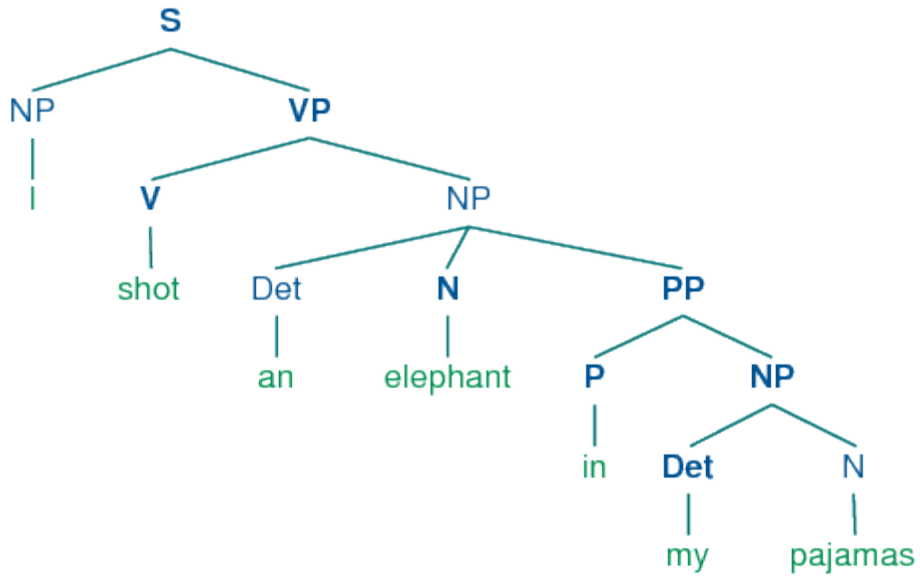
CS 544: Introduction to Natural Language Processing

Unit 3: Syntax and Parsing



March 2010

Liang Huang (lhuang@isi.edu)

Big Picture

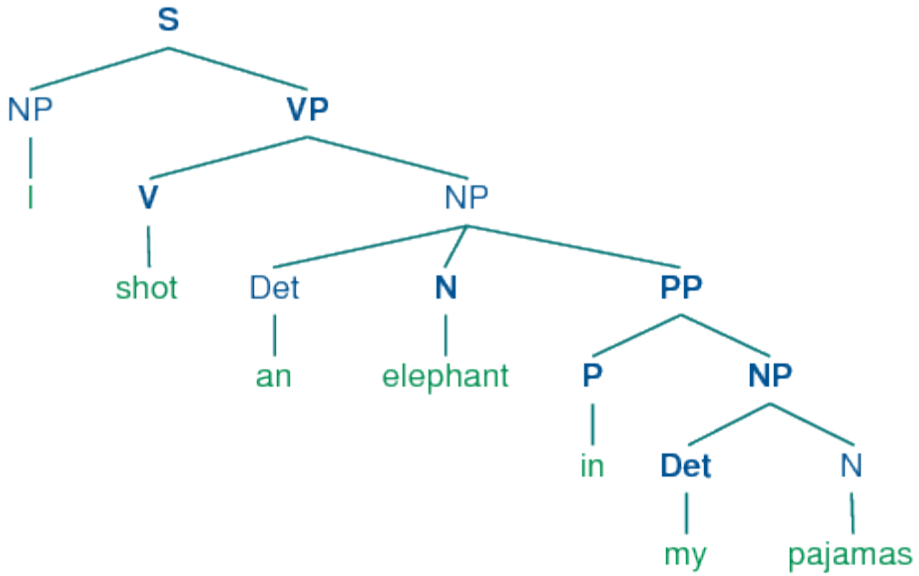
- we have already covered...
  - generation (Hovy)
  - semantics (Hobbs)
  - shift-reduce parsing (guest lectures by Hermajakob and Sagae)
- in this unit we'll look at syntax and parsing, and cover...
  - context-free grammars
  - chomsky hierarchy
  - probabilistic context-free grammars
  - parsing algorithms: CKY and Earley

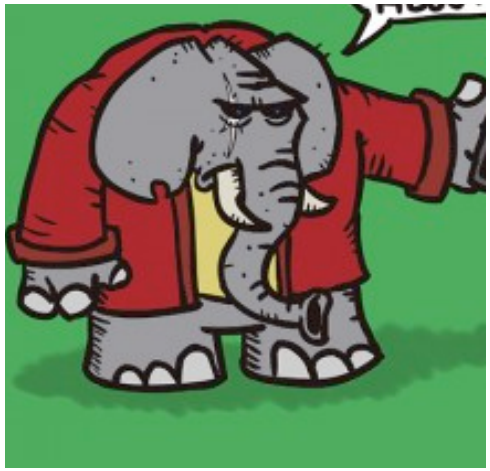
Why do we need syntax?

- because languages are recursive
- and highly ambiguous

but why are human languages evolved to be ambiguous in the first place?

One morning in Africa,
I shot an elephant in my pajamas;
how he got into my pajamas I'll never know.

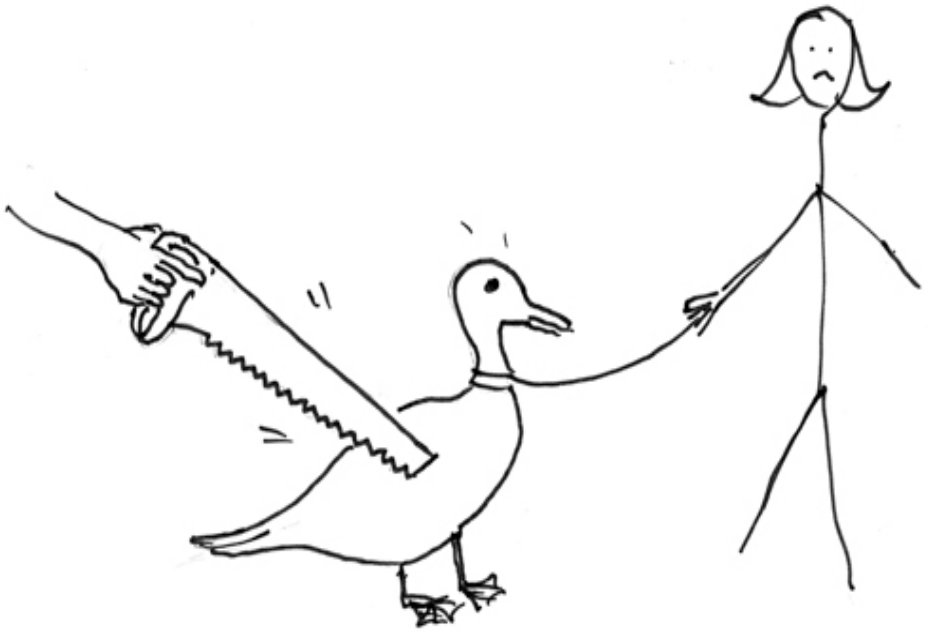




NLP is all about ambiguities

- to middle school kids: what does this sentence mean?

Aravind Joshi



NLP is all about ambiguities

- to middle school kids: what does this sentence mean?

Aravind Joshi

Ambiguities in Translation

zi zhu zhong duan
自 助 终 端

self help terminal device

needs context to disambiguate!

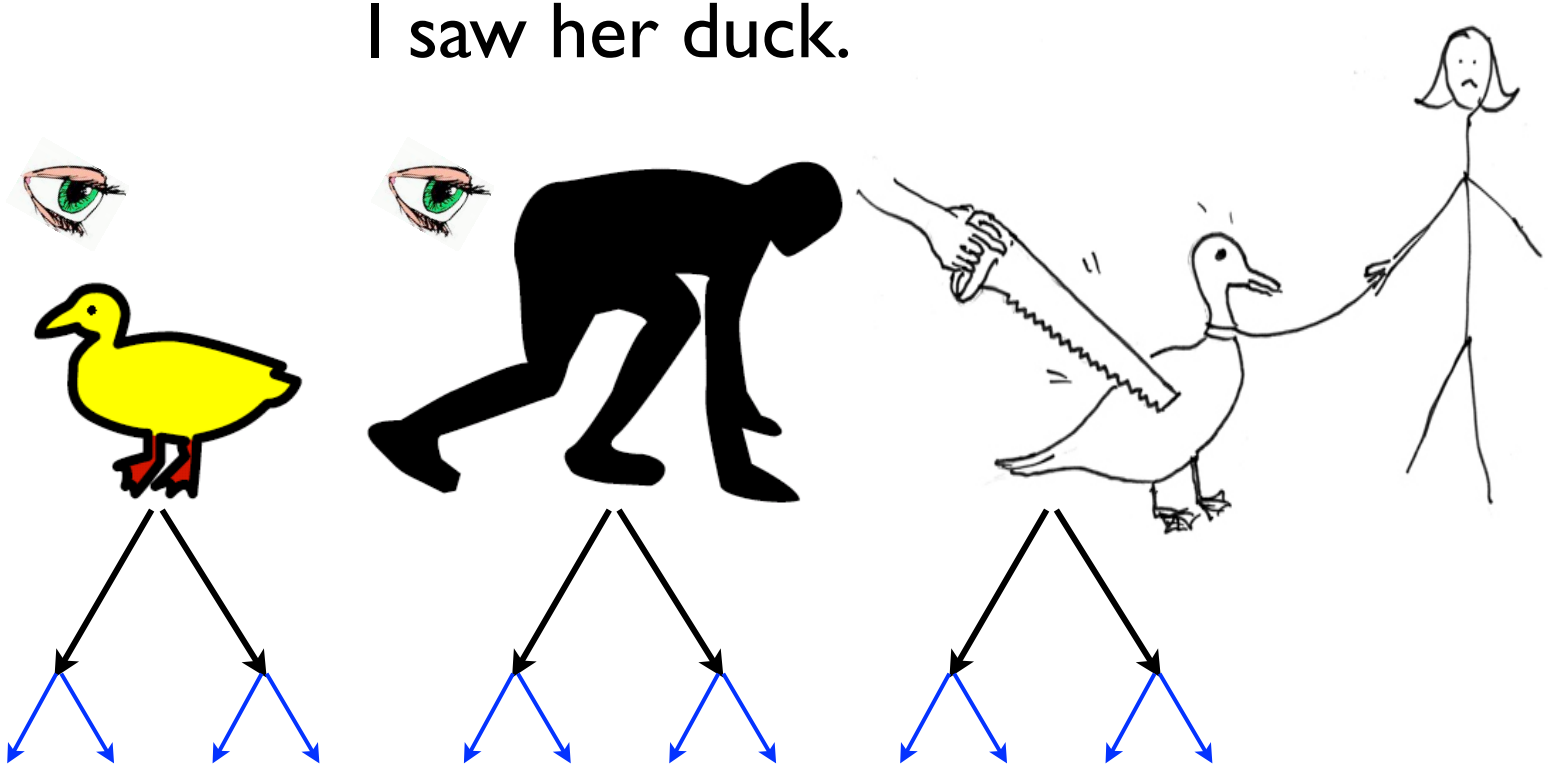
Ambiguities in Translation

or even...

clear evidence that MT is used in real life.

Ambiguity Explosion by Recursion

I saw her duck.



...

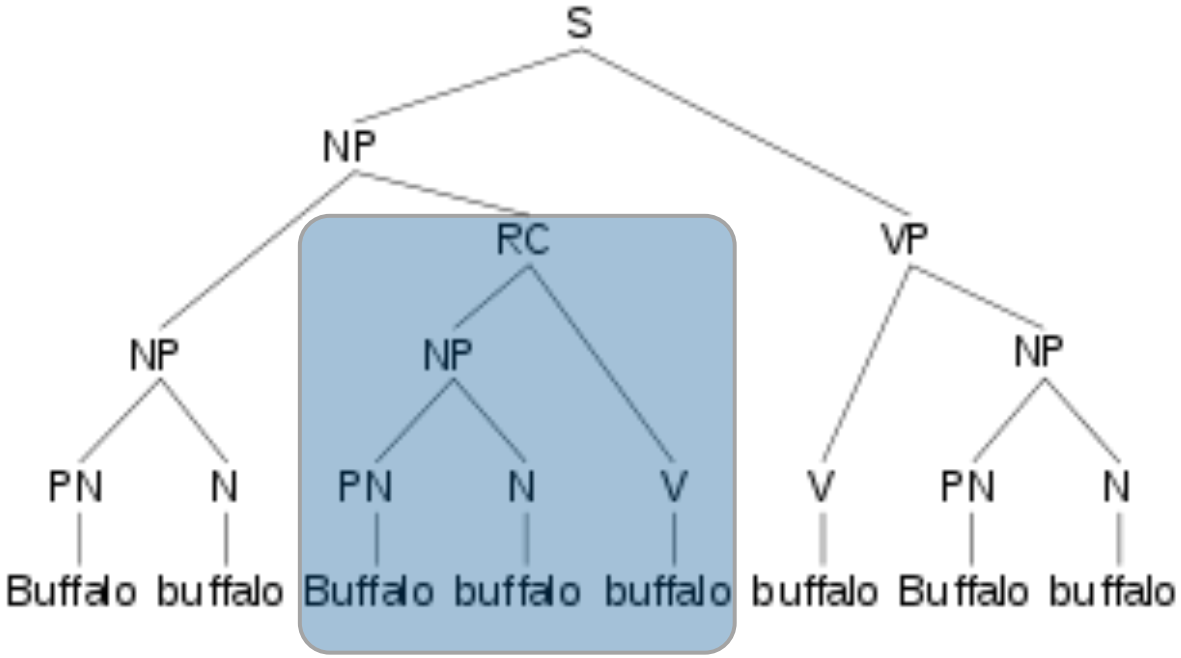
- how about...
  - I saw her duck with a telescope.
  - I saw her duck with a telescope in the garden...

Side Note: Projectivity

- I saw her duck with a telescope in the garden
  - can you attach "with..." to "saw" while "in..." to "duck"?
  - called "crossing dependencies" or "non-projective tree"
  - English speakers generally avoid it, but OK in spoken lang:
    - I saw a dog yesterday with a long tail
  - Slavic and Scandinavian speakers do that more often
  - Chinese speakers simply don't do that
  - crossing dependencies are hard to process, and are gradually fading away in language evolution

Ambiguity Explosion by Recursion

Buffalo buffalo Buffalo buffalo buffalo buffalo Buffalo buffalo



Aravind Joshi

Dogs dogs dog dog dogs.

`Police police police police police`

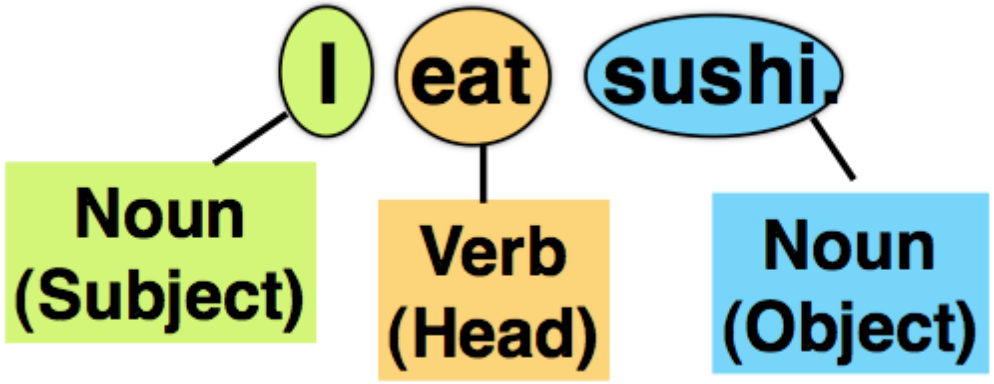
http://www.cse.buffalo.edu/~rapaport/BuffaloBuffalo/buffalobuffalo.html

Limitations of Sequence Models

- can you write an FSA/FST for the following?
  - $\{ (a^n, b^n) \}$ $\{ (a^{2n}, b^n) \}$
  - $\{ a^n\ b^n \}$
  - $\{ w\ w^R \}$
  - $\{ (w, w^R) \}$
- does it matter to human languages?
  - [The woman saw the boy [that heard the man [that left]]].
  - [The claim [that the house [he bought] is valuable] is wrong].
  - but humans can't really process infinite recursions... stack overflow!

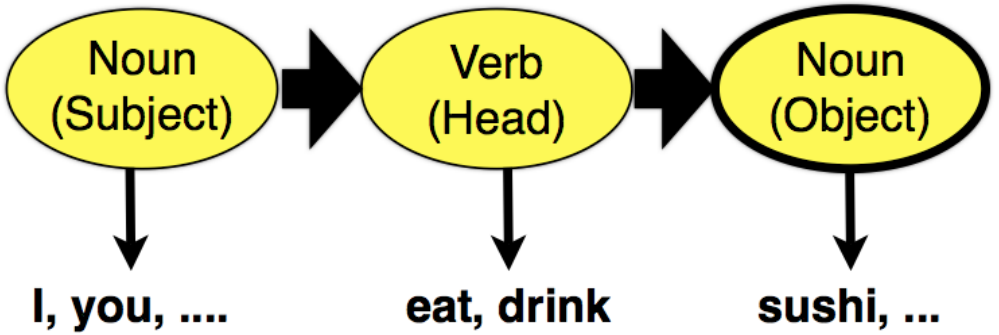
Let's try to write a grammar...

(courtesy of Julia Hockenmaier)



- let's take a closer look...
- we'll try our best to represent English in a FSA...
- basic sentence structure: N, V, N

Subject-Verb-Object



- compose it with a lexicon, and we get an HMM
- so far so good

(Recursive) Adjectives

(courtesy of Julia Hockenmaier)

the ball
the big ball
the big, red ball
the big, red, heavy ball ....

- then add Adjectives, which modify Nouns
- the number of modifiers/adjuncts can be unlimited.
- how about no determiner before noun? "play tennis"

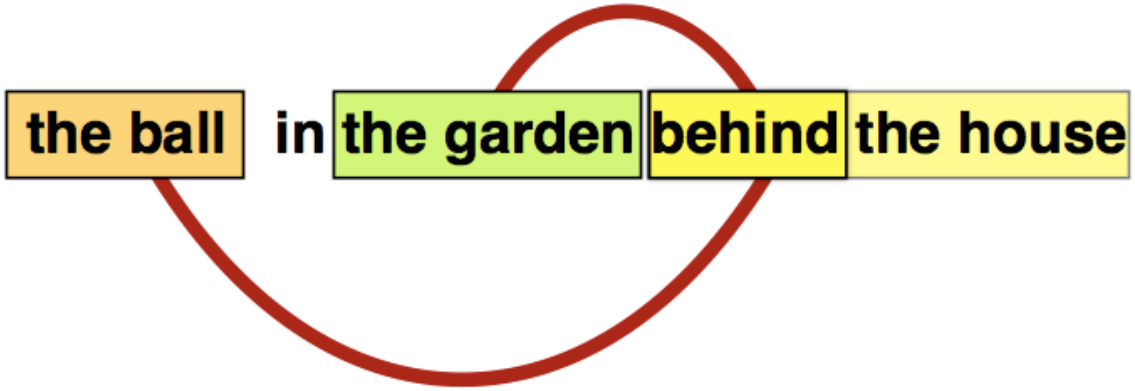
Recursive PPs

(courtesy of Julia Hockenmaier)

the ball
the ball in the garden
the ball in the garden behind the house
the ball in the garden behind the house near the school ...

- recursion can be more complex
- but we can still model it with FSAs!
- so why bother to go beyond finite-state?

FSAs can't go hierarchical!



(courtesy of Julia Hockenmaier)

- but sentences have a hierarchical structure!
  - so that we can infer the *meaning*
  - we need not only strings, but also trees
- FSAs are flat, and can only do tail recursions (i.e., loops)
- but we need real (branching) recursions for languages

FSAs can't do Center Embedding

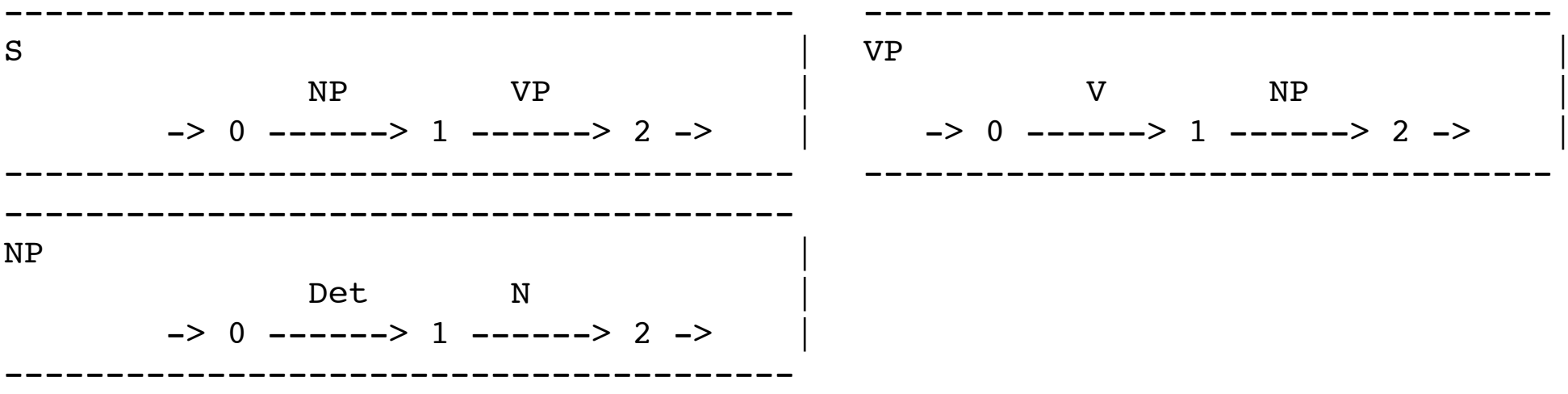
The mouse ate the corn.
The mouse that the snake ate ate the corn.
The mouse that the snake that the hawk ate ate ate the corn.
....
vs.
The claim that the house he bought was valuable was wrong.
vs.
I saw the ball in the garden behind the house near the school.

(courtesy of Julia Hockenmaier)

- in theory, these infinite recursions are still grammatical
  - competence (grammatical knowledge)
- in practice, studies show that English has a limit of 3
  - performance (processing and memory limitations)
- FSAs *can* model *finite* embeddings, but very *inconvenient*.

How about Recursive FSAs?

- problem of FSAs: only tail recursions, no branching recursions
  - can't represent hierarchical structures (trees)
  - can't generate center-embedded strings
- is there a simple way to improve it?
  - recursive transition networks (RTNs)



Context-Free Grammars

- S → **NP VP**
- **NP → Det N**
- **NP → NP PP**
- PP → P NP
- VP → V NP
- VP → VP PP
- …

- **N →** *{ball, garden, house, sushi }*
- **P →** *{in, behind, with}*
- **V → …**
- **Det → …**

Context-Free Grammars

A CFG is a 4-tuple ⟨*N*,Σ,R,S⟩

A set of nonterminals N

(e.g. *N* = {S, NP, VP, PP, Noun, Verb, ...})

A set of terminals Σ

(e.g. Σ = {*I, you, he, eat, drink, sushi, ball,* })

A set of rules R

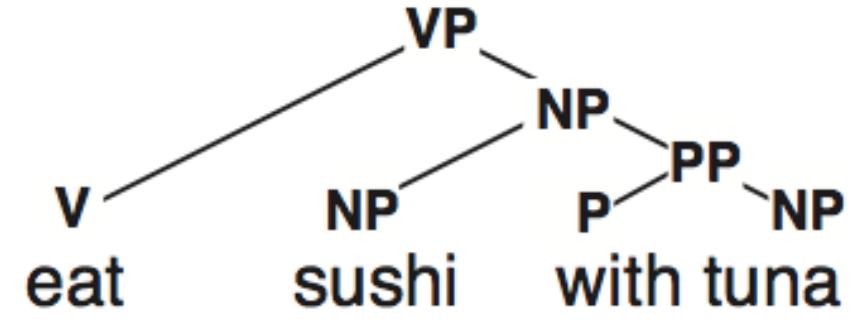
R ⊆ **{**A → β **with left-hand-side (LHS)** A ∈ N

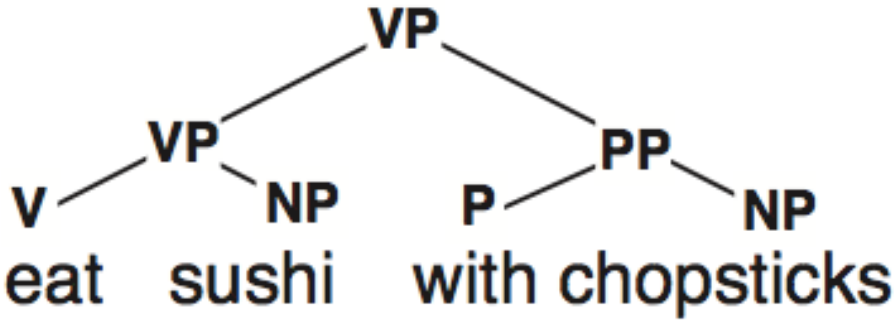
and right-hand-side (RHS) β ∈ **(N** ∪ Σ**)*** **}**

A start symbol S (sentence)

Parse Trees

- **N** → *{sushi, tuna}*
- **P** → *{with}*
- **V** → *{eat}*
- **NP** → **N**
- NP → NP PP
- PP→P NP
- VP→V NP
- VP→VP PP





CFGs for Center-Embedding

The mouse ate the corn.
The mouse that the snake ate ate the corn.
The mouse that the snake that the hawk ate ate ate the corn.
....

- $\{ a^n b^n \}$ $\{ w w^R \}$ *palindrome language:*
  nested dependencies -- easy to process
  (found in many languages, up to 3 levels)
- can you also do $\{ a^n b^n c^n \}$? or $\{ w w \}$? *copy language:*
- $\{ a^n b^n c^m d^m \}$
  cross-serial dependencies -- hard to process
  (only found in Swiss German and Dutch)
- what's the limitation of CFGs?
- CFG for center-embedded clauses:
  - S → **NP ate NP; NP → NP RC; RC → that NP ate**

Chomsky Hierarchy

	Language	Automata	Parsing complexity	Dependencies
Type 3	Regular	Finite-state	linear	adjacent words
Type 2	Context-Free	Pushdown	cubic	nested
Type 1	Context-sensitive	Linear Bounded	exponential	
Type 0	Recursively Enumerable	Turing machine		

computer science and linguistics share the same mathematical foundations.