CS 4650/7650, Lecture 13: Parsing 1

Jacob Eisenstein

September 26, 2013

So far we've explored finite-state models, which correspond to regular languages.

- **representations**: (weighted) finite state automata
- **probabilistic models**: HMMs (as a special case), CRFs
- **algorithms**: viterbi, forward-backward, $\mathcal{O}(NK^2)$ time complexity.
- **linguistic phenomena**:
  - morphology
  - language models
  - part-of-speech disambiguation
  - named entity recognition (chunking)

Is finite state enough?

1 Is English a regular language?

Regular languages are closed under intersection:

- $K \cap L$ is the set of strings in both K and L
- $K \cap L$ is regular iff K and L are regular

How to prove English is not regular:

- Let K be the set of grammatical English sentences
- Let L be some regular language
- Show that the intersection is not regular

We're going to prove this using center embedding:

1. *The cat is fat.*

2. *The cat that the dog chased is fat.*

3. **The cat that the dog is fat.*

4. *The cat that the dog that the monkey kissed chased is fat.*

5. **The cat that the dog that the monkey chased is fat.*

Proof sketch:

- K is the set of grammatical english sentences.
  It excludes sentences (3) and (5).

- L is the regular language *the cat* $(\textit{that } N)_t^+ V_t^+$ *is fat*.

- The language $L \cap K$ is *the cat* $(\textit{that } N)_t^n V_t^n$ *is fat*.

Note that the issue here is not just infinite repetition or productivity; FSAs can handle productive phenomena like *the big red smelly plastic figurine*.

Anyway, what do you think of this argument?

1.1 Is deep center embedding really part of English?

Karlsson (2007) searched for multiple (phrasal) center embeddings in corpora from 7 languages:

- Very few examples of double embedding

- Only 13 examples of triple embedding (none in speech)

- Zero examples of quadruple embeddings

Note that we can build an FSA to accept center-embedding up to any finite depth.

Chomsky and many linguists distinguish between

- **Competence**: the fundamental abilities of the (idealized) human language processing system

- **Performance**: real utterances produced by speakers, subject to non-linguistic factors such as cognitive limitations

Even if English *as performed* is regular, the underlying generative grammar may be context-free... **or beyond**. There is a similar proof that at least some languages are not context-free! I'll post slides with this proof idea.

1.2 How much expressiveness do we need?

- Shieber (1985) makes a similar argument with Swiss-German syntax. In response to the objection that all attested constructions are finite, Shieber writes:

  > Down this path lies tyranny. Acceptance of this argument opens the way to proofs of natural languages as regular, nay, **finite**.

- In practice, many real constructions are much simpler to handle in context-free rather than finite-state representations:

  > *The* ***processor has*** *10 million times fewer transistors on it than todays typical microprocessors,* ***runs*** *much more slowly, and* ***operates*** *at five times the voltage...*

  – The easy way:

  $$\begin{aligned} \text{S} &\rightarrow \text{NN VP} \\ \text{VP} &\rightarrow \text{VP3S} \mid \text{VPN3S} \mid \ldots \\ \text{VP3S} &\rightarrow \text{VP3S}, \text{VP3S}, \textit{and } \text{VP3S} \mid \text{VBZ} \mid \text{VBZ NP} \mid \ldots \end{aligned}$$

  – The hard way: build an FST that basically replicates all of English grammar for VPs with 3S and non-3S subjects.

- Mainstream parsing focuses on CFGs, but there is some work on "mildly" context-sensitive grammars.

2 Context-Free Languages

In the Chomsky hierarchy, context-free languages (CFLs) are a strict generalization of regular languages.

regular	**context-free**
regular expressions	context-free grammars (CFGs)
finite-state machines	pushdown automata
paths	derivations

Context-free grammars define CFLs. They are sets of permissible *productions* which allow you to **derive** strings composed of surface symbols.

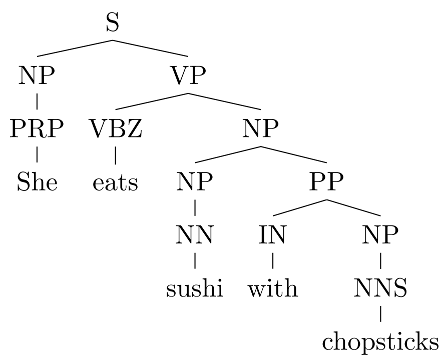
$$\begin{aligned} S &\rightarrow NP\ VP_1 \\ NP &\rightarrow \textit{the } N \mid NP \text{ RELCLAUSE} \\ \text{RELCLAUSE} &\rightarrow \textit{that } NP\ V_t \\ V_t &\rightarrow \textit{ate} \mid \textit{chased} \mid \textit{befriended} \mid \ldots \\ N &\rightarrow \textit{cat} \mid \textit{dog} \mid \textit{monkey} \mid \ldots \\ VP_1 &\rightarrow \textit{is fat} \end{aligned}$$

An important feature of CFGs is *recursion*, in which a nonterminal can be derived from itself.

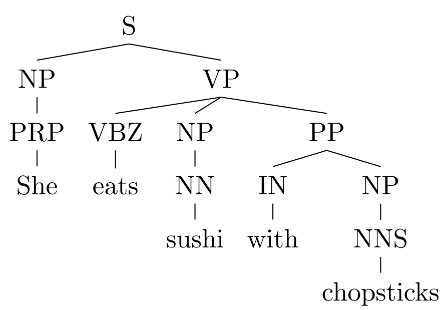
More formally , a CFG is a tuple $\langle N, \Sigma, R, S \rangle$:

N	a set of non-terminals
Σ	a set of terminals (distinct from N)
R	a set of productions, each of the form $A \rightarrow \beta$, where $A \in N$ and $\beta \in (\Sigma \cup N)^*$
S	a designated start symbol

- Context free grammars provide rules for generating strings.
- A surface string can be **parsed** into a series of productions (a **derivation**).
- Parses can be viewed as trees or as bracketings:



$(_S(_{NP}(_{PRP}\ \textit{She})(_{VP}(_{VBZ}\ \textit{eats})$
$(_{NP}(_{NP}(_{NN}\ \textit{sushi}))(_{PP}\ \textit{with}(_{NP}(_{NNS}\ \textit{chopsticks})))))))$



$(_S(_{NP}(_{PRP}\ \textit{She})(_{VP}(_{VBZ}\ \textit{eats})$
$(_{NP}(_{NN}\ \textit{sushi}))$
$(_{PP}\ \textit{with}(_{NP}(_{NNS}\ \textit{chopsticks}))))))$

Semantics Ideally, each derivation will have a distinct semantic interpretation, and all possible interpretations will be represented in some derivation.

$$(_{NP}(_{NP}\ \textit{Ban}\ (_{PP}\ \textit{on}\ (_{NP}\ \textit{nude dancing}\ )))$$
$$(_{PP}\ \textit{on}\ (_{NP}\ \textit{Governor's desk}\ )))$$

$(_{NP}$ *Ban* $(_{PP}$ *on* $(_{NP}(_{NP}$ *nude dancing* $)$
$(_{PP}$ *on* $(_{NP}$ *Governor's desk* $)))))$

Sadly, this is not always the case.

$(_{NP}(_{JJ}$ *nice*$)$ $(_{JJ}$ *little*$)$ $(_{NN}$ *car* $))$
$(_{NP}(_{JJ}$ *nice*$)$ $(_{NP}(_{JJ}$ *little*$)$ $(_{NN}$ *car* $)))$
$(_{NP}(_{JJ}$ *nice*$)$ $(_{NP}(_{JJ}$ *little*$)$ $(_{NP}(_{NN}$ *car* $))))$

3 Constituency

- In natural language grammars, the non-terminals should reflect syntactic categories.
- Bracketed substrings (e.g., *sushi with chopsticks*) are called **constituents**.
- There are several tests for constituency, including:
  - substitution
  - coordination
  - movement

Substitution Constituents generated by the same non-terminal should be substitutable in many contexts:

- $(_{NP}$ *The ban* $)$ *is on the desk.*
- $(_{NP}$ *The Governor's desk* $)$ *is on the desk.*
- $(_{NP}$ *The ban on dancing on the desk* $)$ *is on the desk.*
- *$(_{PP}$ *On the desk* $)$ *is on the desk.*

A more precise test for whether a set of substrings constitute a single category is whether they can be replaced by the same pronouns.

- $(_{NP}$ *It* $)$ *is on the desk.*

What about verbs?

- *I* $(_{V}$ *gave* $)$ *it to Anne.*
- *I* $(_{V}$ *taught* $)$ *it to Anne.*
- *I* $(_{V}$ *gave* $)$ *Anne a fish*
- **I* $(_{V}$ *taught* $)$ *Anne a fish*

This suggests we need nonterminals which distinguish verbs based on the arguments they can take. The technical name for this is *subcategorization.*

Coordination Constituents generated by the same non-terminal can usually be *coordinated* using words like *and* and *or*:

- *We fought* ($_{PP}$ *on the hills*) *and* ($_{PP}$ *in the hedges*).
- *We fought* ($_{ADVP}$ *as well as we could*).
- **We fought* ($_{ADVP}$ *as well as we could*) *and* ($_{PP}$ *in the hedges*).

This too doesn't always work:

- *She* ($_{VP}$ *went*) ($_{PP}$ *to the store*).
- *She* ($_{VP}$ *came*) ($_{PP}$ *from the store*).
- *She* (*went to*) *and* (*came from*) *the store.*

Movement Valid constituents can be moved as a unit, preserving grammaticality.

- Passivization
  - *(The governor) banned (nude dancing on his desk)*
  - *(Nude dancing on his desk) was banned by (the governor)*
- Wh- movement
  - *(Nude dancing was banned) on (the desk).*
  - *(The desk) is where (nude dancing was banned)*
- Topicalization
  - *(He banned nude dancing) to appeal to conservatives.*
  - *To appeal to conservatives, (he banned nude dancing).*

4 A simple grammar of English

4.1 Noun phrases

Let's start with noun phrases:

- ***She*** *sleeps* (Pronoun)
- ***Arlo*** *sleeps* (Proper noun)
- ***Fish*** *sleep* (Mass noun)
- ***The fish*** *sleeps* (determiner + noun)
- ***The blue fish*** *sleeps* (DT + JJ + NN)

- ***The girl from Omaha*** *sleeps* (NP + PP)
- ***The student who ate 15 donuts*** *sleeps* (NP + RelClause)

So overall, we can summarize this fragment as

$$\begin{aligned} \text{NP} &\rightarrow \text{PRP} \mid \text{NNP} \mid \text{DT Nom} \\ \text{Nom} &\rightarrow \text{AdjP Nom} \mid \text{NN} \\ \text{NP} &\rightarrow \text{NP PP} \mid \text{NP RelClause} \end{aligned}$$

We're leaving out some detail, like pluralization and possessives, but you get the idea.

4.2 Adjectival and prepositional phrases

- *Very funny*
- *The* ***large, blue*** *fish*
- *The man* ***from la mancha***

$$\begin{aligned} \text{AdjP} &\rightarrow \text{JJ} \mid \text{RB AdjP} \mid \text{JJ AdjP} \\ \text{PP} &\rightarrow \text{IN NP} \mid \text{TO NP} \end{aligned}$$

4.3 Verb phrases

- *She* ***sleeps***
- *She* ***sleeps restlessly***
- *She* ***sleeps at home***
- *She* ***eats sushi***[1]
- *She* ***gives John sushi***

$$\text{VP} \rightarrow \text{V} \mid \text{VP RB} \mid \text{VP PP} \mid \text{V NP} \mid \text{V NP NP} \mid \text{V NP RB}$$

But what about **She sleeps sushi* or **She speaks John Japanese*?

- Classes of verbs can take different numbers of arguments.
- This is called **subcategorization**

$$\begin{aligned} &\text{VP} \rightarrow \text{V-intrans} \mid \text{V-trans NP} \mid \text{V-ditrans NP NP} \\ &\text{VP} \rightarrow \text{VP RB} \mid \text{VP PP} \end{aligned}$$

We would also need to handle modal and auxiliary verbs that allow us to create complex tenses, like *She will have eaten sushi* but not **She will have eats sushi.*

[1] Sushi examples from Julia Hockenmaier

4.4 Sentences

- *She eats sushi*

$$S \rightarrow NP\ VP$$

- *Sometimes, she eats sushi*

$$S \rightarrow \textsc{AdvP}\ S$$

- *In Japan, she eats sushi*

$$S \rightarrow PP\ S$$

- What about **I eats sushi*, **She eat sushi*??

$$S \rightarrow NP.3S\ VP.3S \mid NP.N3S\ VP.N3S$$

  In general, we need **features** to capture this kind of agreement.

4.5 Conjunctions

- *She eats sushi and candy*

$$NP \rightarrow NP\ \textit{and}\ NP$$

- *She eats sushi and drinks soda*

$$VP \rightarrow VP\ \textit{and}\ VP$$

- *She eats sushi and he drinks soda*

$$S \rightarrow S\ \textit{and}\ S$$

- *fresh and tasty sushi*

$$\textsc{AdjP} \rightarrow JJ\ \textit{and}\ JJ$$

We'd need a little more cleverness to properly cover groups larger than two.

4.6 Odds and ends

- *I gave sushi to the girl* ***who eats sushi***. This is a relative clause,

$$\textsc{RelClause} \rightarrow \textit{who}\ VP \mid \textit{that}\ VP$$

- *I took sushi from the man* ***offering sushi***. This is a gerundive postmodifier.

$$\textsc{Nom} \rightarrow \textsc{Nom}\ \textsc{Gerund}VP$$
$$\textsc{Gerund}VP \rightarrow VBZ \mid VBZ\ NP \mid VBZ\ PP \mid \ldots$$

- ***Can*** *she eat sushi?* (notice it's not *eats*)

$$S \rightarrow \textsc{Aux}\ NP\ VP$$

- ... and many more

5 Grammar design

Our goal is a grammar that avoids

- **Overgeneration**: deriving strings that are not grammatical.
- **Undergeneration**: failing to derive strings that are grammatical.

To avoid undergeneration, we would need thousands of productions.

Typically, grammars are defined in conjunction with large-scale **treebank** annotation projects.

- An annotation guideline specifies the non-terminals and how they go together.
- The annotators then apply these guidelines to data.
- The grammar rules can then be read off the data.

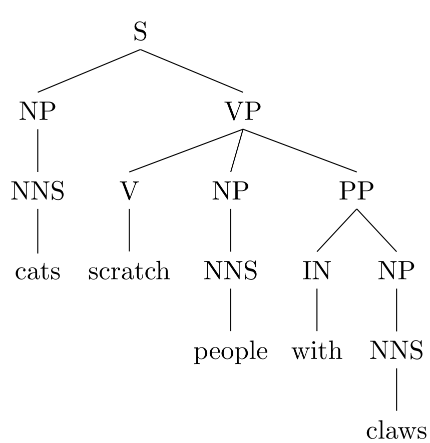
The Penn Treebank contains one million parsed words of Wall Street Journal text from the 1990s.

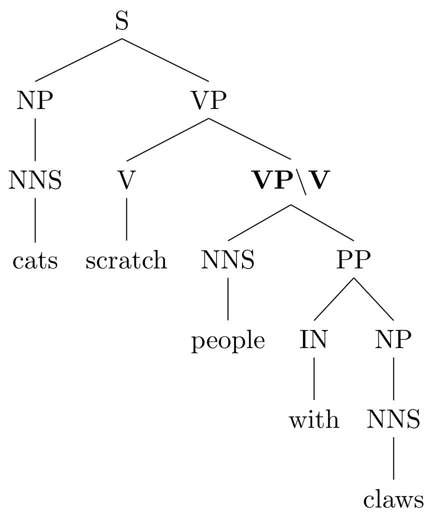
6 Grammar equivalence and normal form

- Grammars are weakly equivalent if they generate the same strings.
- Grammars are strongly equivalent if they generate the same strings **and** assign the same phrase structure to each sentence.
- In Chomsky Normal Form (CNF), all productions are either:

$$A \rightarrow BC$$
$$A \rightarrow a$$

  - All CFGs can be converted into a weakly equivalent grammar in CNF.
  - This is very handy for parsing algorithms.





- Binarization is easy:
  group right children into new non-terminals.
- Un-binarization is important!
  people with claws is not a constituent in the original parse.
- Unary productions are best handled by modifying the algorithm.

7 Parsing

Parsing is the process of determining whether a sentence is in a context-free language, by searching for a legal derivation. Some possibilities:

- **Top-down**: start with the start symbol, and see if we can derive the sentence.
- **Bottom-up**: combine the observed symbols using whatever productions we can, until we reach the start symbol
- **Left-to-right**: move through the input, incrementally building a parse tree

Before we get into these different possibilities, let's see whether exhaustive search is possible. Suppose we only have one non-terminal, X, and it has binary productions

$$\text{X} \rightarrow \text{X X}$$
$$\text{X} \rightarrow \textit{the girl} \mid \textit{ate sushi} \mid \ldots$$

How many different ways could we parse a sentence? This is just equal to the number of binary bracketings of the words in the sentence, which is a Catalan number. Catalan numbers grow **super-exponentially** in the length of the sentence, $C_n = \frac{(2n)!}{(n+1)!n!}$.

8 CKY parsing

CKY is a bottom-up parsing allows us to test whether a sentence is in a context-free language, without considering all possible parses. First we form small constituents, then try to merge them into larger constituents.

Let's start with an example grammar:

$$
\begin{aligned}
S \rightarrow & VP\ NP \\
NP \rightarrow & NP\ PP \mid \textit{we} \mid \textit{sushi} \mid \textit{chopsticks} \\
PP \rightarrow & P\ NP \\
P \rightarrow & \textit{with} \\
VP \rightarrow & VP\ NP \mid VP\ PP \mid \textit{eat}
\end{aligned}
$$

Suppose we encounter the sentence *We eat sushi with chopsticks.*

- The first thing that we notice is that we can apply unary productions to obtain NP VP NP P NP
- Next, we can apply a binary production to merge the first NP VP into an S.
- Or we could merge VP NP into VP
- ... and so on

Let's systematize this. Here is the CKY algorithm:

for j : [1,N] **do**
  $t[j, j-1] \leftarrow \{A | A \rightarrow x_j \in R\}$
  for i : [j-2, 0] **do**
    for k : [i+1, j-1] **do**
      $t[i, j] \leftarrow t[i, j] \cup \{A | A \rightarrow BC \in R, B \in t[i, k], C \in t[k, j]\}$
    end for
  end for
end for

To hand unary transitions, we compute the *unary closure* of each non-terminal.

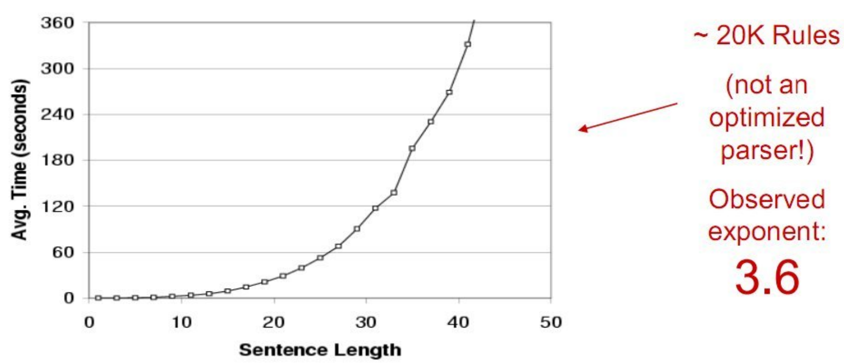
- e.g., if $S \rightarrow VP$, $VP \rightarrow V$, then add $S \rightarrow V$
- At each table entry $t[i, j]$
  - For each non-terminal $A \in t[i, j]$

    * Add all elements of the reflexive unary closure for A

S
|
VP
|
V
|
eat!

Complexity What is the complexity of CKY?

- Space compexity: $\mathcal{O}(L^2|N|)$
- Time complexity: $\mathcal{O}(N^3|R|)$
- L is length of sentence,
  $|N|$ is the number of non-terminals,
  $|R|$ is the number of production rules
- But in practice...



It's worse than worst-case! (figure from Dan Klein)

- Longer sentences "unlock" more of the grammar.

9 Ambiguity in parsing

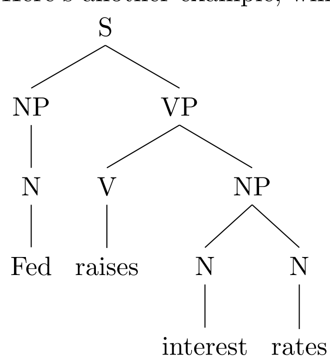
- Syntactic ambiguity is endemic to natural language:[2]
  [¡+-¿]

[2]Examples borrowed from Dan Klein

- Attachment ambiguity: *we eat sushi with chopsticks*, *I shot an elephant in my pajamas.*
- Modifier scope: *southern food store*
- Particle versus preposition: *The puppy tore up the staircase.*
- Complement structure: *The tourists objected to the guide that they couldn't hear.*
- Coordination scope: *"I see," said the blind man, as he picked up the hammer and saw.*
- Multiple gap constructions: *The chicken is ready to eat*

- In morphology, we didn't just want to know which derivational forms are *legal*, we wanted to know which were likely.
- Syntactic parsing is all about choosing among the many, many legal parses for a given sentence.

Here's another example, which we've seen before:



- A minimal grammar permits 36 parses!
- Real-size broad coverage grammars permit millions of parses.

Classical parsers faced a tradeoff:

- broad coverage with tons of ambiguity...
- or limited coverage in exchange for constraints on ambiguity

Consequently, deterministic parsers produced no analysis for many sentences.

9.1 Local solutions

Some ambiguity can be resolved locally:

- [*imposed* [*a ban* [*on asbestos*]]]
- [*imposed* [*a ban*][*on asbestos*]]

- Hindle and Rooth (1990) proposed a likelihood ratio test:

$$LR(v, n, p) = \frac{P(p|v)}{P(p|n)} = \frac{P(on|imposed)}{P(on|ban)}$$

  where we select VERB attachment if $LR(v, n, p) > 1$.

- But the likelihood-ratio approach ignores important information, like the phrase being attached.
  - ...[*it* [*would end* [*its venture* [*with Maserati*]]]]
  - ...[*it* [*would end* [*its venture*][*with Maserati*]]]

- The likelihood ratio gets this wrong
  - $P(with|end) = \frac{607}{5156} = 0.118$
  - $P(with|venture) = \frac{155}{1442} = 0.107$

  Other features (e.g., *Maserati*) argue for noun attachment. How can we add them?

Machine learning solutions Ratnaparkhi et al (1994) propose a maximum-entropy (logistic regression) approach:

$$P(\text{N}|\textit{would end its venture with Maserati}) = \frac{e^{\boldsymbol{w}^\top \boldsymbol{f}(\textit{would end its venture with Maserati})}}{1 + e^{\boldsymbol{w}^\top \boldsymbol{f}(\textit{would end its venture with Maserati})}}$$

Features include n-grams and word classes from hierarchical word clustering; accuracy is roughly 80%.

Collins and Brooks (1995) argued that attachment depends on four **heads**:

- the preposition (*with*)
- the VP attachment site (*end*)
- the NP attachment site (*venture*)
- the NP to be attached (*Maserati*)

They propose a backoff-based approach:

- First, look for counts of the tuple $\langle$*with, Maserati, end, venture*$\rangle$
- If none, try $\langle$*with, Maserati, end*$\rangle$ + $\langle$*with,end,venture*$\rangle$ + $\langle$*with,Maserati, venture*$\rangle$
- If none, try $\langle$*with, Maserati*$\rangle$ + $\langle$*with,end*$\rangle$ + $\langle$*with,venture*$\rangle$
- If none, try $\langle$*with*$\rangle$

Accuracy is roughly 84%. This approach of combining relative frequency estimation, smoothing, and backoff was very characteristic of late 1990s statistical NLP.

9.2 Beyond local solutions

Framing the problem as attachment ambiguity is limiting:

- assumes the parse is mostly done, leaving just a few attachment ambiguities to solve
- But realistic sentences have more than a few syntactic interpretations.
- Attachment decisions are interdependent:
  - *Cats scratch people with claws with knives.*
  - We may want to attach *with claws* to *scratch*.
  - But then we have nowhere to put *with knives*.

The task of statistical parsing is to produce a single analysis that resolves all syntactic ambiguities.

10 PCFGs

We want the parse τ that maximizes $P(\tau|S)$.

$$\begin{aligned}
\arg\max_{\tau} P(\tau|S) =& \arg\max_{\tau} \frac{P(\tau, S)}{P(S)} \\
=& \arg\max_{\tau} P(\tau, S) \\
=& \arg\max_{\tau} P(S|\tau)P(\tau) \\
=& \arg\max_{\tau : S=\text{yield}(\tau)} P(\tau)
\end{aligned}$$

- The **yield** of a tree is the string of terminal symbols that can be read off the leaf nodes.
- The set $\{\tau : S = \text{yield}(\tau)\}$ is exactly the set of all derivations of S in a CFG G.

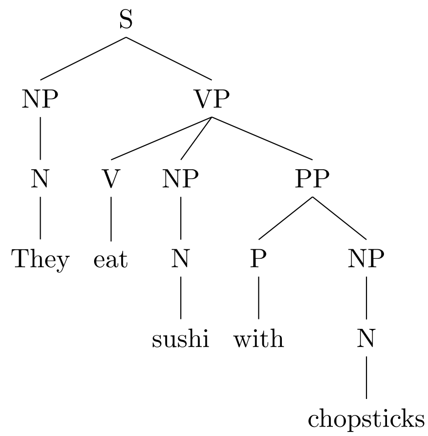
PCFGs extend the CFG by adding probability to each production:

S	$\rightarrow NP\ VP$	0.9
S	$\rightarrow S\ conj\ S$	0.1
NP	$\rightarrow N$	0.2
NP	$\rightarrow DT\ N$	0.3
NP	$\rightarrow N\ NP$	0.2
NP	$\rightarrow JJ\ NP$	0.2
NP	$\rightarrow NP\ PP$	0.1
VP	$\rightarrow V$	0.4
VP	$\rightarrow V\ NP$	0.3
VP	$\rightarrow V\ NP\ NP$	0.1
VP	$\rightarrow VP\ PP$	0.2
PP	$\rightarrow P\ NP$	1.0

The probabilities for all productions involving a single LHS must sum to 1:

$$\sum_{\alpha} P(X \rightarrow \alpha | X) = 1$$

The probability $P(\tau)$ is just the product of all the productions:



10.1 Estimation

- As in supervised HMMs, estimation is easy (for now!).
- PCFG probabilities can be estimated directly from a treebank:

$$P(VP \rightarrow VP\ PP) = \frac{c(VP \rightarrow VP\ PP)}{c(VP)}$$

10.2 Three basic problems for PCFGs

Let $\tau \in T$ be a derivation, S be a sentence, and λ a PCFG.

- **Decoding**: Find $\hat{\tau} = \arg\max_{\tau} P(\tau, S; \lambda)$
- **Likelihood**: Find $P(w; \lambda) = \sum_{\tau} P(\tau, S; \lambda)$
- **(Unsupervised) Estimation**: Find $\arg\max_{\lambda} P(S_{1...N} | \lambda)$