerundV NP* | *GerundV PP* | *GerundV* | *GerundV NP PP*
- *GerundV* → *being* | *preferring* | *arriving* ...

2/9/2009

CSE842, Spring 2009, MSU

21

Infinitives and –ed forms

- the last flight to arrive in Boston
- I need to have dinner served
- which is the aircraft used by this flight?

2/9/2009

CSE842, Spring 2009, MSU

22

Postnominal relative clauses

- Restrictive relative clauses:
 - A flight that serves breakfast
 - Flights that leave in the morning
 - The United flight that arrives in San Jose at ten p.m.
- Rules:
 - *Nominal* → *Nominal RelClause*
 - *RelClause* → (*who* | *that*) *VP*

2/9/2009

CSE842, Spring 2009, MSU

23

Combining post-modifiers

- *A flight from Phoenix to Detroit leaving Monday evening*
- *Evening flights from Nashville to Houston that serve dinner*
- *The earliest American Airlines flight that I can get*

2/9/2009

CSE842, Spring 2009, MSU

24

Recursive Structure

- Rules where the non-terminal on the left-hand side also appears on the right-hand side.
 - $NP \rightarrow NP PP$ (*The flight to Boston*)
 - $VP \rightarrow VP PP$ (*departed Miami at noon*)

2/9/2009

CSE842, Spring 2009, MSU

25

Recursive Structure

- Allow us to the following:
 - *Flights to Miami*
 - *Flights to Miami from Boston*
 - *Flights to Miami from Boston in April*
 - *Flights to Miami from Boston in April on Friday*
 - *Flights to Miami from Boston in April on Friday under \$300*

2/9/2009

CSE842, Spring 2009, MSU

26

Conjunctions

- Any phrasal constituent can be conjoined with a constituent of the same type to form a new constituent of that type
 - $NP \rightarrow NP \text{ and } NP$
 - $S \rightarrow S \text{ and } S$
- $X \rightarrow X \text{ and } X$

2/9/2009

CSE842, Spring 2009, MSU

27

Some Difficulties

- Agreement
- Subcategorization

2/9/2009

CSE842, Spring 2009, MSU

28

Agreement

- By **agreement**, we have in mind constraints that hold among various constituents that take part in a rule or set of rules
- For example, in English, determiners and the head nouns in NPs have to agree in their number.

This flight

*This flights

Those flights

*Those flight

2/9/2009

CSE842, Spring 2009, MSU

29

Problem

- Our earlier NP rules are clearly deficient since they don't capture this constraint
 - $NP \rightarrow \text{Det Nominal}$
 - Accepts, and assigns correct structures, to grammatical examples (*this flight*)
 - But its also happy with incorrect examples (*these flight)
 - Such a rule is said to *overgenerate*.
- How to solve the problem?

2/9/2009

CSE842, Spring 2009, MSU

30

Agreement

- $S \rightarrow \text{Aux NP VP}$
- $S \rightarrow 3\text{sgAux } 3\text{sgNP VP}$
- $S \rightarrow \text{Non3sgAux non3sgNP VP}$

- $3\text{sgAux} \rightarrow \text{does} \mid \text{has} \mid \text{can} \dots$
- $\text{non3sgAux} \rightarrow \text{do} \mid \text{have} \mid \text{can} \dots$

2/9/2009

CSE842, Spring 2009, MSU

31

Agreement

- We now need similar rules for NPs, also for number agreement, etc.
 - $3\text{SgNP} \rightarrow (\text{Det}) (\text{Card}) (\text{Ord}) (\text{Quant}) (\text{AP}) \text{SgNominal}$
 - $\text{Non3SgNP} \rightarrow (\text{Det}) (\text{Card}) (\text{Ord}) (\text{Quant}) (\text{AP}) \text{PlNominal}$
 - $\text{SgNominal} \rightarrow \text{SgNoun} \mid \text{SgNoun SgNoun}$
 - etc.
- Combinatorial explosion

2/9/2009

CSE842, Spring 2009, MSU

32

Verb Phrases

- English *VPs* consist of a head verb along with 0 or more following constituents which we'll call *arguments*.

$VP \rightarrow \text{Verb}$ disappear

$VP \rightarrow \text{Verb NP}$ prefer a morning flight

$VP \rightarrow \text{Verb NP PP}$ leave Boston in the morning

$VP \rightarrow \text{Verb PP}$ leaving on Thursday

2/9/2009

CSE842, Spring 2009, MSU

33

Subcategorization

- But, even though there are many valid VP rules in English, not all verbs are allowed to participate in all those VP rules.
- We can subcategorize the verbs in a language according to the sets of VP rules that they participate in.
- Traditional grammar distinguishes between transitive and intransitive.
- Modern grammars may have 100s or such classes.

2/9/2009

CSE842, Spring 2009, MSU

34

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-VP}
- Told: I was told [United has a flight]_S
- ...

2/9/2009

CSE842, Spring 2009, MSU

35

Subcategorization

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

- As with agreement phenomena, we need a way to formally express the constraints

2/9/2009

CSE842, Spring 2009, MSU

36

Overgeneration

- Right now, 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

2/9/2009

CSE842, Spring 2009, MSU

37

CFG Solution for Agreement

- It works and stays within the power of CFGs
- But its ugly
- And it doesn't scale all that well because of the interaction among the various constraints explodes the number of rules in our grammar.

2/9/2009

CSE842, Spring 2009, MSU

38

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)
 - LFG, HPSG, Construction grammar, XTAG, etc.
 - Chapter 15 explores the unification approach in more detail

2/9/2009

CSE842, Spring 2009, MSU

39

Treebanks

- Treebanks are corpora in which each sentence has been paired with a parse tree (presumably the right one).
- These are generally created
 - By first parsing the collection with an automatic parser
 - And then having human annotators correct each parse as necessary.
- This generally requires detailed annotation guidelines that provide a POS tagset, a grammar and instructions for how to deal with particular grammatical constructions.

2/9/2009

CSE842, Spring 2009, MSU

40

Penn Treebank

- Penn TreeBank is a widely used treebank.
 - Other Treebanks are also available
- Most well known is the Wall Street Journal section of the Penn TreeBank.
 - 1 M words from the 1987-1989 Wall Street Journal.

2/9/2009

CSE842, Spring 2009, MSU

41

Penn Treebank

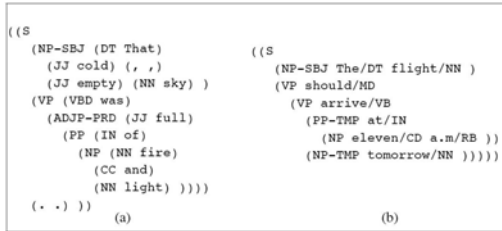
```
( ( S ( ' ' ' ' )
  (S-TPC-2
  (NP-SBJ-1 (PRP We) )
  (VP (MD would)
  (VP (VB have)
  (S
  (NP-SBJ (-NONE- *-1) )
  (VP (TO to)
  (VP (VB wait)
  (SBAR-TMP (IN until)
  (S
  (NP-SBJ (PRP we) )
  (VP (VBP have)
  (VP (VBN collected)
  (PP-CLR (IN on)
  (NP (DT those)(NNS assets)))))))))))))
( , , ) ( ' ' ' ' )
(NP-SBJ (PRP he) )
(VP (VBD said)
(S (-NONE- *T*-2) ))
( . . ) )
```

2/9/2009

CSE842, Spring 2009, MSU

42

Penn Treebank

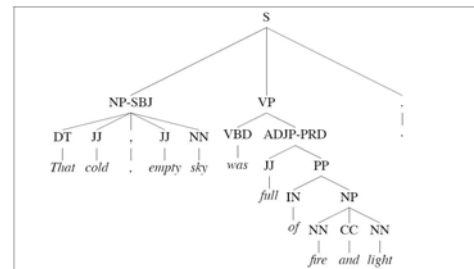


2/9/2009

CSE842, Spring 2009, MSU

43

Penn Treebank



2/9/2009

CSE842, Spring 2009, MSU

44

Treebank Grammars

- Treebanks implicitly define a grammar for the language covered in the treebank.
- Simply take the local rules that make up the sub-trees in all the trees in the collection and you have a grammar.
- Not complete, but if you have decent size corpus, you'll have a grammar with decent coverage.

2/9/2009

CSE842, Spring 2009, MSU

45

Treebank Grammars

- Such grammars tend to be very flat due to the fact that they tend to avoid recursion.
 - To ease the annotators burden
- For example, the Penn Treebank has 4500 different rules for VPs. Among them...

```

VP → VBD PP
VP → VBD PP PP
VP → VBD PP PP PP
VP → VBD PP PP PP PP
  
```

2/9/2009

CSE842, Spring 2009, MSU

46

Heads in Trees

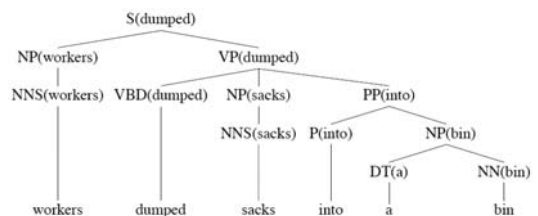
- Finding heads in treebank trees is a task that arises frequently in many applications.
 - Particularly important in statistical parsing
- We can visualize this task by annotating the nodes of a parse tree with the heads of each corresponding node.

2/9/2009

CSE842, Spring 2009, MSU

47

Lexically Decorated Tree



2/9/2009

CSE842, Spring 2009, MSU

48

Head Finding

The standard way to do head finding is to use a simple set of tree traversal rules specific to each non terminal in the grammar.

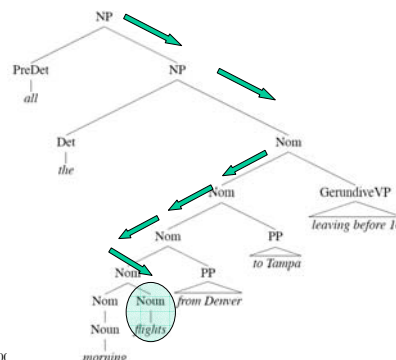
Parent	Direction	Priority List
ADJP	Left	NNS QP NN S ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB
ADVP	Right	RB RBR RBS FW ADVP TO CD JJR JJ IN NP JJS NN
PRN	Left	
PRT	Right	RP
QP	Left	S IN NNS NN JJ RB DT CD NCD QP JJR JJS
S	Left	TO IN VP S SBAR ADJP UCP NP
SBAR	Left	WHNP WHPP WHADV VP WHADJP IN DT S SQ SINV SBAR FRAG
VP	Left	TO VBD VBN MD VBZ VB VBG VBP VP ADJP NN NNS NP

2/9/2009

CSE842, Spring 2009, MSU

49

Noun Phrases



2/9/2009

50

Treebank Uses

- Treebanks (and headfinding) are particularly critical to the development of statistical parsers
 - Chapter 14
- Also valuable to *Corpus Linguistics*
 - Investigating the empirical details of various constructions in a given language

2/9/2009

CSE842, Spring 2009, MSU

51

Summary

- Context free grammars can be used to model various facts about the syntax of a language.
- When paired with parsers, such grammars constitute a critical component in many applications.
- Constituency is a key phenomena easily captured with CFG rules.
 - But agreement and subcategorization do pose significant problems
- Treebanks pair sentences in corpus with their corresponding trees.

2/9/2009

CSE842, Spring 2009, MSU

52

CSE842: Natural Language Processing

Lecture 9: Parsing with Context Free Grammar

2/9/2009

CSE842, Spring 2009, MSU

53

Syntactic Parsing

- **Declarative** formalisms like CFGs define the legal strings of a language but don't specify how to recognize or assign structure to them
- **Parsing algorithms** specify how to recognize the strings of a language and assign each string one or more syntactic structures
- **Parse trees** useful for grammar checking, semantic analysis, MT, QA, information extraction, speech recognition...and almost every task in NLP

2/9/2009

CSE842, Spring 2009, MSU

54

Parsing

- Parsing with CFGs refers to the task of assigning proper trees to input strings
- Proper 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

2/9/2009

CSE842, Spring 2009, MSU

55

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

2/9/2009

CSE842, Spring 2009, MSU

56

Example Grammar

Grammar	Lexicon
$S \rightarrow NP VP$	<i>Det</i> \rightarrow <i>that this a</i>
$S \rightarrow Aux NP VP$	<i>Noun</i> \rightarrow <i>book flight meal money</i>
$S \rightarrow VP$	<i>Verb</i> \rightarrow <i>book include prefer</i>
$NP \rightarrow Pronoun$	<i>Pronoun</i> \rightarrow <i>I she me</i>
$NP \rightarrow Proper-Noun$	<i>Proper-Noun</i> \rightarrow <i>Houston NWA</i>
$NP \rightarrow Det Nominal$	<i>Aux</i> \rightarrow <i>does</i>
$Nominal \rightarrow Noun$	<i>Preposition</i> \rightarrow <i>from to on near through</i>
$Nominal \rightarrow Nominal Noun$	
$Nominal \rightarrow Nominal PP$	
$VP \rightarrow Verb$	
$VP \rightarrow Verb NP$	
$VP \rightarrow Verb NP PP$	
$VP \rightarrow Verb PP$	
$VP \rightarrow VP PP$	
$PP \rightarrow Preposition NP$	

2/9/2009

CSE842, Spring 2009, MSU

57

Top-Down Search

- Since we're trying to find trees rooted with an *S* (Sentences), why not start with the rules that give us an *S*.
- Then we can work our way down from there to the words.

2/9/2009

CSE842, Spring 2009, MSU

58

Top-Down Parser

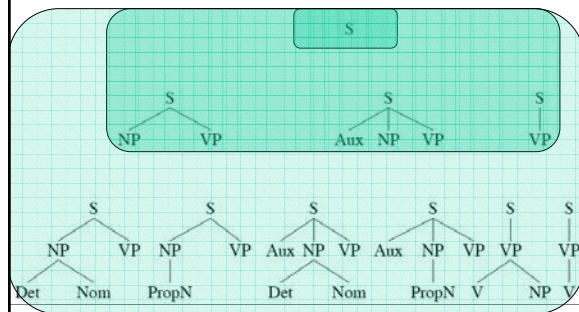
- Builds from the root *S* node to the leaves
- Assuming we build all trees in parallel:
 - Find all trees with root *S*
 - Next expand all constituents in these trees/rules
 - Continue until leaves are pos
 - Candidate trees failing to match pos of input string are rejected

2/9/2009

CSE842, Spring 2009, MSU

59

Top Down Space



2/9/2009

CSE842, Spring 2009, MSU

60

Bottom-Up Parsing

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

2/9/2009

CSE842, Spring 2009, MSU

61

Bottom-Up Parsing

- Parser begins with words of input and builds up trees, applying grammar rules w/rhs match

– Book that flight

N	Det	N	V	Det	N
Book	that	flight	Book	that	flight

– 'Book' ambiguous

– Parse continues until an S root node reached or no further node expansion possible

2/9/2009

CSE842, Spring 2009, MSU

62

Bottom-Up Search

Book that flight

2/9/2009

CSE842, Spring 2009, MSU

63

Bottom-Up Search

Verb Det Noun
| | |
Book that flight

2/9/2009

CSE842, Spring 2009, MSU

64

Bottom-Up Search

Nominal
|
Noun
|
Verb Det Noun
| | |
Book that flight

2/9/2009

CSE842, Spring 2009, MSU

65

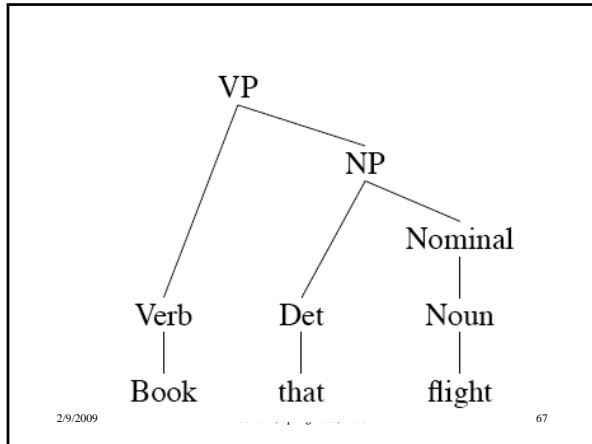
Bottom-Up Search

NP
/ \
Verb Det Nominal
| | |
Book that Noun
|
flight

2/9/2009

CSE842, Spring 2009, MSU

66



Comparing Top-Down and Bottom-Up

- **Top Down parsers** never explore illegal parses (e.g. can't form an S) - - but waste time on trees that can never match the input
- **Bottom Up parsers** never explore trees inconsistent with input - - but waste time exploring illegal parses (no S root)
- 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

2/9/2009 68

Backtracking

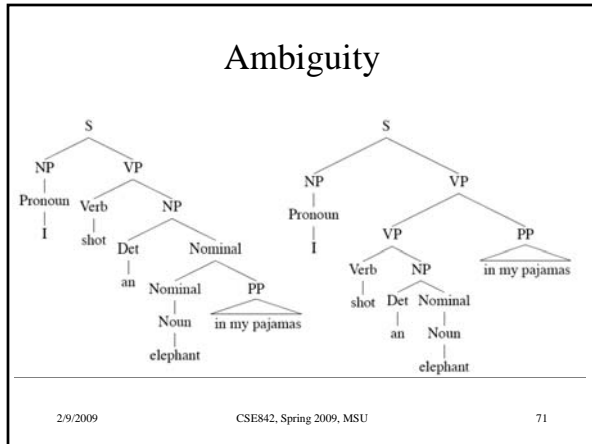
- One approach is called backtracking.
 - Make a choice, if it works out then fine
 - If not then back up and make a different choice
- Backtracking methods are doomed because of two problems
 - Ambiguity
 - Shared subproblems

2/9/2009 69

Structural ambiguity:

- Multiple legal structures
 - Attachment (e.g. **I saw a man on a hill with a telescope**)
 - Coordination (e.g. **younger cats and dogs**)
 - NP bracketing (e.g. **Spanish language teachers**)

2/9/2009 70



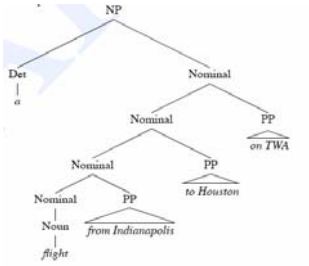
Shared Sub-Problems

- No matter what kind of search (top-down or bottom-up or mixed) that we choose.
 - We don't want to redo work we've already done.
 - Unfortunately, naïve backtracking will lead to duplicated work.

2/9/2009 72

Shared Sub-Problems

- Consider
 - A flight from Indianapolis to Houston on TWA



2/9/2009

73

Shared Sub-Problems

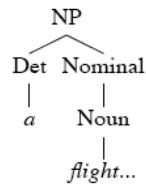
- Assume a top-down parse making choices among the various Nominal rules.
- In particular, between these two
 - Nominal- >Noun
 - Nominal- >Nominal PP
- Statically choosing the rules in this order leads to the following bad results...

2/9/2009

CSE842, Spring 2009, MSU

74

Shared Sub-Problems

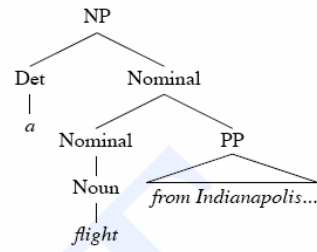


2/9/2009

CSE842, Spring 2009, MSU

75

Shared Sub-Problems

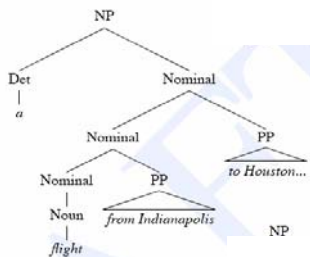


2/9/2009

CSE842, Spring 2009, MSU

76

Shared Sub-Problems

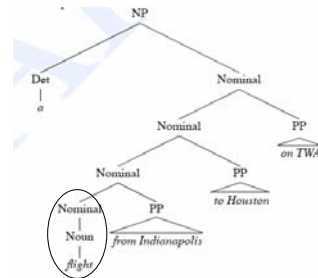


2/9/2009

CSE842, Spring 2009, MSU

77

Shared Sub-Problems



2/9/2009

CSE842, Spring 2009, MSU

78

Dynamic Programming

- Create table of solutions to sub problems (e.g. subtrees) as parse proceeds
- Look up subtrees for each constituent rather than re parsing, avoiding repeated work.
- Since all parses implicitly stored, all available for later disambiguation
- We will look at two approaches corresponding to top down and bottom up
 - CYK: Cocke-Younger-Kasami (CYK) (1960),
 - Earley: Earley (1970)