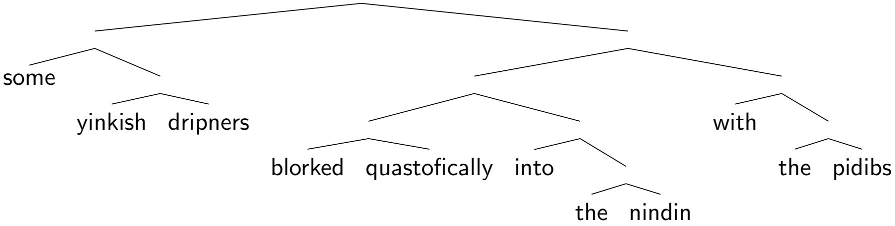
L90: Overview of Natural Language Processing

Lecture 4: Phrase Structure and Structured Prediction

Simone Teufel

Department of Computer Science and Technology
University of Cambridge

Michaelmas 2021/22



Words are organized into nested blocks

Lecture 4: Phrase Structure and Structured Prediction

1. Phrase structure
2. Structured prediction
3. Context-free grammars
4. Probabilistic context-free grammars

Slides by
Weiwei Sun

Phrase Structure

Carefully, very carefully...

(1) a. the guy who fixed the car very carefully packed his tools

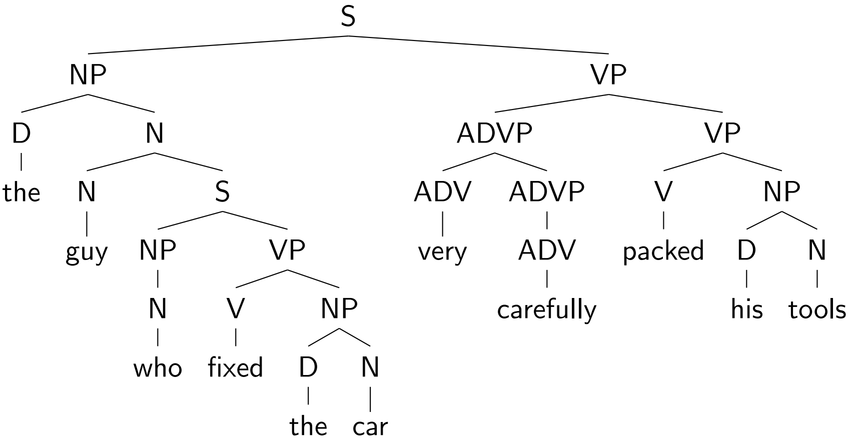
b. very carefully, the guy who fixed the car packed his tools

c. *very carefully, the guy who fixed the car is tall

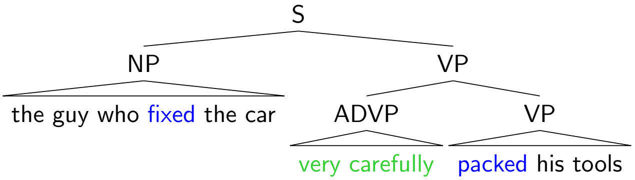
Constituency (phrase structure)

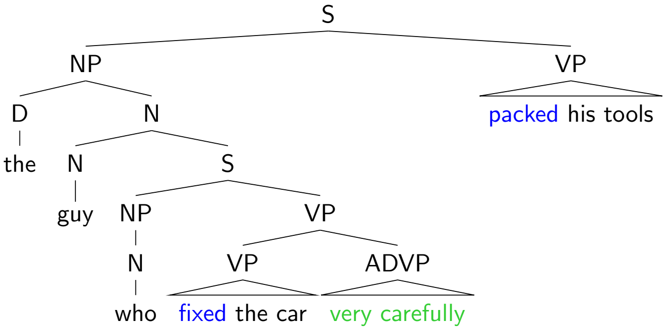
The basic idea

Phrase structure organizes words into *nested* *constituents*, which can be represented as **a tree**.



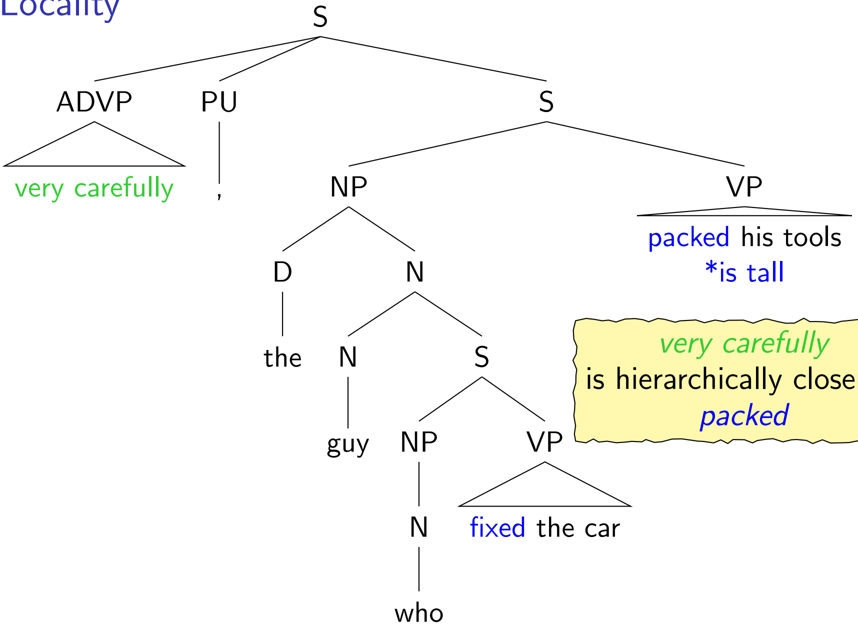
Different structures, different meaning





Results by a cool parser: `http://erg.delph-in.net/logon`

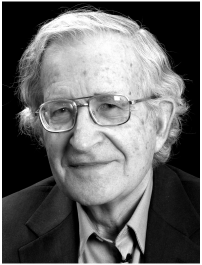
Locality



very carefully
is hierarchically closer to
packed

Interview of Noam Chomsky by Lex Fridman

I think the deepest property of language and puzzling property that's been discovered is what is sometimes called structure dependence. [...] Linear closeness is an easy computation, but here you're doing a much more, what looks like a more complex computation.



Noam Chomsky: Language, Cognition, and Deep Learning

www.youtube.com/watch?v=cMscNuSUy0I

Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

- Machine translation
- Information extraction
- Grammar checking
- etc.

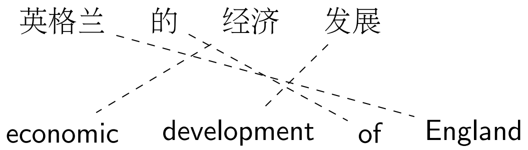
Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

- Machine translation
- Information extraction
- Grammar checking
- etc.

Translate “英格兰的经济发展” into English



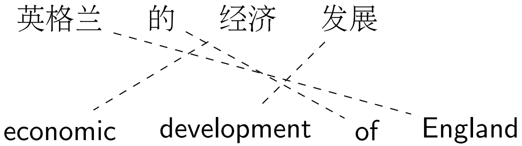
Applications of parsing

Modern parsers are quite accurate

For some languages, automatic syntactic parsing is good enough to help in a range of NLP tasks

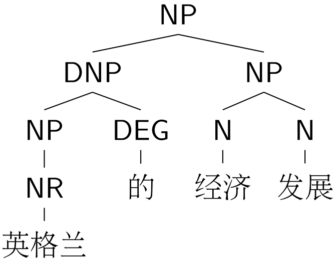
- Machine translation
- Information extraction
- Grammar checking
- etc.

Translate “英格兰的经济发展” into English



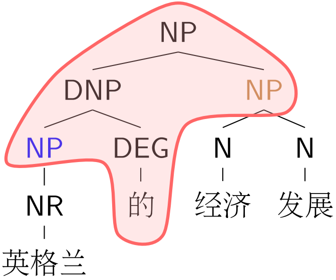
finite-state transducer?

An example: Machine translation



R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

An example: Machine translation

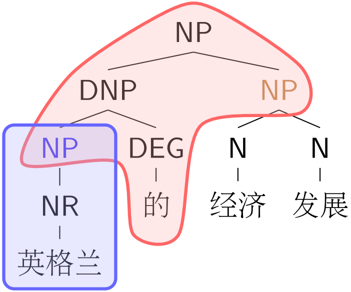


⇒NP of NP

▷R2

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

An example: Machine translation

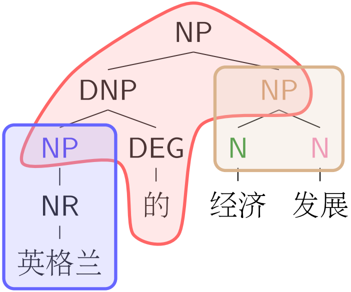


⇒NP of NP ▷R2

⇒NP of England ▷R1

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

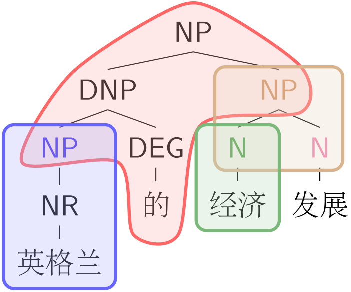
An example: Machine translation



$\Rightarrow$NP of NP ▷R2
$\Rightarrow$NP of England ▷R1
$\Rightarrow$N N of England ▷R3

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

An example: Machine translation



⇒NP of NP ▷R2

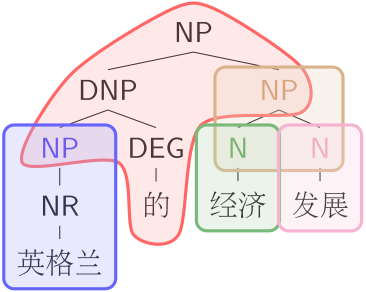
⇒NP of England ▷R1

⇒N N of England ▷R3

⇒economic N of England ▷R4

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

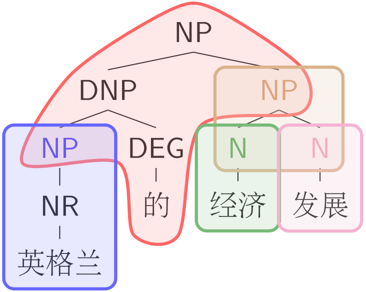
An example: Machine translation



⇒NP of NP ▷R2
⇒NP of England ▷R1
⇒N N of England ▷R3
⇒economic N of England ▷R4
⇒economic development of England ▷R5

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

An example: Machine translation



⇒NP of NP ▷R2

⇒NP of England ▷R1

⇒N N of England ▷R3

⇒economic N of England ▷R4

⇒economic development of England ▷R5

recursive form transformation

R1	NP NR 英格兰	England
R2	NP DNP NP NP DEG 的	NP of NP
R3	NP N N	N N
R4	N 经济	economic
R5	N 发展	development

Structured Prediction

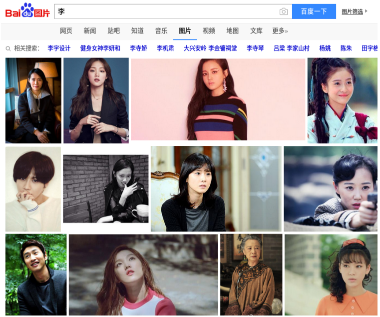
watch this video

www.youtube.com/watch?v=bjUwSHGsG9o

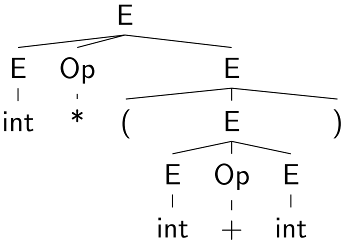
Muhammad Li

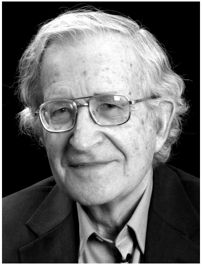
Howard Who's Muhammad Li?

Sheldon Muhammad is the most common first name in the world, Li, the most common surname. As I didn't know the answer, I thought that gave me a mathematical edge.



Two perspectives ≈ Possible vs Probable




Noam Chomsky

[...] Therefore the true logic for this world is the calculus of probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind.


James C. Maxwell

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\boldsymbol{y}^*(\boldsymbol{x}; \theta) = \operatorname*{arg\,max}_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \textsc{Score}(\boldsymbol{x}, \boldsymbol{y})$$

- Decoding: find the analysis that obtains the highest score ▷L95
- Parameter estimation: find good parameters ▷L101

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\boldsymbol{y}^*(\boldsymbol{x};\ \theta\ ) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \textsc{Score}(\boldsymbol{x}, \boldsymbol{y})$$

- Decoding: find the analysis that obtains the highest score ▷L95
- Parameter estimation: find good parameters ▷L101

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\boldsymbol{y}^*(\boldsymbol{x}; \theta) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \textsc{Score}(\boldsymbol{x}, \boldsymbol{y})$$

- Decoding: find the analysis that obtains the highest score ▷L95
- Parameter estimation: find good parameters ▷L101

Linguistic structure prediction

As a structured prediction problem

- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\boldsymbol{y}^*(\boldsymbol{x};\theta) = \arg\max_{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})} \textsc{Score}(\boldsymbol{x}, \boldsymbol{y})$$

- Decoding: find the analysis that obtains the highest score ▷L95
- Parameter estimation: find good parameters ▷L101

Linguistic structure prediction

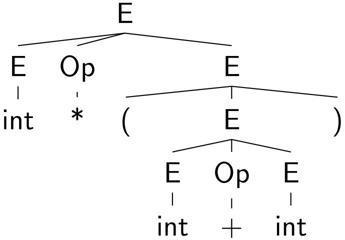
As a structured prediction problem

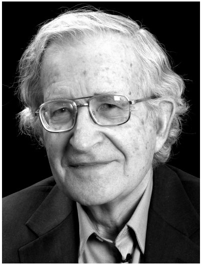
- Search space: Is this analysis possible? ▷CFG (today)
- Measurement: Is this analysis *good*? ▷PCFG (today)

$$\boldsymbol{y}^*(\boldsymbol{x}; \theta) = \underset{\boldsymbol{y} \in \mathcal{Y}(\boldsymbol{x})}{\arg\max} \ \textsc{Score}(\boldsymbol{x}, \boldsymbol{y})$$

- Decoding: find the analysis that obtains the highest score ▷L95
- Parameter estimation: find good parameters ▷L101

Context-Free Grammar





A formal language is a set of strings over an alphabet

Strings and languages

- A string of length n over an alphabet Σ is an ordered n-tuple of elements of Σ.
- Σ^* denotes the set of all strings over Σ of finite length.
- Given an alphabet Σ any subset of Σ^* is a formal language over alphabet Σ.

Goal

- Define the subset of strings $S \subseteq \Sigma^*$ in such a way that finite *rules* can generate all of it.

Formal grammars

Formally specify a grammar that can generate all and only the acceptable sentences of a natural language.

Formal grammars

Formally specify a grammar that can generate all and only the acceptable sentences of a natural language.

A grammar G consists of the following components:

1. A finite set Σ of terminal symbols.
2. A finite set N of nonterminal symbols that is disjoint from Σ.
3. A distinguished nonterminal symbol that is the START symbol.
4. A finite set R of production rules, each rule of the form

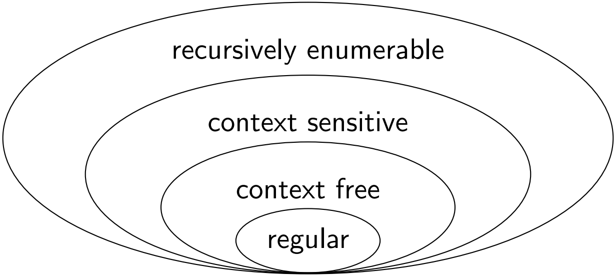
   $$(\Sigma \cup N)^{+} \rightarrow (\Sigma \cup N)^{*}$$

   Each production rule maps from one string of symbols to another.

Chomsky Hierarchy

Grammar	Languages	Production rules
Type-0	Recursively enumerable	$\gamma \rightarrow \alpha$
Type-1	Context-sensitive	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Type-2	Context-free	$A \rightarrow \alpha$
Type-3	Regular	$A \rightarrow a$
		$A \rightarrow aB$

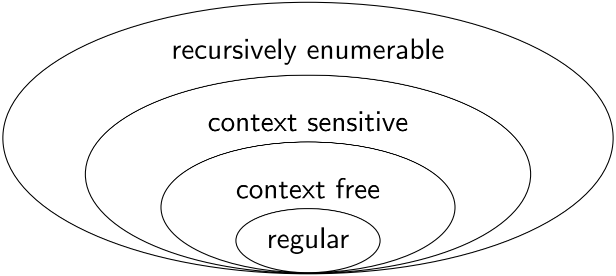
$$A, B \in N, a \in \Sigma;\ \alpha, \beta \in (N \cup \Sigma)^*, \gamma \in (N \cup \Sigma)^+$$



Chomsky Hierarchy

Grammar	Languages	Production rules
Type-0	Recursively enumerable	$\gamma \to \alpha$
Type-1	Context-sensitive	$\alpha A \beta \to \alpha \gamma \beta$
Type-2	Context-free	$A \to \alpha$
Type-3	Regular	$A \to a$
		$A \to aB$

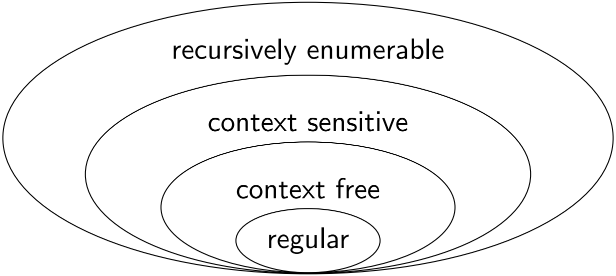
$$A, B \in N, a \in \Sigma;\ \alpha, \beta \in (N \cup \Sigma)^*, \gamma \in (N \cup \Sigma)^+$$



Chomsky Hierarchy

Grammar	Languages	Production rules
Type-0	Recursively enumerable	$\gamma \to \alpha$
Type-1	Context-sensitive	$\alpha A \beta \to \alpha \gamma \beta$
Type-2	Context-free	$A \to \alpha$
Type-3	Regular	$A \to a$
		$A \to aB$

$$A, B \in N, a \in \Sigma;\ \alpha, \beta \in (N \cup \Sigma)^*, \gamma \in (N \cup \Sigma)^+$$



Context-Free Grammars

1. N: variables
2. Σ: terminals
3. R: productions

$$A \rightarrow (N \cup \Sigma)^*$$

$$A \in N$$

4. S: START

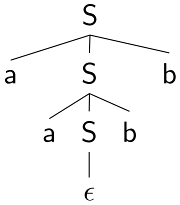
Question

What does *context-free* mean?

An example

- $L = \{a^n b^n | n \in \mathbb{N}\}$
- S$\rightarrow$aSb$|\epsilon$

$$
\begin{aligned}
& \text{S} \\
\Rightarrow\ & \text{aSb} \\
\Rightarrow\ & \text{aaSbb} \\
\Rightarrow\ & \text{aa}\epsilon\text{bb}
\end{aligned}
$$



A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP VP→VP AdvP VP→V AdvP→Adv	NP→AdjP NP NP→N AdjP→Adj
Adj→*colourless* N→*ideas* Adv→*furiously*	Adj→*green* V→*sleep*

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP VP→VP AdvP VP→V AdvP→Adv	NP→AdjP NP NP→N AdjP→Adj

Adj→*colourless* N→*ideas* Adv→*furiously*	Adj→*green* V→*sleep*

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP VP→VP AdvP VP→V AdvP→Adv	NP→AdjP NP NP→N AdjP→Adj
Adj→*colourless* N→*ideas* Adv→*furiously*	Adj→*green* V→*sleep*

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP VP→VP AdvP VP→V AdvP→Adv	NP→AdjP NP NP→N AdjP→Adj
Adj→*colourless* N→*ideas* Adv→*furiously*	Adj→*green* V→*sleep*

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP
⇒ ideas sleep AdvP

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj

Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP
⇒ ideas sleep AdvP
⇒ ideas sleep Adv

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

S→NP VP VP→VP AdvP VP→V AdvP→Adv	NP→AdjP NP NP→N AdjP→Adj
Adj→*colourless* N→*ideas* Adv→*furiously*	Adj→*green* V→*sleep*

$S = \mathsf{S}$

We can derive the structure of a string.

S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP
⇒ ideas sleep AdvP
⇒ ideas sleep Adv
⇒ ideas sleep furiously

A linguistic example (1)

$N = \{\mathsf{S}, \mathsf{NP}, \mathsf{VP}, \mathsf{AdjP}, \mathsf{AdvP}\} \cup \{\mathsf{N}, \mathsf{Adj}, \mathsf{Adv}\}$

$\Sigma = \{$*colourless*, *green*, *ideas*, *sleep*, *furiously*$\}$

R

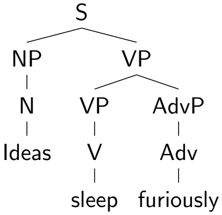
S→NP VP	NP→AdjP NP
VP→VP AdvP	
VP→V	NP→N
AdvP→Adv	AdjP→Adj
Adj→*colourless*	Adj→*green*
N→*ideas*	V→*sleep*
Adv→*furiously*	

$S = \mathsf{S}$

Generative Grammar

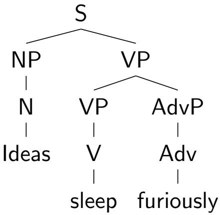
We can derive the structure of a string.

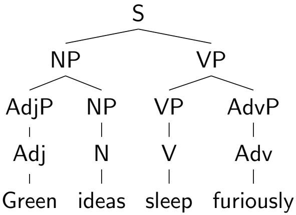
S ⇒ NP VP
⇒ N VP
⇒ ideas VP
⇒ ideas VP AdvP
⇒ ideas V AdvP
⇒ ideas sleep AdvP
⇒ ideas sleep Adv
⇒ ideas sleep furiously

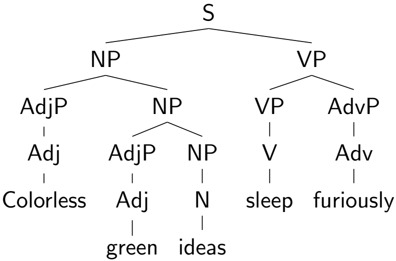


A linguistic example (2)

We can define the language of a grammar by applying the productions.







Recursion

from *Inception* (https://www.imdb.com/title/tt1375666/)

Recursion:
place one component inside another component of the same type

Recursion

> *We hypothesize that FLN (faculty of language in the narrow sense) only includes recursion and is the only uniquely human component of the faculty of language.*
>
> *M Hauser, N Chomsky and W Fitch (2002)*
>
> *science.sciencemag.org/content/298/5598/1569*

(2) a. The dog bit the cat [which chased the mouse [which died]]. (right)
    b. [[the dog] 's owner] 's friend (left)
    c. The mouse [the cat [the dog bit] chased] died. (center)

Large-coverage grammars

English Treebank

- Penn Treebank = ca. 50,000 sentences with associated trees
- Usual set-up: ca. 40,000 training sentences, ca. 2,400 test sentences

Probabilistic Context-Free Grammars

[...] Therefore the true logic for this world is the calculus of probabilities, which takes account of the magnitude of the probability which is, or ought to be, in a reasonable man's mind.

Probabilistic CFGs

Probability of a tree t with rules $A_1 \to \beta_1, A_2 \to \beta_2, \ldots$ is

$$p(t) = \prod_{i=1}^{n} q(A_i \to \beta_i)$$

where $q(A_i \to \beta_i)$ is the probability for rule $A_i \to \beta_i$ given A_i.

- When we expand A_i, how likely is it that we choose $A_i \to \beta_i$?
- For each nonterminal A_i,

$$\sum_{\beta} q(A_i \to \beta) = 1$$

- PCFG generates stochastic derivations of CFG.
- Each event (expanding nonterminal by production rules) is statistically independent of all the others.

An example (1)

S	$\rightarrow$	NP VP	0.8
S	$\rightarrow$	Aux NP VP	0.15
S	$\rightarrow$	VP	0.05
NP	$\rightarrow$	AdjP NP	0.2
NP	$\rightarrow$	D N	0.7
NP	$\rightarrow$	N	0.1
VP	$\rightarrow$	VP AdvP	0.3
VP	$\rightarrow$	V	0.2
VP	$\rightarrow$	V NP	0.3
VP	$\rightarrow$	V NP NP	0.2
AdvP	$\rightarrow$	Adv	1.0
AdjP	$\rightarrow$	Adj	1.0

Adj	$\rightarrow$	*colorless*	0.4
Adj	$\rightarrow$	*green*	0.6
N	$\rightarrow$	*ideas*	1.0
V	$\rightarrow$	*sleep*	1.0
Adv	$\rightarrow$	*furiously*	1.0

An example (2)

S

S$\rightarrow$NP VP 0.8

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1
⇒	N VP	N→*ideas*	1.0
⇒	ideas VP	VP→VP AdvP	0.3
⇒	ideas VP AdvP	VP→V	0.2
⇒	ideas V AdvP	V→*sleep*	1.0
⇒	ideas sleep AdvP	AdvP→Adv	1.0
⇒	ideas sleep Adv	Adv→*furiously*	1.0
⇒	ideas sleep furiously		

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1
⇒	N VP	N→*ideas*	1.0
⇒	ideas VP	VP→VP AdvP	0.3
⇒	ideas VP AdvP	VP→V	0.2
⇒	ideas V AdvP	V→*sleep*	1.0
⇒	ideas sleep AdvP	AdvP→Adv	1.0
⇒	ideas sleep Adv	Adv→*furiously*	1.0
⇒	ideas sleep furiously		

$$0.8 \times 0.1 \times 1.0 \times 0.3 \times 0.2 \times 1.0 \times 1.0 \times 1.0$$

An example (2)

	S	S→NP VP	0.8
⇒	NP VP	NP→N	0.1
⇒	N VP	N→*ideas*	1.0
⇒	ideas VP	VP→VP AdvP	0.3
⇒	ideas VP AdvP	VP→V	0.2
⇒	ideas V AdvP	V→*sleep*	1.0
⇒	ideas sleep AdvP	AdvP→Adv	1.0
⇒	ideas sleep Adv	Adv→*furiously*	1.0
⇒	ideas sleep furiously		

$$0.8 \times 0.1 \times 1.0 \times 0.3 \times 0.2 \times 1.0 \times 1.0 \times 1.0$$

Generative model

Properties of PCFGs

- A probability is assigned to each parse-tree licensed by the underlying CFG
- Say we have a sentence s, set of derivations for that sentence is $\mathcal{T}(s)$, as defined by a CFG. Then a PCFG assigns a probability $p(t)$ to each member of $\mathcal{T}(s)$.

- We now have a $\textsc{Score}$ function (probability) that can rank trees.
- The most likely parse tree for a sentence s is

$$\boxed{\arg\max_{t \in \mathcal{T}(s)} p(t)}$$

"correct" means more probable parse tree

"grammaticality" is the inclusion criterion into the language, but is not a binary property

Deriving a PCFG from a Treebank

Given a set of example trees (a treebank), the underlying CFG can simply be all rules seen in the corpus

Maximum Likelihood Estimates

$$q_{ML}(\alpha \rightarrow \beta) = \frac{\textsc{count}(\alpha \rightarrow \beta)}{\textsc{count}(\alpha)}$$

The counts are taken from a training set of example trees.

If the training data is generated by a PCFG, then as the training data size goes to infinity, the maximum-likelihood PCFG will converge to the same distribution as the "true" PCFG.

Challenge

Cross-serial dependencies in Swiss German

... das mer em Hans es huus hälfed aastriiche
... that we Hans$_{Dat}$ house$_{Acc}$ help paint
... that we helped Hans paint the house

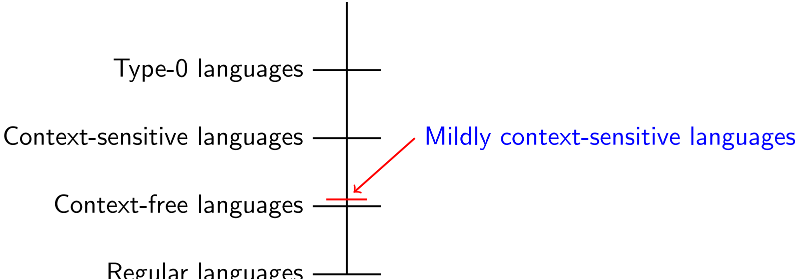
... das mer d'chind em Hans es huus lönd hälfe aastriiche
... that we the children$_{Acc}$ Hans$_{Dat}$ house$_{Acc}$ let help paint
... that we let the children help Hans paint the house

Cross-serial dependencies in Dutch

... dat Wim Jan Marie de kinderen zag helpen leren zwemmen
... that Wim Jan Marie the children saw help teach swim
... that Wim saw Jan help Marie teach the children to swim

Mildly Context-Sensitive Languages

Natural languages are provably non-context-free.
Natural languages = mildly context-sensitive languages?



Reading

- Ann's lecture notes.
  `https://www.cl.cam.ac.uk/teaching/1920/NLP/materials.html`
- D Jurafsky and J Martin. *Speech and Language Processing*.
  - Chapter 12. Constituency Grammars.
    `https://web.stanford.edu/~jurafsky/slp3/12.pdf`
  - Chapter 14. Statistical Constituency Parsing.
    `https://web.stanford.edu/~jurafsky/slp3/14.pdf`