NLP

Introduction to NLP

Statistical Parsing

Need For Probabilistic Parsing

- Time flies like an arrow
  - Many parses
  - Some (clearly) more likely than others
  - Need for a probabilistic ranking method

Probabilistic Context-free Grammars

- Just like (deterministic) CFG, a 4-tuple (N, Σ, R, S)
  - N: non-terminal symbols
  - Σ: terminal symbols (disjoint from N)
  - R: rules $(A \rightarrow \beta)$ [p]
    - β is a string from $(\Sigma \cup N)^*$
    - p is the probability $P(\beta|A)$
  - S: start symbol (from N)

Example

```
S -> NP VP
NP -> DT N | NP PP
PP -> PRP NP
VP -> V NP | VP PP
DT -> 'a' | 'the'
N -> 'child' | 'cake' | 'fork'
PRP -> 'with' | 'to'
V -> 'saw' | 'ate'
```

Example

```
S -> NP VP
NP -> DT N
NP -> NP PP
PP -> PRP NP
VP -> V NP
VP -> VP PP
DT -> 'a'
DT -> 'the'
N -> 'child'
N -> 'cake'
N -> 'fork'
PRP -> 'with'
PRP -> 'to'
V -> 'saw'
V -> 'ate'
```

Example

```
S -> NP VP
NP -> DT N
NP -> NP PP
PP -> PRP NP
VP -> V NP
VP -> VP PP
DT -> 'a'
DT -> 'the'
N -> 'child'
N -> 'cake'
N -> 'fork'
PRP -> 'with'
PRP -> 'to'
V -> 'saw'
V -> 'ate'
```

Example

```
S -> NP VP        [p0=1]
NP -> DT N        [p1]
NP -> NP PP       [p2]
PP -> PRP NP      [p3=1]
VP -> V NP        [p4]
VP -> VP PP       [p5]
DT -> 'a'         [p6]
DT -> 'the'       [p7]
N -> 'child'      [p8]
N -> 'cake'       [p9]
N -> 'fork'       [p10]
PRP -> 'with'     [p11]
PRP -> 'to'       [p12]
V -> 'saw'        [p13]
V -> 'ate'        [p14]
```

Probability Of A Parse Tree

- The probability of a parse tree *t* given all *n* productions used to build it:

$$p(t) = \prod_{i=1}^{n} p(\alpha_i \rightarrow \beta_i)$$

- The most likely parse is determined as follows:

$$\underset{t \in T(s)}{\arg\max}\ p(t)$$

- The probability of a sentence is the sum of the probabilities of all of its parses

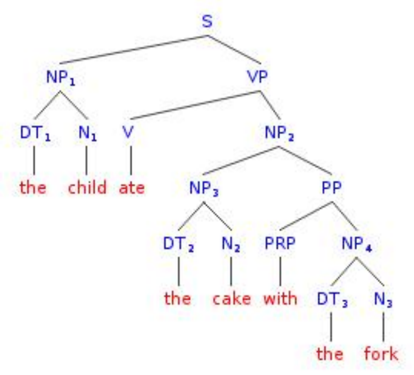
Example

```
S -> NP VP          [p0=1]
NP -> DT N          [p1]
NP -> NP PP         [p2]
PP -> PRP NP        [p3=1]
VP -> V NP          [p4]
VP -> VP PP         [p5]
DT -> 'a'           [p6]
DT -> 'the'         [p7]
N -> 'child'        [p8]
N -> 'cake'         [p9]
N -> 'fork'         [p10]
PRP -> 'with'       [p11]
PRP -> 'to'         [p12]
V -> 'saw'          [p13]
V -> 'ate'          [p14]
```

Example

```
S -> NP VP          [p0=1]
NP -> DT N          [p1]
NP -> NP PP         [p2]
PP -> PRP NP        [p3=1]
VP -> V NP          [p4]
VP -> VP PP         [p5]
DT -> 'a'           [p6]
DT -> 'the'         [p7]
N -> 'child'        [p8]
N -> 'cake'         [p9]
N -> 'fork'         [p10]
PRP -> 'with'       [p11]
PRP -> 'to'         [p12]
V -> 'saw'          [p13]
V -> 'ate'          [p14]
```

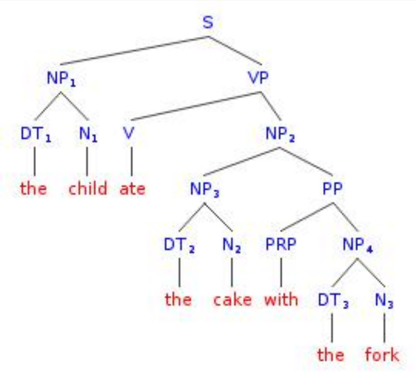
t_1



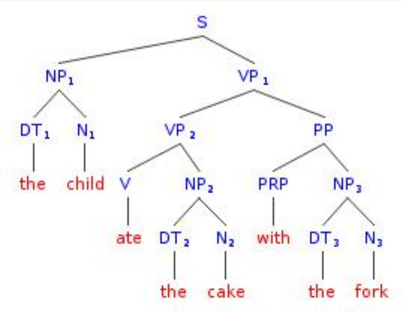
Example

```
S -> NP VP          [p0=1]
NP -> DT N          [p1]
NP -> NP PP         [p2]
PP -> PRP NP        [p3=1]
VP -> V NP          [p4]
VP -> VP PP         [p5]
DT -> 'a'           [p6]
DT -> 'the'         [p7]
N -> 'child'        [p8]
N -> 'cake'         [p9]
N -> 'fork'         [p10]
PRP -> 'with'       [p11]
PRP -> 'to'         [p12]
V -> 'saw'          [p13]
V -> 'ate'          [p14]
```

t_1



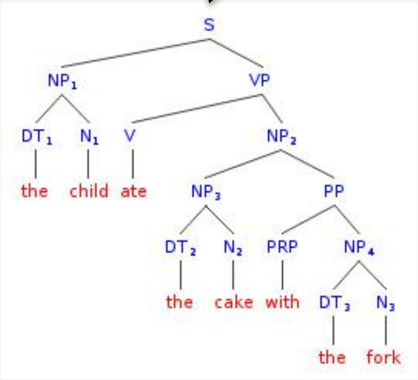
t_2



Example

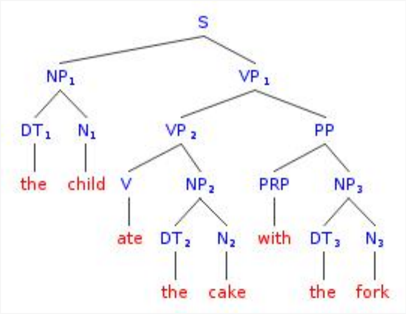
```
S -> NP VP          [p0=1]
NP -> DT N          [p1]
NP -> NP PP         [p2]
PP -> PRP NP        [p3=1]
VP -> V NP          [p4]
VP -> VP PP         [p5]
DT -> 'a'           [p6]
DT -> 'the'         [p7]
N -> 'child'        [p8]
N -> 'cake'         [p9]
N -> 'fork'         [p10]
PRP -> 'with'       [p11]
PRP -> 'to'         [p12]
V -> 'saw'          [p13]
V -> 'ate'          [p14]
```

t_1



$$p(t_1) = p_0 p_1 p_4 p_7 p_8 p_{14} p_2 p_1 p_3 p_7 p_9 p_{11} p_1 p_7 p_{10}$$

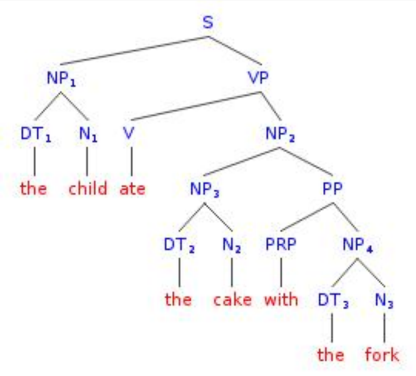
t_2



Example

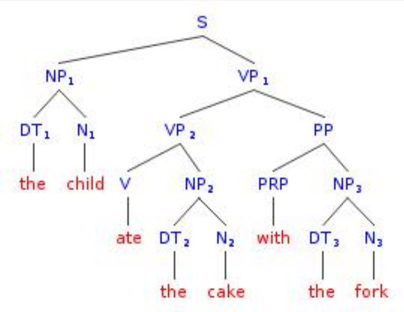
```
S -> NP VP        [p0=1]
NP -> DT N        [p1]
NP -> NP PP       [p2]
PP -> PRP NP      [p3=1]
VP -> V NP        [p4]
VP -> VP PP       [p5]
DT -> 'a'         [p6]
DT -> 'the'       [p7]
N -> 'child'      [p8]
N -> 'cake'       [p9]
N -> 'fork'       [p10]
PRP -> 'with'     [p11]
PRP -> 'to'       [p12]
V -> 'saw'        [p13]
V -> 'ate'        [p14]
```

t_1



$$p(t_1) = p_0 p_1 p_4 p_7 p_8 p_{14} p_2 p_1 p_3 p_7 p_9 p_{11} p_1 p_7 p_{10}$$

t_2

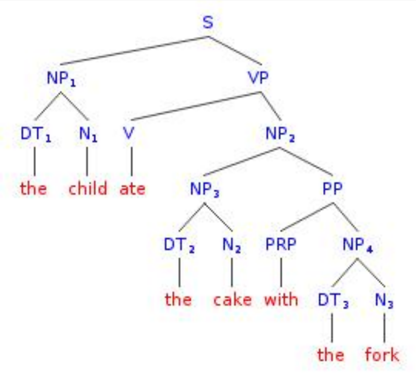


$$p(t_2) = p_0 p_1 p_5 p_7 p_8 p_4 p_3 p_{14} p_1 p_{11} p_1 p_7 p_9 p_7 p_{10}$$

Example

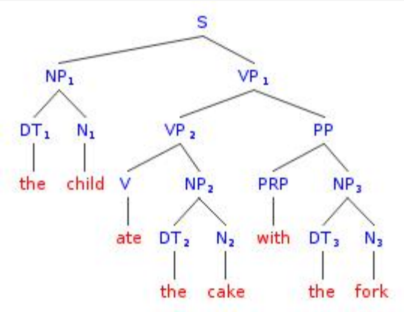
```
S -> NP VP         [p0=1]
NP -> DT N         [p1]
NP -> NP PP        [p2]
PP -> PRP NP       [p3=1]
VP -> V NP         [p4]
VP -> VP PP        [p5]
DT -> 'a'          [p6]
DT -> 'the'        [p7]
N -> 'child'       [p8]
N -> 'cake'        [p9]
N -> 'fork'        [p10]
PRP -> 'with'      [p11]
PRP -> 'to'        [p12]
V -> 'saw'         [p13]
V -> 'ate'         [p14]
```

t_1



$$p(t_1) = p_0 p_1 p_4 p_7 p_8 p_{14} p_2 p_1 p_3 p_7 p_9 p_{11} p_1 p_7 p_{10}$$

t_2

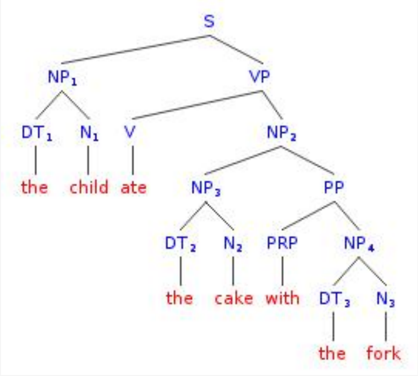


$$p(t_2) = p_0 p_1 p_5 p_7 p_8 p_4 p_3 p_{14} p_1 p_{11} p_1 p_7 p_9 p_7 p_{10}$$

Example

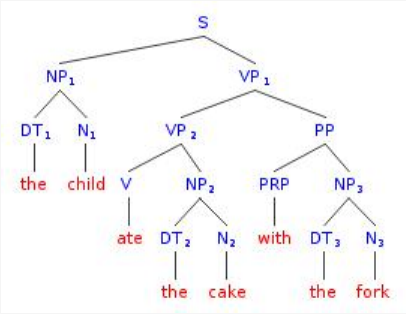
```
S -> NP VP          [p0=1]
NP -> DT N          [p1]
NP -> NP PP         [p2]
PP -> PRP NP        [p3=1]
VP -> V NP          [p4]
VP -> VP PP         [p5]
DT -> 'a'           [p6]
DT -> 'the'         [p7]
N -> 'child'        [p8]
N -> 'cake'         [p9]
N -> 'fork'         [p10]
PRP -> 'with'       [p11]
PRP -> 'to'         [p12]
V -> 'saw'          [p13]
V -> 'ate'          [p14]
```

t_1



$p(t_1) = p_0 p_1 p_4 p_7 p_8 p_{14} p_2 p_1 p_3 p_7 p_9 p_{11} p_1 p_7 p_{10}$

t_2



$p(t_2) = p_0 p_1 p_5 p_7 p_8 p_4 p_3 p_{14} p_1 p_{11} p_1 p_7 p_9 p_7 p_{10}$

Main Tasks With Pcfgs

- Given a grammar G and a sentence s, let T(s) be all parse trees that correspond to s
- Task 1
  - find which tree t among T(s) maximizes the probability p(t)
- Task 2
  - find the probability of the sentence p(s) as the sum of all possible tree probabilities p(t)

Probabilistic Parsing Methods

- Probabilistic Earley algorithm
  - Top-down parser with a dynamic programming table
- Probabilistic Cocke-Kasami-Younger (CKY) algorithm
  - Bottom-up parser with a dynamic programming table

Probabilistic Grammars

- Probabilities can be learned from a training corpus (Treebank)
- Intuitive meaning
  - Parse #1 is twice as probable as parse #2
- Possible to do reranking
- Possible to combine with other stages
  - E.g., speech recognition, translation

Maximum Likelihood Estimates

- Use the parsed training set for getting the counts
  - $P_{ML}(\alpha \rightarrow \beta) = Count(\alpha \rightarrow \beta) / Count(\alpha)$

- Example:
  - $P_{ML}(S \rightarrow NP\ VP) = Count(S \rightarrow NP\ VP) / Count(S)$

Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.15] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flights$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.30]	$Verb \rightarrow book$ [.30] | $include$ [.30]
$NP \rightarrow Det\ Nominal$	[.20]	| $prefer$; [.40]
$NP \rightarrow Nominal$	[.15]	$Pronoun \rightarrow I$ [.40] | she [.05]
$Nominal \rightarrow Noun$	[.75]	| me [.15] | you [.40]
$Nominal \rightarrow Nominal\ Noun$	[.20]	$Proper\text{-}Noun \rightarrow Houston$ [.60]
$Nominal \rightarrow Nominal\ PP$	[.05]	| NWA [.40]
$VP \rightarrow Verb$	[.35]	$Aux \rightarrow does$ [.60] | can [40]
$VP \rightarrow Verb\ NP$	[.20]	$Preposition \rightarrow from$ [.30] | to [.30]
$VP \rightarrow Verb\ NP\ PP$	[.10]	| on [.20] | $near$ [.15]
$VP \rightarrow Verb\ PP$	[.15]	| $through$ [.05]
$VP \rightarrow Verb\ NP\ NP$	[.05]	
$VP \rightarrow VP\ PP$	[.15]	
$PP \rightarrow Preposition\ NP$	[1.0]	

Example from Jurafsky and Martin

Example

```
S -> NP VP        [p0=1]
NP -> DT N        [p1=.8]
NP -> NP PP       [p2=.2]
PP -> PRP NP      [p3=1]
VP -> V NP        [p4=.7]
VP -> VP PP       [p5=.3]
DT -> 'a'         [p6=.25]
DT -> 'the'       [p7=.75]
N -> 'child'      [p8=.5]
N -> 'cake'       [p9=.3]
N -> 'fork'       [p10=.2]
PRP -> 'with'     [p11=.1]
PRP -> 'to'       [p12=.9]
V -> 'saw'        [p13=.4]
V -> 'ate'        [p14=.6]
```

the								
	child							
		ate						
			the					
				cake				
					with			
						the		
							fork	

the	DT .75							
child								
ate								
the								
cake								
with								
the								
fork								

the	DT .75							
child		N .5						
ate								
the								
cake								
with								
the								
fork								

the	DT .75	NP .8						
	child	N .5						
		ate						
			the					
				cake				
					with			
						the		
							fork	

the	DT .75	NP .8*.5*.75						
child		N .5						
ate								
the								
cake								
with								
the								
fork								

Question

- How, on your own, could you compute the probability of the entire sentence using Probabilistic CKY?
- Don’t forget that there may be multiple parses, so you will need to add the corresponding probabilities.

NLP