Context Free Grammars

Lecture #5

SNU 4th Industrial Revolution Academy:
Artificial Intelligence Agent

Grammaticality

Doesn't depend on

- Having heard the sentence before
- The sentence being true
  - I saw a unicorn yesterday
- The sentence being meaningful
  - Colorless green ideas sleep furiously
  - *Furiously sleep ideas green colorless

Grammatically is a formal property that we can investigate and describe

Syntax

By syntax, we mean various aspects of how words are strung together to form components of sentences and how those components are strung together to form sentences

- New Concept: Constituency
- Groups of words may behave as a single unit or constituent
- E.g., noun phrases
- Evidence
  - Whole group appears in similar syntactic environment
  - E.g., before a verb
  - Preposed/postposed constructions
  - Note: notions of meaning play no role in syntax (sort-of)

What is Syntax?

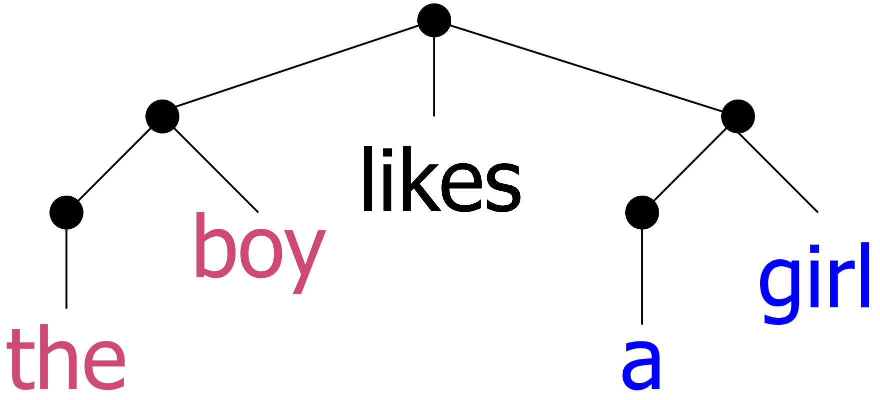
- Study of structure of language
- Specifically, goal is to relate surface form (e.g., interface to phonological component) to semantics (e.g., interface to semantic component)
- Morphology, phonology, semantics farmed out (mainly), issue is word order and structure
- Representational device is **tree structure**

What About Chomsky?

- At birth of formal language theory (comp sci) and formal linguistics
- Major contribution: syntax is **cognitive** reality
- Humans able to learn languages quickly, but not all languages $\Rightarrow$ **universal grammar** is biological
- Goal of syntactic study: find universal **principles and** language-specific **parameters**
- Specific Chomskyan theories change regularly
- These ideas adopted by almost all contemporary syntactic theories ("principles-and-parameters-type theories")

From Substrings to Trees

- (((the) boy) likes ((a) girl))

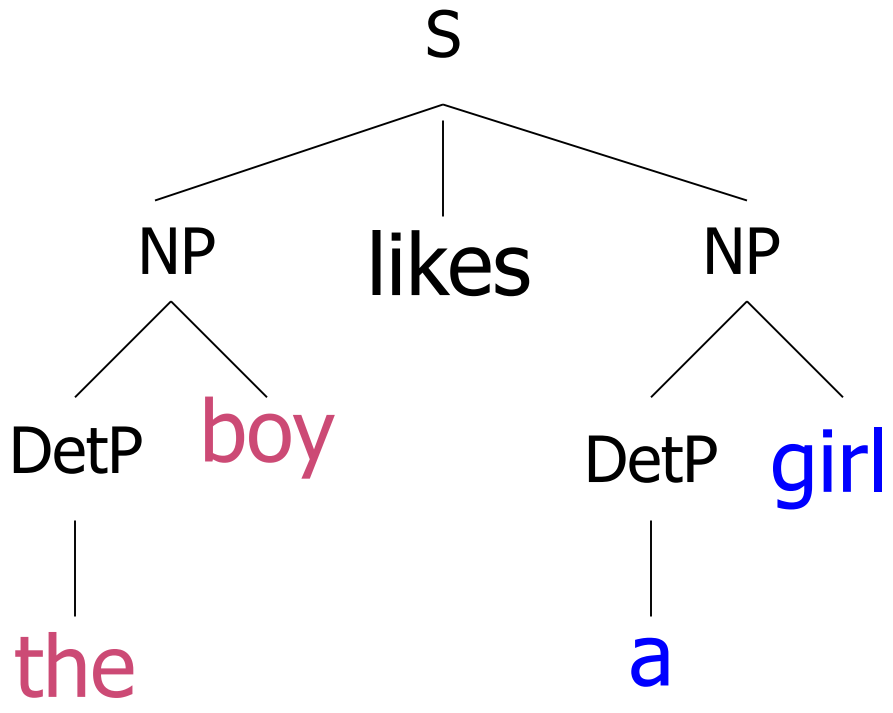


Node Labels?

- (((the) boy) likes ((a) girl))
- Choose constituents so each one has one non-bracketed word: the **head**
- Group words by distribution of constituents they head (part-of-speech, POS):
  - Noun (N), verb (V), adjective (Adj), adverb (Adv), determiner (Det)
- Category of constituent: XP, where X is POS
  - NP, S, AdjP, AdvP, DetP

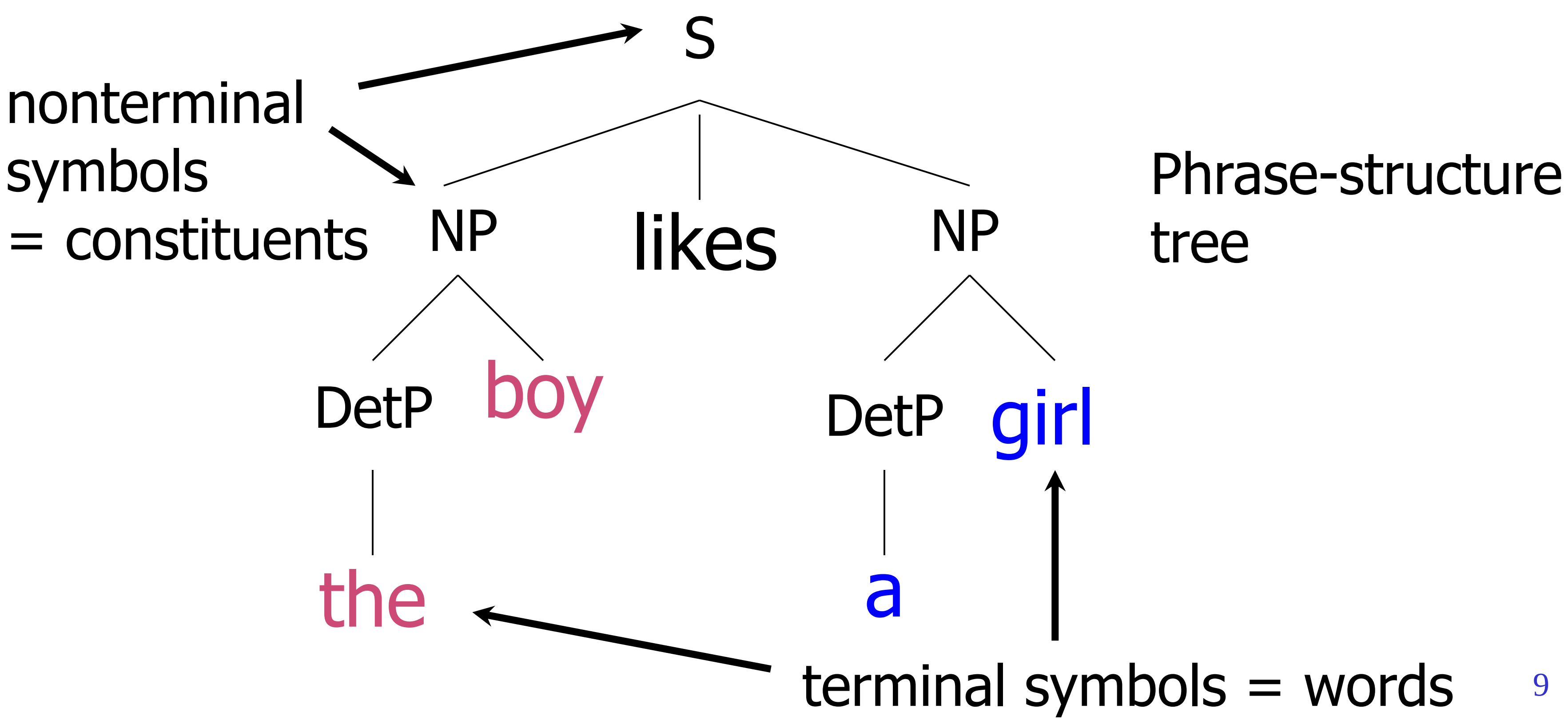
Node Labels

- (((the/Det) boy/N) likes/V ((a/Det) girl/N))



Types of Nodes

- (((the/Det) boy/N) likes/V ((a/Det) girl/N))



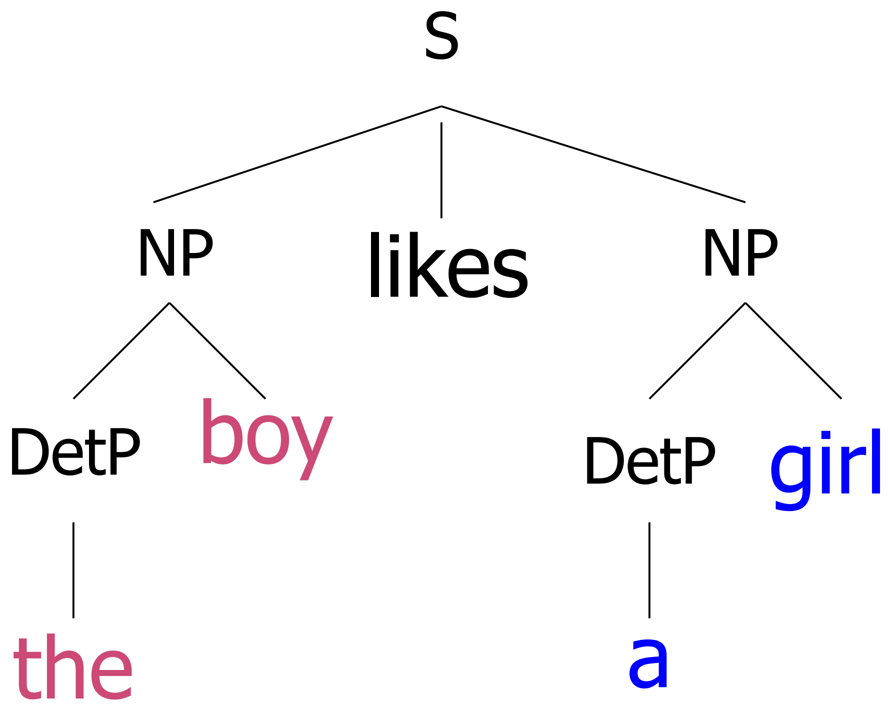
Determining Part-of-Speech

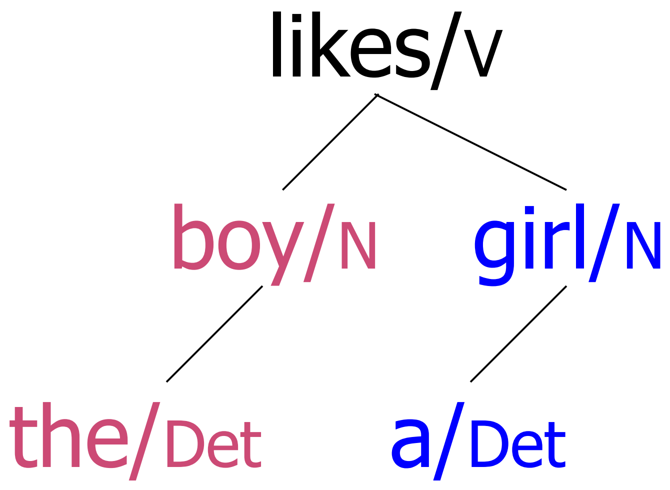
- noun or adjective?
  - a **child** seat
  - a blue seat
  - *a very **child** seat
  - *this seat is **child**
  - It's a noun!
- preposition or particle?
  - he threw the garbage **out** the door
  - *he threw the garbage the door **out**
  - he threw **out** the garbage
  - he threw the garbage **out**

Word Classes (=POS)

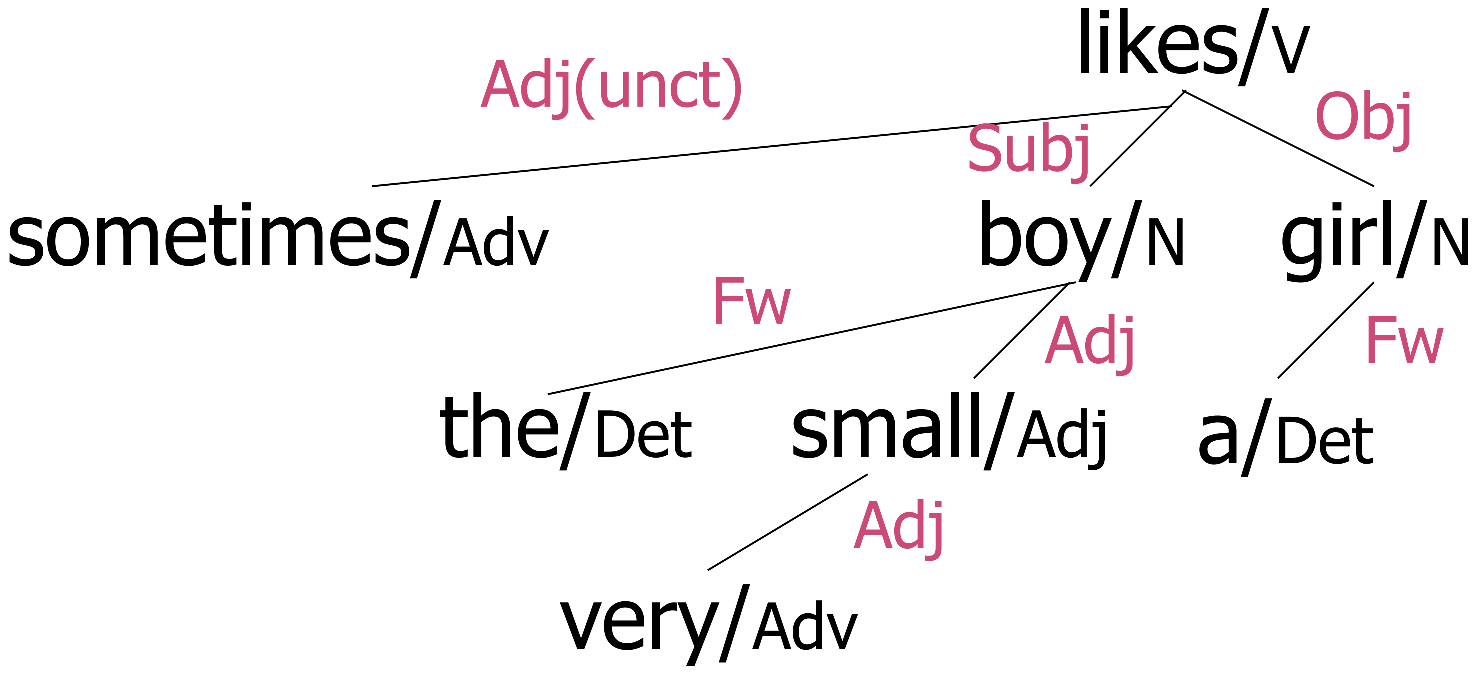
- Heads of constituents fall into distributionally defined classes
- Additional support for class definition of word class comes from morphology

Phrase Structure and Dependency Structure





Types of Dependency



Grammatical Relations

- Types of relations between words
  - Arguments: subject, object, indirect object, prepositional object
  - Adjuncts: temporal, locative, causal, manner, …
  - Function Words

Subcategorization

- List of arguments of a word (typically, a verb), with features about realization (POS, perhaps case, verb form etc)
- In canonical order Subject-Object-IndObj
- Example:
  - like: N-N, N-V(to-inf)
  - see: N, N-N, N-N-V(inf)
- Note: J&M talk about subcategorization only within VP

Context-Free Grammars

- Defined in formal language theory (comp sci)
- Terminals, nonterminals, start symbol, rules
- String-rewriting system
- Start with start symbol, rewrite using rules, done when only terminals left
- NOT A LINGUISTIC THEORY, just a formal device

Context-Free Grammars

Context-Free Grammars

- A CFG is a 4-tuple: (N, T, P, S), where
  - N is a set of non-terminal symbols,
  - T is a set of terminal symbols which can include the empty string ϵ. T is analogous to Σ the alphabet in FSAs.
  - P is a set of rules of the form $A \rightarrow \alpha$, where $A \in N$ and $\alpha \in \{N \cup T\}^*$
  - S is a set of start symbols, $S \in N$

Chomsky Hierarchy

Grammar	Languages	Automaton	Production rules
Type-0	Recursively enumerable	Turing machine	$\alpha \rightarrow \beta$ (no restrictions)
Type-1	Context-sensitive	Linear-bounded non-deterministic Turing machine	$\alpha A\beta \rightarrow \alpha\gamma\beta$
Type-2	Context-free	Non-deterministic pushdown automaton	$A \rightarrow \gamma$
Type-3	Regular	Finite state automaton	$A \rightarrow a$ and $A \rightarrow aB$

CFG: Example

- Many possible CFGs for English, here is an example (fragment):
  - S $\rightarrow$ NP VP
  - VP $\rightarrow$ V NP
  - NP $\rightarrow$ DetP N | AdjP NP
  - AdjP $\rightarrow$ Adj | Adv AdjP
  - N $\rightarrow$ boy | girl
  - V $\rightarrow$ sees | likes
  - Adj $\rightarrow$ big | small
  - Adv $\rightarrow$ very
  - DetP $\rightarrow$ a | the

the very small boy likes a girl

Derivations in a CFG

S

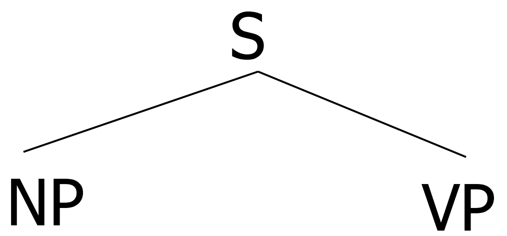
S → NP VP
VP → V NP
NP → DetP N | AdjP NP
AdjP → Adj | Adv AdjP
N → boy | girl
V → sees | likes
Adj → big | small
Adv → very
DetP → a | the

S

Derivations in a CFG

NP VP

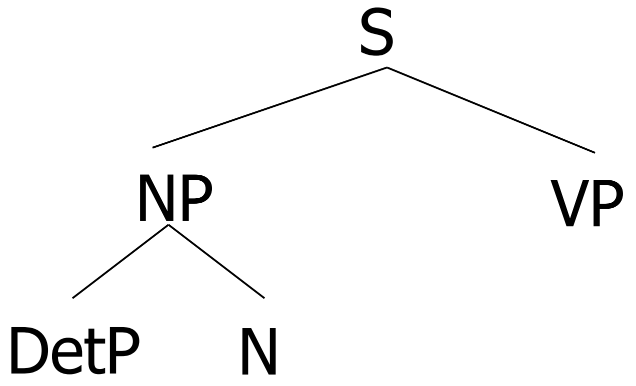
S → NP VP
VP → V NP
NP → DetP N | AdjP NP
AdjP → Adj | Adv AdjP
N → boy | girl
V → sees | likes
Adj → big | small
Adv → very
DetP → a | the



Derivations in a CFG

DetP N VP

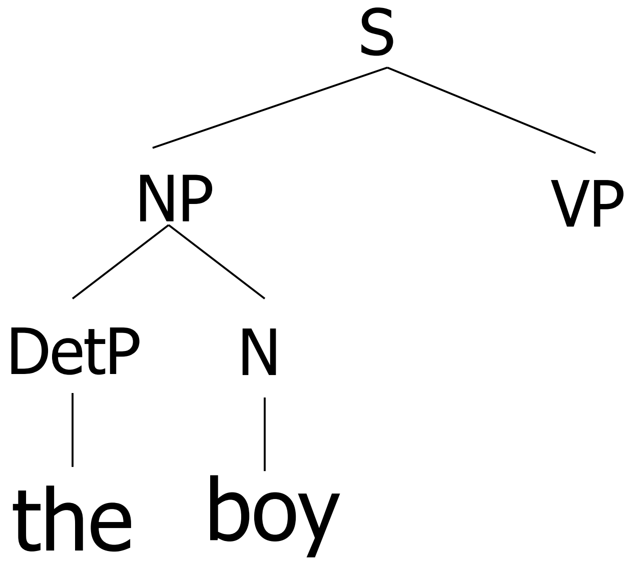
$S \rightarrow$ NP VP
VP $\rightarrow$ V NP
NP $\rightarrow$ DetP N | AdjP NP
AdjP $\rightarrow$ Adj | Adv AdjP
N $\rightarrow$ boy | girl
V $\rightarrow$ sees | likes
Adj $\rightarrow$ big | small
Adv $\rightarrow$ very
DetP $\rightarrow$ a | **the**



Derivations in a CFG

the boy VP

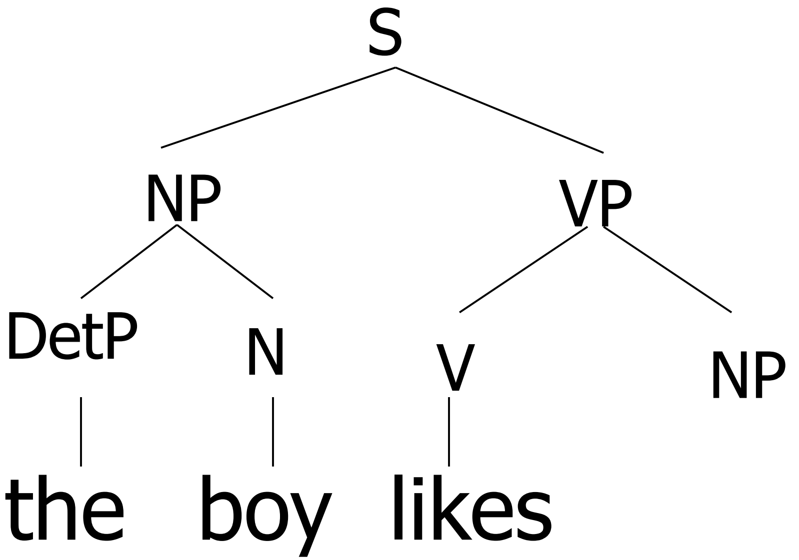
$S \rightarrow$ NP VP
VP $\rightarrow$ V NP
NP $\rightarrow$ DetP N | AdjP NP
AdjP $\rightarrow$ Adj | Adv AdjP
N $\rightarrow$ boy | girl
V $\rightarrow$ sees | **likes**
Adj $\rightarrow$ big | small
Adv $\rightarrow$ very
DetP $\rightarrow$ a | the



Derivations in a CFG

the boy likes NP

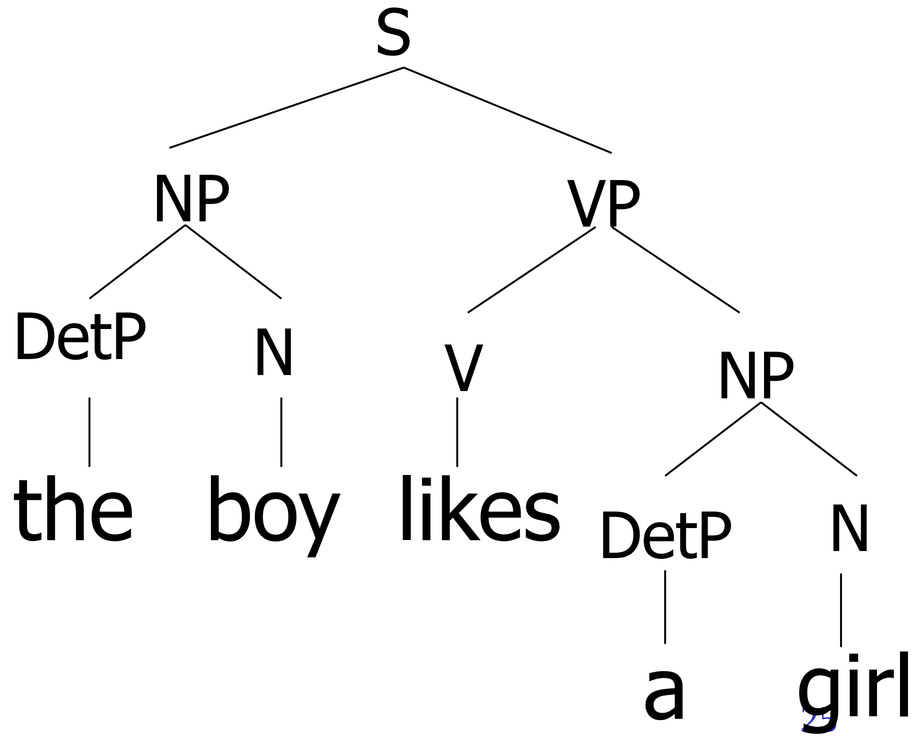
S $\rightarrow$ NP VP
VP $\rightarrow$ V NP
NP $\rightarrow$ DetP N | AdjP NP
AdjP $\rightarrow$ Adj | Adv AdjP
N $\rightarrow$ boy | **girl**
V $\rightarrow$ sees | likes
Adj $\rightarrow$ big | small
Adv $\rightarrow$ very
DetP $\rightarrow$ a | the



Derivations in a CFG

the boy likes a girl

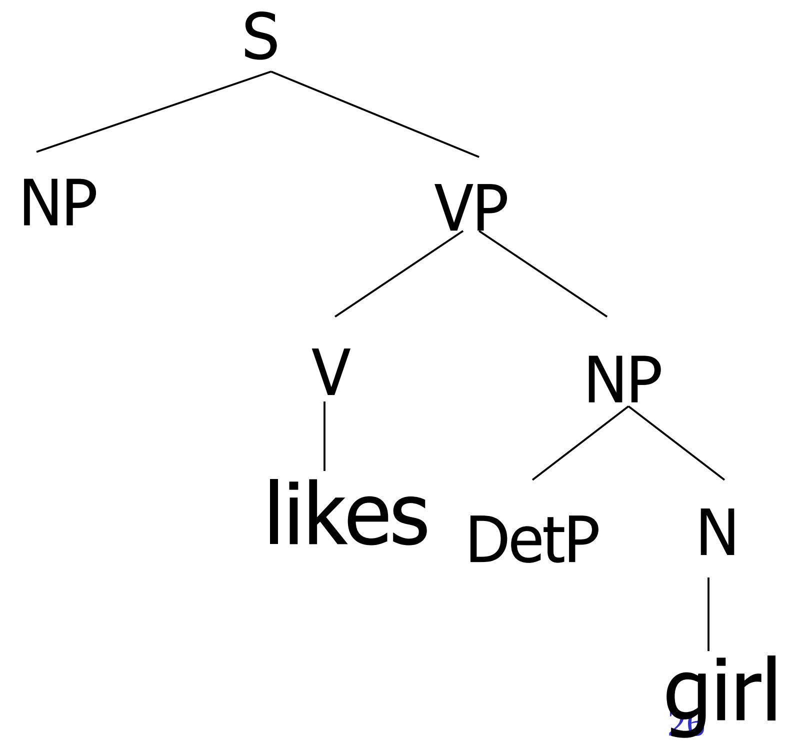
S $\rightarrow$ NP VP
VP $\rightarrow$ V NP
NP $\rightarrow$ DetP N | AdjP NP
AdjP $\rightarrow$ Adj | Adv AdjP
N $\rightarrow$ boy | girl
V $\rightarrow$ sees | likes
Adj $\rightarrow$ big | small
Adv $\rightarrow$ very
DetP $\rightarrow$ a | the



Derivations in a CFG; Order of Derivation Irrelevant

NP likes DetP girl

S → NP VP
VP → V NP
NP → DetP N | AdjP NP
AdjP → Adj | Adv AdjP
N → boy | girl
V → sees | likes
Adj → big | small
Adv → very
DetP → a | the



Derivations of CFGs

- String rewriting system: we derive a string (=**derived** structure)
- But derivation history represented by phrase-structure tree (=**derivation** structure)!

Grammar Equivalence

- Can have different grammars that generate same set of strings (weak equivalence)
  - Grammar 1: NP $\rightarrow$ DetP N and DetP $\rightarrow$ a | the
  - Grammar 2: NP $\rightarrow$ a N | NP $\rightarrow$ the N
- Can have different grammars that have same set of derivation trees (strong equivalence)
  - With CFGs, possible only with useless rules
  - Grammar 2': DetP $\rightarrow$ many
- Strong equivalence implies weak equivalence

Normal Forms &c

- There are weakly equivalent normal forms (Chomsky Normal Form, Greibach Normal Form)
- There are ways to eliminate useless productions and so on

Generative Grammar

- Formal languages: formal device to generate a set of strings (such as a CFG)
- Linguistics (Chomskyan linguistics in particular): approach in which a linguistic theory enumerates all possible strings/structures in a language (=competence)
- Chomskyan theories do not really use formal devices – they use CFG + informally defined transformations

Nobody Uses CFGs Only (Except Intro NLP Courses)

- All major syntactic theories (Chomsky, LFG, HPSG) represent both phrase structure and dependency, in one way or another
- All successful parsers currently use statistics about phrase structure and about dependency
- Derive dependency through “head percolation”: for each rule, say which daughter is head

What about Computational Complexity – Options to CFG

- Regular Grammars – generally claimed to be too weak to capture linguistic generalizations
- Context Sentsitive Grammars – generally regarded as too strong
- Recursively Enumerable (Type 0) Grammars – generally regarded as way too strong

- Approaches that are TOO STRONG have the power to predict/describe/capture syntactic structures that don't exist in human languages. (But CFG probably not enough)
- Computational processes associated with stronger formalisms are not as efficient as those associated with weaker methods

Massive Ambiguity of Syntax

- For a standard sentence, and a grammar with wide coverage, there are 1000s of derivations!
- Example:
  - The large head painter told the delegation that he gave money orders and shares in a letter on Wednesday

Types of syntactic constructions

- Is this the same construction?
  - An elf **decided** to clean the kitchen
  - An elf **seemed** to clean the kitchen

  An elf cleaned the kitchen
- Is this the same construction?
  - An elf **decided** to be in the kitchen
  - An elf **seemed** to be in the kitchen

  An elf was in the kitchen

Types of syntactic constructions (ctd)

- Is this the same construction?

  There is an elf in the kitchen

  - *There **decided** to be an elf in the kitchen
  - There **seemed** to be an elf in the kitchen
- Is this the same construction?

  It is raining/it rains

  - ??It **decided** to rain/be raining
  - It **seemed** to rain/be raining

Types of syntactic constructions (ctd)

Conclusion:

- *to seem:* whatever is embedded surface subject can appear in upper clause
- *to decide:* only full nouns that are referential can appear in upper clause
- Two types of verbs

Types of syntactic constructions: Analysis

to seem: lower surface subject **raises** to upper clause; **raising verb**

seems there to be an elf in the kitchen

there seems *t* to be an elf in the kitchen

it seems (that) there is an elf in the kitchen

Types of syntactic constructions: Analysis (ctd)

- *to decide:* subject is in upper clause and co-refers with an empty subject in lower clause; **control verb**

an elf decided an elf to clean the kitchen

an elf decided to clean the kitchen

an elf decided (that) he cleans/should clean the kitchen

*it decided (that) he cleans/should clean the kitchen

Lessons Learned from the Raising/Control Issue

- Use distribution of data to group phenomena into classes
- Use different underlying structure as basis for explanations
- Allow things to "move" around from underlying structure -> **transformational grammar**
- Check whether explanation you give makes predictions

Developing Grammars

- We saw with the previous example a complex structure
- Let's back off to simple English Structures and see how we would capture them with Context Free Grammars
- Developing a grammar of any size is difficult.

Key Constituents (English)

- Sentences
- Noun phrases
- Verb phrases
- Prepositional phrases

See text for examples of these!

Common Sentence Types

- Declaratives: John left

  S -> NP VP
- Imperatives: Leave!

  S -> VP
- Yes-No Questions: Did John leave?

  S -> Aux NP VP
- WH Questions (who, what, where, when, which, why, how): When did John 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).

NP -> NP PP	[[The flight] [to Boston]]
VP -> VP PP	[[departed Miami] [at noon]]

Recursion

- Can make things interesting. Consider the rule:
- NP -> NP PP

  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

[[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

- VP -> V NP
- I hate

  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

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

Problems

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

Agreement

- This dog
- Those dogs

- This dog eats
- Those dogs eat

- *This dogs
- *Those dog

- *This dog eat
- *Those dogs eats

Handing Number Agreement in CFGs

- To handle, would need to expand the grammar with multiple sets of rules – but it gets rather messy quickly.
- NP_sg → Det_sg N_sg
- NP_pl → Det_pl N_pl
- …..
- VP_sg → V_sg NP_sg
- VP_sg → V_sg NP_pl
- VP_pl → V_pl NP_sg
- VP_pl → V_pl NP_pl

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 type of the argument it wants to take

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

Possible CFG Solution

- VP -> V
- VP -> V NP
- VP -> V NP PP
- …

- VP -> IntransV
- VP -> TransV NP
- VP -> TransPP NP PP
- …

Movement

- Core 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)