English Syntax and Parsing Algorithms

Adam Meyers

New York University

Outline

- Context Free Grammar for part of English
- A Simple Generator
- How are CFGs used to model linguistic concepts?
- Recognizers and Parsers
- More linguistics: Heads & Subcategorization
- Some alternative frameworks
- A More Thorough English Grammar
- Chunking: Parsing on the Cheap

English'
A 'Model' of a Subset of English

- N = {S, VP, NP, POSSP, PP, N, P, D, POSS}
- T ={the, a, this, that, food, clam, discussion, group, table, room, Bill, Mary, sincerity, knowledge, redness, of, about, on, in, 's, ate, saw, had, put}
- S=Sent
- R = Rules listed on next slide

Grammar for English' NonTerminal Rules

- Sent(ence) → NP VP
- VP → V(erb)
- VP → V(erb) NP
- VP → V(erb) NP PP
- NP → POSSP N(oun) PP
- NP → POSSP N(oun)
- POSSP → NP POSS
- PP → P(reposition) NP
- NP → D(eterminer) N(oun)
- NP → N(oun)

Grammar of English' Terminal Rules

- D → the, D → a, D → this, D → that,
- N → food, N → clam, N → discussion, N → group, N → table, N → room, N → Bill, N → Mary, N → sincerity, N → knowledge, N → redness
- P → of, P → about, P → on, P → in, P → with
- POSS → 's
- V→ ate, V→ saw, V→ had, V→ put

A Random Sentence Generater

- Algorithm
  - Set Output to S
  - Repeat Until Output Contains no NonTerminals
    - Replace the 1st Nonterminal (Q) in Current String
      - Find all phrase structure rules with Q on the left hand side
        - » Choose a rule Q → α, where α is a string of terminals and/or nonterminals
      - Replace Q with α
- What kinds of objects get generated?
  - Are they all well-formed?
  - If ill-formed, how are they ill-formed?

Implementation of Generator

- random_sentence3.py
- Slightly different (bigger grammar)
  - Variable ***rules*** set to a list of lists
    - 1st item in each list is a nonterminal
    - Remaining items are possible expansions of nonterminal
    - Example: ***DetP --> (NP pos) | (determiner)***
  - The file: words3.py assigns POS to words
- Possible expansions for each symbol stored in table
- generate_random_phrase
  - unexpanded nonterms in stack (starting w/initial symbol)
  - repeat until stack empty
    - remove top item
      - if terminal, add to result,
      - if non-terminal, add expansion stack (left most item on top)

Evaluating Strings of Terminals

- A formal language is a set of strings of symbols
- Members of English' and English
  - ***Mary ate food***
- Member of English', but not member of English
  - ***Clam saw the Mary***
  - English' is an imperfect model of English
- Semantically ill-formed
  - ***The redness ate sincerity***
- Not a member of English' or English (Word Salad)
  - ***Clam discussion the the knowledge***

Phrase Structure Typically Models the Distribution of Words and Sequences

- Most linguistic theories assume ways of testing if 2 units belong to the same category or if a sequence forms a unit (or constituent). To some degree these syntactic units correspond to semantic ones.
- There is some variation among accounts as to whether most, some or any of these tests are being modeled by the grammar
- Alternate view: the grammar is adequate iff all and only the strings of the language can be generated by the grammar. This may not necessitate that constituents or category labels model anything linguistic or semantic.
  - Example: It can be useful to convert the grammar to Chomsky normal form (CNF), converting all rules to either:
    - Nonterminal → Terminal
    - Nonterminal1 → Nonterminal2 Nonterminal3

    Automatic mappings to Nonterminal2 and 3 do not typically model linguistic features. This is assumed by the CKY parsing algorithm. Similar views are implied in some linguistic theories (theories that impose a syntactic template on all phrases)

Same Category Tests

- If 2 units A and B have the same phrase label (or POS), then exchanging A for B or B for A, should not change whether a sentence is syntactically well-formed.
  - *NP read the book* | NP ∈{*Mary, The pianist with a hat, She, The uncooked eggplant,* …}
  - *They VP and VP* | VP ∈{*saw a movie, wrote poetry, are politicians, ...*}
- *Two words that participate in the same inflectional paradigm are probably the same part of speech*
  - *{kill,kills,killed,killing}; {love,loves,loved,loving}*
  - *{big,bigger,biggest}; {silly,sillier,silliest}* …
  - *{house, houses}; {nose, noses}* …

Constituency Tests

- Sequences behave as units across corresponding sentences.
  - *[In this story], there are 3 pigs ↔ There are 3 pigs [in this story]*
  - *The clam scared [the tourist with a wig] ↔ [The tourist with a wig] was scared of the clam.*
- Only units can conjoin; should be of the same type
  - *[eat] and [drink]; [read books] and [see movies]; [in the house] and [on the porch]*
  - Exceptions (with explanations):
    - *She is a thief and extremely wealthy*
    - *He is angry and in a bad mood*
    - *John did eat and Mary did not eat dinner*
- Units can occur as fragments, e.g., answers to questions
  - *What do you want? **A toothbrush with golden bristles***
- Anaphora resolution
  - *Fish really do **eat their children**. Humans wouldn't do **that**.*

Convert English' NonTerminal Rules to CNN

- Replace ***VP* → *V*** & ***NP* → *N*** with nonbranching rules expanding VP and NP to actual words:
  - ***VP* → *ate, VP* → *had, NP* → *food, …***
- Replace ***VP* → *V NP PP*** with 2 rules:
  - ***VP* → *VG PP***
  - ***VG* → *V NP***
- Replace: ***NP* → *POSSP N PP*** with 2 rules:
  - ***NP* → *NG PP***
  - ***NG* → *POSSP N***
- NG = Noun Group and VG = Verb Group

Math Terms: Set Theory

- Quick Overview: set, member of, subset, superset, proper subset, proper superset
- Cartesian Product
  - Set A = {1,2,3}, Set B = {a,b,c}
  - A × B = {{1,a},{1,b},{1,c},{2,a},{2,b},{2,c},{3,a},{3,b},{3,c}}
- Partition = division into multiple non-overlapping subsets, including all members of original set

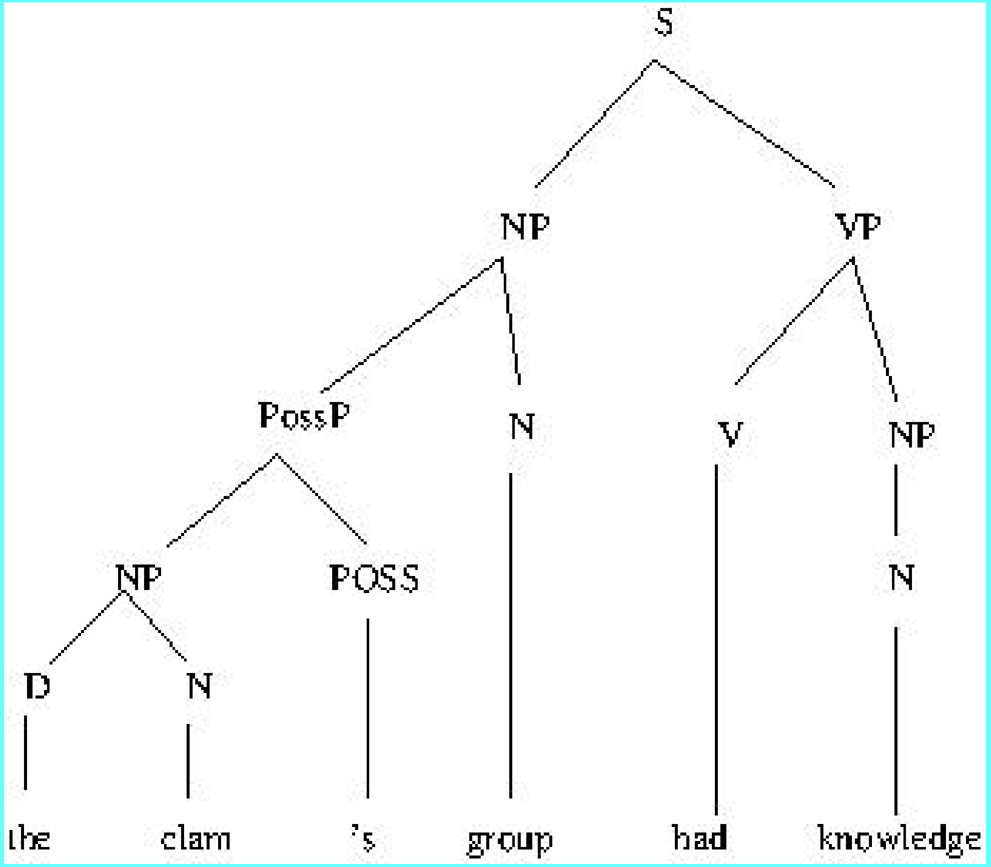
Math Terms: Set Theory 2

- Binary Relation = set of ordered pairs
- Properties of Relations
  - Reflexive: (X,X) is always in relation
  - Irreflexive: (X,X) is never in relation for any X
  - Transitive: (X,Y) and (Y,Z) $\rightarrow$ (X,Z)
  - Antisymmetric: (X,Y) and (Y,X) $\rightarrow$ X=Y
  - Asymetric: If (X,Y), then not (Y,X)

Linguistic Phrase Structure Tree

- Phrase Structure Tree = <N,Q,D,P,L>
  - N = set of nodes
  - Q = set of labels
  - D = weak partial order on N X N (the dominance relation)
    - Reflexive, transitive & antisymmetric (X & Y dominate each other iff X = Y)
  - P = strict partial order on N X N (the precedence relation)
    - Irreflexive, transitive & asymmetric (if X precedes Y & Y cannot precede X)
  - L = function from N into Q (labeling function)
- Conditions
  - Single Root (tree dominated by a single (start) nonterminal symbol)
  - A node pair cannot be both part of D and part of P
    - Only nonterminals symbols can label nonterminal nodes
  - Branches cannot cross
    - Branch A dominates nodes a1 and a2
    - Branch B dominates nodes b1 and b2
    - If a1 precedes b1, b2 cannot precede a2
  - [A CFG and a simple generator should not produce a tree violating these]

Sample English' Linguistic Phrase Structure Tree



Bracketing Representation of Linguistic Tree

```
(S (NP (POSSP (NP (D the) (N clam))
                      (POSS 's)
        (N group))
   (VP (V had)
         (NP (N knowledge))))
```

- Commonly used for treebanks and parsers

English" NonTerminal Rules Used in Example

- Sent(ence) → NP VP
- VP → V(erb)
- VP → V(erb) NP
- NP → POSSP N(oun)
- POSSP → NP POSS
- PP → P(reposition) NP
- NP → D(eterminer) N(oun)
- NP → N(oun)

CKY Recognizer and Parser

- Create a (triangular) table representing all spans in the sentence from 0 (the position before the first word) to N the position after sentence of length N
- For j from 1 to N do:

```
Fill in one span of length 1 using a POS rules, e.g., V → ate
    On different iterations these will be [0, 1], [1, 2], …, [N-1, N]
For i from 0 to j-2 do:              ### Note: J&M does this in reverse
    for k from i+1 to j-1:
        Add all matching nonterminals to [i,j] in table * ## fill in rest of column j
```

- * A nonterminal matches iff
  - There is some rule of the form A--> BC in the grammar
  - [i,k] includes the label B and [k,j] includes the label C

  Note that the inner 2 loops identify all [i,k] amd [k,j] such that i is between 0 and j-2, i<k and k<j, thus identifying the possible binary partitions of [0,j]
- Parser needs 2 additional things:
  - Differentiate different expansions of same nonterminal
  - Record pointers to matched children of nonterminals

Python Code Trace

- Do trace with python-CKY-loop.py
- Code outlines lookup instance in the CKY algorithm:
  - Looking up lexical items in outer loop
  - Looking up rule matches in loop 3
- Algorithm hinges on binary branching rules
- Adjustments of pseudo-code for python
  - range(N) of python is a list of numbers that is N long starting with 0
    - Range(5) = [0,1,2,3,4]
    - Implementing the pseudo-code, requires adding 1 to some of the variables as a result

Recognize with CKY & Grammar' Outer Loop 1st Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]					
1						
2						
3						
4						
5						

2nd Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]	NP [0,2]				
1	■	N, NP [1,2]				
2	■	■				
3	■	■	■			
4	■	■	■	■		
5	■	■	■	■	■	

3rd Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]	NP [0,2]	POSSP [0,3]			
1		N, NP [1,2]	POSSP [1,3]			
2			POSS [2,3]			
3						
4						
5						

4th Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]	NP [0,2]	POSSP [0,3]	NP [0,4]		
1		N, NP [1,2]	POSSP [1,3]	NP [1,4]		
2			POSS [2,3]			
3				N, NP [3,4]		
4						
5						

5th Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]	NP [0,2]	POSSP [0,3]	NP [0,4]	S [0,5]	
1		N, NP [1,2]	POSSP [1,3]	NP [1,4]	S [1,5]	
2			POSS [2,3]			
3				N,NP [3,4]	S [3,5]	
4					V, VP [4,5]	
5						

6th Iteration

	The	clam	's	group	had	knowledge
	1	2	3	4	5	6
0	D [0,1]	NP [0,2]	POSSP [0,3]	NP [0,4]	S [0,5]	S [0,6]
1		N, NP [1,2]	POSSP [1,3]	NP [1,4]	S [1,5]	S [1,6]
2			POSS [2,3]			
3				N,NP [3,4]	S [3,5]	S [3,6]
4					V, VP [4,5]	VP [4,6]
5						N,NP [5,6]

Chart Parsers/Recognizers

- States (or Edges) represent partial processing of the sentence
  - Phrase structure rule + text span + indication of part of rule is used
  - A Dot indicates how much of rule is used:
    - **NP → . det Adjp noun [0,0]** ## nothing has been recognized
    - **NP → det . Adjp noun [0,1]** ## a sentence initial det was found
- New states are created
  - to account for new or larger portions of input string
  - based on previously created states
- States are recorded in a "chart"
- A Final State
  - Represents the whole sentence
  - Instantiates a completed expansion of the start symbol

A Generic Chart Parser (13.4 in J&M)

- chart-parse(words,grammar, strategy):

```
Initialize(chart,agenda,words): Add initial “edges” (or states) into chart and agenda
while agenda is non-empty:
        current_edge ← pop_next_edge(agenda)
        process_edge(current-edge)   # may change agenda & chart
return(chart)
```

- process_edge(**edge**):

```
Add edge to chart (if not already there)
IF incomplete(edge):
    For each complete_edge  that: a) starts where edge ends and b) matches the symbol after the dot in edge
    Add a modified edge to agenda, such that the dot in edge is advanced past the next symbol
ELSE if edge spans the input string and matches the start symbol: then do nothing
        ### [Possibly end processing if only 1 parse is desired]
ELSE:
    For each incomplete_edge such that left_side(edge) is after the dot:
      – Add modified incomplete_edge to agenda with dot advanced one symbol
Make_prediction(edge)
    Add additional edges to agenda based on edge – details depend on parsing strategy
```

Alternative Parsing Strategies

- Direction: Left to Right, Right to Left (order in which rules beginning with terminals are placed on agenda)
- Search Strategies
  - Bottom Up: complete constituents 1st, start w/terminals
  - Top Down: nonterminals 1st, start with start symbol
  - Breadth 1st – investigate constituents in rule in parallel
  - Depth 1st – investigate constituents in rule 1 at a time
  - Others: Left corner parsing, head-driven parsing, etc.
- Optimizations
  - Dynamic Programming

The concept of Head

- Head of a phrase – word that determines the category of the phrase
  - *the **book** about Fred* – NP because *book* is a noun
  - ***ate** a sandwich* – VP because *ate* is a verb
  - ***in** the room* – PP because *in* is a preposition
- Some phrases arguably don't have heads
  - *John and Mary*
  - *Adam Meyers*
  - *the bigger, the better*
- Heads typically subcategorize (select) other members of their phrase
  - $VP \rightarrow V_{put}\ NP\ PP{+}LOC$ *He put the book on the table*
  - $VP \rightarrow V_{laugh}\ PP\text{-}at$ *She laughed at the poodle*
- CFG representations can break up POS into subcategories which correspond to these different selectional possibilities

Typical Criteria for Head

- The single word that determines the distribution of the phrase
  - Bloomfield 1933
- The head of the phrase selects for the other members of the phrase
  - Subcategorization
  - Semantic selection, e.g., ***eat*** requires that its object be tangible (one cannot eat an idea). In NLP, a notion of statistical selection is used instead, e.g., typical objects of ***eat*** are edible (not rocks).
- Heads tend to determine the agreement properties of phrases (number and gender)
- Problem: Sometimes different items in a phrase fullfill these roles, e.g., an adjective selects the head noun, e.g., ***angry*** modified sentient head nouns.

Subcategorization Dictionaries

- NYU built one of the widely used dictionaries of this type
  - http://nlp.cs.nyu.edu/comlex/
  - Subcategorization for nouns, verbs, adjectives
  - Modification frames for adverbs
- Examples:

```
(NOUN :ORTH “authority”
        :SUBC ((NOUN-FOR-TO-INF)
               (NOUN-PP :PVAL (“by” “for” of” “over” …))))
(VERB :ORTH “put”
       :SUBC ((NP-PP :PVAL (“on” “in” “into” “off” “out” …))
              (PART-PP :ADVAL (“up” “in”) :PVAL (“with” “for”))
              (NP-ADVP) …))
(ADVERB :ORTH “right”
        :MODIF ((PRE-PREP :PVAL (“outside” “in” “into” “on” “out of” …))
                (PRE-ADV)))
```

Argument Sharing Properties in the Lexicon

- Raising (subj-to-sub, subj-to-obj—aka ECM)
  - Verbs/Adjectives/Nouns link predicates and arguments
    - *The student is **likely** to leave (the student's likelihood of leaving)*
    - *The student **seemed** to leave*
    - *The student **believed** the test to have only 1 question.*
- Equi (aka control)
  - Arguments are shared between two predicates
    - *The student **desires** to leave (the student's **desire** to leave*)
    - She ***promised*** her mother to be good.

Adverbs

- Not well-defined part of speech in some systems
  - “adverb” often assigned to words not fitting other categories
- Mostly “adverb” refers that “modify” a non-noun and/or express locative, verbal, evaluative concepts.
  - by modify, I mean not selected by head of phrase
- One way adverbs are classified in COMLEX is by what they modify
- Examples of modifiers of determiners or numbers
  - *all, barely, just, only, quite, scarcely*
  - *all 5 children, scarcely a solution, …*
- Examples of modifiers of prepositions or adverbs
  - *all, just, less, quite, really, rather, right, somewhat, tightly, very*
  - *He climbed right up the wall, They walked rather slowly, …*

Adverbs 2

- Examples of modifiers of adjectives
  - *absolutely, deeply, legally, only, not, slightly, very*
  - *He was **deeply** hurt, She was **legally** liable, They were both **very** angry*
- Some adverb modifiers of adjectives take complements as well
  - *The type was **too** small **to read***
- Examples of modifiers of verbs/sentences
  - *easily, probably, never, not, actively, then, immediately*
  - *She **actively** looked for her glasses, She **probably** found them, **Then**, I saw them at the bottom of the stairs*
  - There are further distinctions for these:
    - variation where they can occur
      - intially, finally, between subject and verb, etc.
    - Sentential adverbs are distinguished from verbal ones, but there is controversy in the details.
      - E.g., Epistemic (probably, possibly, ...) are usually considered sentential

Frameworks in CL that go beyond CFGs

- Dependency Grammar: Words are linked to form graphs
  - No concept of phrase
  - Nonheads depend on heads (so all constructions must have heads, even for arguably non-headed constructions, e.g., conjoined phrases)
- Categorial Grammar
  - There is a syntax of POS and Category names
    - Solves some of the problems with the notion head
    - An adjective is a category of type N/N because it is an item that combines with an N to produce an N
      - N/N X N = N
    - A VP is a S/NP because it combines with an NP (subject) to form an S
      - S/NP X NP = S
- Feature Structure Grammars (HPSG, LFG, etc.)
  - Generalize CFG to large attribute value structures
  - See GLARF (nlp.cs.nyu.edu/meyers/GLARF.html)

A More Complete English Grammar

- Scope of this discussion:
  - Surface Syntax (which could conceivably be captured by phrase structure rules)
  - Subphrases of Sentences: (NP, VP, PP, ADJP, ADVP ...)
  - Ignoring for now:
    - Grammar beyond the “surface”: paraphrase relations (e.g., between active and passive, filling in of 'gaps', etc.), semantics, pragmatics, etc.
    - Nondeclarative clauses (questions, exclamations, ...) & exotic constructions
- Open Class vs Closed Class
  - Open class: parts of speech with an opened set of members. Newly coined words usually fall into these classes: nouns, verbs, adjectives, some adverb classes
  - Closed class: words with highly idiosyncratic grammatical functions. New words rarely fall into these classes. Ex: prepositions, subordinate conjunctions, coordinate conjunctions, infinitival “to”, “as”, etc.

Noun Phrases

- NP → Determiners Adjectives Nouns-as-Modifiers Head-Noun Noun-Complements Noun-post-modifiers
  - *[The happy [ice cream truck] salesman [sitting on the chair]]*
  - *[All the little piggies [that live piggy lives]]*
  - All elements except the noun are optional (sort of)
- Determiner (and DetPs) – additional rules required for multiple determiners
  - possessive phrases (NP + 's)
  - articles and demonstratives: *the, a, an, this, that, these, those*
  - numbers: *1, 2, 3, 4, 53, 175, one hundred thirty, five hundred and thirty*
  - Quantifiers: *every, all, each,* some, ….
- Adjectives (and adjective phrases and verb particples): angry, very angry
- Noun-Phrases-as-Modifiers: *the [ice cream] man, a [car insurance] policy*
  - An NP as a modifier typically: lacks a determiner and has a singular form
    - ** the [the car insurance] policy*
    - Exception: *the [three woman] band*

Right Modifers of Nouns

- Complements selected by nouns (PP or Clause)
  - *The fact [that one plus one equals two]*
  - *His anger [about the situation]*
- Non-complement PPs (ownership, location, time)
  - *The book [on the table], the book [of John's]*
- Relative clauses (missing subject, object or other)
  - *The fact [that she forgot]* --- missing object of *forgot*
- Reduced relative clauses (participles or adjectives with relative like meanings)
  - *The book [__ sitting on the table]*
  - *The people [__ angry at Simon]*
- Appositive Phrase: A second noun phrase modifying the first. It is typically separated by a pause (punctuation), and has a “that is” type of meaning.
  - *John Smith, the new president of ACME*
  - *The palamino, the horse-shaped subatomic particle*

Verb Phrases

- Verb + Complements + Adverbial Modifiers
  - *put [the book] [on the table] quietly*
- Complements: big variety – see COMLEX website http://nlp.cs.nyu.edu/comlex
  - Complements for verbs can be obligatory
- Auxilliaries: closed class words that in some grammars are labeled as types of verbs. Most theories assume a binary structure like this:
  - (VP aux (VP aux .. (VP verb …)))
  - infinitival ***to*** and modals ***(may, can, could***, …) are followed by the base form of a verb
  - Forms of ***be*** are followed by either ***-ing*** forms of verbs (progressive) or past participle forms (passive) (past particples end in ***-en*** or ***-ed***)
  - Forms of ***have*** are followed by past participle forms of verbs (perfect)

Adjective Phrases

- ADJP → (ADV) ADJ Complement
  - *very angry about the situation*
- Most adjectives can occur either:
  - Attributively (as noun modifiers): ***the angry bird***
  - or as predicates (after ***be*** or other special verbs)
    - *The bird was angry, The bird got angry, They considered the bird stupid*
- Some adjectives can only occur one way
  - *The bird was alive. *That is an alive bird*
  - *He is the former president. *That president is former.*
- Prenominal ADJP do not include their complements, except for special adjectives that allow a complement after the noun
  - **The easy politician to please**
  - **??The easy to please politician** ## In written text, better if hyphenated
  - ***The angry at them people** ## Always bad

Some Other Closed Class Words

- Coordinate Conjunctions: *and*, *or*, *but*
  - Combine like constituents
    - XP → XP and XP
    - X → X and X
- Subordinate Conjunctions: *if*, *while*, *before*, ..
  - Combine clauses in special logical, temporal or discourse relationships (not equal like coordinate conjunctions)
- Prepositions:
  - Can have purely formal functions or semantic ones (similar to subordinate conjunctions)
  - Link NPs with other words/phrases

Chunking

- Linguistic transduction that selects short approximations of linguistic phrases or chunks
- Most Common Chunks:
  - Verb Group: A unit beginning at an auxilliary verb and ending at a matrix verb or particle (that immediately follows the matrix verb) – semantic justification possible, but syntactic one is not:
    - *The cars [may have been slowly stripped down]*
    - *They [could have stolen] the cars instead*
  - Noun Group: Unit consisting of a noun & its pre-modifiers
    - *[The big bad wolf] that huffed and puffed*
    - *[President G. W. Bush], [a goofy president] who …*
- It is possible to create quick & accurate processors

Chunking 2: Methodology

- Finite state transducer
  - Using a left regular grammar similar to English'
- ML Chunker described in J & M
  - Assumes a pre-processing stage of POS tagging
  - Assumes an annotated corpus for “training”
    - Four types of tags for each chunk type:
      - Beginning (B_NG)
      - Interior (I_NG)
      - Outside (O)
    - *The*/**B_NG** *big*/**I_NG** *cat*/**I_NG** *without*/**O** *a*/**B_NG** *tail*/**I_NG** *caught*/**O** *a*/**B_NG** *mouse*/**I_NG**
  - 13-tuple for each word includes: Chunk tag, word, word's POS, 2 words to left, 2 words to right, and POS info for 2 words to left and right
  - Frequency of subsets of these fields are recorded in training and used for final system to predict tags (most frequent matching pattern is used)

Chunking 3: Sample Regexp Rules for Chunks

- NG → (Det)* (AdjP)* (Noun)* Noun
- AdjP → (Adv)* Adj
- VG → (Adv)* (Aux)* (Adv)* (Verb | VerbPart)
- VerbPart → (call up) | (break up) | (call out) |…
  - disjunction of 500 verb particle combinations and their inflectional varients
- Det → the|a|each|every|this|that|these|…
  - Noun, Adv, Adj, Aux, Verb are also defined as disjunctions of words taken from a dictionary

Some Additional Sources

- Partee, et. al. (1990) *Mathematical Methods in Linguistics* – Gives good mathematical background to formalize linguistic concepts
- Descriptive Grammars
  - McCawley (1988) *The Syntactic Phenomena of English* (Vol 1 and 2)
  - Huddleson and Pullum (2002) *The Cambridge Grammar of the English Language*
- Introductory Books for Syntactic Theories
  - Carnie (2006) *Syntax* – Introduction to Chomskian linguistics
  - Borsley (1996) *Modern Phrase Structure Grammar* – Introduction to feature structure approaches to Syntax
  - Wood (1993) *Categorial Grammars*
  - Dependency Grammar – still looking for good intro: Prague Dependency Treebank, Kyoto Corpus, …
- Parsing and Chunking: See citations in J & M
- Interesting application of CFG generation: http://pdos.csail.mit.edu/scigen/
  - Randomly generates CS papers for submission to marginal conferences

Readings

- Jurafsky and Martin, Chapters 12 and 13
- NLTK – Chapter 8 – You need to read at least enough to try out the parser (as per the next slide).

Homework #3 – Slide 1

- 1. Manual Parse
  - Find a long sentence (at least 20 words) from a news article
  - Assign it a phrase structure and list the rules you would need to account for it.
  - You may base the rules on English', Jurafsky and Martin section 12.3 or a combination
  - Note any interesting problems you come across
  - Try to have your nonterminals approximately model semantic units and/or obey constituent tests, but do not get stuck on this
- 2. Try your sentence with the CFG parser from NLTK, chapter 8
  - You should use a set of rules that you believe should make the parser handle your sentence.
  - If something goes wrong, describe how it goes wrong and hypothesize why.
    - Ambiguity: ***I shot an elephant in my pajamas***
    - Garden Paths: *The horse raced past the barn fell* → *The horse that was raced past the barn fell*
    - Some combinations of sentences, grammars and parsers will not terminate in a reasonable amount of time (or at all) – if your sentence does not terminate for a particular parser/grammar, try to figure out why. or simply document it.
    - Other problem
  - Try other sets of rules to fix any problem, but it is OK if the system does not properly parse your sentence. You should make an attempt at understanding what happens and try to describe it.
- 3. Try sentence with the shift reduce parser and left corner parsers and discuss differences in performance.

Homework #3 – Slide 2

- 3. Optional: Modify the random sentence generator downloadable from this website or write your own using a different strategy and/or different programming language. Include improvements of your choice. Example improvements are:
  - Provide a way to prefer some expansions over others and modify the grammar to exploit this, e.g., noun noun modification is not so frequent.
  - Account for any of the following:
    - subcategorization of verbs, nouns and/or adjectives
    - coordination
    - compatibility of determiners with singular/plural distinction of nouns
    - Subject/verb agreement
    - Or anything else that would make the output more likely to be well-formed