Introduction to NLP

Context-free grammars

Context-free grammars

- A context-free grammar is a 4-tuple (N, Σ, R, S)
  - N: non-terminal symbols
  - Σ: terminal symbols (disjoint from N)
  - R: rules ($A \rightarrow \beta$), where β is a string from $(\Sigma \cup N)^*$
  - S: start symbol from N

Example

```
["the", "child", "ate", "the", "cake", "with", "the", "fork"]
```

```
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

```
["the", "child", "ate", "the", "cake", "with", "the", "fork"]
```

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'

Heads marked in bold face

Phrase-structure grammars (1/2)

- Sentences are not just bags of words
  - Alice bought Bob flowers
  - Bob bought Alice flowers
- Context-free view of language
  - A prepositional phrase looks the same whether it is part of the subject NP or part of the VP
- Constituent order
  - SVO (subject verb object)
  - SOV (subject object verb)

Phrase-structure grammars (2/2)

- Auxiliary verbs
  - The dog may have eaten my homework
- Imperative sentences
  - Leave the book on the table
- Interrogative sentences
  - Did the customer have a complaint?
  - Who had a complaint?
- Negative sentences
  - The customer didn’t have a complaint

A longer example

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```

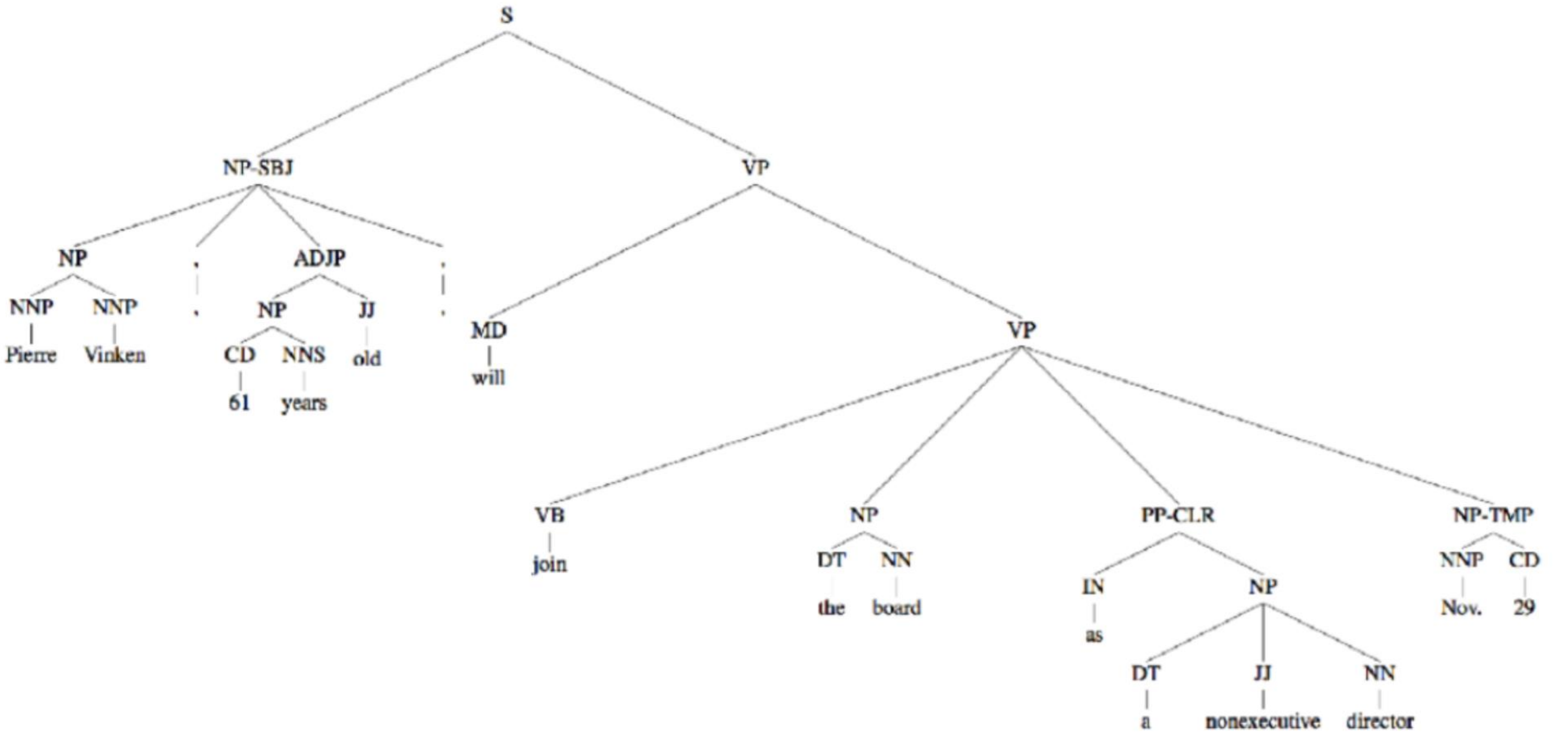
What changes were made to the grammar?

A longer example

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```

A longer example

```
S -> NP VP | Aux NP VP | VP
NP -> PRON | Det Nom
Nom -> N | Nom N | Nom PP
PP -> PRP NP
VP -> V | V NP | VP PP
Det -> 'the' | 'a' | 'this'
PRON -> 'he' | 'she'
N -> 'book' | 'boys' | 'girl'
PRP -> 'with' | 'in'
V -> 'takes' | 'take'
```


S
NP-SBJ
VP
NP
,
ADJP
,
NNP
NNP
NP
JJ
MD
VP
Pierre
Vinken
CD
NNS
old
will
61
years
VB
NP
PP-CLR
NP-TMP
join
DT
NN
IN
NP
NNP
CD
the
board
as
Nov.
29
DT
JJ
NN
a
nonexecutive
director

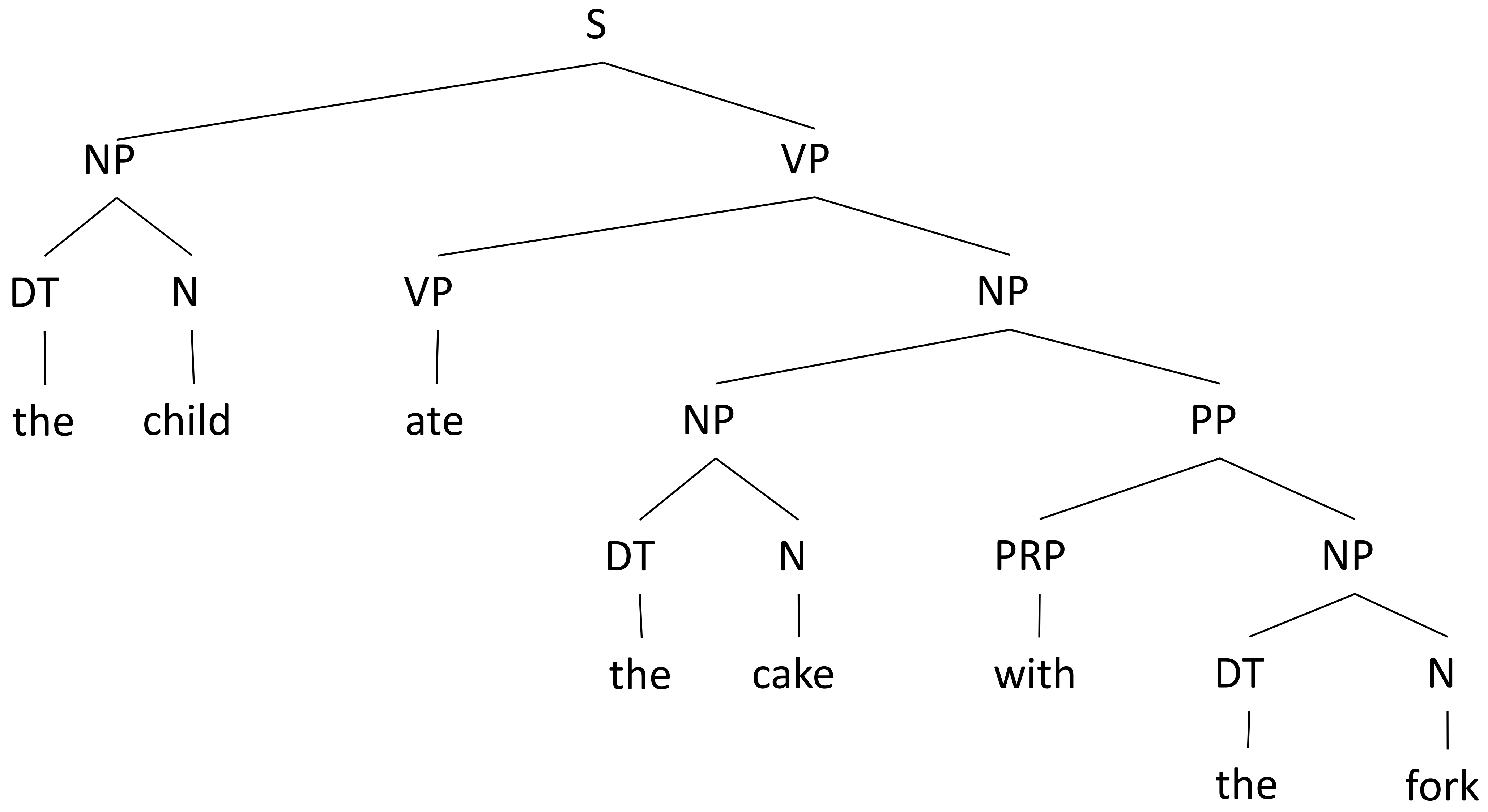
Penn Treebank Example

```
( (S
    (NP-SBJ
      (NP (NNP Pierre) (NNP Vinken) )
      (, ,)
      (ADJP
        (NP (CD 61) (NNS years) )
        (JJ old) )
      (, ,) )
    (VP (MD will)
      (VP (VB join)
        (NP (DT the) (NN board) )
        (PP-CLR (IN as)
          (NP (DT a) (JJ nonexecutive) (NN director) ))
        (NP-TMP (NNP Nov.) (CD 29) )))
    (. .) ))
( (S
    (NP-SBJ (NNP Mr.) (NNP Vinken) )
    (VP (VBZ is)
      (NP-PRD
        (NP (NN chairman) )
        (PP (IN of)
          (NP
            (NP (NNP Elsevier) (NNP N.V.) )
            (, ,)
            (NP (DT the) (NNP Dutch) (VBG publishing) (NN group) )))))
    (. .) ))
```

CFGs are equivalent to PDAs

- PDA = Pushdown Automata
- Example: consider the language L={$x^n y^n$}
  - stack is empty,   input=xxxyyy
  - push * onto stack, input=xxyyy
  - push * onto stack, input=xyyy
  - push * onto stack, input=yyy
  - pop * from stack,   input=yy
  - pop * from stack,   input=y
  - pop * from stack,   input=“”

Leftmost derivation



Introduction to NLP

Issues with Context-free Grammars

Agreement

- Number
  - Chen is/people are
- Person
  - I am/Chen is
- Tense
  - Chen was reading/Chen is reading/Chen will be reading
- Case
  - not in English but in many other languages such as German, Russian, Greek
- Gender
  - not in English but in many other languages such as German, French, Spanish

Combinatorial explosion

- Many combinations of rules are needed to express agreement
  - S $\rightarrow$ NP VP
  - S $\rightarrow$ 1sgNP 1sgVP
  - S $\rightarrow$ 2sgNP 2sgVP
  - S $\rightarrow$ 3sgNP 3sgVP
  - …
  - 1sgNP $\rightarrow$ 1sgN
  - ...

Subcategorization frames

- Direct object
  - The dog ate a sausage
- Prepositional phrase
  - Mary left the car in the garage
- Predicative adjective
  - The receptionist looked worried
- Bare infinitive
  - She helped me buy this place
- To-infinitive
  - The girl wanted to be alone
- Participial phrase
  - He stayed crying after the movie ended
- That-clause
  - Ravi doesn’t believe that it will rain tomorrow
- Question-form clauses
  - She wondered where to go
- Empty (ϕ)
  - She slept

CFG independence assumptions

- Non-independence
  - All NPs
    - 11% NP PP, 9% DT NN, 6% PRP
  - NPs under S
    - 9% NP PP, 9% DT NN, 21% PRP
  - NPs under VP
    - 23% NP PP, 7% DT NN, 4% PRP
  - example from Dan Klein

NLP

Introduction to NLP

241.

Syntax

Syntax

- Is language more than just a “bag of words”?
  - Grammatical rules apply to categories and groups of words, not individual words.
- Example
  - a sentence includes a subject and a predicate. The subject is a noun phrase and the predicate is a verb phrase.
  - Noun phrases:
    - The cat, Samantha, She
  - Verb phrase:
    - arrived, went away, had dinner
- When people learn a new word, they learn its syntactic usage.
  - Examples: wug (n), cluvious (adj) – use them in sentences
  - Hard to come up with made up words: forkle, vleer, etc. all taken.


TABLES ARE
FOR EATING
CUSTOMERS
ONLY
NO LOITERING

Defining Parts of Speech

- What do nouns typically have in common?
  - E.g., *can* be preceded by “the”.
- What about verbs?
  - Verbs can be preceded by “can’t”.
- Adjectives can come between “the” and a noun.
  - How is this different from grade school definitions?
- Determiners
  - a, the, many, no, five
- Prepositions
  - for, to, in, without, before

Constituents

- Constituents are continuous
- Constituents are non-crossing
  - if two constituents share one word, then one of them must completely contain the other.
- Each word is a constituent

Constituent Tests

- “coordination” test
  - She bought a bagel and three chocolate croissants
- “pronoun” test
  - A small dog is barking in the park.
  - It is barking in the park
- “question by repetition” test:
  - I have seen blue elephants
  - Blue elephants?
  - * Seen blue?
  - Seen blue elephants?
- “topicalization” test:
  - Blue elephants, I have seen.
- “question” test:
  - *What* have I seen?
- “deletion” test
  - Last year I saw <u>a blue elephant in the zoo</u>.
- “semantic” test
- “intuitition” test

How to generate sentences

- One way: tree structure
  - Generate the tree structure first
  - Then fill the leaf nodes with terminals

A Simple Syntactic Rule

- The simplest rule for a sentence, e.g. “Birds fly”

  $S \rightarrow N\ V$

Simplest Grammar

S → N V

`N → Samantha | Min | Jorge`

`V → left | sang | walked`

Sample sentences:

Samantha sang

Jorge left

Syntax

- The verbs so far were intransitive (no direct object)
- What rules are needed next?
  - Transitive verbs and direct objects ("Jorge saw Samantha")
  - Determiners ("the cats")
- Combinatorial explosion (even for the simplest form of sentences)
  - Need for noun phrases
  - Ditto for verb phrases


DUDE, WHY DO YOU TALK BACKWARDS ALL THE TIME?
BACKWARDS-TALKING I AM NOT. ENGLISH-CENTRIC VIEW OF GRAMMATICAL STRUCTURE, YOU HAVE
©2012 BY DOUG SAVAGE

Latest Grammar

```
S → NP VP
NP → DT N
VP → V NP
DT → the | a
N → child | cat | dog
V → took | saw | liked | scared | chased
```

Sample sentences:

a dog chased the cat

the child saw a dog

Alternatives

- Different expansions of a category are delineated with " | "
  - `NP → PN | DT CN`
- One rule for proper nouns and another for common nouns

Latest Grammar

S → NP VP

NP → DT CN

NP → PN

VP → V NP

DT → the | a

CN → child | cat | dog

PN → Samantha | Jorge | Min

V → took | saw | liked | scared | chased

Sample sentences:

a child scared Jorge

Min took the child

Optional categories

- Wherever `N` is allowed in a sentence,
  - `DT N`
  - `JJ N`
  - `DT JJ N`

  are also allowed
- We can use the notation for alternatives
  - `NP → N | DT N | JJ N | DT JJ N`
- Optional categories can be also marked using parentheses:
  - `NP → (DT) (JJ) N`

Verb Phrases

- Samantha ran.
- Samantha ran to the park.
- Samantha ran away.
- Samantha bought a cookie.
- Samantha bought a cookie for John.
- Overall structure
  - $\mathrm{VP} \rightarrow \mathrm{V}(\mathrm{NP})(\mathrm{P})(\mathrm{NP})$

Latest Grammar

```
S  →  NP VP
NP →  DT CN
NP →  PN
VP →  V (NP) (P) (NP)
DT →  the | a
CN →  child | cat | dog
PN →  Samantha | Jorge | Min
P  →  to | for | from | in
V  →  took | saw | liked | scared | chased | gave
```

Sample sentences:

Samantha saw the cat

Jorge gave the cat to Min

Prepositional Phrases

- Examples:
  - Mary bought a book for John **in a bookstore**.
  - The bookstore sells magazines.
  - The bookstore **on Main St**. sells magazines.
  - Mary ran away.
  - Mary ran **down the hill**.
- Changes are needed to both NP and VP to accommodate prepositional phrases
  - Wherever a preposition is allowed, it can be followed by a noun phrase.
  - Run up
  - NP can contain any number of PPs but only up to two NPs.
- How do we revise the grammar accordingly?

The Rules So Far

- **`S`** $\rightarrow$ **`NP VP`**
- **`NP`** $\rightarrow$ **`(DT) (JJ) N (PP)`**
- **`VP`** $\rightarrow$ **`V (NP) (PP)`**
- **`PP`** $\rightarrow$ **`P (NP)`**

PP Ambiguity

- The boy saw the woman with the telescope.

```
PP → PREP NP
VP → V NP PP
VP → V NP
NP → DT N
NP → DT N PP
```

Repetition (*)

- (JJ*) = a sequence of zero or more JJ
- Are all sequences of adjectives allowed?
  - a big red house
  - * a red big house
- Adjective ordering in English depends on semantics!

Exercise

- The Little Red Riding Hood
- Three Little Pigs
- The Three Musketeers
- The Steadfast Tin Soldier
- The French Connection
- Old Macdonald
- Five Golden Rings
- The Ancient Mariner

Adjective ordering

- **Det**
- Number
- Strength
- Size
- Age
- Shape
- Color
- Origin
- Material
- Purpose
- **Noun**

- det < number < size < color < purpose < noun
- strength < material < noun
- origin < noun

Nested Sentences

- Examples:
  - I don’t recall whether I took the dog out.
  - Do you know if the mall is still open?
- **`VP → V (NP) (NP) (C S) (PP*)`**
- Can (C S) appear inside an NP?
  - Whether he will win the elections remains to be seen.

Recursion

- S can generate VP, VP can generate S
- NP can generate PP, PP can generate NP
- What does recursion allow?
- Is there a longest sentence in English?
- Conjunction of NPs:

  `NP → NP and NP`

- Conjunction of PPs:

  `PP → PP and PP`

- Conjunction of VPs:

  `VP → VP and VP`

Meta-patterns

- $S \to NP\ VP$
  - $NP \to (DT)\ (JJ)\ N\ (PP)$
  - $VP \to V\ (NP)\ (PP)$
  - $PP \to P\ (NP)$
- Is there a meta-pattern here?
  - $XP \to$ (specifier) X’
  - $X' \to X$ (complement)
- Example: $NP \to\ \ DT\ N'$
- X-bar Theory
  - http://www.unlweb.net/wiki/X-bar_theory

Meta-rules for Conjunctions

- Conjunction
  - `X → X and X`
- This kind of rule even covers entire sentences
  - `S → S and S`

Auxiliaries

- Is “Aux V” a constituent?
  - I have seen blue elephants and will remember them forever.
- Recursion:
  - VP -> Aux VP
  - Raj may have been sleeping.
- Is such recursion unlimited?

Exercise

- Grammar:
  - `S → NP VP | CP VP`
  - `NP → (DT) (JJ*) N (CP) (PP*)`
  - `VP → V (NP) (NP) (PP*) | V (NP) (CP) (PP*)`
  - `PP → P NP`
  - `CP → C S`
- What rules are needed to generate these three sentences:
  - 1. The small dog of the neighbors brought me an old tennis ball.
  - 2. That wugs have three eyes is unproven by scientists.
  - 3. I saw the gift that the old man gave me at the meeting.

Notes

- Syntax helps with sentences like

  * The milk drank the cat

  The milk is drunk by the cat
- Overgeneration

  The girl saw
- Undergeneration
- Grammar – between the two

Arguments vs. Adjuncts

- Arguments
    - Mandatory (e.g., “* Romeo likes”, “*likes Juliet”)
    - Cannot be repeated (e.g., “* Juliet likes Romeo John”)
    - Verbs can have more than one subcategorization frame
- Adjuncts
    - Optional
    - Typically prepositional phrases or adverbs
    - Can be repeated (e.g., “Apparently Candace ate pizza yesterday at the restaurant with pleasure”)

Introduction to NLP

251.

Probabilistic Grammars

Need for Probabilistic Parsing

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

Probabilistic CFG

- 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:

$$\operatorname*{argmax}_{t \in T(s)} p(t)$$

- The probabilities are obtained using MLE from the training corpus
- The probability of a *sentence* is the *sum* of the probabilities of all of its parses

A		β	P(β | NP)
NP	→	NP PP	0.092
NP	→	DT NN	0.087
NP	→	NN	0.047
NP	→	NNS	0.042
NP	→	DT JJ NN	0.035
NP	→	NNP	0.034
NP	→	NNP NNP	0.029
NP	→	JJ NNS	0.027
NP	→	QP -NONE-	0.018
NP	→	NP SBAR	0.017
NP	→	NP PP-LOC	0.017
NP	→	JJ NN	0.015
NP	→	DT NNS	0.014
NP	→	CD	0.014
NP	→	NN NNS	0.013
NP	→	DT NN NN	0.013
NP	→	NP CC NP	0.013

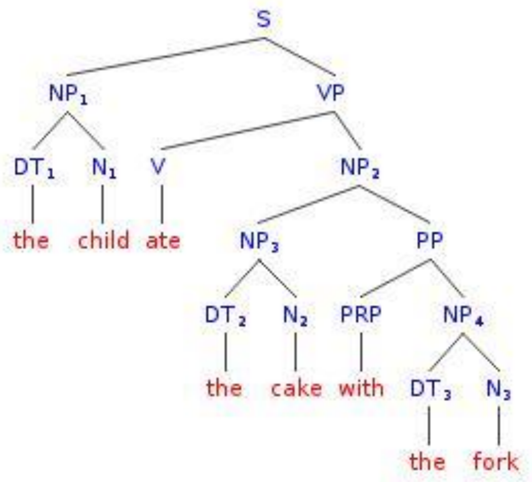
Example

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]
```

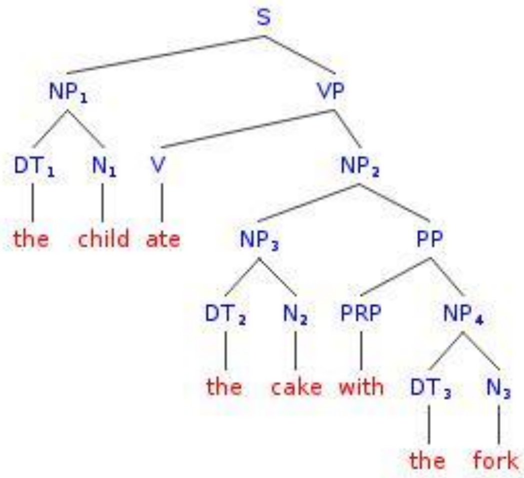
t_1



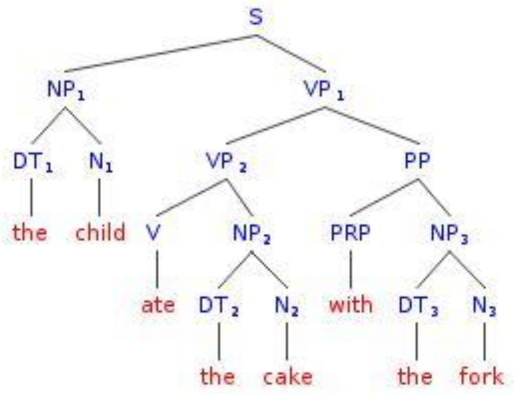
Example

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



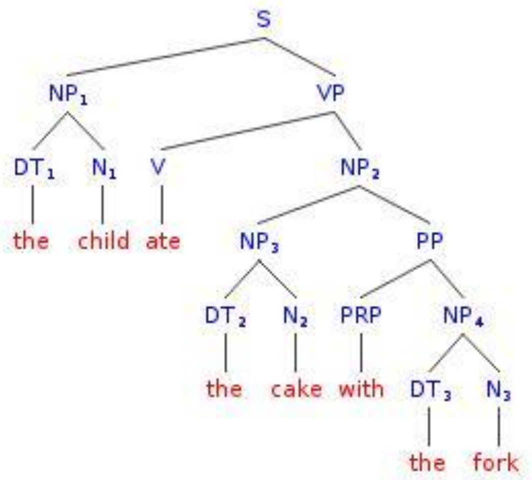
t_2



Example

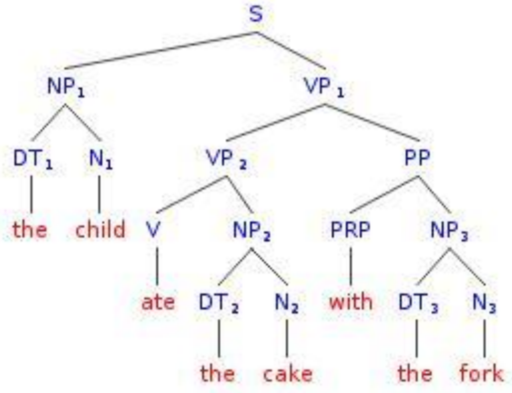
S -> NP VP [p0=1]
NP -> DT N [p1]
NP -> NP PP [p2]
PP -> PRP NP [p3=1]
VP -> V NP [p4]
VP -> VP PP [p5]
DT -> 'a' [p6]
DT -> 'the' [p7]
N -> 'child' [p8]
N -> 'cake' [p9]
N -> 'fork' [p10]
PRP -> 'with' [p11]
PRP -> 'to' [p12]
V -> 'saw' [p13]
V -> 'ate' [p14]

t_1



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

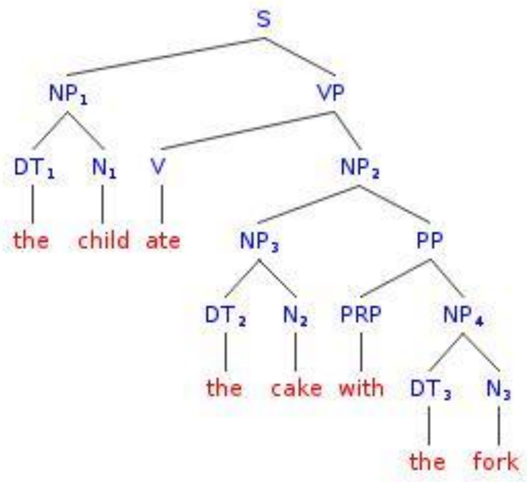
t_2



Example

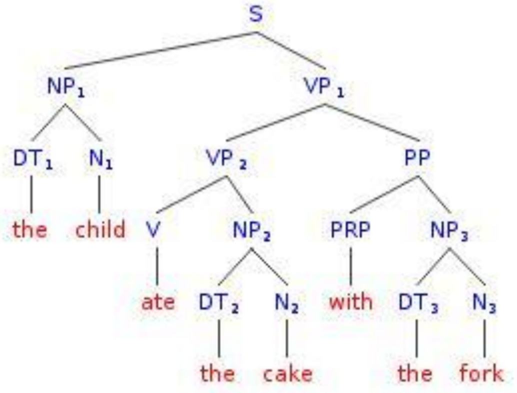
S -> NP VP [p0=1]
NP -> DT N [p1]
NP -> NP PP [p2]
PP -> PRP NP [p3=1]
VP -> V NP [p4]
VP -> VP PP [p5]
DT -> 'a' [p6]
DT -> 'the' [p7]
N -> 'child' [p8]
N -> 'cake' [p9]
N -> 'fork' [p10]
PRP -> 'with' [p11]
PRP -> 'to' [p12]
V -> 'saw' [p13]
V -> 'ate' [p14]

t_1



$$p(t_1)=p_0p_1p_4p_7p_8p_{14}p_2p_1p_3p_7p_9p_{11}p_1p_7p_{10}$$

t_2

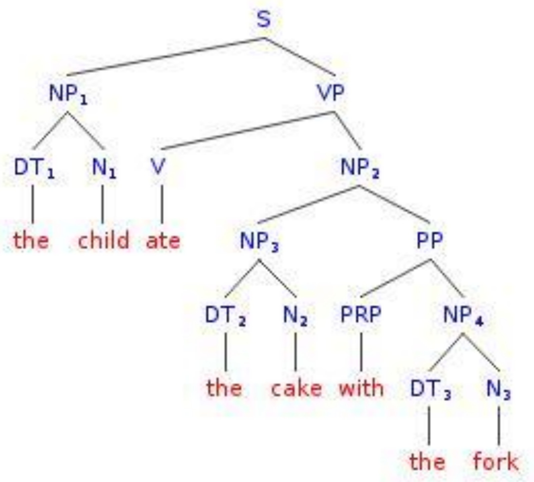


$$p(t_2)=p_0p_1p_5p_7p_8p_4p_3p_{14}p_1p_{11}p_1p_7p_9p_7p_{10}$$

Example

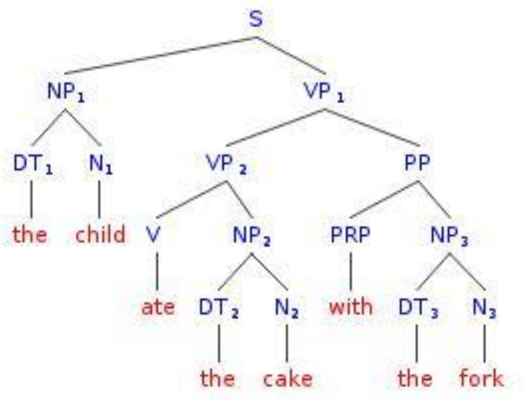
S -> NP VP [p0=1]
NP -> DT N [p1]
NP -> NP PP [p2]
PP -> PRP NP [p3=1]
VP -> V NP [p4]
VP -> VP PP [p5]
DT -> 'a' [p6]
DT -> 'the' [p7]
N -> 'child' [p8]
N -> 'cake' [p9]
N -> 'fork' [p10]
PRP -> 'with' [p11]
PRP -> 'to' [p12]
V -> 'saw' [p13]
V -> 'ate' [p14]

t_1



$$p(t_1)=p_0p_1p_4p_7p_8p_{14}p_2p_1p_3p_7p_9p_{11}p_1p_7p_{10}$$

t_2

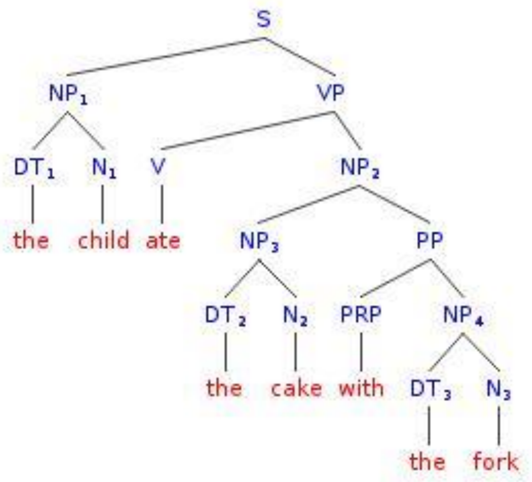


$$p(t_2)=p_0p_1p_5p_7p_8p_4p_3p_{14}p_1p_{11}p_1p_7p_9p_7p_{10}$$

Example

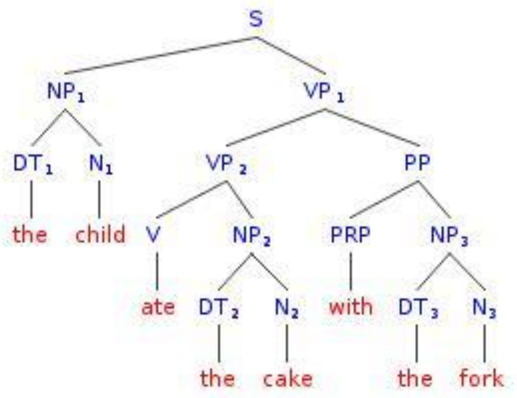
```
S -> NP VP       [p0=1]
NP -> DT N       [p1]
NP -> NP PP      [p2]
PP -> PRP NP     [p3=1]
VP -> V NP       [p4]
VP -> VP PP      [p5]
DT -> 'a'        [p6]
DT -> 'the'      [p7]
N -> 'child'     [p8]
N -> 'cake'      [p9]
N -> 'fork'      [p10]
PRP -> 'with'    [p11]
PRP -> 'to'      [p12]
V ->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



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

Introduction to NLP

252.

Probabilistic Parsing

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

Sample Probabilistic Grammar

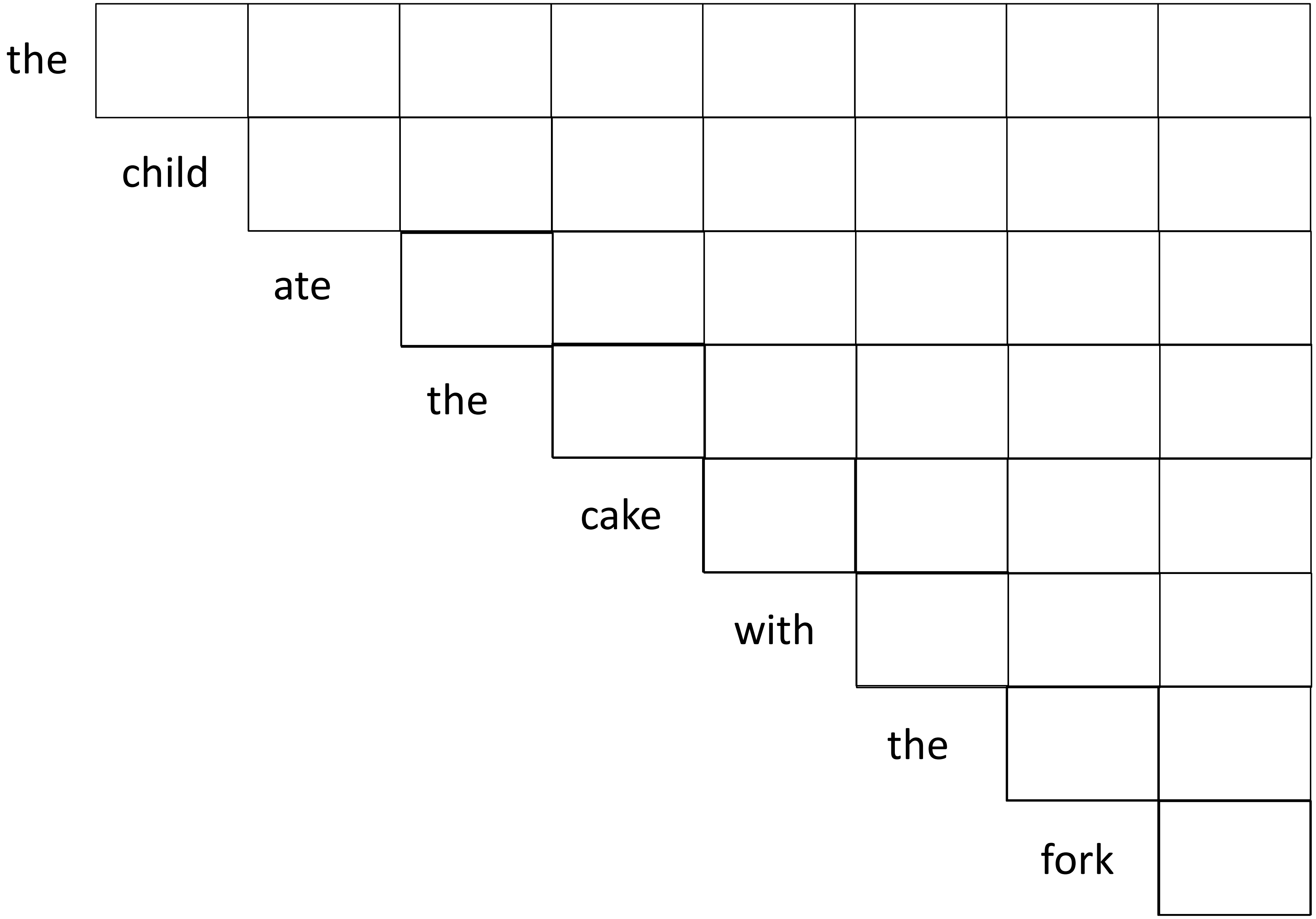
Grammar		Lexicon
$S \rightarrow NP\ VP$	[.80]	$Det \rightarrow that$ [.10] | a [.30] | the [.60]
$S \rightarrow Aux\ NP\ VP$	[.15]	$Noun \rightarrow book$ [.10] | $flight$ [.30]
$S \rightarrow VP$	[.05]	| $meal$ [.15] | $money$ [.05]
$NP \rightarrow Pronoun$	[.35]	| $flights$ [.40] | $dinner$ [.10]
$NP \rightarrow Proper\text{-}Noun$	[.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

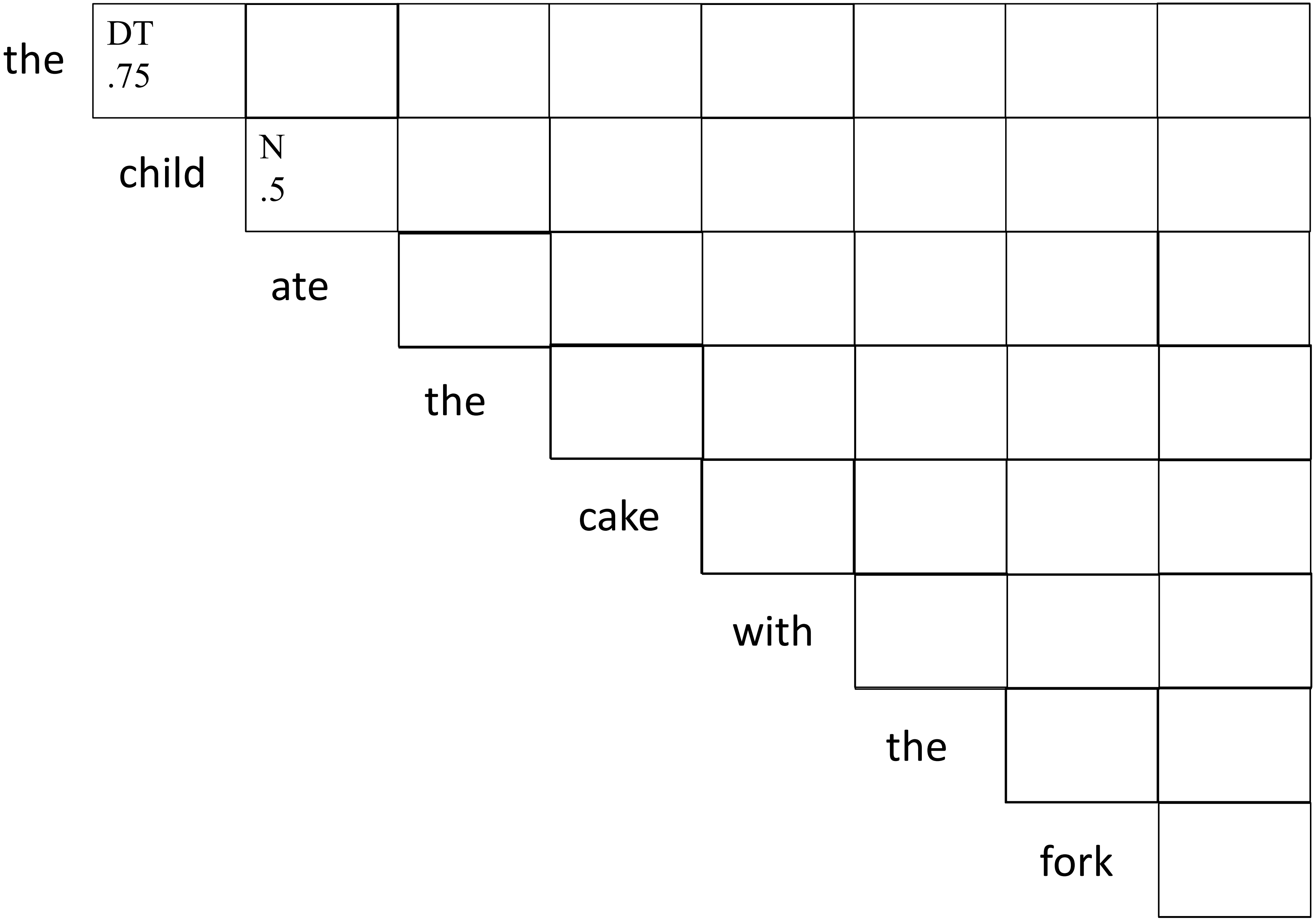
Example

```
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]
```

```
function PROBABILISTIC-CKY(words,grammar) returns most probable parse
                                                   and its probability
  for j ← from 1 to LENGTH(words) do
     for all { A | A → words[j] ∈ grammar}
        table[j−1, j,A] ← P(A → words[j])
     for i ← from j−2 downto 0 do
        for k ← i+1 to j−1 do
              for all { A | A → BC ∈ grammar,
                                and table[i,k,B] > 0 and table[k, j,C] > 0 }
                    if (table[i,j,A] < P(A → BC) × table[i,k,B] × table[k,j,C]) then
                       table[i,j,A] ← P(A → BC) × table[i,k,B] × table[k,j,C]
                       back[i,j,A] ← {k,B,C}
     return BUILD_TREE(back[1, LENGTH(words), S]), table[1, LENGTH(words), S]
```

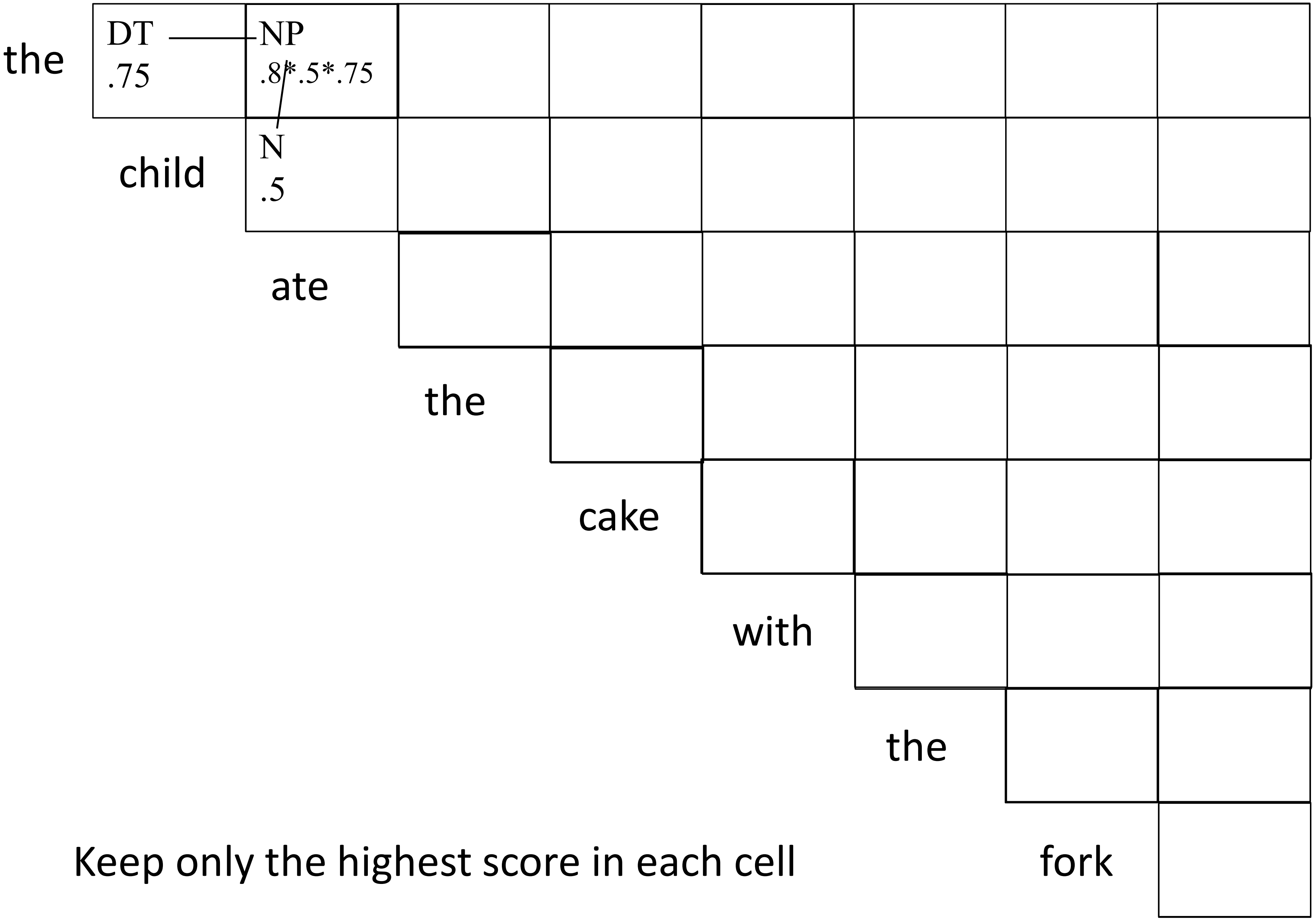

the
child
ate
the
cake
with
the
fork

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	



Keep only the highest score in each cell

Exercise

- Now, on your own, 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.

Notes

- Stanford Demo
  - http://nlp.stanford.edu:8080/parser/
- PTB statistics
  - 50,000 sentences (40,000 training; 2,400 testing)
- PTB peculiarities
  - includes traces and other null elements
  - Flat NP structure (e.g., NP -> DT JJ JJ NNP NNS)
- Parent transformation
  - Subject NPs are more likely to have modifiers than object NPs
  - E.g., replace NP with NP^S

Introduction to NLP

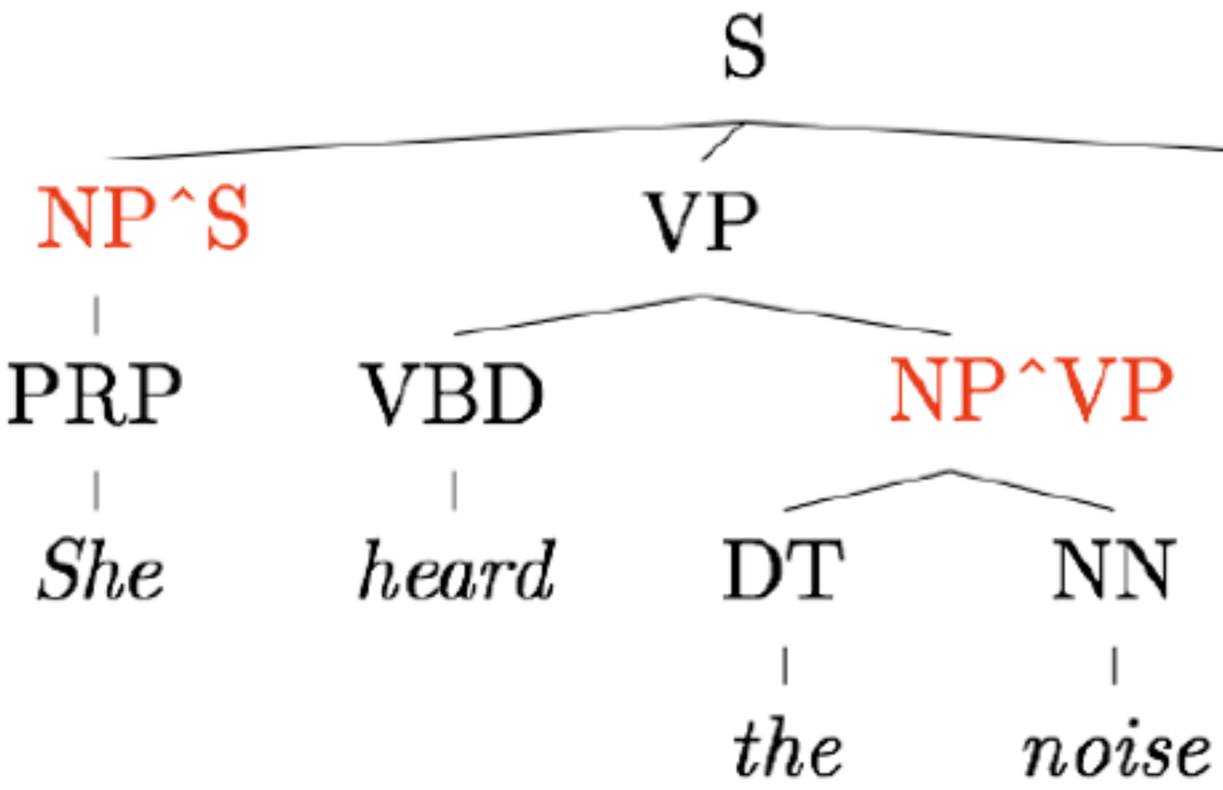
253.

Lexicalized Parsing

Limitations of PCFGs

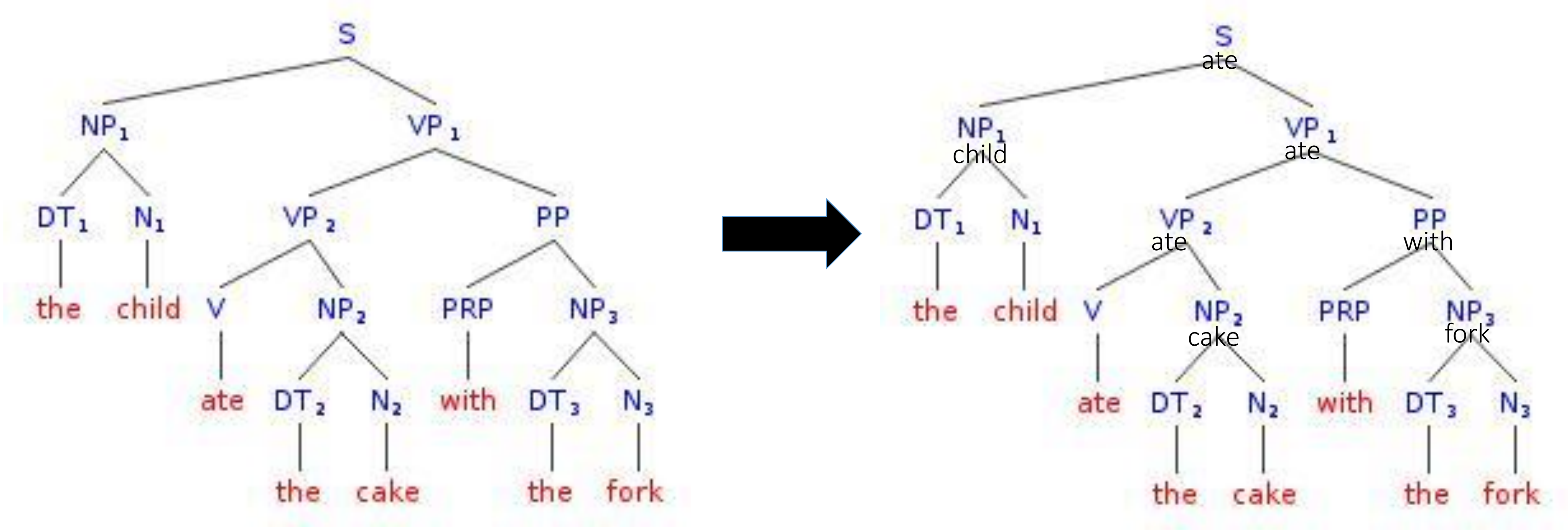
- The probabilities don’t depend on the specific words
  - E.g., *give* someone something (2 arguments) vs. *see* something (1 argument)
- It is not possible to disambiguate sentences based on semantic information
  - E.g., eat pizza with *pepperoni* vs. eat pizza with *fork*
- Lexicalized grammars - idea
  - Use the head of a phrase as an additional source of information
  - VP[ate] -> V[ate]
  - Fundamental idea in syntax, cf. X-bar theory, HPSG
  - Constituents receive their heads from their head child

Parent Annotation



[Johnson 1998]

Lexicalization



Head Extraction Example (Collins)

- NP -> DT NNP NN (rightmost)
- NP -> DT NN NNP (rightmost)
- NP -> NP PP (leftmost)
- NP -> DT JJ (rightmost)
- NP -> DT (rightmost leftover child)

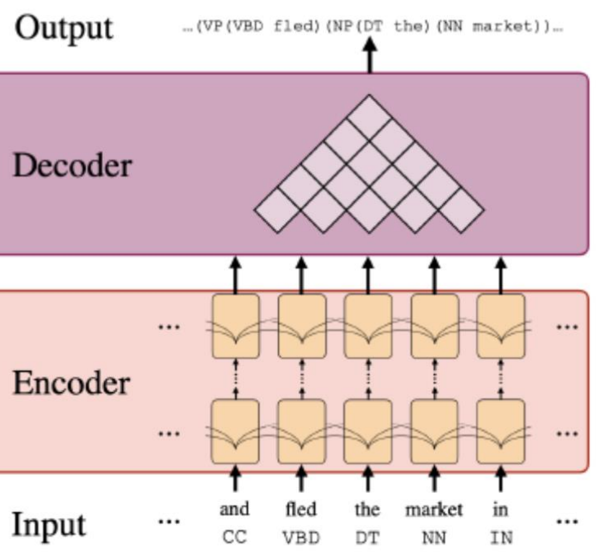
Notes

- Complexity of lexicalized parsing
  - $O(N^5 g^3 V^3)$, instead of $O(N^3)$ because of the lexicalization
    - N = sentence length
    - g = number of non-terminals
    - V = vocabulary size
  - Use beam search (Charniak; Collins)
- Sparse data
  - 40,000 sentences; 12,409 rules (Collins)
  - 15% of all test sentences contain a rule not seen in training (Collins)

Recent results

Labeled precision/recall on PTB-WSJ

- Vanilla PCFG: 70.6% recall, 74.8% precision
- Lexicalized PCFG: 88.1% recall, 88.3% precision
- Neuralized constituency parsing (Kitaev and Klein, 2018): 94.9% recall, 95.4% precision



Neural encoding followed by max marginal decoding: no independence assumption (read the paper)

[Karl Stratos]

Recent results (Label Attention Layer)

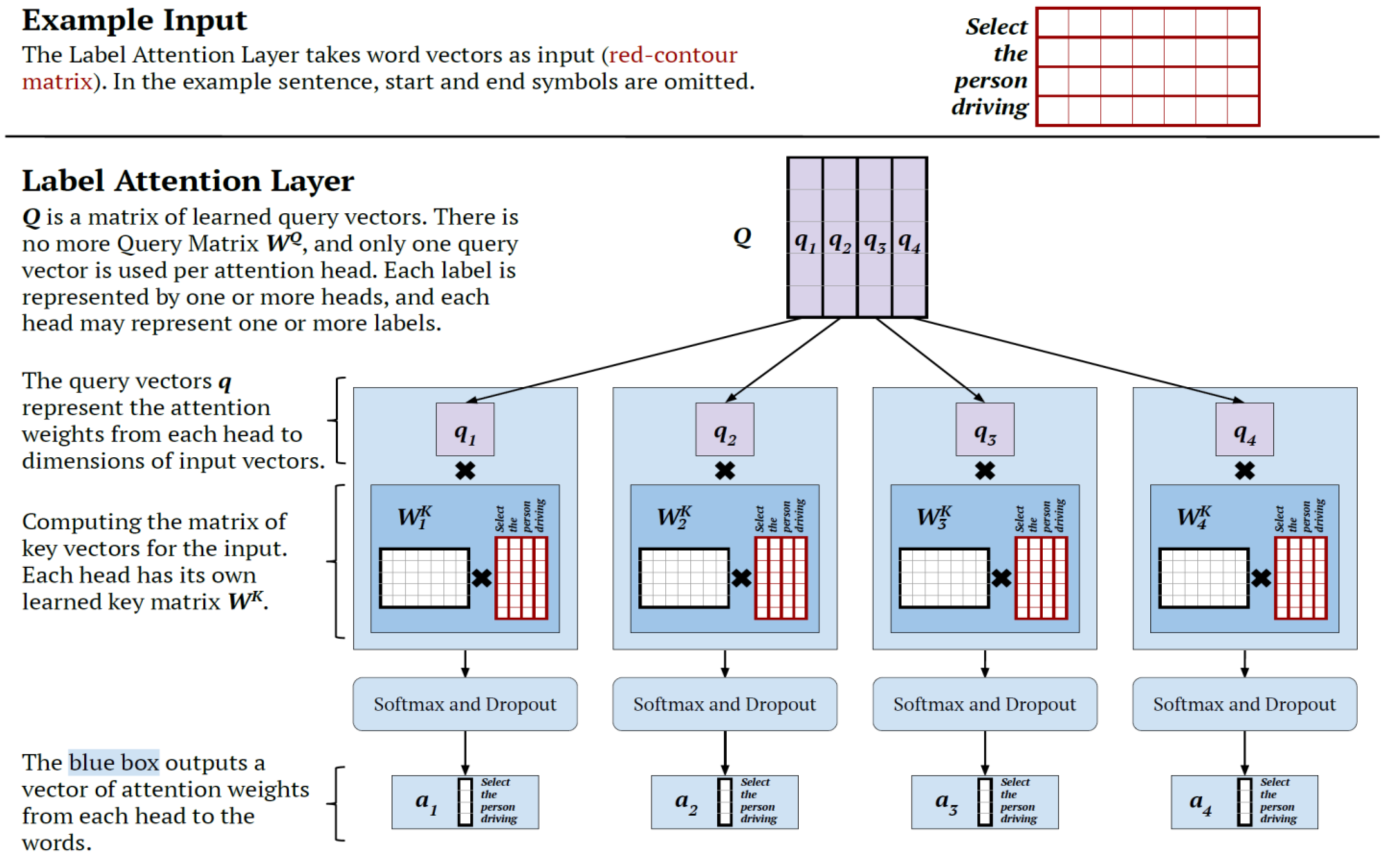


Figure 2: The architecture of the top of our proposed Label Attention Layer. In this figure, the example input sentence is "*Select the person driving*".

https://paperswithcode.com/sota/constituency-parsing-on-penn-treebank