Natural Language Processing I

lecture 7: constituent parsing

Ivan Titov

Institute for Logic, Language and Computation
Outline

- Syntax: intro, CFGs, PCFGs
  - PCFGs: Estimation
  - CFGs: Parsing
  - PCFGs: Parsing
  - Parsing evaluation
  - Unsupervised estimation of PCFGs (?)
  - Grammar refinement (producing a state-of-the-art parser) (?)
Syntactic parsing

- **Syntax**
  - The study of the patterns of formation of sentences and phrases from words
  - Borders with **semantics** and **morphology** sometimes blurred

- **Parsing**
  - The process of predicting syntactic representations

- **Syntactic Representations**
  - Different types of syntactic representations are possible, for example:

  - **Constuent (a.k.a. phrase-structure) tree**

  - **Dependency tree**

  > Afyonkarahisarlılaştırabilirdiklerimizdenmişsinizcesineee in Turkish mean "as if you are one of the people that we thought to be originating from Afyonkarahisar" [wikipedia]
Constituent trees

- Internal nodes correspond to phrases

S – a sentence

NP (Noun Phrase):  My dog, a sandwich, lakes, ..

VP (Verb Phrase):  ate a sausage, barked, …

PP (Prepositional phrases):  with a friend, in a car, …

- Nodes immediately above words are PoS tags (aka preterminals)

PN – pronoun
D – determiner
V – verb
N – noun
P – preposition
Bracketing notation

It is often convenient to represent a tree as a bracketed sequence

(S
  (NP (PN My) (N Dog))
  (VP (V ate)
    (NP (D a) (N sausage)))
)

We will use this format in the assignment
Nodes are words (along with PoS tags)

Directed arcs encode syntactic dependencies between them

Labels are types of relations between the words

- poss – possessive
- dobj – direct object
- nsubj - subject
- det - determiner
Constituent and dependency representations

- Constituent trees can (potentially) be converted to dependency trees
  
  One potential rule to extract nsubj dependency. (In practice, they are specified a bit differently)

- Dependency trees can (potentially) be converted to constituent trees

Recovering labels is harder

Roughly: every word along with all its dependents corresponds to a phrase = to an inner node in the constituent tree
Recovering shallow semantics

- Some **semantic information** can be (approximately) derived from syntactic information
  - Subjects (nsubj) are (often) **agents** ("initiator / doers for an action")
  - Direct objects (dobj) are (often) **patients** ("affected entities")
Recovering shallow semantics

- Some semantic information can be (approximately) derived from syntactic information
  - Subjects (nsubj) are (often) agents ("initiator / doers for an action")
  - Direct objects (dobj) are (often) patients ("affected entities")

- But even for agents and patients consider:
  - *Mary is baking a cake in the oven***
  - *A cake is baking in the oven***

- In general it is not trivial even for the most shallow forms of semantics
  - E.g., consider prepositions: in can encode direction, position, temporal information, …
Brief history of parsing

- Before mid-90s syntactic **rule-based parsers** producing rich linguistic information
  - Provide low coverage (though hard to evaluate)
  - Predict much more information than the formalisms we have just discussed
- Realization that basic **PCFGs do not result in accurate predictive models**
  - We will discuss this in detail
- Mid-90s first **accurate statistical parsers** (e.g., [Magerman 1994, Collins 1996])
  - No handcrafted grammars, estimated from large datasets (Penn Treebank WSJ)
- Now: better models, more efficient algorithms, more languages, …
Ambiguity

- Ambiguous sentence:
  - I saw a girl with a telescope
- What kind of ambiguity it has?
- What implications it has for parsing?
Why parsing is hard?  Ambiguity

- Prepositional phrase attachment ambiguity

How serious is this problem in practice?

What kind of clues would be useful?

PP-attachment ambiguity is just one example type of ambiguity
Why parsing is hard? Ambiguity

- Example with 3 preposition phrases, 5 interpretations:
  - *Put the block* ((in the box on the table) in the kitchen)
  - *Put the block* (in the box (on the table in the kitchen))
  - *Put* ((the block in the box) on the table) in the kitchen.
  - *Put* (the block (in the box on the table)) in the kitchen.
  - *Put* (the block in the box) (on the table in the kitchen)
Why parsing is hard? Ambiguity

- Example with 3 preposition phrases, 5 interpretations:
  - Put the block ((in the box on the table) in the kitchen)
  - Put the block (in the box (on the table in the kitchen))
  - Put ((the block in the box) on the table) in the kitchen.
  - Put (the block (in the box on the table)) in the kitchen.
  - Put (the block in the box) (on the table in the kitchen)

- A general case:
  \[ \text{Cat}_n = \binom{2n}{n} - \binom{2n}{n-1} \sim \frac{4^n}{n^{3/2} \sqrt{\pi}} \]

1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, ...
Why parsing is hard? Ambiguity

- **Example with 3 preposition phrases, 5 interpretations:**
  - Put the block ((in the box on the table) in the kitchen)
  - Put the block (in the box (on the table in the kitchen))
  - Put ((the block in the box) on the table) in the kitchen.
  - Put (the block (in the box on the table)) in the kitchen.
  - Put (the block in the box) (on the table in the kitchen)

- **A general case:**
  \[
  Cat_n = \binom{2n}{n} - \binom{2n}{n-1} \sim \frac{4^n}{n^{3/2} \sqrt{\pi}}
  \]
  
  1, 2, 5, 14, 42, 132, 429, 1430, 4862, 16796, 58786, ... 

This is only one type of ambiguity but you have many types in almost all sentences
Why parsing is hard? Ambiguity

- A typical tree from a standard dataset (Penn treebank WSJ)

Canadian Utilities had 1988 revenue of $1.16 billion, mainly from its natural gas and electric utility businesses in Alberta, where the company serves about 800,000 customers.
Key problems

- **Recognition problem**: is the sentence grammatical?
- **Parsing problem**: what is a (most plausible) derivation (tree) corresponding the sentence?
- Parsing problem encompasses the recognition problem

A more interesting question from practical point of view
Context-Free Grammar

- Context-free grammar is a tuple of 4 elements \( G = (V, \Sigma, R, S) \)

- \( V \) - the set of **non-terminals**

- \( \Sigma \) - the set of **terminals**

- \( R \) - the set of **rules** of the form \( X \rightarrow Y_1, Y_2, \ldots, Y_n \), where \( n \geq 0 \), \( X \in V \), \( Y_i \in V \cup \Sigma \)

- \( S \) is a dedicated start symbol

In our case: phrase categories (VP, NP, ..) and PoS tags (N, V, .. – aka **preterminals**)
An example grammar

\[ V = \{S, VP, NP, PP, N, V, PN, P\} \]

\[ \Sigma = \{\text{girl}, \text{telescope}, \text{sandwich}, I, \text{saw}, \text{ate}, \text{with}, \text{in}, a, \text{the}\} \]

\[ S = \{S\} \]

\[ R : \]

\[ S \rightarrow NP \ VP \quad (\text{NP A girl}) \ (\text{VP ate a sandwich}) \]

\[ VP \rightarrow V \]

\[ VP \rightarrow V \ NP \quad (V \text{ ate}) \ (\text{NP a sandwich}) \]

\[ VP \rightarrow VP \ PP \quad (\text{VP saw a girl}) \ (\text{PP with a telescope}) \]

\[ NP \rightarrow NP \ PP \quad (\text{NP a girl}) \ (\text{PP with a sandwich}) \]

\[ NP \rightarrow D \ N \quad (D \ a) \ (N \text{ sandwich}) \]

\[ NP \rightarrow PN \]

\[ PP \rightarrow P \ NP \quad (P \text{ with}) \ (\text{NP with a sandwich}) \]

Preterminal rules (correspond to the HMM emission model \textbf{aka task model})

\[ N \rightarrow \text{girl} \]

\[ N \rightarrow \text{telescope} \]

\[ N \rightarrow \text{sandwich} \]

\[ PN \rightarrow I \]

\[ V \rightarrow \text{saw} \]

\[ V \rightarrow \text{ate} \]

\[ P \rightarrow \text{with} \]

\[ P \rightarrow \text{in} \]

\[ D \rightarrow a \]

\[ D \rightarrow \text{the} \]
CFGs

\[ S \rightarrow NP \ VP \]

\[ VP \rightarrow V \]
\[ VP \rightarrow V \ NP \]
\[ VP \rightarrow VP \ PP \]
\[ NP \rightarrow NP \ PP \]
\[ NP \rightarrow D \ N \]
\[ NP \rightarrow PN \]
\[ PP \rightarrow P \ NP \]

\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

S \rightarrow NP \ VP

VP \rightarrow V

VP \rightarrow V \ NP

VP \rightarrow VP \ PP

NP \rightarrow NP \ PP

NP \rightarrow D \ N

NP \rightarrow PN

PP \rightarrow P \ NP

N \rightarrow girl

N \rightarrow telescope

N \rightarrow sandwich

PN \rightarrow I

V \rightarrow saw

V \rightarrow ate

P \rightarrow with

P \rightarrow in

D \rightarrow a

D \rightarrow the
$$S \rightarrow NP \ VP$$

$$VP \rightarrow V$$

$$VP \rightarrow V \ NP$$

$$VP \rightarrow VP \ PP$$

$$NP \rightarrow NP \ PP$$

$$NP \rightarrow D \ N$$

$$NP \rightarrow PN$$

$$PP \rightarrow P \ NP$$

$$N \rightarrow girl$$

$$N \rightarrow telescope$$

$$N \rightarrow sandwich$$

$$PN \rightarrow I$$

$$V \rightarrow saw$$

$$V \rightarrow ate$$

$$P \rightarrow with$$

$$P \rightarrow in$$

$$D \rightarrow a$$

$$D \rightarrow the$$
CFGs

\[
S \rightarrow NP\ VP \\
NP \rightarrow NP\ PP \\
PN \rightarrow I \\
I \rightarrow I \\
VP \rightarrow V NP \\
VP \rightarrow VP PP \\
VP \rightarrow V N \\
VP \rightarrow VP PP \\

N \rightarrow girl \\
N \rightarrow telescope \\
N \rightarrow sandwich \\
PN \rightarrow I \\
V \rightarrow saw \\
V \rightarrow ate \\
P \rightarrow with \\
P \rightarrow in \\
D \rightarrow a \\
D \rightarrow the \\
\]
CFGs

\[ S \rightarrow NP \ VP \]
\[ V \rightarrow saw \]
\[ V \rightarrow ate \]
\[ N \rightarrow girl \]
\[ N \rightarrow telescope \]
\[ N \rightarrow sandwich \]
\[ PN \rightarrow I \]
\[ P \rightarrow with \]
\[ P \rightarrow in \]
\[ D \rightarrow a \]
\[ D \rightarrow the \]
CFGs

\[
S \rightarrow NP \ VP \\
NP \rightarrow D \ N \\
NP \rightarrow PN \\
VP \rightarrow VP \ PP \\
VP \rightarrow V \ NP \\
VP \rightarrow V \\
PN \rightarrow I \\
V \rightarrow saw \\
V \rightarrow ate \\
P \rightarrow with \\
P \rightarrow in \\
D \rightarrow a \\
D \rightarrow the
\]
CFGs

\[
S \rightarrow NP\ VP \\
NP \rightarrow NP\ PP \\
NP \rightarrow D\ N \\
PP \rightarrow P\ NP
\]

\[
N \rightarrow girl \\
N \rightarrow telescope \\
N \rightarrow sandwich \\
PN \rightarrow I \\
V \rightarrow saw \\
V \rightarrow ate \\
P \rightarrow with \\
P \rightarrow in \\
D \rightarrow a \\
D \rightarrow the
\]
CFGs

\[
S \to NP \ VP \\
VP \to V \\
VP \to VP \ NP \\
VP \to VP \ PP \\
NP \to NP \ PP \\
NP \to D \ N \\
NP \to PN \\
PP \to P \ NP \\
P \to with \\
D \to a \\
D \to the \\
\]

\[
N \to girl \\
N \to telescope \\
N \to sandwich \\
PN \to I \\
V \to saw \\
V \to ate \\
P \to in \\
D \to a \\
\]
CFGs

CFG defines both:

- a set of substrings (a language)
- structures used to represent sentences (constituent trees)
Why context-free?

What can be a sub-tree is only affected by what the phrase type is (VP) but not the context.
Why context-free?

What can be a sub-tree is only affected by what the phrase type is (VP) but not the context.

Not grammatical

The context-free nature of the phrase tree means that the structure is determined by the phrase types and their relationships, not by the specific context in which the phrase is used.

For example, in the sentence "The dog ate a sandwich," the sub-tree that includes "dog" and "ate" is unaffected by the context of whether the dog is eating a sandwich or not. Similarly, in the sentence "The dog often ate a sandwich," the sub-tree that includes "dog" and "ate" is again unaffected by the context of frequency.

This property is important in natural language processing, as it allows for the creation of parsers that can handle a wide variety of sentence structures without being overly sensitive to the context in which they are used.
Why context-free?

What can be a sub-tree is only affected by what the phrase type is (VP) but not the context.

Not grammatical

Matters if we want to generate language (e.g., language modeling) but is this relevant to parsing?
Introduced coordination ambiguity

Here, the coarse VP and NP categories cannot enforce subject-verb agreement in number resulting in the coordination ambiguity.

"Bark" can refer both to a noun or a verb.
Introduced coordination ambiguity

- Here, the coarse VP and NP categories cannot enforce subject-verb agreement in number resulting in the coordination ambiguity.

"Bark" can refer both to a noun or a verb.

This tree would be ruled out if the context would be somehow captured (subject-verb agreement).

Coordination

Even more detailed PoS tags are not going to help here.
Key problems

- **Recognition problem:** does the sentence belong to the language defined by CFG?
  - That is: is there a derivation which yields the sentence?

- **Parsing problem:** what is a (most plausible) derivation (tree) corresponding the sentence?

- Parsing problem encompasses the recognition problem
How to deal with ambiguity?

- There are (exponentially) many derivation for a typical sentence

```
S

VP

NP  PP

Put  NP  PP

D  N  P

the  box  in

VP

NP  PP

Put  NP  PP

D  N  P

the  box  in

Put the block in the box on the table in the kitchen

S  VP

V  PP

NP

NP

D  N  P

the  box  on

S  VP

V  PP

NP

NP

D  N  P

the  box  in

S  VP

V  PP

NP

NP

D  N  P

the  box  in
```

- We want to score all the derivations to encode how plausible they are
An example probabilistic CFG

Associate probabilities with the rules $p(X \rightarrow \alpha)$:

$$\forall X \rightarrow \alpha \in R : \quad 0 \leq p(X \rightarrow \alpha) \leq 1$$
$$\forall X \in N : \quad \sum_{\alpha: X \rightarrow \alpha \in R} p(X \rightarrow \alpha) = 1$$

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
<th>Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP \ VP$</td>
<td>1.0</td>
<td>(NP A girl) (VP ate a sandwich)</td>
</tr>
<tr>
<td>$VP \rightarrow V$</td>
<td>0.2</td>
<td>(VP ate) (NP a sandwich)</td>
</tr>
<tr>
<td>$VP \rightarrow V \ NP$</td>
<td>0.4</td>
<td>(VP saw a girl) (PP with …)</td>
</tr>
<tr>
<td>$NP \rightarrow NP \ PP$</td>
<td>0.3</td>
<td>(NP a girl) (PP with ….)</td>
</tr>
<tr>
<td>$NP \rightarrow D \ N$</td>
<td>0.5</td>
<td>(D a) (N sandwich)</td>
</tr>
<tr>
<td>$NP \rightarrow PN$</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>$PP \rightarrow P \ NP$</td>
<td>1.0</td>
<td>(P with) (NP with a sandwich)</td>
</tr>
<tr>
<td>$N \rightarrow girl$</td>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>$N \rightarrow telescope$</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>$N \rightarrow sandwich$</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>$PN \rightarrow I$</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>$V \rightarrow saw$</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>$V \rightarrow ate$</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>$P \rightarrow with$</td>
<td>0.6</td>
<td></td>
</tr>
<tr>
<td>$P \rightarrow in$</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>$D \rightarrow a$</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>$D \rightarrow the$</td>
<td>0.7</td>
<td></td>
</tr>
</tbody>
</table>

Now we can score a tree as a product of probabilities corresponding to the used rules.
### CFGs

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP\ VP$</td>
<td>$1.0$</td>
</tr>
<tr>
<td>$VP \rightarrow V$</td>
<td>$0.2$</td>
</tr>
<tr>
<td>$VP \rightarrow V\ NP$</td>
<td>$0.4$</td>
</tr>
<tr>
<td>$VP \rightarrow VP\ PP$</td>
<td>$0.4$</td>
</tr>
<tr>
<td>$NP \rightarrow NP\ PP$</td>
<td>$0.3$</td>
</tr>
<tr>
<td>$NP \rightarrow D\ N$</td>
<td>$0.5$</td>
</tr>
<tr>
<td>$NP \rightarrow PN$</td>
<td>$0.2$</td>
</tr>
<tr>
<td>$PP \rightarrow P\ NP$</td>
<td>$1.0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N \rightarrow girl$</td>
<td>$0.2$</td>
</tr>
<tr>
<td>$N \rightarrow telescope$</td>
<td>$0.7$</td>
</tr>
<tr>
<td>$N \rightarrow sandwich$</td>
<td>$0.1$</td>
</tr>
<tr>
<td>$PN \rightarrow I$</td>
<td>$1.0$</td>
</tr>
<tr>
<td>$V \rightarrow saw$</td>
<td>$0.5$</td>
</tr>
<tr>
<td>$V \rightarrow ate$</td>
<td>$0.5$</td>
</tr>
<tr>
<td>$P \rightarrow with$</td>
<td>$0.6$</td>
</tr>
<tr>
<td>$P \rightarrow in$</td>
<td>$0.4$</td>
</tr>
<tr>
<td>$D \rightarrow a$</td>
<td>$0.3$</td>
</tr>
<tr>
<td>$D \rightarrow the$</td>
<td>$0.7$</td>
</tr>
</tbody>
</table>

$p(T) =$
\[ p(T) = 1.0 \times \]

**CFGs**

\[
\begin{align*}
S \rightarrow NP \ VP & \quad 1.0 \\
VP \rightarrow V & \quad 0.2 \\
VP \rightarrow V \ NP & \quad 0.4 \\
VP \rightarrow VP \ PP & \quad 0.4 \\
NP \rightarrow NP \ PP & \quad 0.3 \\
NP \rightarrow D \ N & \quad 0.5 \\
NP \rightarrow PN & \quad 0.2 \\
PP \rightarrow P \ NP & \quad 1.0 \\
N \rightarrow girl & \quad 0.2 \\
N \rightarrow telescope & \quad 0.7 \\
N \rightarrow sandwich & \quad 0.1 \\
PN \rightarrow I & \quad 1.0 \\
V \rightarrow saw & \quad 0.5 \\
V \rightarrow ate & \quad 0.5 \\
P \rightarrow with & \quad 0.6 \\
P \rightarrow in & \quad 0.4 \\
D \rightarrow a & \quad 0.3 \\
D \rightarrow the & \quad 0.7 
\end{align*}
\]
\[
p(T) = 1.0 \times 0.2 \times 0.5 \times 0.4 = 0.4
\]
"CFGs"

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times \]

\[ S \rightarrow NP \ VP \ 1.0 \]
\[ VP \rightarrow V \ 0.2 \]
\[ VP \rightarrow V \ NP \ 0.4 \]
\[ VP \rightarrow VP \ PP \ 0.4 \]
\[ NP \rightarrow NP \ PP \ 0.3 \]
\[ NP \rightarrow D \ N \ 0.5 \]
\[ NP \rightarrow PN \ 0.2 \]
\[ PP \rightarrow P \ NP \ 1.0 \]

\[ N \rightarrow girl \ 0.2 \]
\[ N \rightarrow telescope \ 0.7 \]
\[ N \rightarrow sandwich \ 0.1 \]
\[ PN \rightarrow I \ 1.0 \]
\[ V \rightarrow saw \ 0.5 \]
\[ V \rightarrow ate \ 0.5 \]
\[ P \rightarrow with \ 0.6 \]
\[ P \rightarrow in \ 0.4 \]
\[ D \rightarrow a \ 0.3 \]
\[ D \rightarrow the \ 0.7 \]
\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times \]
$p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times$

$S \rightarrow NP \; VP \; 1.0$

$VP \rightarrow V \; 0.2$

$VP \rightarrow V \; NP \; 0.4$

$VP \rightarrow VP \; PP \; 0.4$

$NP \rightarrow NP \; PP \; 0.3$

$NP \rightarrow D \; N \; 0.5$

$NP \rightarrow PN \; 0.2$

$PP \rightarrow P \; NP \; 1.0$

$N \rightarrow girl \; 0.2$

$N \rightarrow telescope \; 0.7$

$N \rightarrow sandwich \; 0.1$

$PN \rightarrow I \; 1.0$

$V \rightarrow saw \; 0.5$

$V \rightarrow ate \; 0.5$

$P \rightarrow with \; 0.6$

$P \rightarrow in \; 0.4$

$D \rightarrow a \; 0.3$

$D \rightarrow the \; 0.7$
$p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3\times$

CFGs

$S \rightarrow NP \ VP \ 1.0$
$VP \rightarrow V \ 0.2$
$VP \rightarrow V \ NP \ 0.4$
$VP \rightarrow VP \ PP \ 0.4$
$NP \rightarrow NP \ PP \ 0.3$
$NP \rightarrow D \ N \ 0.5$
$NP \rightarrow PN \ 0.2$
$PP \rightarrow P \ NP \ 1.0$

$N \rightarrow \text{girl} \ 0.2$
$N \rightarrow \text{telescope} \ 0.7$
$N \rightarrow \text{sandwich} \ 0.1$
$PN \rightarrow I \ 1.0$
$V \rightarrow \text{saw} \ 0.5$
$V \rightarrow \text{ate} \ 0.5$
$P \rightarrow \text{with} \ 0.6$
$P \rightarrow \text{in} \ 0.4$
$D \rightarrow a \ 0.3$
$D \rightarrow the \ 0.7$
$$p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times$$

CFGs

$$S \rightarrow NP \ VP \ 1.0$$

$$VP \rightarrow V \ 0.2$$

$$VP \rightarrow V \ NP \ 0.4$$

$$VP \rightarrow VP \ PP \ 0.4$$

$$NP \rightarrow NP \ PP \ 0.3$$

$$NP \rightarrow D \ N \ 0.5$$

$$NP \rightarrow PN \ 0.2$$

$$PP \rightarrow P \ NP \ 1.0$$

$$N \rightarrow girl \ 0.2$$

$$N \rightarrow telescope \ 0.7$$

$$N \rightarrow sandwich \ 0.1$$

$$PN \rightarrow I \ 1.0$$

$$V \rightarrow saw \ 0.5$$

$$V \rightarrow ate \ 0.5$$

$$P \rightarrow with \ 0.6$$

$$P \rightarrow in \ 0.4$$

$$D \rightarrow a \ 0.3$$

$$D \rightarrow the \ 0.7$$
CFGs

\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times 0.3 \times \]

```
S \rightarrow NP \ VP \ 1.0
VP \rightarrow V \ 0.2
VP \rightarrow V \ NP \ 0.4
VP \rightarrow VP \ PP \ 0.4
NP \rightarrow NP \ PP \ 0.3
NP \rightarrow D \ N \ 0.5
NP \rightarrow PN \ 0.2
PP \rightarrow P \ NP \ 1.0
N \rightarrow girl \ 0.2
N \rightarrow telescope \ 0.7
N \rightarrow sandwich \ 0.1
PN \rightarrow I \ 1.0
V \rightarrow saw \ 0.5
V \rightarrow ate \ 0.5
P \rightarrow with \ 0.6
P \rightarrow in \ 0.4
D \rightarrow a \ 0.3
D \rightarrow the \ 0.7
```

```
S
  \_NP_ 1.0
  \_VP_ 0.4
  \_PN_ 0.3
    \_V_ 0.5
    \_saw
    \_NP_ 0.5
      \_D_ 0.3
      \_a
  \_PP
```
\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times 0.3 \times 0.2 \]
\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times 0.3 \times 0.2 \times 1.0 \times 0.6 \times 0.5 \times 0.3 \times 0.7 \]

\[ = 2.26 \times 10^{-5} \]
Distribution over trees

- We defined a distribution over production rules for each nonterminal.
- Our goal was to define a distribution over parse trees.

Unfortunately, not all PCFGs give rise to a proper distribution over trees, i.e. the sum over probabilities of all trees the grammar can generate may be less than 1: \[ \sum_T P(T) < 1 \]
Distribution over trees

- We defined a distribution over production rules for each nonterminal.
- Our goal was to define a distribution over parse trees.

Unfortunately, not all PCFGs give rise to a proper distribution over trees, i.e. the sum over probabilities of all trees the grammar can generate may be less than 1:

$$\sum_T P(T) < 1$$

- Good news: any PCFG estimated with the maximum likelihood procedure are always proper [Chi and Geman, 98]
Distribution over trees

- Let us denote by $G(x)$ the set of derivations for the sentence $x$.
- The probability distribution defines the scoring $P(T)$ over the trees $T \in G(x)$.
- Finding the best parse for the sentence according to PCFG:

$$\arg \max_{T \in G(x)} P(T)$$

We will look at this at the next class.
Outline

- Syntax: intro, CFGs, PCFGs
- PCFGs: Estimation
- CFGs: Parsing
- PCFGs: Parsing
- Parsing evaluation
- Unsupervised estimation of PCFGs (?)
- Grammar refinement (producing a state-of-the-art parser) (?)
ML estimation

- A treebank: a collection sentences annotated with constituent trees
ML estimation

- A treebank: a collection sentences annotated with constituent trees

- An estimated probability of a rule (maximum likelihood estimates)

\[ p(X \rightarrow \alpha) = \]
ML estimation

- A treebank: a collection of sentences annotated with constituent trees

- An estimated probability of a rule (maximum likelihood estimates)

\[ p(X \rightarrow \alpha) = \frac{C(X \rightarrow \alpha)}{C(X)} \]

- The number of times the rule used in the corpus

- The number of times the nonterminal X appears in the treebank
ML estimation

- A treebank: a collection sentences annotated with constituent trees

- An estimated probability of a rule (maximum likelihood estimates)

\[ p(X \rightarrow \alpha) = \frac{C(X \rightarrow \alpha)}{C(X)} \]

- Smoothing is helpful
  - Especially important for preterminal rules, i.e. generation of words (\(=\) task mode in PoS tagging)
  - See smoothing techniques in Section 4.5 of the J&M book (+ section 5.8.2 for some techniques specifically dealing with to unknown words)
ML estimation: an example

- A toy treebank:

  \[
  n_1 \times \begin{array}{c}
  S \\
  B \quad C \\
  a \quad a \\
  \end{array}
  \]

  \[
  n_2 \times \begin{array}{c}
  S \\
  C \\
  a \quad a \\
  \end{array}
  \]

  \[
  n_3 \times \begin{array}{c}
  S \\
  B \\
  a \\
  \end{array}
  \]

- Without smoothing:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Count</th>
<th>Prob. estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S \rightarrow B \ C)</td>
<td>(n_1)</td>
<td>(n_1 / (n_1 + n_2 + n_3))</td>
</tr>
<tr>
<td>(S \rightarrow C)</td>
<td>(n_2)</td>
<td>(n_2 / (n_1 + n_2 + n_3))</td>
</tr>
<tr>
<td>(S \rightarrow B)</td>
<td>(n_3)</td>
<td>(n_3 / (n_1 + n_2 + n_3))</td>
</tr>
<tr>
<td>(B \rightarrow a \ a)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(B \rightarrow a)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(C \rightarrow a \ a)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(C \rightarrow a \ a \ a)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ML estimation: an example

- A toy treebank:

  \[
  S \rightarrow B \quad \text{with count } n_1 \times \\
  B \quad \text{with count } n_2 \times \\
  C \quad \text{with count } n_3 \times \\
  a \quad a \quad a \quad a \\
  a \quad a \quad a \\
  B \quad a \\
  B \quad a \\
  C \quad a \quad a \\
  C \quad a \quad a \quad a
  \]

- Without smoothing:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Count</th>
<th>Prob. estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow B \ C$</td>
<td>$n_1$</td>
<td>$n_1/(n_1 + n_2 + n_3)$</td>
</tr>
<tr>
<td>$S \rightarrow C$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S \rightarrow B$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B \rightarrow a \ a$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$B \rightarrow a$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C \rightarrow a \ a$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$C \rightarrow a \ a \ a$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ML estimation: an example

A toy treebank:

```
  S
 / \      S
B   C     C
/ \    / |
a a  a  a   a
```

Without smoothing:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Count</th>
<th>Prob. estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow B C$</td>
<td>$n_1$</td>
<td>$n_1/(n_1 + n_2 + n_3)$</td>
</tr>
<tr>
<td>$S \rightarrow C$</td>
<td>$n_2$</td>
<td>$n_2/(n_1 + n_2 + n_3)$</td>
</tr>
<tr>
<td>$S \rightarrow B$</td>
<td>$n_3$</td>
<td>$n_3/(n_1 + n_2 + n_3)$</td>
</tr>
<tr>
<td>$B \rightarrow a a$</td>
<td>$n_1$</td>
<td>$n_1/(n_1 + n_3)$</td>
</tr>
<tr>
<td>$B \rightarrow a$</td>
<td>$n_3$</td>
<td>$n_3/(n_1 + n_3)$</td>
</tr>
<tr>
<td>$C \rightarrow a a$</td>
<td>$n_1$</td>
<td>$n_1/(n_1 + n_2)$</td>
</tr>
<tr>
<td>$C \rightarrow a a a$</td>
<td>$n_2$</td>
<td>$n_2/(n_1 + n_2)$</td>
</tr>
</tbody>
</table>
Penn Treebank: peculiarities

- Wall street journal: around 40,000 annotated sentences, 1,000,000 words
- Fine-grain part of speech tags (45), e.g., for verbs
  - VBD: Verb, past tense
  - VBG: Verb, gerund or present participle
  - VBP: Verb, present (non-3rd person singular)
  - VBZ: Verb, present (3rd person singular)
  - MD: Modal
- Flat NPs (no attempt to disambiguate NP attachment)

```
NP
  /\    /
 DT JJ NN NN NN
 / \ / \ / \
 a hot dog food cart
```
Outline

- Unsupervised estimation: forward-backward
- Syntax: intro, CFGs, PCFGs
- PCFGs: Estimation
  - CFGs: Parsing
- PCFGs: Parsing
- Parsing evaluation
- Unsupervised estimation of PCFGs (?)
- Grammar refinement (producing a state-of-the-art parser) (?)
Parsing

- **Parsing is search** through the space of all possible parses
  - e.g., we may want either any parse, all parses or the highest scoring parse (if PCFG):
    \[
    \arg\max_{T \in G(x)} P(T)
    \]
    The probability by the PCFG model
    Set of all trees given by the grammar for the sentence \(x\)

- **Bottom-up:**
  - One starts from words and attempt to construct the full tree

- **Top-down**
  - Start from the start symbol and attempt to expand to get the sentence
CKY algorithm (aka CYK)

- Cocke-Kasami-Younger algorithm
  - Independently discovered in late 60s / early 70s
- An efficient bottom up parsing algorithm for (P)CFGs
  - can be used both for the recognition and parsing problems
- Very important in NLP (and beyond)

- We will start with the non-probabilistic version
Constraints on the grammar

- The basic CKY algorithm supports only rules in the **Chomsky Normal Form (CNF)**:

  \[ C \rightarrow x \]
  \[ C \rightarrow C_1 C_2 \]

  **Unary preterminal** rules (generation of words given PoS tags) \( N \rightarrow \text{telescope} , D \rightarrow \text{the} , \ldots \)

  **Binary inner** rules (e.g., \( S \rightarrow NP \ VP \), \( NP \rightarrow D \ N \))
Constraints on the grammar

- The basic CKY algorithm supports only rules in the \textit{Chomsky Normal Form (CNF)}:
  \[
  C \rightarrow x \\
  C \rightarrow C_1 C_2
  \]
  - **Unary preterminal** rules (generation of words given PoS tags \( N \rightarrow \text{telescope}, D \rightarrow \text{the}, \ldots \))
  - **Binary inner** rules (e.g., \( S \rightarrow NP \ VP, \ NP \rightarrow D \ N \))

- Any CFG can be converted to an equivalent CNF
  - Equivalent means that they define \textit{the same language}
  - However (syntactic) \textit{trees will look differently}
  - It is possible to address it but defining such transformations that allows for easy reverse transformation

Makes linguists unhappy

Equivalent means that they define \textit{the same language}

However (syntactic) \textit{trees will look differently}

It is possible to address it but defining such transformations that allows for easy \textit{reverse transformation}
Transformation to CNF form

What one need to do to convert to CNF form

- Get rid of empty (aka epsilon) productions: $C \rightarrow \epsilon$
- Get rid of unary rules: $C \rightarrow C_1$
- N-ary rules: $C \rightarrow C_1 C_2 \ldots C_n \ (n > 2)$

Generally not a problem as there are not empty production in the standard (postprocessed) treebanks

Not a problem, as our CKY algorithm will support unary rules

Crucial to process them, as required for efficient parsing
Transformation to CNF form: binarization

- Consider \( NP \rightarrow DT \ NNP \ VBG \ NN \)

```
NP
  
DT    NNP    VBG    NN
 the    Dutch    publishing    group
```

- How do we get a set of binary rules which are equivalent?
Transformation to CNF form: binarization

- Consider \( NP \rightarrow DT\ NNP\ VBG\ NN \)

  NP
  ├── DT
  │    └── the
  ├── NNP
  │    └── Dutch
  ├── VBG
  │    └── publishing
  └── NN
        └── group

- How do we get a set of binary rules which are equivalent?

  \[
  NP \rightarrow DT\ X \\
  X \rightarrow NNP\ Y \\
  Y \rightarrow VBG\ NN
  \]
Transformation to CNF form: binarization

- **Consider** \[ NP \rightarrow DT \ NNP \ VBG \ NN \]

  NP
  
  DT \quad NNP \quad VBG \quad NN
  
  the \quad Dutch \quad publishing \quad group

- **How do we get a set of binary rules which are equivalent?**

  \[ NP \rightarrow DT \ X \]
  \[ X \rightarrow NNP \ Y \]
  \[ Y \rightarrow VBG \ NN \]

- **A more systematic way to refer to new non-terminals**

  \[ NP \rightarrow DT \ @NP|DT \]
  \[ @NP|DT \rightarrow NNP \ @NP|DT.NNP \]
  \[ @NP|DT.NNP \rightarrow VBG \ NN \]
Instead of binarizing rules we can binarize trees on preprocessing:

Also known as lossless Markovization in the context of PCFGs

Can be easily reversed on postprocessing
Instead of binarizing rules we can binarize trees on preprocessing:
CKY: Parsing task

- We are given
  - a grammar \( G = (V, \Sigma, R, S) \)
  - a sequence of words \( w = (w_1, w_2, \ldots, w_n) \)
- Our goal is to produce a parse tree for \( w \)
CKY: Parsing task

- We are given
  - a grammar \( G = (V, \Sigma, R, S) \)
  - a sequence of words \( w = (w_1, w_2, \ldots, w_n) \)
- Our goal is to produce a parse tree for \( w \)
- We need an easy way to refer to substrings of \( w \)

**span** \((i, j)\) refers to words between fenceposts \(i\) and \(j\)
Key problems

- **Recognition problem**: does the sentence belong to the language defined by CFG?
  - Is there a derivation which yields the sentence?

- **Parsing problem**: what is a derivation (tree) corresponding the sentence?
  - Probabilistic parsing: what is the most probable tree for the sentence?
Parsing one word

\[ C \rightarrow w_i \]
Parsing one word

\[ C \rightarrow w_i \]
Parsing one word

\[ C \rightarrow w_i \]

covers all words between \( i-1 \) and \( i \)
Parsing longer spans

\[ C \rightarrow C_1 \quad C_2 \]

Check through all \( C_1, C_2, \text{mid} \)

covers all words btw \textit{min} and \textit{mid}  
covers all words btw \textit{mid} and \textit{max}
Parsing longer spans

$C \rightarrow C_1 \ C_2$

Check through all $C_1, C_2, mid$

covers all words btw $min$ and $mid$
covers all words btw $mid$ and $max$
Parsing longer spans

covers all words between min and max
Applications of rules is independent of inner structure of a parse tree
We only need to know the corresponding span and the root label of the tree

Its signature \([min, max, C]\)

Also known as an edge
CKY idea

- Compute for every span a set of admissible labels (may be empty for some spans)
  - Start from small trees (single words) and proceed to larger ones
CKY idea

- Compute for every span a set of admissible labels (may be empty for some spans)
  - Start from small trees (single words) and proceed to larger ones
- When done, how do we detect that the sentence is grammatical?
Compute for every span a set of admissible labels (may be empty for some spans)

- Start from small trees (single words) and proceed to larger ones

When done, check if $S$ is among admissible labels for the whole sentence, if yes – the sentence belong to the language

- That is if a tree with signature $[0, n, S]$ exists
CKY idea

- Compute for every span a set of admissible labels (may be empty for some spans)
  - Start from small trees (single words) and proceed to larger ones
- When done, check if $S$ is among admissible labels for the whole sentence, if yes – the sentence belong to the language
  - That is if a tree with signature $[0, n, S]$ exists
- Unary rules?
CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Preterminal rules

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$S \rightarrow NP \ VP$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$VP \rightarrow M \ V$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$VP \rightarrow V$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$NP \rightarrow N$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$NP \rightarrow N \ NP$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Inner rules

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$N \rightarrow can$</td>
<td></td>
</tr>
<tr>
<td>$N \rightarrow lead$</td>
<td></td>
</tr>
<tr>
<td>$N \rightarrow poison$</td>
<td></td>
</tr>
<tr>
<td>$M \rightarrow can$</td>
<td></td>
</tr>
<tr>
<td>$M \rightarrow must$</td>
<td></td>
</tr>
<tr>
<td>$V \rightarrow poison$</td>
<td></td>
</tr>
<tr>
<td>$V \rightarrow lead$</td>
<td></td>
</tr>
</tbody>
</table>
CKY in action

<table>
<thead>
<tr>
<th>min = 0</th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>min = 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>min = 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

$S \rightarrow NP \ VP$
$VP \rightarrow M \ V$
$VP \rightarrow V$
$NP \rightarrow N$
$NP \rightarrow N \ NP$

$N \rightarrow can$
$N \rightarrow lead$
$N \rightarrow poison$

$M \rightarrow can$
$M \rightarrow must$

$V \rightarrow poison$
$V \rightarrow lead$

Chart (aka parsing triangle)
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules

\[ S \rightarrow NP \ VP \]

Inner rules

\[ VP \rightarrow M \ V \]
\[ VP \rightarrow V \]

\[ NP \rightarrow N \]
\[ NP \rightarrow N \ NP \]

\[ N \rightarrow can \]
\[ N \rightarrow lead \]
\[ N \rightarrow poison \]

\[ M \rightarrow can \]
\[ M \rightarrow must \]

\[ V \rightarrow poison \]
\[ V \rightarrow lead \]
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$
$VP \rightarrow V$

$NP \rightarrow N$
$NP \rightarrow N \ NP$

$N \rightarrow can$
$N \rightarrow lead$
$N \rightarrow poison$

$M \rightarrow can$
$M \rightarrow must$

$V \rightarrow poison$
$V \rightarrow lead$
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
### CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

---

#### Preterminal rules

- $S \rightarrow NP VP$
- $VP \rightarrow M V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N NP$

#### Inner rules

- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1
max = 2
max = 3

min = 0
min = 1
min = 2

S

Preterminal rules

Inner rules

S → NP VP

VP → M V
VP → V

NP → N
NP → N NP

N → can
N → lead
N → poison

M → can
M → must

V → poison
V → lead
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  
max = 2  
max = 3

min = 0

min = 1

min = 2

\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N \ NP
\]

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

Preterminal rules
- $N \rightarrow \text{can}$
- $N \rightarrow \text{lead}$
- $N \rightarrow \text{poison}$
- $M \rightarrow \text{can}$
- $M \rightarrow \text{must}$
- $V \rightarrow \text{poison}$
- $V \rightarrow \text{lead}$

Inner rules

$$S \rightarrow NP \ VP$$

$$VP \rightarrow M \ V$$

$$VP \rightarrow V$$

$$NP \rightarrow N$$

$$NP \rightarrow N \ NP$$

$$\min = 0$$
- 1

$$\min = 1$$
- 2

$$\min = 2$$
- 3

max = 1     max = 2     max = 3

1

2

3
### CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>min</th>
<th>max = 1</th>
<th>max = 2</th>
<th>max = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>$N, V$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$N, M$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>$N, V$</td>
<td></td>
</tr>
</tbody>
</table>

### Preterminal rules
- $N \rightarrow \text{can}$
- $N \rightarrow \text{lead}$
- $N \rightarrow \text{poison}$
- $M \rightarrow \text{can}$
- $M \rightarrow \text{must}$
- $V \rightarrow \text{poison}$
- $V \rightarrow \text{lead}$

### Inner rules
- $S \rightarrow NP \ VP$
- $VP \rightarrow M \ V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N \ NP$
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0

1  $N, V$
   $NP, VP$

min = 1

2  $N, M$
   $NP$

min = 2

3  $N, V$
   $NP, VP$

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$
CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

Preterminal rules

$S \rightarrow NP \ VP$

$VP \rightarrow M \ V$

$VP \rightarrow V$

$NP \rightarrow N$

$NP \rightarrow N \ NP$

Inner rules

$N \rightarrow can$

$N \rightarrow lead$

$N \rightarrow poison$

$M \rightarrow can$

$M \rightarrow must$

$V \rightarrow poison$

$V \rightarrow lead$

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
</table>
| 1 | $N, V$
|   | $NP, VP$
| 2 | $N, M$
|   | $NP$
| 3 | $N, V$
|   | $NP, VP$
| 4 |   | ?

max = 1  max = 2  max = 3

min = 0  min = 1  min = 2
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules

- $S \rightarrow NP \ VP$
- $VP \rightarrow M \ V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N NP$

Inner rules

- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$

<table>
<thead>
<tr>
<th>min = 0</th>
<th>min = 1</th>
<th>min = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$N, V$</td>
<td>$N, M$</td>
<td>$N, V$</td>
</tr>
<tr>
<td>$NP, VP$</td>
<td>$NP$</td>
<td>$NP, VP$</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3
CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0

\[
\begin{array}{c}
\text{1} & N, V & NP, VP \\
2 & N, M & NP \\
3 & N, V & NP, VP \\
4 & NP & \\
\end{array}
\]

\[
S \rightarrow NP \ VP
\]

\[
VP \rightarrow M \ V
\]

\[
VP \rightarrow V
\]

\[
NP \rightarrow N
\]

\[
NP \rightarrow N\ NP
\]

\[
N \rightarrow can
\]

\[
N \rightarrow lead
\]

\[
N \rightarrow poison
\]

\[
M \rightarrow can
\]

\[
M \rightarrow must
\]

\[
V \rightarrow poison
\]

\[
V \rightarrow lead
\]
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>N, V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>N, M</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N, V</td>
<td></td>
</tr>
<tr>
<td>N, V</td>
<td>VP</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>NP, VP</td>
<td></td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>NP</td>
<td></td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>N</td>
<td></td>
</tr>
</tbody>
</table>

Check about unary rules: no unary rules here
## CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>max = 1</th>
<th>max = 2</th>
<th>max = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>(N, V)</td>
<td>(NP)</td>
<td></td>
</tr>
<tr>
<td>(NP, VP)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>min = 0</th>
<th>min = 1</th>
<th>min = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>(N, M)</td>
<td>(?)</td>
<td></td>
</tr>
<tr>
<td>(NP)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Preterminal rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>(S \rightarrow NP \ VP)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Inner rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>(VP \rightarrow M \ V)</td>
</tr>
<tr>
<td>(VP \rightarrow V)</td>
</tr>
<tr>
<td>(NP \rightarrow N)</td>
</tr>
<tr>
<td>(NP \rightarrow N \ NP)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Preterminal rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>(N \rightarrow can)</td>
</tr>
<tr>
<td>(N \rightarrow lead)</td>
</tr>
<tr>
<td>(N \rightarrow poison)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Preterminal rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>(M \rightarrow can)</td>
</tr>
<tr>
<td>(M \rightarrow must)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Preterminal rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>(V \rightarrow poison)</td>
</tr>
<tr>
<td>(V \rightarrow lead)</td>
</tr>
</tbody>
</table>
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

max = 1  max = 2  max = 3

min = 0  

1

2

3

4

5

N, V  
NP, VP

N, M  
NP

N, V  
NP, VP

N  
P

NP, VP

S, VP, NP

S  
NP, VP

NP  

Preterminal rules

VP → M V
VP → V

NP → N
NP → N NP

Inner rules

N → can
N → lead
N → poison
M → can
M → must
V → poison
V → lead
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

- **Lead**
  - 0
  - 1
  - 2
  - 3

- **Can**
  - 0
  - 1
  - 2
  - 3

- **Poison**
  - 0
  - 1
  - 2
  - 3

**Preterminal rules**

- $S \rightarrow NP \ VP$

**Inner rules**

- $VP \rightarrow M \ V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N \ NP$

**Nonterminal rules**

- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$

**Check about unary rules:** no unary rules here.
### CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>min = 0</th>
<th>max = 1</th>
<th>max = 2</th>
<th>max = 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$N, V$</td>
<td>$NP$</td>
<td></td>
<td>$?$</td>
</tr>
<tr>
<td>2</td>
<td>$N, M$</td>
<td>$NP$</td>
<td>$S, VP, NP$</td>
<td>$?$</td>
</tr>
<tr>
<td>3</td>
<td>$N, V$</td>
<td>$NP, VP$</td>
<td></td>
<td>$?$</td>
</tr>
<tr>
<td>4</td>
<td>$NP$</td>
<td></td>
<td></td>
<td>$?$</td>
</tr>
<tr>
<td>5</td>
<td>$S, VP, NP$</td>
<td></td>
<td></td>
<td>$?$</td>
</tr>
<tr>
<td>6</td>
<td>$?$</td>
<td></td>
<td></td>
<td>$?$</td>
</tr>
</tbody>
</table>

**Preterminal rules**

- $S \rightarrow NP \ VP$
- $VP \rightarrow M \ V$
- $VP \rightarrow V$
- $NP \rightarrow N$
- $NP \rightarrow N \ NP$

**Inner rules**

- $N \rightarrow can$
- $N \rightarrow lead$
- $N \rightarrow poison$
- $M \rightarrow can$
- $M \rightarrow must$
- $V \rightarrow poison$
- $V \rightarrow lead$
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

\[
\begin{array}{c|cc}
\text{min} = 0 & N, V & \text{NP, VP} \\
\text{min} = 1 & N, M & \text{NP} \\
\text{min} = 2 & N, V & \text{NP, VP} \\
\end{array}
\]

max = 1 max = 2 max = 3

\[
\begin{array}{c|cc}
\text{max} = 1 & N, V & \text{NP} \\
\text{max} = 2 & N, M & \text{NP} \\
\text{max} = 3 & S, V, P, & ? \\
\end{array}
\]

Preterminal rules

\begin{align*}
S & \rightarrow \text{NP} \; \text{VP} \\
VP & \rightarrow M \; V \\
VP & \rightarrow V \\
NP & \rightarrow N \\
NP & \rightarrow N \; NP \\
N & \rightarrow \text{can} \\
N & \rightarrow \text{lead} \\
N & \rightarrow \text{poison} \\
M & \rightarrow \text{can} \\
M & \rightarrow \text{must} \\
V & \rightarrow \text{poison} \\
V & \rightarrow \text{lead}
\end{align*}
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

\[
\begin{array}{ccc}
\text{max} = 1 & \text{max} = 2 & \text{max} = 3 \\
1 & N, V & NP, VP \\
4 & NP & \\
6 & S, NP & \\
2 & N, M & NP \\
5 & S, VP, NP & \\
3 & N, V & NP, VP \\
\end{array}
\]

mid = 1

\[
S \rightarrow NP \ VP
\]

\[
\begin{align*}
VP & \rightarrow M \ V \\
VP & \rightarrow V \\
NP & \rightarrow N \\
NP & \rightarrow N \ NP \\
N & \rightarrow \text{can} \\
N & \rightarrow \text{lead} \\
N & \rightarrow \text{poison} \\
M & \rightarrow \text{can} \\
M & \rightarrow \text{must} \\
V & \rightarrow \text{poison} \\
V & \rightarrow \text{lead}
\end{align*}
\]
CKY in action

<table>
<thead>
<tr>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

Preterminal rules:
- \( S \rightarrow NP \ VP \)

Inner rules:
- \( VP \rightarrow M \ V \)
- \( VP \rightarrow V \)
- \( NP \rightarrow N \)
- \( NP \rightarrow N \ NP \)

Terminal rules:
- \( N \rightarrow can \)
- \( N \rightarrow lead \)
- \( N \rightarrow poison \)
- \( M \rightarrow can \)
- \( M \rightarrow must \)
- \( V \rightarrow poison \)
- \( V \rightarrow lead \)
CKY in action

<table>
<thead>
<tr>
<th></th>
<th>lead</th>
<th>can</th>
<th>poison</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>

min = 0

mid = 2

max = 0

max = 1

max = 2

max = 3

S → NP VP

VP → M V

VP → V

NP → N

NP → N NP

N → can

N → lead

N → poison

M → can

M → must

V → poison

V → lead

Apparently the sentence is ambiguous for the grammar: (as the grammar overgenerates)
Ambiguity

No subject-verb agreement, and *poison* used as an intransitive verb

Apparently the sentence is ambiguous for the grammar: (as the grammar overgenerates)
CKY more formally

- Chart can be represented by a Boolean array \( \text{chart}[\text{min}][\text{max}][\text{C}] \)
- Relevant entries have \( 0 < \text{min} < \text{max} \leq n \)
- \( \text{chart}[\text{min}][\text{max}][\text{C}] = \text{true if the signature } (\text{min}, \text{max}, \text{C}) \) is already added to the chart; \( \text{false otherwise.} \)

Here we assume that labels \( (\text{C}) \) are integer indices.
CKY more formally

- Chart can be represented by a Boolean array \( \text{chart}[\text{min}][\text{max}][C] \)
- Relevant entries have \( 0 < \text{min} < \text{max} \leq n \)
- \( \text{chart}[\text{min}][\text{max}][C] = \text{true} \) if the signature \((\text{min}, \text{max}, C)\) is already added to the chart; \( \text{false} \) otherwise.

Here we assume that labels \((C)\) are integer indices.
Implementation: preterminal rules

for each \( w_i \) from left to right

for each preterminal rule \( C \rightarrow w_i \)

\[ \text{chart}[i - 1][i][C] = \text{true} \]
Implementation: binary rules

for each max from 2 to n

  for each min from max - 2 down to 0

    for each syntactic category C

      for each binary rule C \rightarrow C_1 C_2

        for each mid from min + 1 to max - 1

          if chart[min][mid][C_1] and chart[mid][max][C_2] then

            chart[min][max][C] = true
Unary rules

- How to integrate unary rules $C \rightarrow C_1$?
Implementation: unary rules

for each max from 1 to n

for each min from max - 1 down to 0

// First, try all binary rules as before.
...

// Then, try all unary rules.

for each syntactic category C

for each unary rule C → C₁

if chart[min][max][C₁] then

chart[min][max][C] = true
Implementation: unary rules

for each max from 1 to n

for each min from max - 1 down to 0

// First, try all binary rules as before.
...

// Then, try all unary rules.

for each syntactic category C

for each unary rule C -> C₁

if chart[min][max][C₁] then

chart[min][max][C] = true

But we forgot something!
Unary closure

- What if the grammar contained 2 rules:
  
  \[ A \to B \]
  
  \[ B \to C \]

- But \( C \) can be derived from \( A \) by a chain of rules:
  
  \[ A \to B \to C \]

- One could support chains in the algorithm but it is easier to extend the grammar, to get the transitive closure:
  
  \[ A \to B \]
  
  \[ B \to C \]

\[ \Rightarrow \]

\[ A \to B \]

\[ B \to C \]

\[ A \to C \]
Unary closure

- What if the grammar contained 2 rules:
  \[ A \rightarrow B \]
  \[ B \rightarrow C \]

- But \( C \) can be derived from \( A \) by a chain of rules:
  \[ A \rightarrow B \rightarrow C \]

- One could support chains in the algorithm but it is easier to extend the grammar, to get the reflexive transitive closure

  \[ A \rightarrow B \]
  \[ B \rightarrow C \]
  \[ \Rightarrow \]
  \[ A \rightarrow A \]
  \[ A \rightarrow B \]
  \[ B \rightarrow C \]
  \[ B \rightarrow B \]
  \[ A \rightarrow C \]
  \[ C \rightarrow C \]

Convenient for programming reasons in the PCFG case
Implementation: skeleton

// int n = number of words in the sequence
// int m = number of syntactic categories in the grammar
// int s = the (number of the) grammar’s start symbol

boolean[][][] chart = new boolean[n + 1][n + 1][m]

// Recognize all parse trees built with preterminal rules.
// Recognize all parse trees built with inner rules.

return chart[0][n][s]
Algorithm analysis

- Time complexity?
Algorithm analysis

- Time complexity?

  ```
  for each max from 2 to n
    for each min from max - 2 down to 0
      for each syntactic category C
        for each binary rule C -> C₁ C₂
          for each mid from min + 1 to max - 1
  ```
Algorithm analysis

- **Time complexity?**

  \[
  \text{for each max from 2 to } n \\
  \quad \text{for each min from max - 2 down to 0} \\
  \quad \text{for each syntactic category } C \\
  \quad \quad \text{for each binary rule } C \rightarrow C_1 C_2 \\
  \quad \quad \quad \text{for each mid from min + 1 to max - 1}
  \]

- \( \Theta(n^3 |R|) \), where \(|R|\) is the number of rules in the grammar

A few seconds for sentences under < 20 words for a non-optimized parser
Algorithm analysis

- Time complexity?

  \[
  \text{for each } \max \text{ from 2 to } n
  
  \text{for each } \min \text{ from } \max - 2 \text{ down to } 0
  
  \text{for each syntactic category } C
  
  \text{for each binary rule } C \rightarrow C_1 \ C_2
  
  \text{for each } \mid \text{mid} \mid \text{ from } \min + 1 \text{ to } \max - 1
  
  \theta(n^3 \mid R\mid), \text{ where } \mid R\mid \text{ is the number of rules in the grammar}

- There exist algorithms with better asymptotical time complexity but the `constant' makes them slower in practice (in general)
Practical time complexity

- Time complexity? (for the PCFG version)

~ $n^{3.6}$
Outline

- Syntax: intro, CFGs, PCFGs
- PCFGs: Estimation
- CFGs: Parsing
- PCFGs: Parsing
- Parsing evaluation
- Unsupervised estimation of PCFGs (?)
- Grammar refinement (producing a state-of-the-art parser) (?)
Probabilistic parsing

- We discussed the recognition problem:
  - check if a sentence is parsable with a CFG
- Now we consider parsing with PCFGs
  - Recognition with PCFGs: what is the probability of the most probable parse tree?
  - Parsing with PCFGs: What is the most probable parse tree?
\[ p(T) = 1.0 \times 0.2 \times 1.0 \times 0.4 \times 0.5 \times 0.3 \times 0.5 \times 0.3 \times 0.2 \times 1.0 \times 0.6 \times 0.5 \times 0.3 \times 0.7 \]
\[ = 2.26 \times 10^{-5} \]
Distribution over trees

- Let us denote by $G(x)$ the set of derivations for the sentence $x$
- The probability distribution defines the scoring $P(T)$ over the trees $T \in G(x)$
- Finding the best parse for the sentence according to PCFG:

$$\arg \max_{T \in G(x)} P(T)$$

This will be another Viterbi / Max-Product algorithm!
CKY with PCFGs

- Chart is represented by a **double array** `chart[min][max][C]`
  - It stores probabilities for the most probable subtree with a given signature
  - `chart[0][n][S]` will store the probability of the most probable full parse tree
For every $C$ choose $C_1$, $C_2$ and mid such that

$$P(T_1) \times P(T_2) \times P(C \rightarrow C_1C_2)$$

is maximal, where $T_1$ and $T_2$ are left and right subtrees.
Implementation: preterminal rules

for each $w_i$ from left to right

for each preterminal rule $C \rightarrow w_i$

$\text{chart}[i-1][i][C] = p(C \rightarrow w_i)$
Implementation: binary rules

for each max from 2 to n

for each min from max - 2 down to 0

for each syntactic category C

    double best = undefined

    for each binary rule C -> C₁ C₂

        for each mid from min + 1 to max - 1

            double t₁ = chart[min][mid][C₁]

            double t₂ = chart[mid][max][C₂]

            double candidate = t₁ * t₂ * p(C -> C₁ C₂)

        if candidate > best then

            best = candidate

        chart[min][max][C] = best
Unary rules

- Similarly to CFGs: after producing scores for signatures \((c, i, j)\), try to improve the scores by applying unary rules (and rule chains)
  - If improved, update the scores
Unary (reflexive transitive) closure

\[
A \rightarrow B \quad 0.1 \quad \Rightarrow \quad A \rightarrow B \quad 0.1 \\
B \rightarrow C \quad 0.2 \\
\rightarrow \quad B \rightarrow C \quad 0.2 \\
A \rightarrow C \quad 0.2 \times 0.1 \\
\rightarrow \quad C \rightarrow C \quad 1
\]

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.
Unary (reflexive transitive) closure

\[
\begin{align*}
A & \rightarrow B \quad 0.1 \\
B & \rightarrow C \quad 0.2
\end{align*}
\Rightarrow
\begin{align*}
A & \rightarrow B \quad 0.1 \\
B & \rightarrow C \quad 0.2 \\
A & \rightarrow C \quad 0.2 \times 0.1
\end{align*}
\]

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.

The fact that the rule is composite needs to be stored to recover the true tree.
Unary (reflexive transitive) closure

\[
\begin{align*}
A &\rightarrow B \quad 0.1 \\
B &\rightarrow C \quad 0.2 \\
A &\rightarrow C \quad 1.e-5 \\
\Rightarrow \\
A &\rightarrow B \quad 0.1 \\
B &\rightarrow C \quad 0.2 \\
A &\rightarrow C \quad 1.e-5 \\
\Rightarrow \\
A &\rightarrow B \quad 0.1 \\
B &\rightarrow C \quad 0.1 \\
A &\rightarrow C \quad 0.02 \\
\end{align*}
\]

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.

The fact that the rule is composite needs to be stored to recover the true tree.
Unary (reflexive transitive) closure

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow B$</td>
<td>0.1</td>
</tr>
<tr>
<td>$B \rightarrow C$</td>
<td>0.2</td>
</tr>
<tr>
<td>$A \rightarrow C$</td>
<td>$1.e-5$</td>
</tr>
</tbody>
</table>

The fact that the rule is composite needs to be stored to recover the true tree.

Note that this is not a PCFG anymore as the rules do not sum to 1 for each parent.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A \rightarrow B$</td>
<td>0.1</td>
</tr>
<tr>
<td>$B \rightarrow C$</td>
<td>0.1</td>
</tr>
<tr>
<td>$A \rightarrow C$</td>
<td>0.02</td>
</tr>
</tbody>
</table>

What about loops, like: $A \rightarrow B \rightarrow A \rightarrow C$?
Recovery of the tree

- For each signature we store backpointers to the elements from which it was built (e.g., rule and, for binary rules, midpoint)
  - start recovering from $[0, n, S]$  

- Be careful with unary rules
  - Basically you can assume that you always used an unary rule from the closure (but it could be the trivial one $C' ightarrow C'$)
Speeding up the algorithm (approximate search)

Any ideas?
Speeding up the algorithm (approximate search)

- **Basic pruning (roughly):**
  - For every span \((i,j)\) store only labels which have the probability at most \(N\) times smaller than the probability of the most probable label for this span.
  - Check not all rules but only rules yielding subtree labels having non-zero probability.

- **Coarse-to-fine pruning**
  - Parse with a smaller (simpler) grammar, and precompute (posterior) probabilities for each spans, and use only the ones with non-negligible probability from the previous grammar.
Outline

- Syntax: intro, CFGs, PCFGs
- PCFGs: Estimation
- CFGs: Parsing
- PCFGs: Parsing
- Parsing evaluation
- Unsupervised estimation of PCFGs (?)
- Grammar refinement (producing a state-of-the-art parser) (?)
Parser evaluation

Intrinsic evaluation:

- **Automatic**: evaluate against annotation provided by human experts (*gold standard*) according to some predefined measure
- **Manual**: … according to human judgment

Extrinsic evaluation: score syntactic representation by comparing how well a system using this representation performs on some task

- E.g., use syntactic representation as input for a semantic analyzer and compare results of the analyzer using syntax predicted by different parsers.

Though has many drawbacks it is easier and allows to track state-of-the-art across many years.
Standard evaluation setting in parsing

- **Automatic intrinsic evaluation is used**: parsers are evaluated against gold standard by provided by linguists

- **There is a standard split into the parts**:
  - **training set**: used for estimation of model parameters
  - **development set**: used for tuning the model (initial experiments)
  - **test set**: final experiments to compare against previous work
Automatic evaluation of constituent parsers

- **Exact match**: percentage of trees predicted correctly
- **Bracket score**: scores how well individual phrases (and their boundaries) are identified
- **Crossing brackets**: percentage of phrases boundaries crossing
- **Dependency metrics**: scores dependency structure corresponding to the constituent tree (percentage of correctly identified heads)

The most standard measure; we will focus on it.
Brackets scores

- The most standard score is **bracket score**
- It regards a tree as a collection of brackets: $[\text{min}, \text{max}, C]$
- The set of brackets predicted by a parser is compared against the set of brackets in the tree annotated by a linguist
- Precision, recall and F1 are used as scores
Bracketing notation

The same tree as a bracketed sequence

(S
  (NP (PN My) (N Dog) )
  (VP (V ate)
    (NP (D a ) (N sausage) )
  )
)

(S
  (NP (PN My) (N Dog) )
  (VP (V ate)
    (NP (D a ) (N sausage) )
  )
)
Brackets scores

\[ Pr = \frac{\text{number of brackets the parser and annotation agree on}}{\text{number of brackets predicted by the parser}} \]

\[ Re = \frac{\text{number of brackets the parser and annotation agree on}}{\text{number of brackets in annotation}} \]

\[ F_1 = \frac{2 \times Pr \times Re}{Pr + Re} \]

Harmonic mean of precision and recall
The best results reported (as of 2012)
Looked into this in BSc KI at UvA

We will look into this one