CS626: Speech, NLP and the Web

Chart Parsing, CYK Parsing, Probabilistic Parsing, Start of Dependency Parsing

Pushpak Bhattacharyya

Computer Science and Engineering Department

IIT Bombay

Week of 4th October, 2021

Grammar and Parsing Algorithms

A simplified grammar

- $S \rightarrow NP\ VP$
- $NP \rightarrow DT\ N \mid N$
- $VP \rightarrow V\ ADV \mid V$

- The above captures declarative sentences
- 4 kinds of sentences as per traditional grammar
  - Declarative (Sun rises in the east)
  - Interrogative (Does sun rise in the east?)
  - Imperative (Rise in the east please)
  - Exclamatory (Oh, sun rises in the east!)

Example Sentence

People laugh

1 2 3

These are positions

Lexicon:

People - N, V

Laugh - N, V

This indicate that both Noun and Verb is possible for the word “People”

Top-Down Parsing

State	Backup State	Action
1. ((S) 1)	-	-
2. ((NP VP)1)	-	-
3a. ((DT N VP)1	((N VP) 1)	-
3b. ((N VP)1)	-	-
4. ((VP)2)	-	Consume “People”
5a. ((V ADV)2)	((V)2)	-
6. ((ADV)3)	((V)2)	Consume “laugh”
5b. ((V)2)	-	-
6. ((.)3)	-	Consume “laugh”

Position of input pointer

Termination Condition : All inputs over. No symbols remaining.

Note: Input symbols can be pushed back.

Exercise

- Construct examples of Top-Down parsing failure by
  - Input over but stack not empty
  - Stack empty but input not over

Discussion for Top-Down Parsing

- This kind of searching is goal driven.
- Gives importance to textual precedence (rule precedence).
- No regard for data, a priori (useless expansions made).

Bottom-Up Parsing

Some conventions:

N_{12}

Represents positions

$S_{1?} \rightarrow NP_{12} \circ VP_{2?}$

End position unknown

Work on the LHS done, while the work on RHS remaining

Bottom-Up Parsing (pictorial representation)

$S \rightarrow NP_{12}\ VP_{23}\ °$

People   Laugh

1   2   3

N_{12}

V_{12}

$NP_{12} \rightarrow N_{12}\ °$

$VP_{12} \rightarrow V_{12}\ °$

$S_{1?} \rightarrow NP_{12}\ °\ VP_{2?}$

N_{23}

V_{23}

$NP_{23} \rightarrow N_{23}\ °$

$VP_{23} \rightarrow V_{23}\ °$

Problem with Top-Down Parsing

- Left Recursion
  - Suppose you have A-> AB rule.

    Then we will have the expansion as follows:
    - ((A)K) -> ((AB)K) -> ((ABB)K) ……..

Combining top-down and bottom-up strategies

Top-Down Bottom-Up Chart Parsing

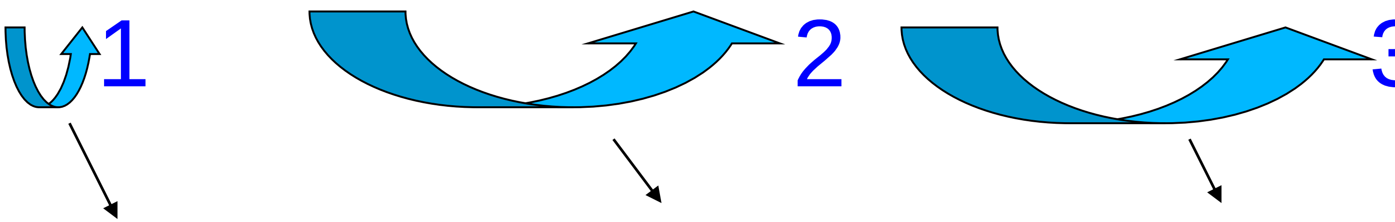
- Combines advantages of top-down & bottom-up parsing.
- Does not work in case of left recursion.
  - *e.g.* – “People laugh”
    - People – noun, verb
    - Laugh – noun, verb
  - Grammar – $S \rightarrow NP\ VP$

    $NP \rightarrow DT\ N \mid N$

    $VP \rightarrow V\ ADV \mid V$

Transitive Closure

People laugh



S → •NP VP

NP → •DT N

NP → •N

NP → N•

S → NP•VP

VP → •V ADV

VP → •V

VP → V •

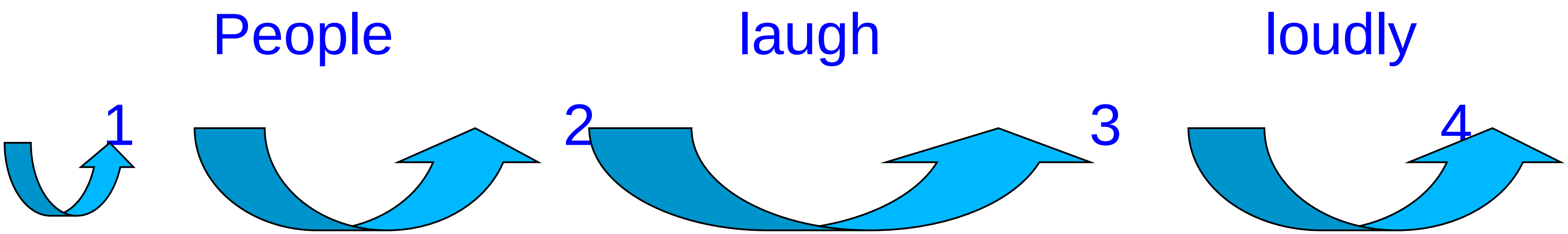
S → NP VP •

success

Arcs in Parsing

- Each arc represents a chart which records
  - Completed work (left of •)
  - Expected work (right of •)

Example



S → • NP VP
NP → • DT N
NP → • N

NP → N•
S → NP•VP
VP → •V ADV
VP → •V

VP → V•
VP → V•ADV
S → NP VP•

VP → V ADV•
S → NP VP•

An important parsing algo

Illustrating CYK [Cocke, Younger, Kashmi] Algo

- S → NP VP 1.0
- NP → DT NN 0.5
- NP → NNS 0.3
- NP → NP PP 0.2
- PP → P NP 1.0
- VP → VP PP 0.6
- VP → VBD NP 0.4

- DT → the 1.0
- NN → gunman 0.5
- NN → building 0.5
- VBD → sprayed 1.0
- NNS → bullets 1.0

CYK: Start with (0,1)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To **From**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT						
1	-------						
2	-------	-------- -					
3	-------	-------- -	--------				
4	------- -	-------- -	--------	-------- -			
5	------- -	-------- -	--------	-------- -	-------- -		
6	------- -	-------- -	--------	-------- -	-------- -	-------- -	

CYK: Keep filling diagonals

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT						
1	-------	NN					
2	-------	---------					
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: Try getting higher level structures

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP					
1	-------	NN					
2	-------	---------					
3	-------	---------	--------				
4	---------	---------	--------	---------			
5	---------	---------	--------	---------	---------		
6	---------	---------	--------	---------	---------	---------	

CYK: Diagonal continues

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7**.

To From	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	DT	NP					
1	-------	NN					
2	-------	-------- -	VBD				
3	-------	-------- -	--------				
4	------- -	-------- -	--------	-------- -			
5	------- -	-------- -	--------	-------- -	-------- -		
6	------- -	-------- -	--------	-------- -	-------- -	-------- -	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------				
1	-------	NN	--------				
2	-------	---------	VBD				
3	-------	---------	--------				
4	--------	---------	--------	---------			
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------				
1	-------	NN	--------				
2	-------	---------	VBD				
3	-------	---------	--------	DT			
4	---------	---------	--------	---------			
5	---------	---------	--------	---------	---------		
6	---------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------			
3	-------	---------	--------	DT			
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: starts filling the 5th column

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------			
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------			
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* **1** *gunman* **2** *sprayed* **3** *the* **4** *building* **5** *with* **6** *bullets* **7** *.*

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------			
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK: S found, but NO termination!

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S		
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------		
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S		
1	-------	NN	--------	---------	---------		
2	-------	---------	VBD	---------	VP		
3	-------	---------	--------	DT	NP		
4	--------	---------	--------	---------	NN		
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	

CYK: Control moves to last column

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	
4	--------	---------	--------	---------	NN	---------	
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	
1	-------	NN	--------	---------	---------	---------	
2	-------	---------	VBD	---------	VP	---------	
3	-------	---------	--------	DT	NP	---------	NP
4	--------	---------	--------	---------	NN	---------	---------
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK (cont…)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7*.*

To From	1	2	3	4	5	6	7
0	DT	NP	--------	-------- -	S	-------- -	
1	-------	NN	--------	-------- -	-------- -	-------- -	
2	-------	-------- -	VBD	-------- -	VP	-------- -	VP
3	-------	-------- -	--------	DT	NP	-------- -	NP
4	------- -	-------- -	--------	-------- -	NN	-------- -	-------- -
5	------- -	-------- -	--------	-------- -	-------- -	P	PP
6	------- -	-------- -	--------	-------- -	-------- -	-------- -	NP NNS

CYK: filling the last column

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	-------- -	S	-------- -	
1	-------	NN	--------	-------- -	-------- -	-------- -	-------- -
2	-------	-------- -	VBD	-------- -	VP	-------- -	VP
3	-------	-------- -	--------	DT	NP	-------- -	NP
4	------- -	-------- -	--------	-------- -	NN	-------- -	-------- -
5	------- -	-------- -	--------	-------- -	-------- -	P	PP
6	------- -	-------- -	--------	-------- -	-------- -	-------- -	NP NNS

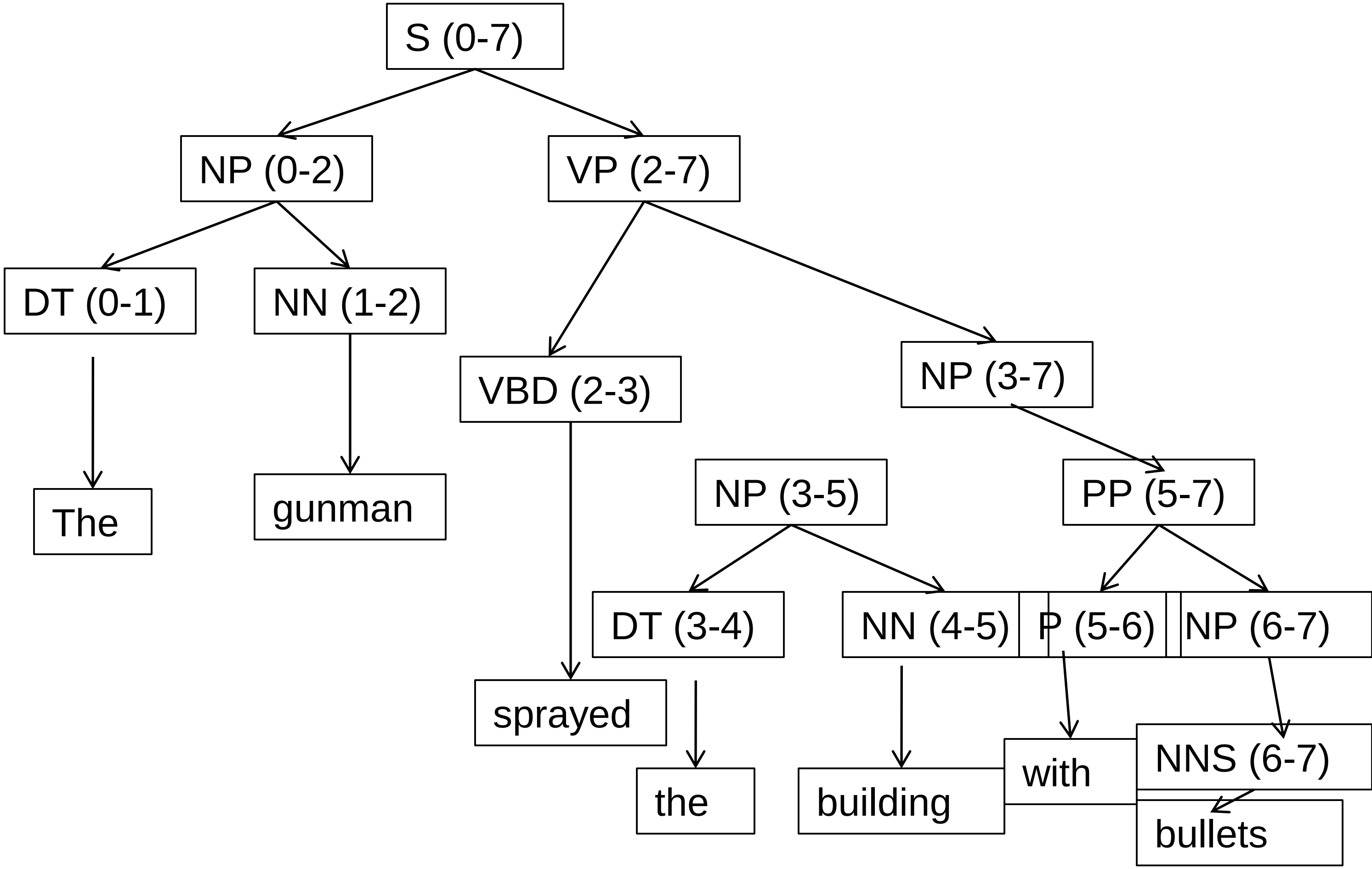
CYK: terminates with S in (0,7)

0 *The* 1 *gunman* 2 *sprayed* 3 *the* 4 *building* 5 *with* 6 *bullets* 7.

To From	1	2	3	4	5	6	7
0	DT	NP	--------	---------	S	---------	S
1	-------	NN	--------	---------	---------	---------	---------
2	-------	---------	VBD	---------	VP	---------	VP
3	-------	---------	--------	DT	NP	---------	NP
4	--------	---------	--------	---------	NN	---------	---------
5	--------	---------	--------	---------	---------	P	PP
6	--------	---------	--------	---------	---------	---------	NP NNS

CYK: Extracting the Parse Tree

- The parse tree is obtained by keeping back pointers.



Exercise

- Sit with CYK table that is filled and uncover all the parse trees
- Understand the tree extraction procedure

Probabilistic parsing

Example of Sentence labeling: Parsing

[$_{S1}$[$_{S}$[$_{S}$[$_{VP}$[$_{VB}$Come][$_{NP}$[$_{NNP}$July]]]]]

[$_{,}$,]

[$_{CC}$ and]

[$_{S}$ [$_{NP}$ [$_{DT}$ the] [$_{JJ}$ IIT] [$_{NN}$ campus]]

[$_{VP}$ [$_{AUX}$ is]

[$_{ADJP}$ [$_{JJ}$ abuzz]

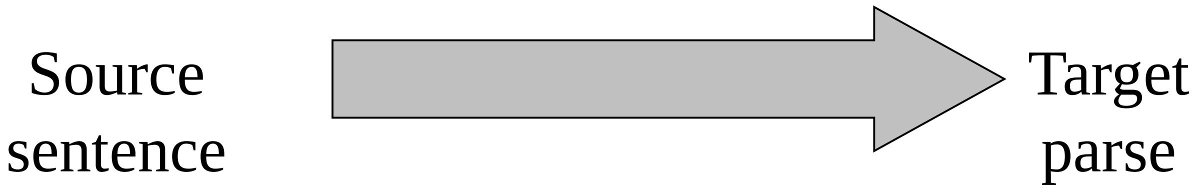
[$_{PP}$[$_{IN}$ with]

[$_{NP}$[$_{ADJP}$ [$_{JJ}$ new] [$_{CC}$ and] [$_{VBG}$ returning]]

[$_{NNS}$ students]]]]]]

[$_{.}$.]]]

Noisy Channel Modeling

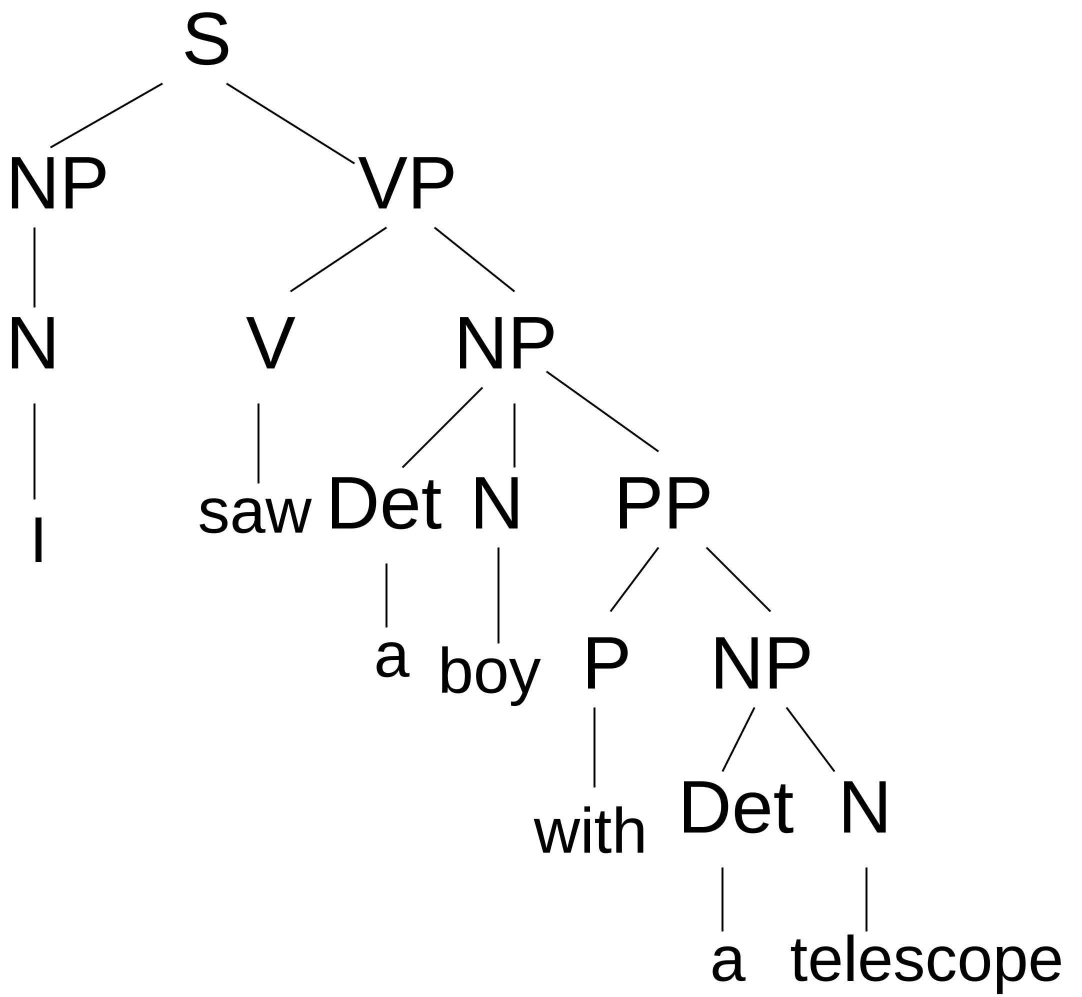


$$T^* = \underset{T}{\operatorname{argmax}} [P(T|S)]$$

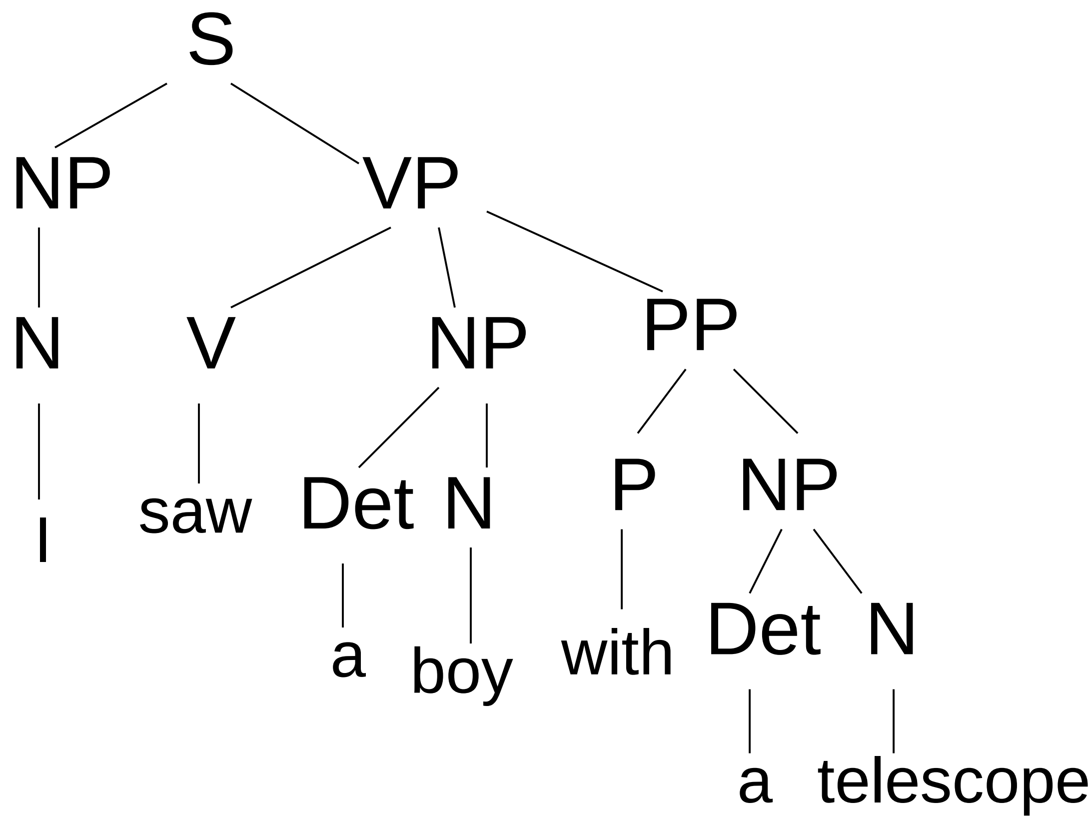
$$= \underset{T}{\operatorname{argmax}} [P(T).P(S|T)]$$

$$= \underset{T}{\operatorname{argmax}} [P(T)],$$ ***since given the parse the sentence is completely determined and P(S|T)=1***

I saw a boy with a telescope: Tree - 1



Constituency Parse Tree -2



Formal Definition of PCFG

- A PCFG consists of
  - A set of terminals $\{w_k\}$, k = 1,….,V

    $\{w_k\}$ = { child, teddy, bear, played…}
  - A set of non-terminals $\{N^i\}$, i = 1,…,n

    $\{N_i\}$ = { NP, VP, DT…}
  - A designated start symbol N^1
  - A set of rules $\{N^i \rightarrow \zeta^j\}$, where ζ^j is a sequence of terminals & non-terminals

    NP $\rightarrow$ DT NN
  - A corresponding set of rule probabilities

Rule Probabilities

- Rule probabilities are such that for the same non terminal all production rules sum to1.

  E.g., P(NP $\rightarrow$ DT NN) = 0.2
  P(NP $\rightarrow$ NNS) = 0.5
  P(NP $\rightarrow$ NP PP) = 0.3

- P(NP $\rightarrow$ DT NN) = 0.2
  - Means 20 % of the training data parses use the rule NP $\rightarrow$ DT NN

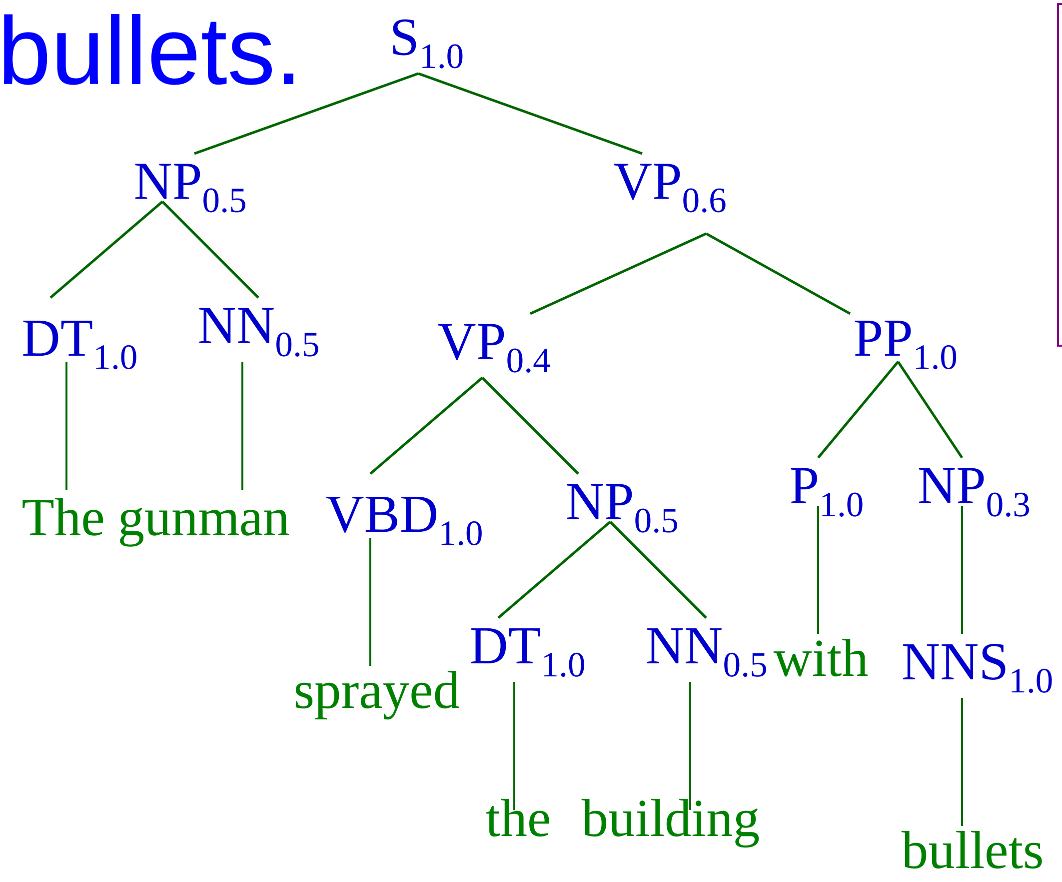
Probabilistic Context Free Grammars

- S → NP VP 1.0
- NP → DT NN 0.5
- NP → NNS 0.3
- NP → NP PP 0.2
- PP → P NP 1.0
- VP → VP PP 0.6
- VP → VBD NP 0.4

- DT → the 1.0
- NN → gunman 0.5
- NN → building 0.5
- VBD → sprayed 1.0
- NNS → bullets 1.0

Example Parse t_1`

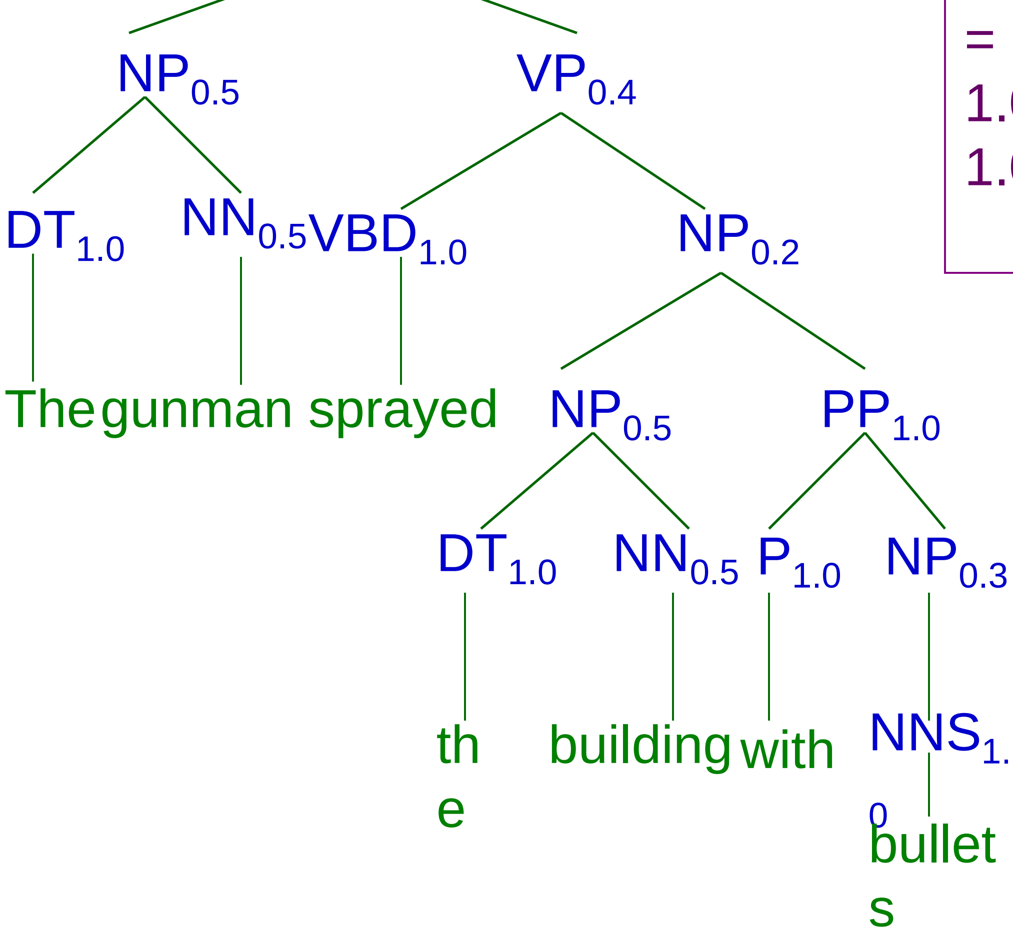
- The gunman sprayed the building with bullets.



$P(t_1) = 1.0 * 0.5 * 1.0 * 0.5 * 0.6 * 0.4 * 1.0 * 0.5 * 1.0 * 0.5 * 1.0 * 1.0 * 0.3 * 1.0 = 0.00225$

Another Parse t_2

- The gunman sprayed the building with bullets.



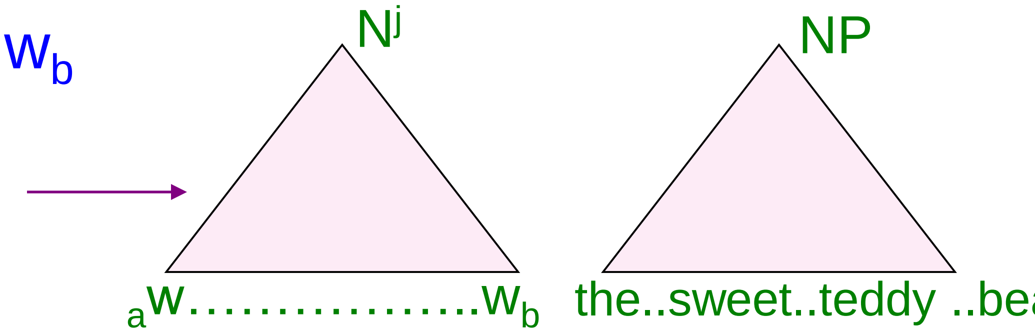
P (t_2)
= 1.0 * 0.5 * 1.0 * 0.5 * 0.4 * 1.0 * 0.2 * 0.5 * 1.0 * 0.5 * 1.0 * 1.0 * 0.3 * 1.0
= 0.0015

Probability of a sentence

Notation: (*a,b* etc. are BETWEEN word indices

- w_{ab} – subsequence $_a w \ldots . w_b$
- N^j dominates $_a w \ldots . w_b$

  or yield(N^j) = $_a w \ldots . w_b$



Probability of a sentence = $P(w_{0,l})$ *(0 is the index before the first word and l the index after the last word. All other indices are between words)*

$=\Sigma_t (P(w_{0,l}, t)$

$=\Sigma_t (P(t). (P(w_{0,l} | t)$

$=\Sigma_t P(t).1$

If t is a parse tree for the sentence $w_{0,l}$, this will be 1 !!

Where *t* is a parse tree of the sentence

Assumptions of the PCFG model

- Place invariance :

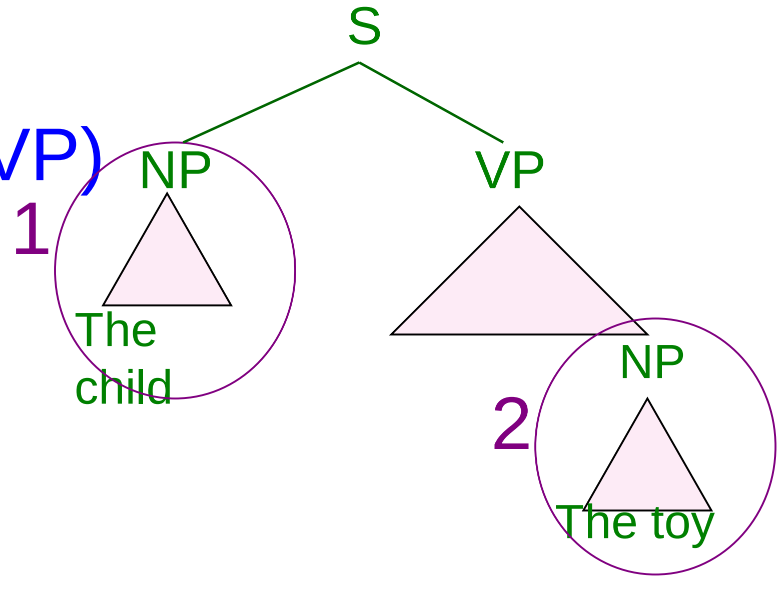
  P(NP $\rightarrow$ DT NN) is same in locations 1 and 2

- Context-free :

  P(NP $\rightarrow$ DT NN | anything outside “The child”)
  = P(NP $\rightarrow$ DT NN)

- Ancestor free : At 2,

  P(NP $\rightarrow$ DT NN|its ancestor is VP)
  = P(NP $\rightarrow$DT NN)

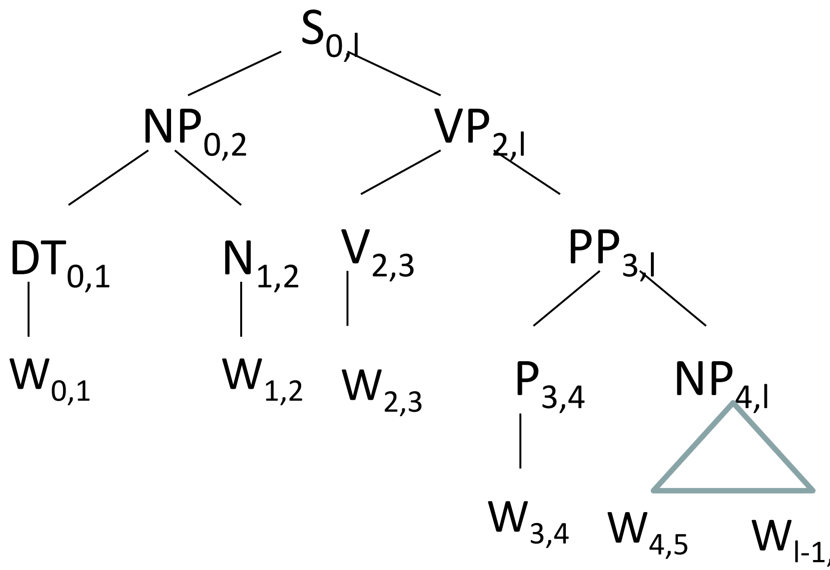


Probability of a parse tree

- *Domination* :We say N_j dominates from *k* to *l*, symbolized as $N^j_{k,l}$, if $W_{k,l}$ is derived from N_j
- P (tree |sentence) = P (tree | $S_{1,l}$)

  where $S_{0,l}$ means that the start symbol S dominates the word sequence $W_{0,l}$
- P (t |s) approximately equals joint probability of constituent non-terminals dominating the sentence fragments (next slide)

Probability of a parse tree (cont.)



$P(t|s) = P(t \mid S_{0,l})$

$= P(NP_{0,2}, DT_{0,1}, w_{0,1}, N_{1,2}, w_{1,2}, VP_{2,l}, V_{2,3}, w_{2,3}, PP_{3,l}, P_{3,4}, w_{3,4}, NP_{4,l}, w_{4,l} \mid S_{0,l})$

$= P(NP_{0,2}, VP_{2,l} \mid S_{0,l}) * P(DT_{0,1}, N_{1,2} \mid NP_{0,2}) * P(w_{0,1} \mid DT_{0,1}) * P(w_{1,2} \mid N_{1,2}) * P(V_{2,3}, PP_{3,l} \mid VP_{2,l}) * P(w_{2,3} \mid V_{2,3}) * P(P_{3,4}, NP_{4,l} \mid PP_{3,l}) * P(w_{3,4} \mid P_{3,4}) * P(w_{4,l} \mid NP_{4,l})$

(Using Chain Rule, Context Freeness and Ancestor Freeness)

Why probability in Parsing

Why probability in parsing? (1/2)

- What is randomness in tree?
  - At every position of the sentence there is a potential ambiguity with respect to whatever phrase structure can be built till and from that point
  - This leads to ambiguity in the parse tree
  - The root of a subtree covering a segment of the sentence is said to dominate that segment
  - The ambiguity in deciding domination leads to randomness

Why probability in parsing? (2/2)

- Example:
- In the earthquake **old** men and women were taken to safe locations

In the earthquake **old** men and women were taken to safe locations.

  - Ambiguity:
    - old -> “men” or old -> “men and women”
    - Adjective phrase dominates “men” or “men and women”
  - This example illustrates ambiguity, uncertainly, and calls for probability to be used in this situation

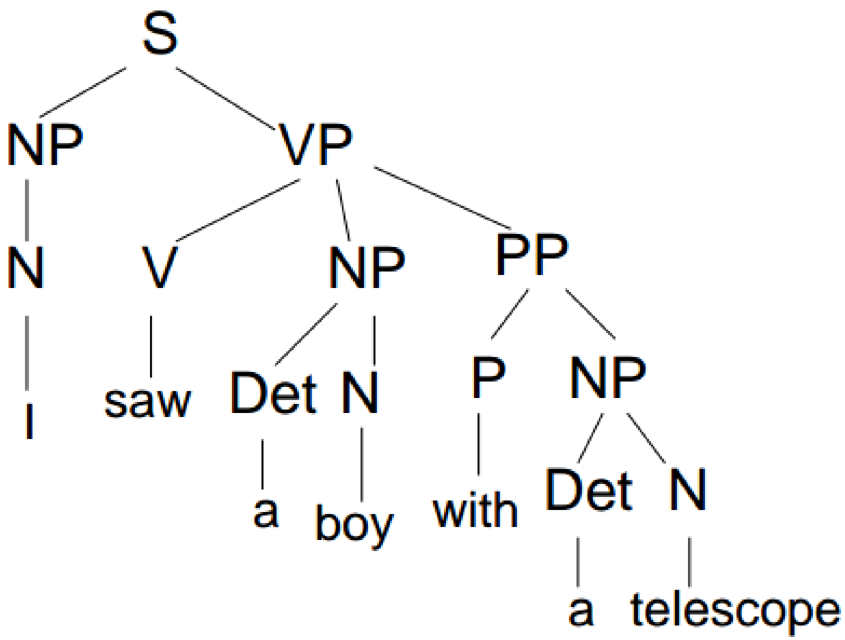
Domination

- A sentence is dominated by the symbol S through domination of segments by phrases
- Analogy
  - The capital of a country dominates the whole country.
  - The capital of a state dominates the whole state.
  - The district headquarter dominates the district.

- Another analogy

  - IIT Bombay is dominated by the administration of IIT Bombay.
  - Administration dominates Heads of Depts
  - The department is dominated by head of the department.

Domination: Example

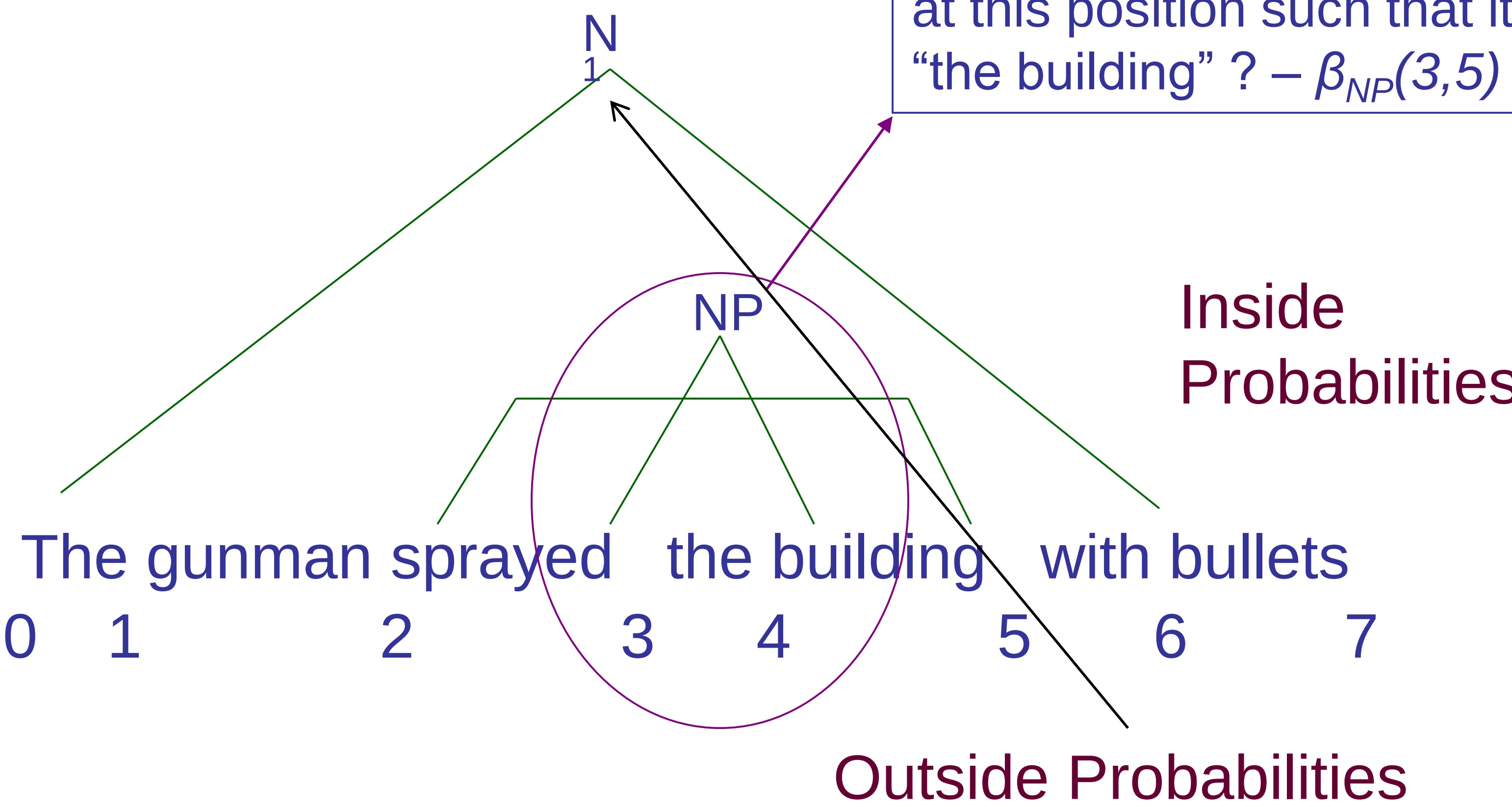


- Dominations
  - NP dominates “a telescope”
  - VP dominates “saw a boy with a telescope
  - S dominates the whole sentence
- Domination is composed of many sub-domination.

- I saw a boy with a telescope
  - Meaning: I used the telescope to see the boy

Interesting Probabilities

What is the probability of having a NP at this position such that it will derive "the building" ? – $\beta_{NP}(3,5)$



What is the probability of starting from N^1 and deriving "The gunman sprayed", a NP and "with bullets" ? - $\alpha_{NP}(3,5)$

Parse tree for the given sentence using probabilistic CYK parsing

$_0$ The $_1$ gunman $_2$ sprayed $_3$ the $_4$ building $_5$ with $_6$ bullets $_7$

- Two parse trees are possible because the sentence has attachment ambiguity .
- Total 16 multiplications are required to make both the parse trees using probabilistic CYK.
- Number of multiplications is less in comparison to a probabilistic parsing which prepares the two parse trees independently
- with 28 multiplication.

	The 1	gunman 2	Sprayed 3	the 4	Building 5	with 6	Bullets 7
0	β_{DT} (0-1) =1.0	β_{NP} (0-2) =0.25					β_{S}(0-7) =0.006
1		β_{NN} (1-2) =0.5					
2			β_{VBD}(2-3) =1.0		β_{VP} (2-5) =0.1		β_{VP}(2-7) =0.024
3				β_{DT}(3-4) =1.0	β_{NP} *(3-5) =0.25*		β_{NP}(3-7) =0.015
4					β_{NN} (4-5) =0.5		
5						β_{P}(5-6) =1.0	β_{PP}(5-7) =0.3
6							$\beta_{NP/NNS}$(6-7) =1.0

Calculation of values for each non terminal occuring in the CYK table

β_{DT} (0-1) =1.0 (From Grammar rules)

β_{NN} (1-2) =0.5 (From Grammar rules)

β_{NP} (0-2) = P(the gunman | NP_{0-2} , G)
= P(NP->DT NN)* β_{DT} (0-1) * β_{NN} (1-2)
= 0.5 * 1.0 * 0.5
=0.25

β_{VBD}(2-3) =1.0 (From Grammar rules)

β_{DT}(3-4) =1.0 (From Grammar rules)

β_{NN} (4-5) =0.5 (From Grammar rules)

β_{NP} (3-5) = P(the building | NP_{3-5} , G)
= P(NP->DT NN)* β_{DT} (3-4) * β_{NN} (4-5)
= 0.5 * 1.0 * 0.5
=0.25

β_{VP} (2-5) = P(VP->VBD NP)* β_{VBD} (2-3) * β_{NN} (3-5)

= 0.4 * 1 * 0.25

= 0.1

β_{P}(5-6) = 1.0 (From Grammar rules)

$\beta_{NP/NNS}$(6-7) =1.0*0.3 (From Grammar rules)=0.3

β_{PP}(5-7) = P(PP->P NP) * β_{P}(5-6) * $\beta_{NP/NNS}$(6-7)

= 1.0 * 1.0 * 0.3

= 0.3

β_{NP}(3-7) = P(NP->NP PP)* β_{NP}(3-5) * β_{PP}(5-7)

= 0.2 * 0.25 * 0.3

=0.015

β_{VP}(2-7) =(P(VP->VBD NP)* β_{VBD} (2-3) * β_{NP} (3-7) + P(VP->VP PP) * β_{VP} (2-5) * β_{PP} (5-7))

= 0.4 * 1 * 0.015 + 0.6 * 0.1 * 0.3

= 0.024

β_{S}(0-7) =P(S->NP VP) * β_{NP} (0-2) * β_{VP} (2-7)

= 1 * 0.25 * 0.024

= 0.006

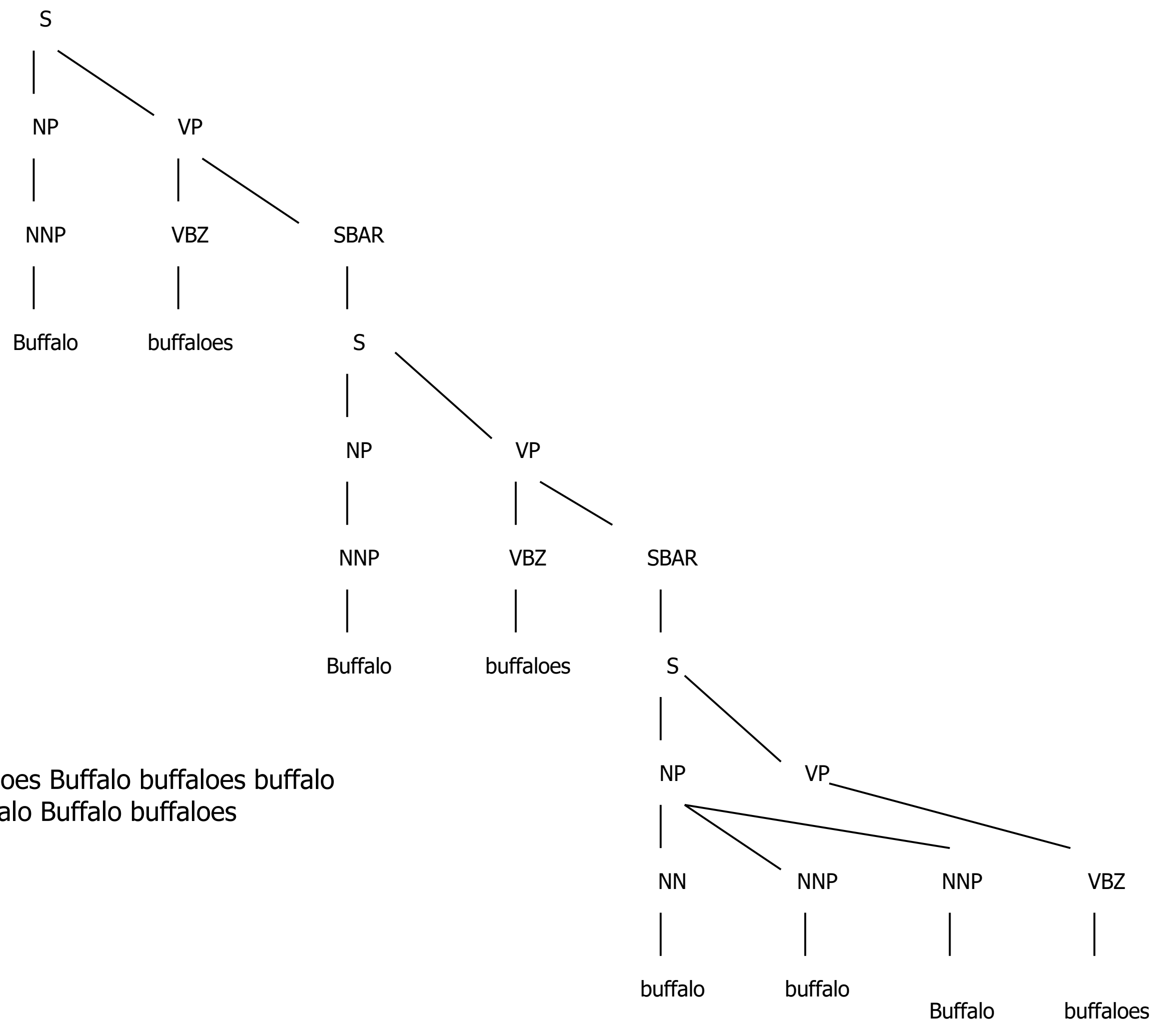
Stress Test for Parsing: A very difficult parsing situation!

Repeated Word handling

Sentence on Buffaloes!

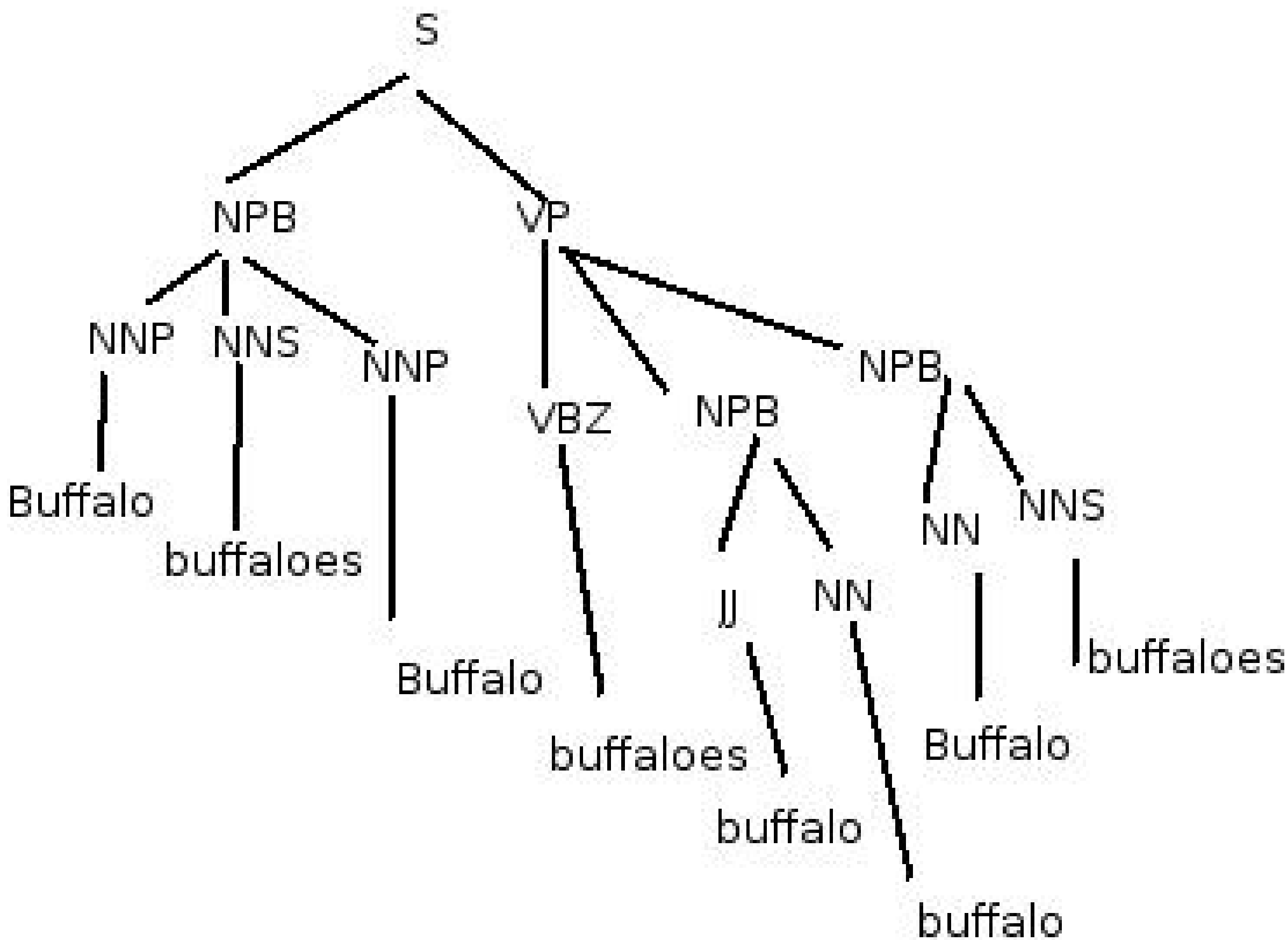
Buffaloe buffaloes Buffaloe buffaloes buffaloe buffaloe Buffaloe buffaloes

Charniak

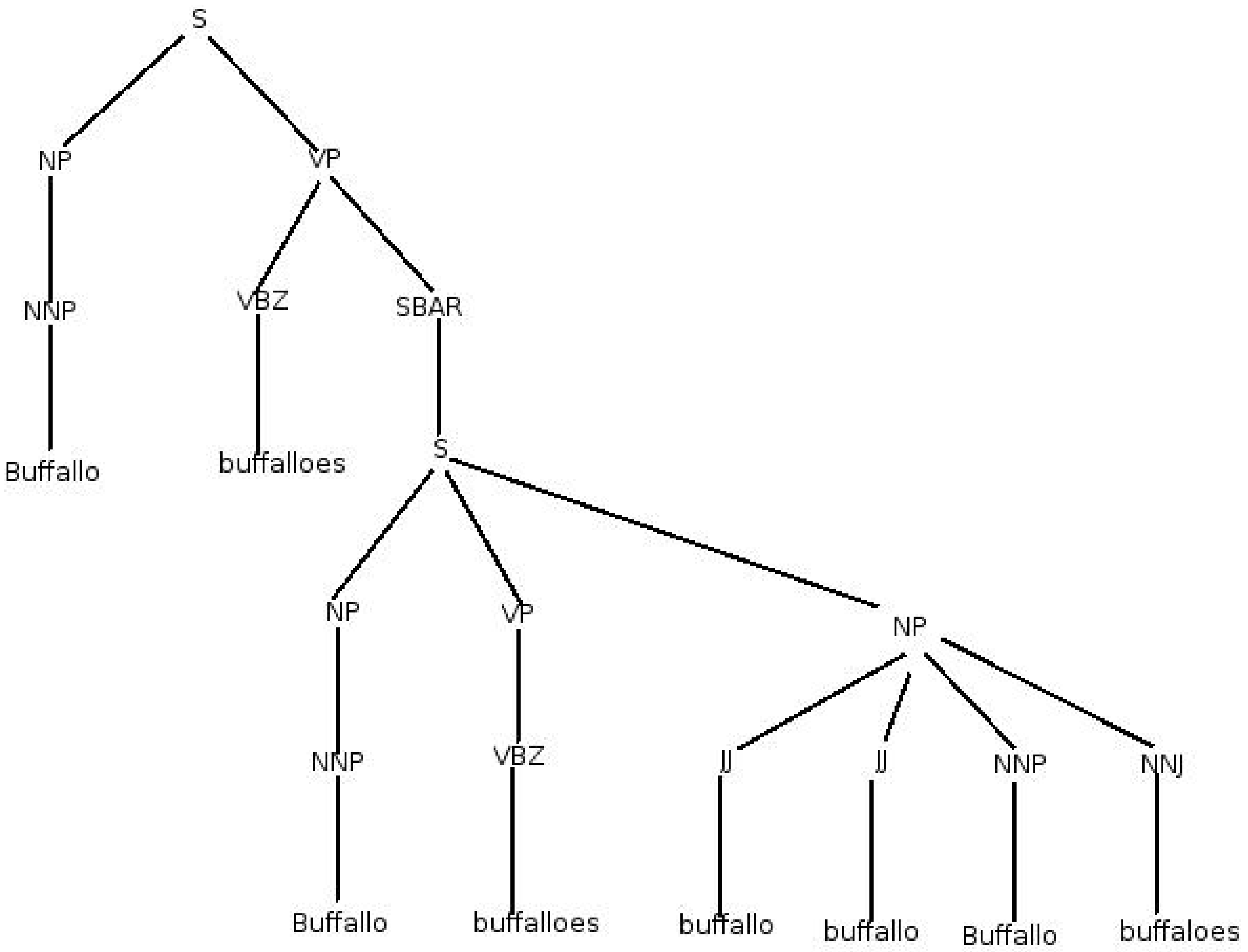


Buffalo buffaloes Buffalo buffaloes buffalo buffalo Buffalo buffaloes

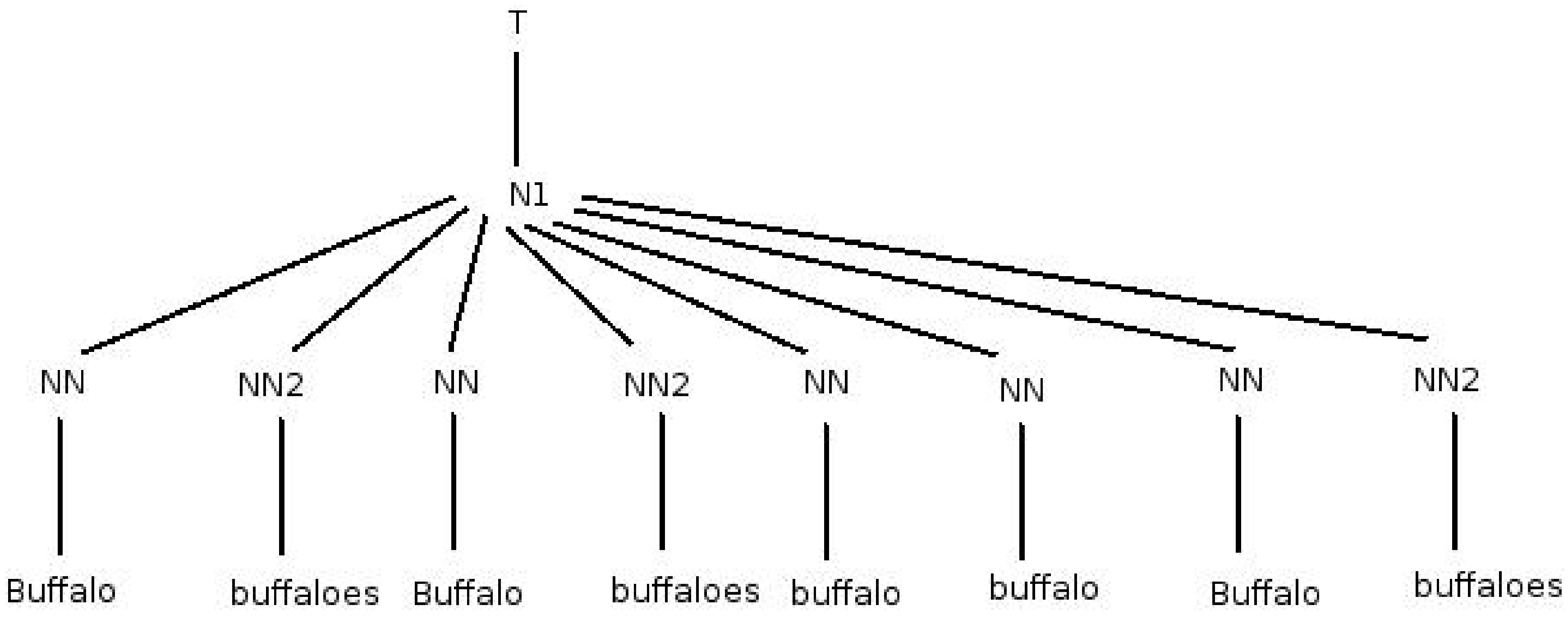
Collins


S
NPB
VP
NNP
NNS
NNP
VBZ
NPB
NPB
Buffalo
buffaloes
JJ
NN
NN
NNS
Buffalo
buffaloes
buffaloes
buffalo
Buffalo
buffalo

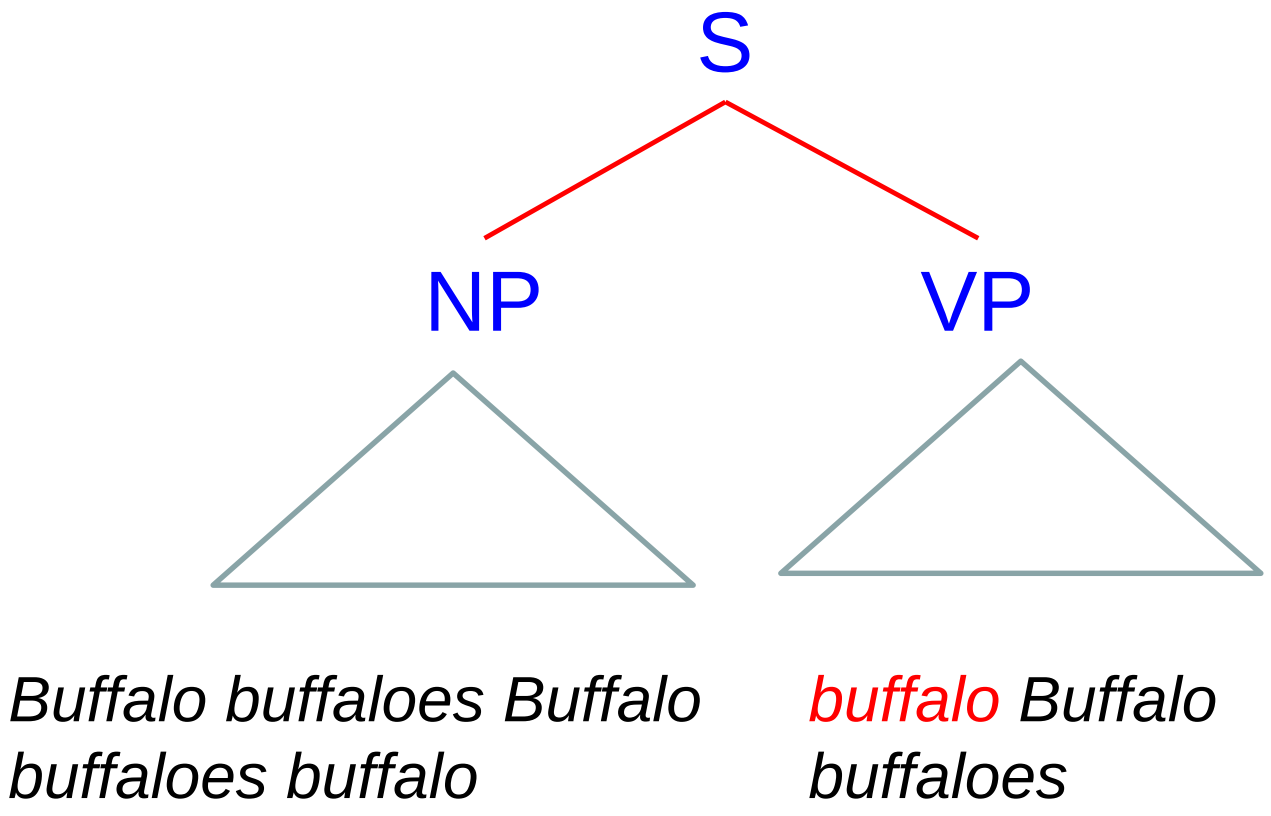
Stanford


S
NP
VP
NNP
VBZ
SBAR
Buffallo
buffalloes
S
NP
VP
NP
NNP
VBZ
JJ
JJ
NNP
NNJ
Buffallo
buffalloes
buffallo
buffallo
Buffallo
buffaloes

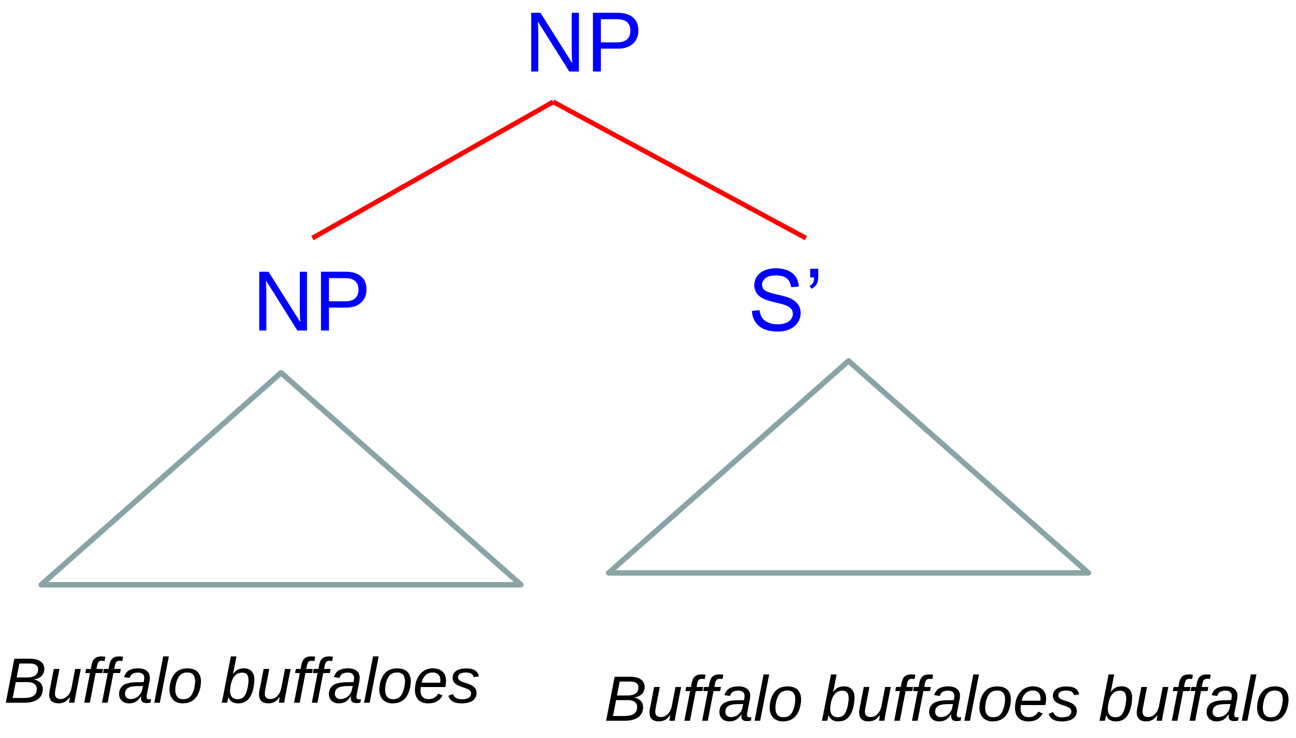
RASP


T
N1
NN
NN2
NN
NN2
NN
NN
NN
NN2
Buffalo
buffaloes
Buffalo
buffaloes
buffalo
buffalo
Buffalo
buffaloes

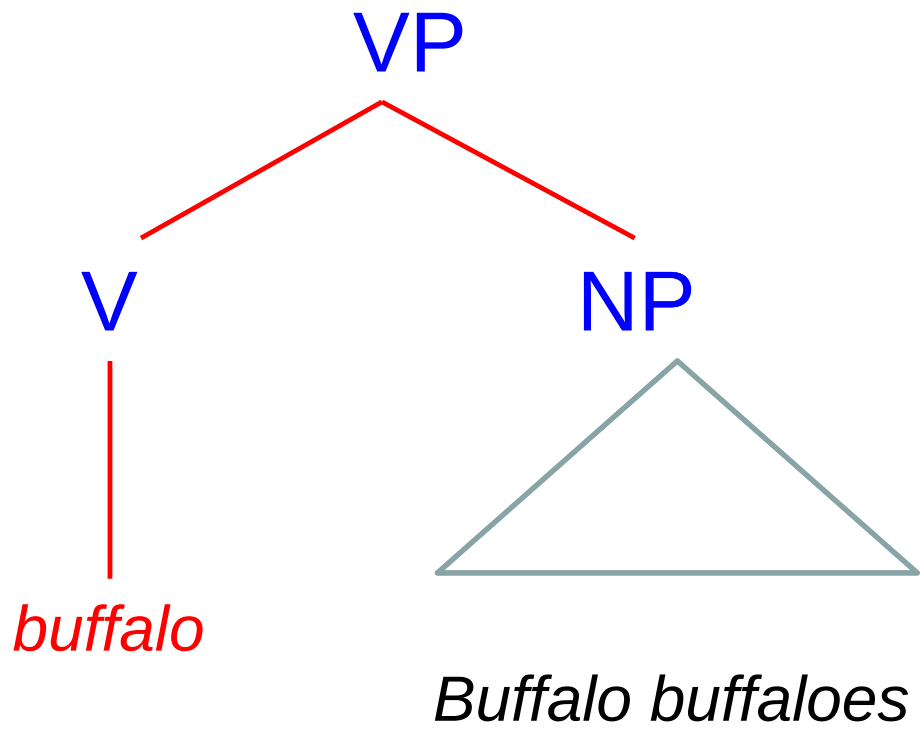
S: Buffalo buffaloes Buffalo buffaloes buffalo buffalo Buffalo buffaloes



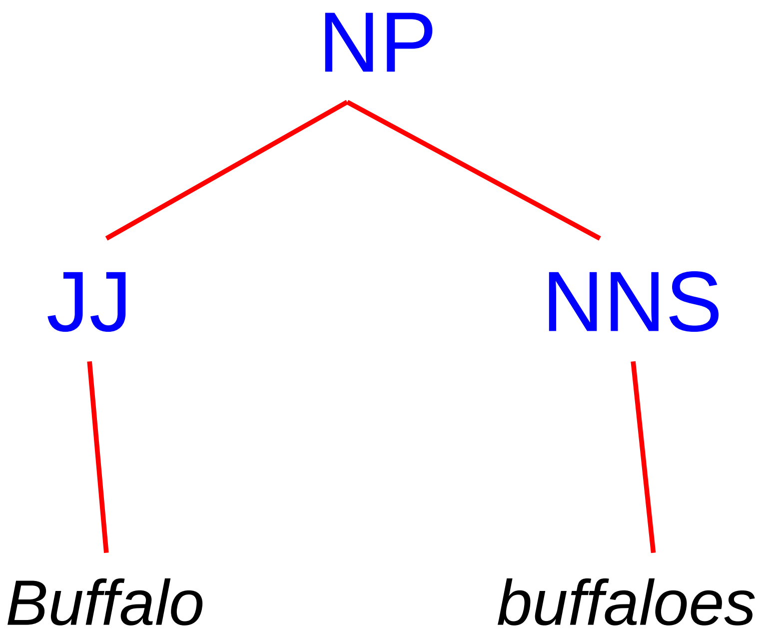
NP: Buffalo buffaloes Buffalo buffaloes buffalo



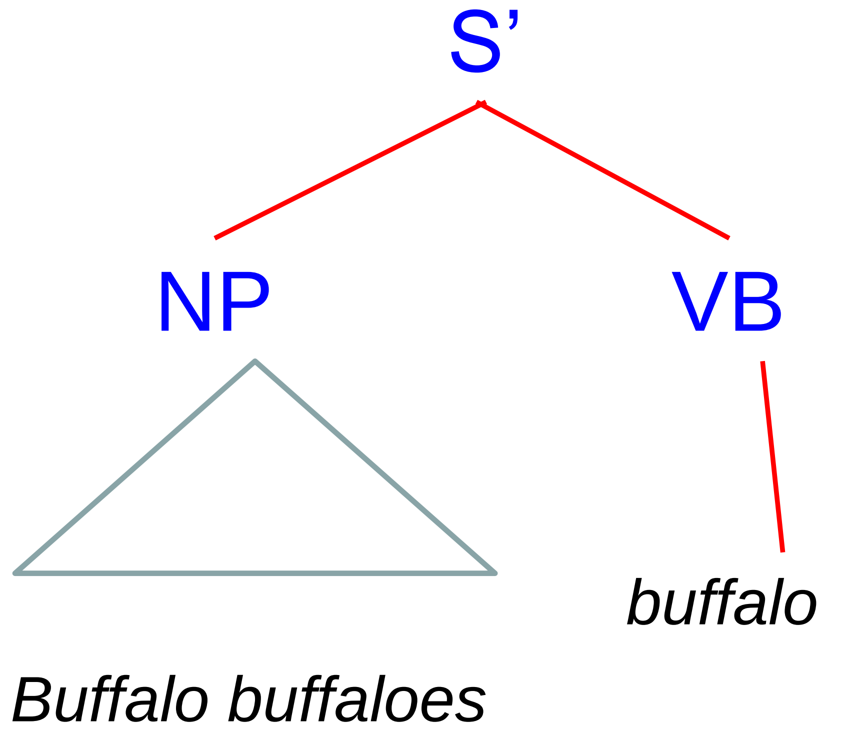
VP: buffalo Buffalo buffaloes



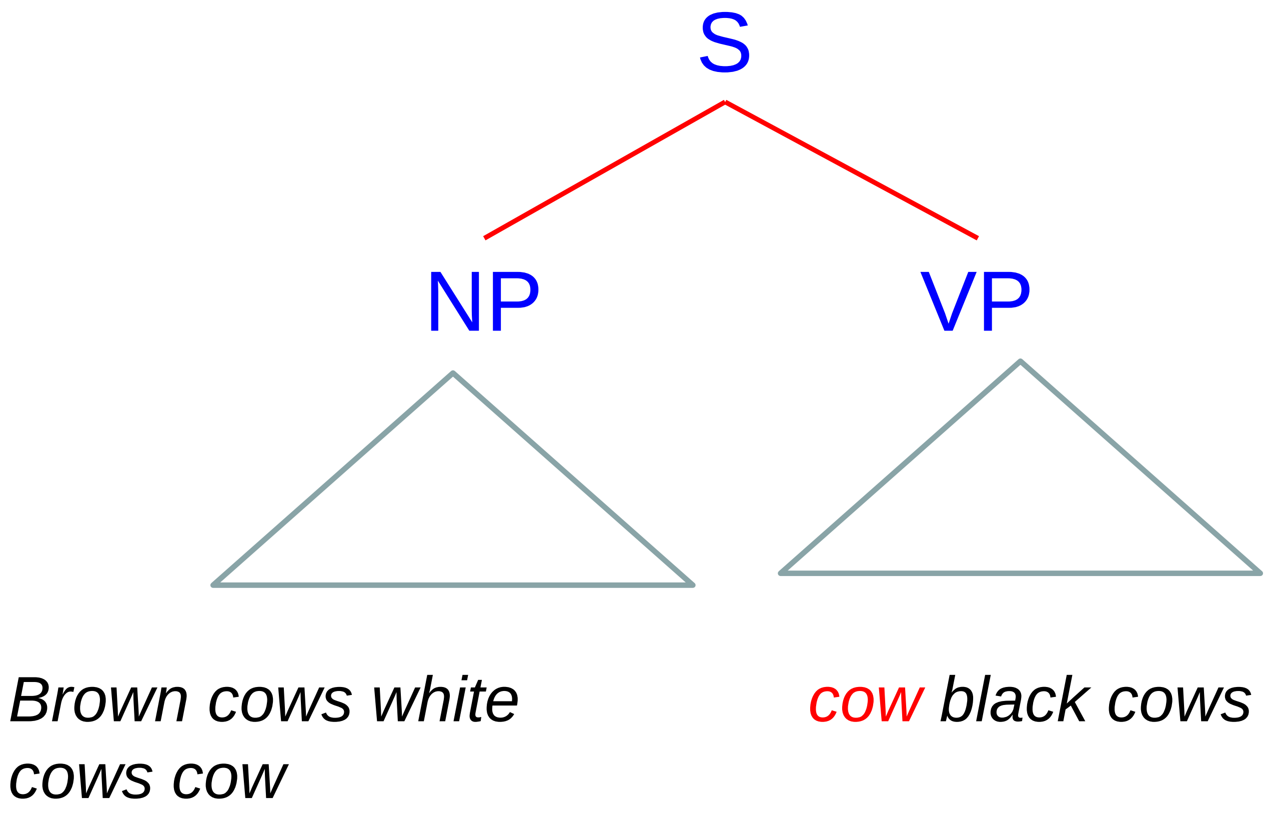
NP: Buffalo buffaloes



S’: Buffalo buffaloes buffalo



Another similar sentence: Brown cows white cows cow cow black cows

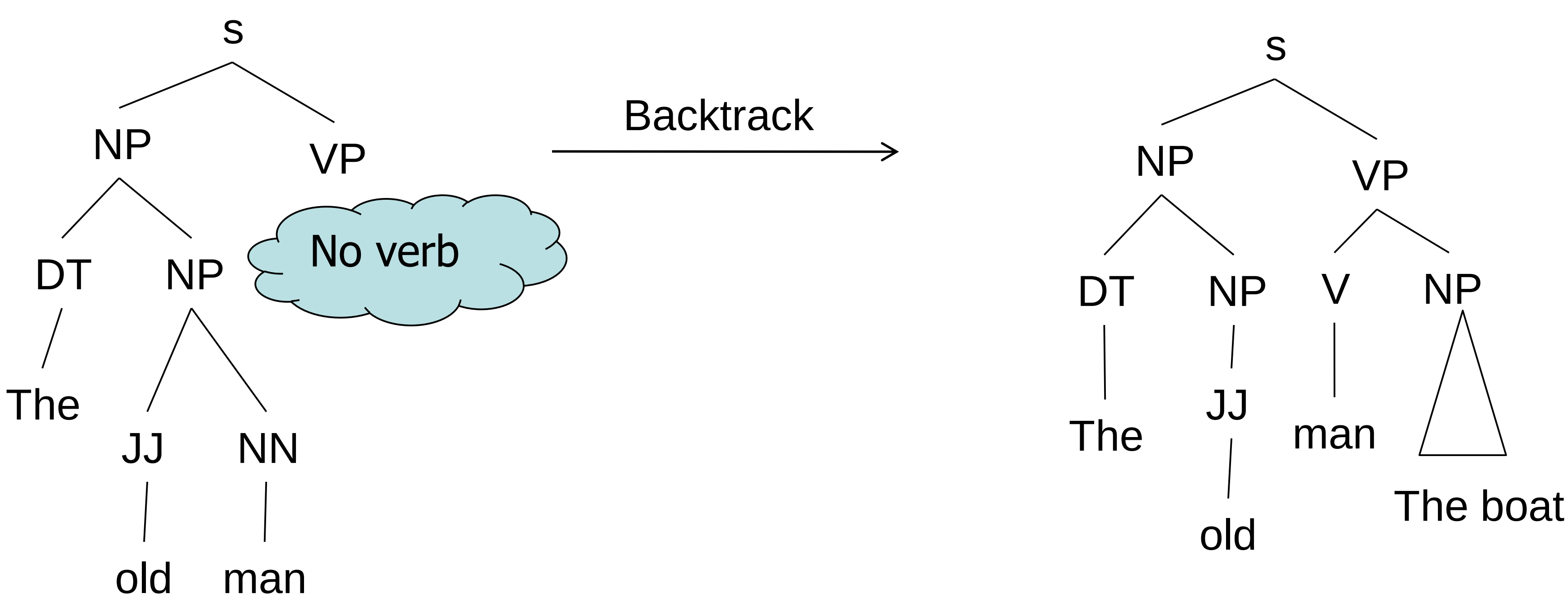


Observation

- Collins and Charniak come close to producing the correct parse.
- RASP tags all the words as nouns.

Another phenomenon: Garden pathing

e.g. The old man the boat.



Another example: The horse raced past the garden fell.

Introducing dependency parsing

Example: raw sentence

The strongest rain shut down the financial hub of Mumbai

(from: Stanford parser https://nlp.stanford.edu/software/lex-parser.shtml)

Example: POS Tagged sentence

The/DT strongest/JJS rain/NN shut/VBD down/RP the/DT financial/JJ hub/NN of/IN Mumbai/NNP

Constituency parse

```
(S
  (NP
      (DT The)
       (JJS strongest)
          (NN rain))
   )

 (VP
     …
)

(VP
  (VP
     (VBD shut)
     (PRT (RP down))
  (NP
     (NP
        (DT the) (JJ financial)
        (NN hub))
     (PP (IN of)
           (NP (NNP Mumbai)))))
```

Dependency Parse

root(ROOT-0, shut-4)

nsubj(shut-4, rain-3)

prt(shut-4, down-5)

det(rain-3, the-1)

amod(rain-3, strongest-2)

dobj(shut-4, hub-8)

det(hub-8, the-6)

amod(hub-8, financial-7)

prep(hub-8, of-9)

pobj(of-9, Mumbai-10)

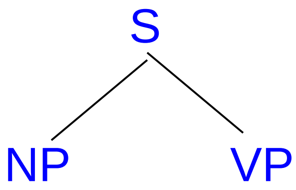
Note: dependency parsing chooses to remain shallow; prepositions are NOT Disambiguated wrt their semantic roles.

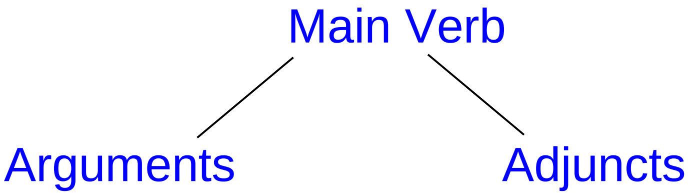
Examples to illustrate difference between DP and Semantic Role Labeling (SRL)

Sentence	Shallow relation from Dependency Parsing	Deeper relation from Semantic Role Labeling
John broke the window	nsubj	Agent
The stone broke the window	nsubj	Instrument
The window broke	nsubj	Object
1947 saw the freedom of India	nsubj	Time
Delhi saw bloodshed when Nadir Shah attacked Delhi	nsubj	Place

Disambiguation is needed to convert shallow DP relations to semantic roles.

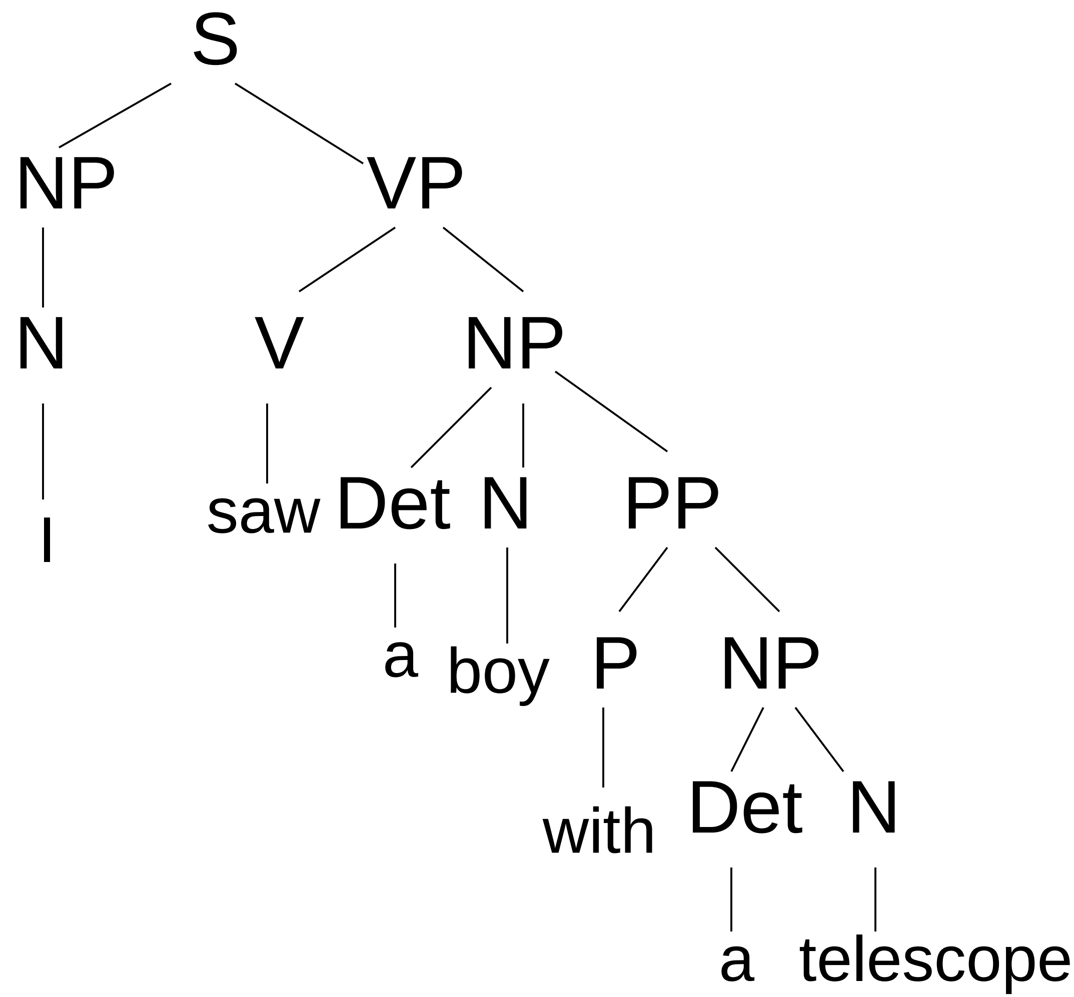
Two kinds of parse representations: Constituency Vs. Dependency



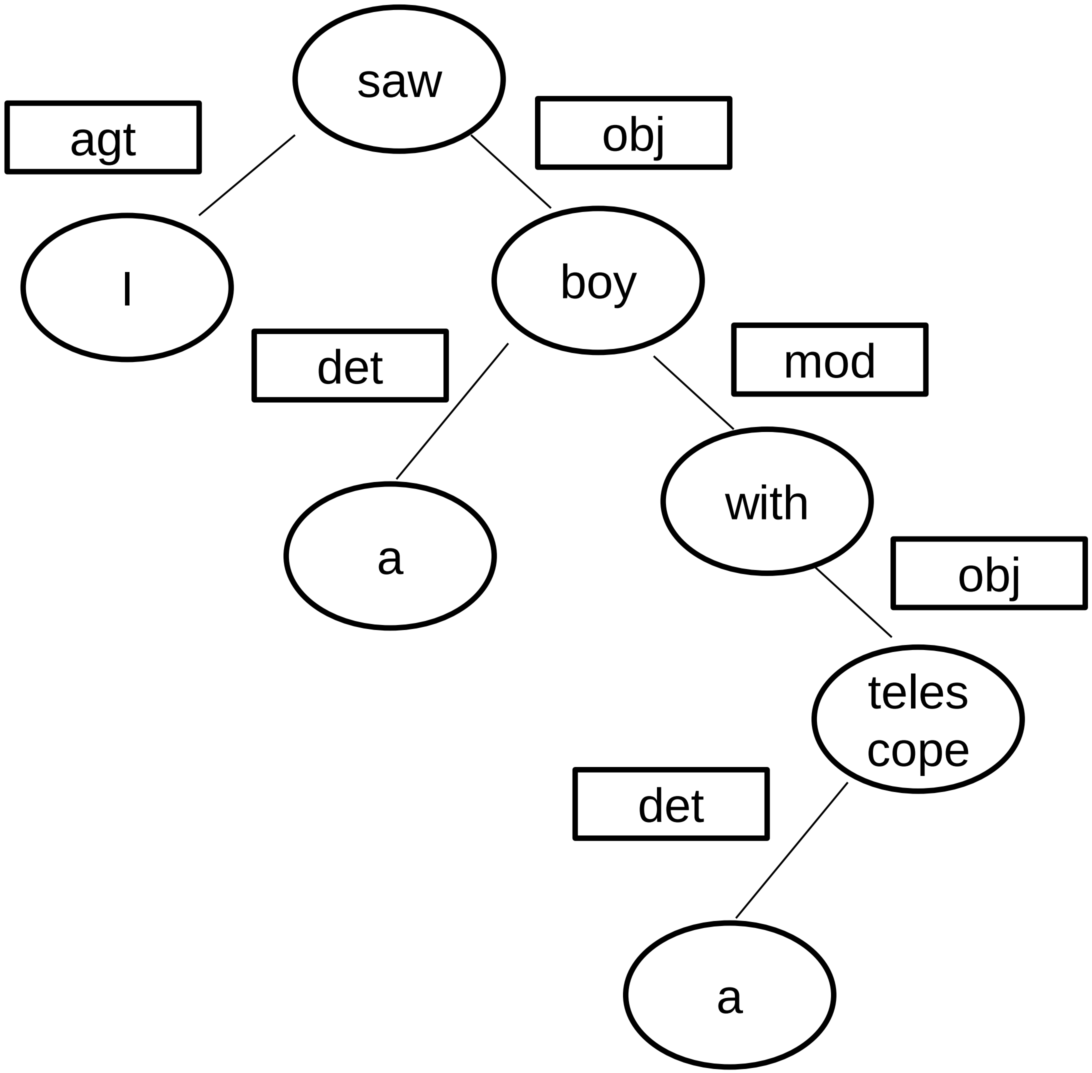


- Penn Constituency Treebank
  - http://www.cis.upenn.edu/~treebank/
- Prague Dependency Treebank
  - http://ufal.mff.cuni.cz/pdt2.0/

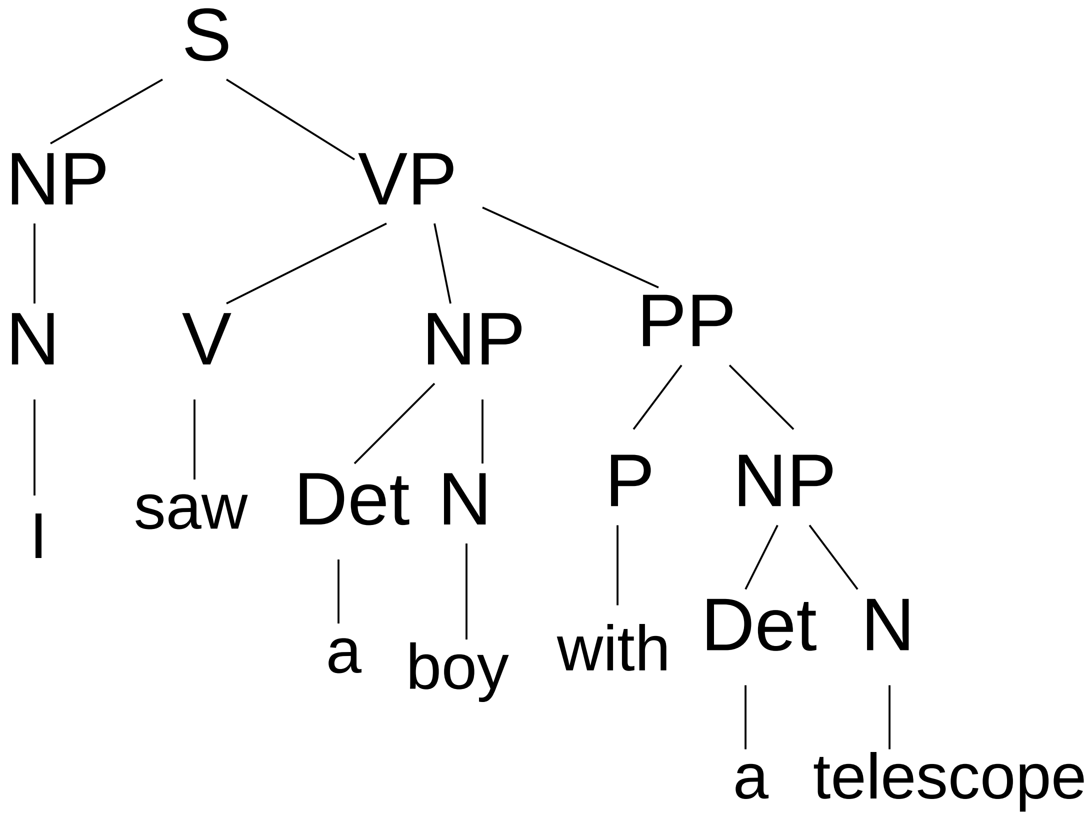
"I saw the boy with a telescope": Constituency parse-1: *telescope with boy*



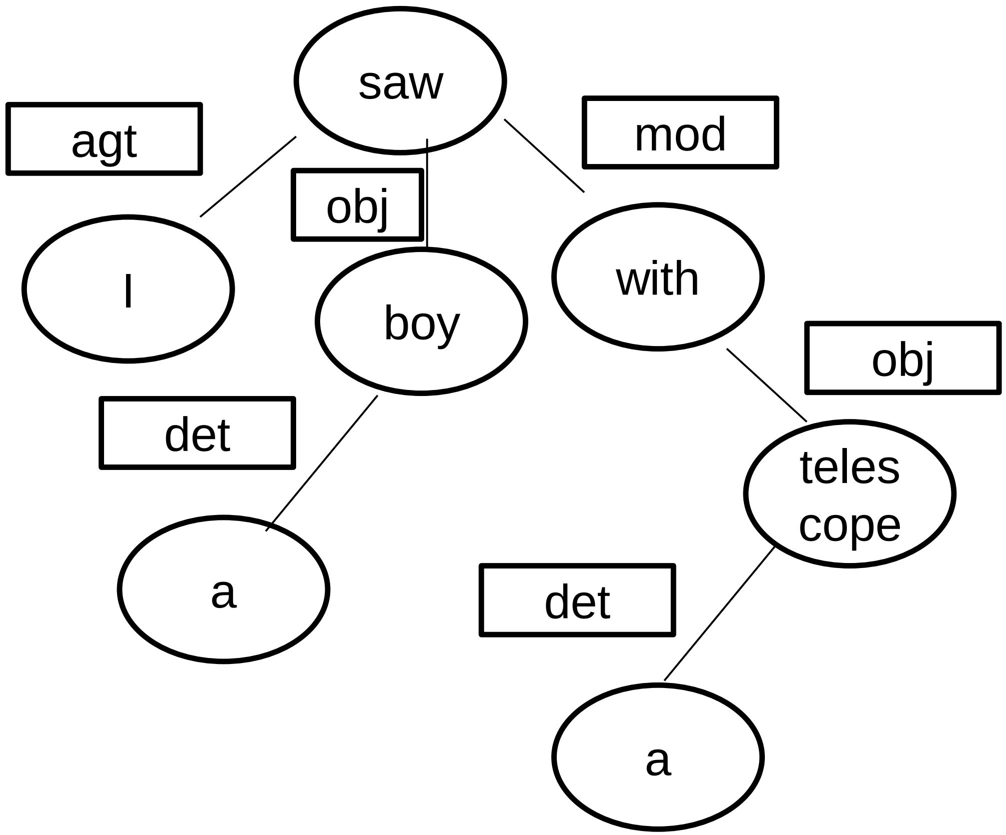
"I saw the boy with a telescope": Dependency Parse Tree-1



Constituency Parse Tree-2: *telescope with me*



Dependency Parse Tree-2



Advantage of DP over CP

- Related entities are closer in DP than in CP: in terms of path length

- Free word order does not affect DP; CP needs additional rules

- Additional rules may overgeneralize!!

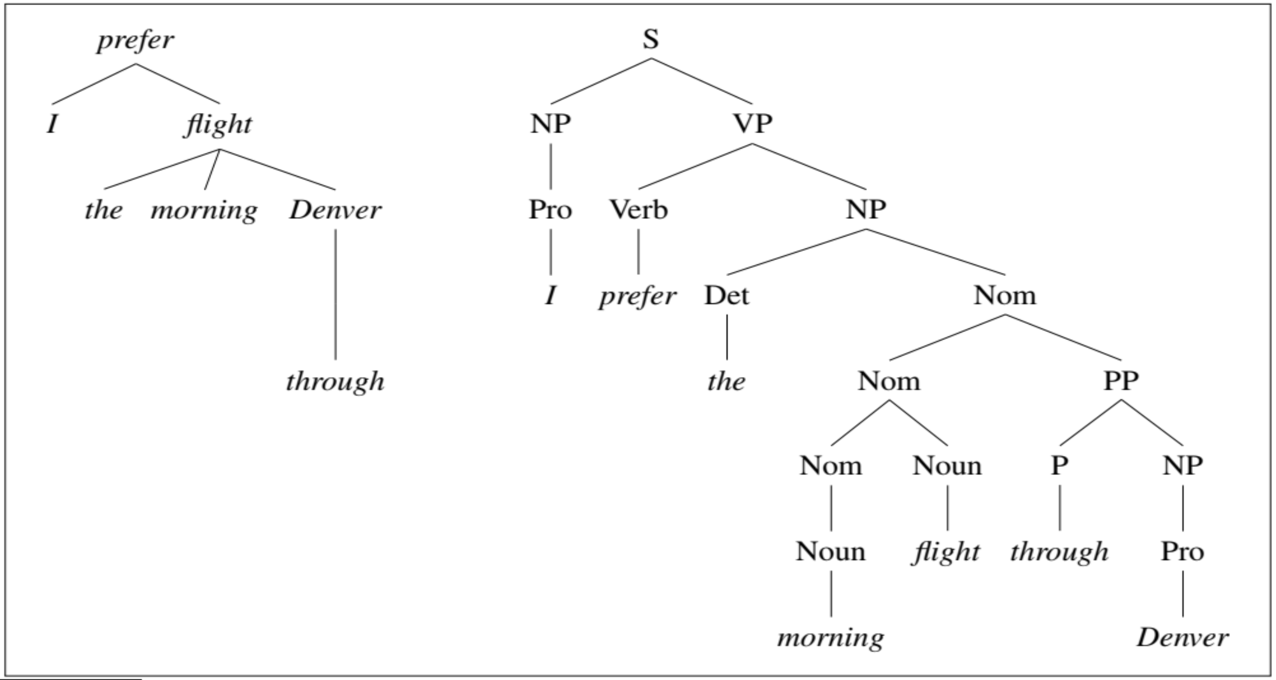
…CP needs additional rules

- *I saw the boy with a telescope*
  - *S → NP VP*
  - *VP → VBD NP PP*
- *With a telescope I saw the boy*
  - *S → NP VP*
  - *S → PP NP VP ???*

Impact of free word order on constituency parsing

- Constituency parse fundamentally use adjacency information.
- Word order disturbs the adjacency
- Chomsky normal form demands that
  - The deduction should happen by linking together two adjacent entities.
- Example:
  - राम ने श्याम को देखा | (Ram ne Shyam ko dekha)
    - श्याम को देखा =VP
  - श्याम को राम ने देखा | (Shyam ko Ram ne dekha)
    - VP is discontinuous
    - Constituency parsing fails here
  - The agent and object is reversed in the above example.
  - CP needs additional rules

Arguments are immediately linked



J & M, Chapter 15, 3rd Edition

Prefer: who prefers? “I”; what is preferred?: “flight”.

On the other hand, phrases are like *suitcases* that put all related things **at one place**: “The morning flight through Denver”

Subset of Dependency Relations: from Universal Dependency Project (Nivre et all 2016)

Clausal Argument Relations	Description
NSUBJ	Nominal subject
DOBJ	Direct object
IOBJ	Indirect object
CCOMP	Clausal complement
XCOMP	Open clausal complement
Nominal Modifier Relations	**Description**
NMOD	Nominal modifier
AMOD	Adjectival modifier
NUMMOD	Numeric modifier
APPOS	Appositional modifier
DET	Determiner
CASE	Prepositions, postpositions and other case markers
Other Notable Relations	**Description**
CONJ	Conjunct
CC	Coordinating conjunction

Examples to illustrate Dependency Relations

- NSUBJ, DOBJ, IOBJ- "*Ram gave a book to Shyam*"
  - Main Verb (MV): *gave*
  - NSUBJ: *Ram;* DOBJ: *book;* IOBJ: *Shyam*
- CCOMP, XCOMP: "I said that he should go", "I told him to go"
  - CCOMP: *said*$\rightarrow$*go*
  - XCOMP: *told*$\rightarrow$*go*

A note on CCOMP and XCOMP

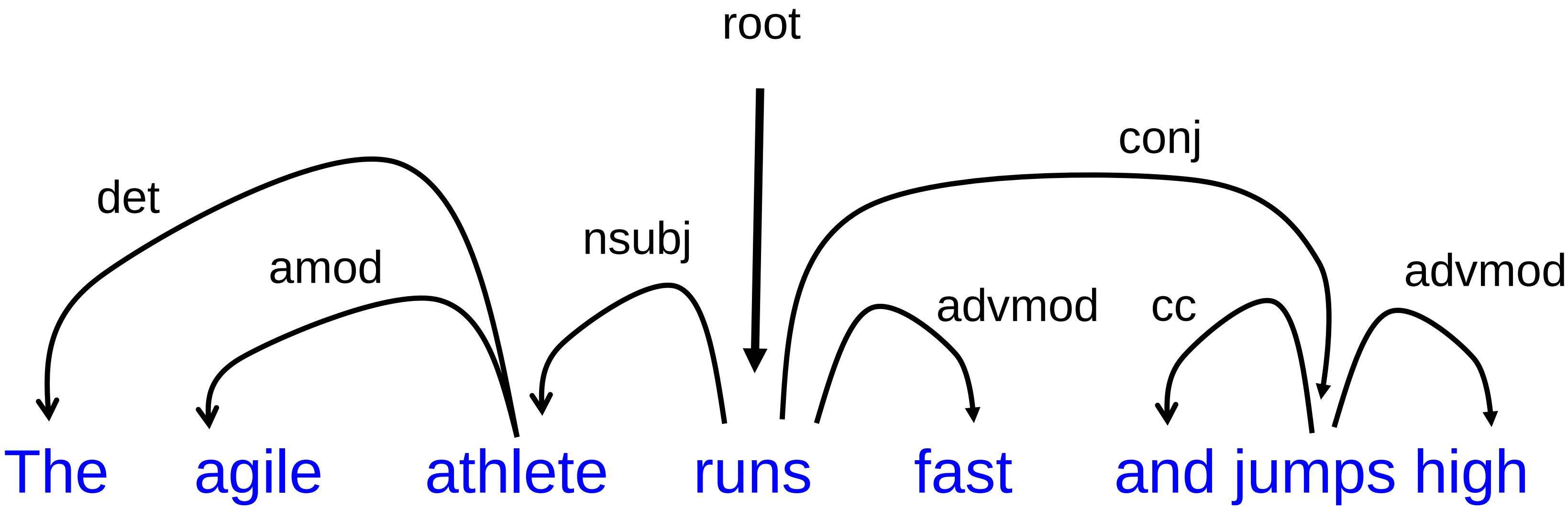
- CCOMP links the main verb with the finite verb
- XCOMP links main verb with an infinite verb
- Finite verb means: “takes GNPTAM marking”
- Infinite verb: remains in lemma form
- E.g. “told him to *go”: ‘*go’ will not change form (infinite form)
- “said he should go/be_going”: ‘go’ can change form

Illustration of DRs cntd.

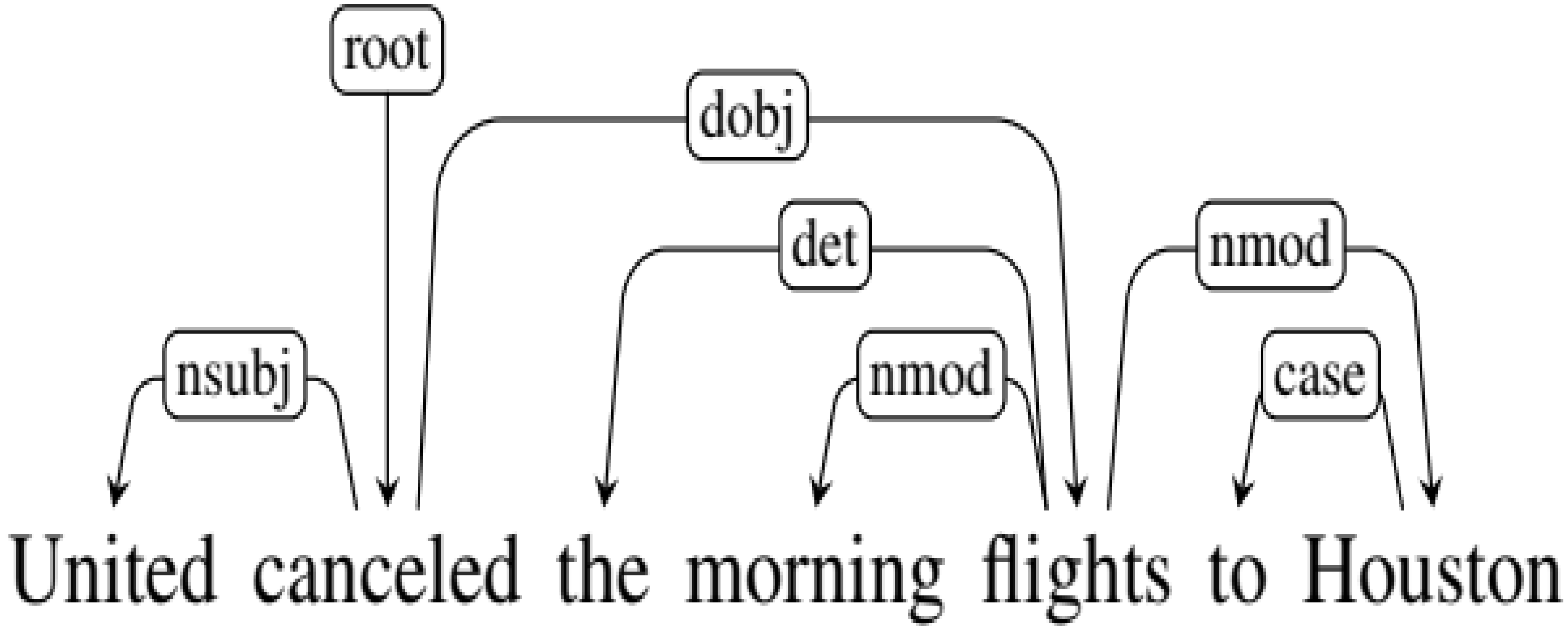
- NMOD (nominal modifier), AMOD (adjective modifier), NUMMOD (numerical modifier), APPOS (appositional modifier)
  - NMOD: *The bungalow of the Director: Director* ← *bungalow*
  - AMOD: *The large bungalow: large* ← *bungalow*
  - NUMMOD: *Three cups: three* ← *cups*
  - APPOS: *covid19, the pandemic: covid19* ← *pandemic*

Illustration of DRs cntd.

- DET (determiner), CASE (preposition, postposition and other case markers), CONJ (conjunct), CC (coordinating conjuct)
  - DET: *The bungalow: The→bungalow*
  - CASE: *The bungalow of Director: of→Director*
  - CONJ: *He is sincere and honest: sincere→honest*
  - CC: *He is sincere and honest: honest→and*


root
conj
det
nsubj
amod
advmod
advmod
cc
The
agile
athlete
runs
fast
and jumps high

Head → Modifier, e.g., *morning → flight*



Dependency Tree

- (1) There is a single designated root node that has no incoming arcs.
- (2) With the exception of the root node, each vertex has exactly one incoming arc.
- (3). There is a unique path from the root node to each vertex in *V*.