Natural Language Processing
自然語言處理

黃瀚萱
Department of Computer Science
National Chengchi University
2020 Fall

Parsing

Schedule

Date	Topic
9/16	Introduction
9/23	Linguistic Essentials
9/30	Collocation
10/7	Language Model
10/14	Performance Evaluation and Word Sense Disambiguation
10/21	Text Classification (HW1 will be assigned)
10/28	Invited Talk: NLP and Cybersecurity (Term Project)
11/4	POS Tagging
11/11	Midterm Exam

Schedule

Date	Topic
11/18	Chinese Word Segmentation
11/25	Word Embeddings
12/2	Neural Networks for NLP
12/9	Semi-supervised Learning
12/16	Discussion about your Final Project
12/23	Invited Talk
12/30	**Syntactic & Discourse Parsing**
1/6	**Final Project Presentation II**
1/13	Final Exam

Agenda

- Linguistic structure
  - Constituency parsing
  - Dependency parsing
  - Discourse parsing
- Approaches to parsing
  - CKY parsing
  - Shift-reduce parsing
- Resources

Syntactic Parsing

Structure of Wording

- Human beings write/say different kinds of things in different ways, but the different expressions have some structure and regularity.

- To isolate the structure in writing/conversation:
  - **Syntax**（語法）

- Describing the ordering and arrangement of words in
  - Words
  - Word categories

Word Order: Syntax

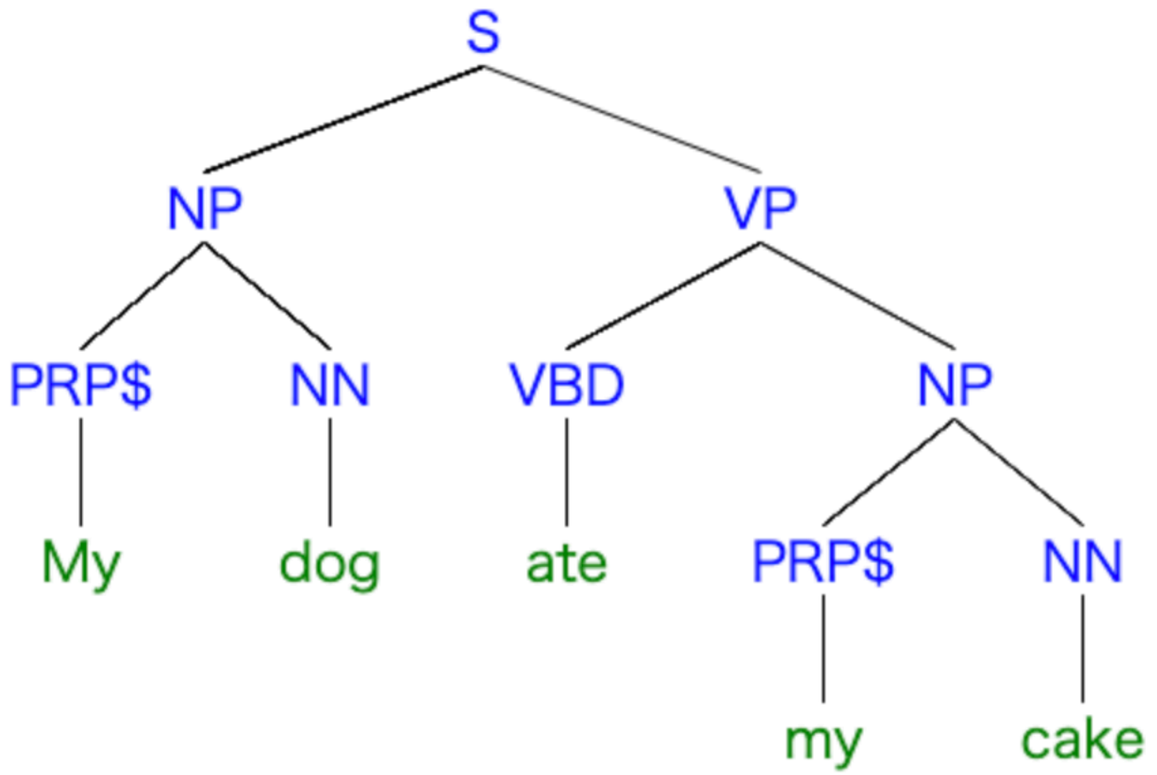
- Languages have constraints on word order.
  - the dog ate my cake
  - ate cake dog my
  - the cake ate my dog
- A sequence of words without a proper order results an uninterpretable sentence.
- Syntax: the study of the regularities and constraints of word order and phrase structure.

Syntax vs N-grams/POS sequence

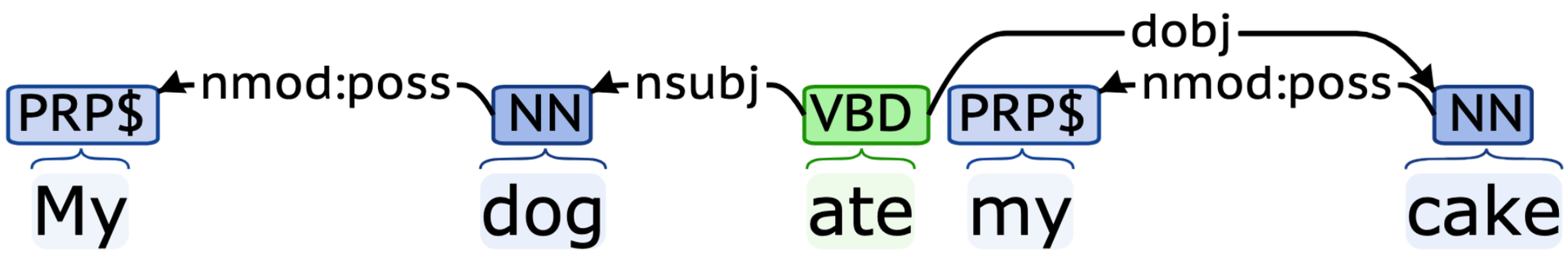
- Both attempt to capture the word order information
- Ngrams and POS sequences capture the ordering and the arrangement in the **linear** form
    - My dog ate my cake => My dog, dog ate, ate my, my cake
    - PRP$ NN VBD PRP$ NN
- Syntax capture the information in a **tree structure**
    - More than just linear order

Two Kinds of Structures

- Constituency parsing

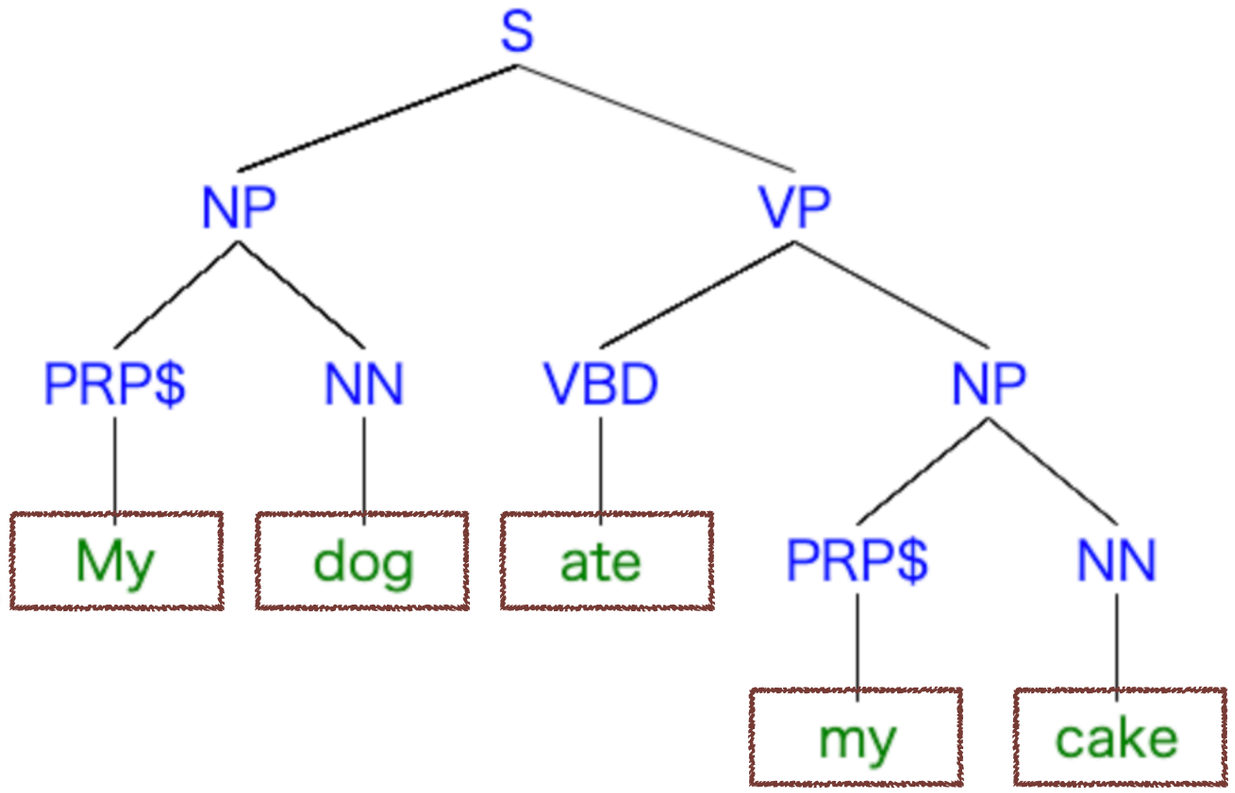


- Dependency parsing



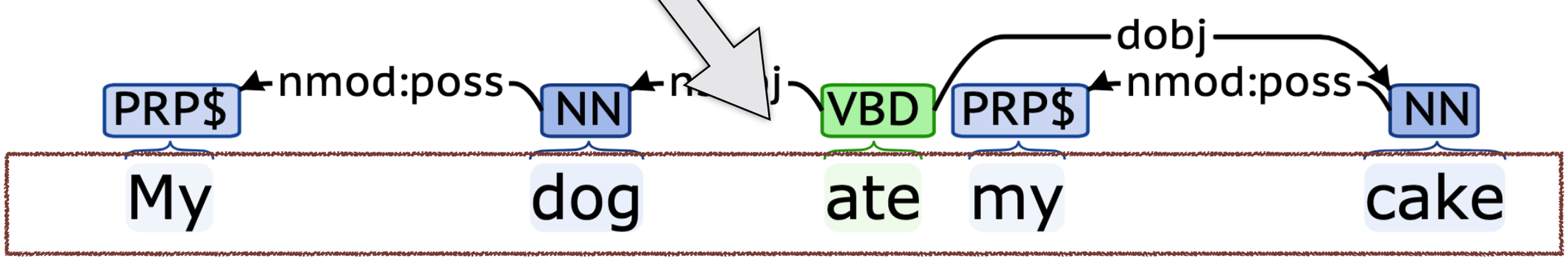
Two Kinds of Structures

- Constituency parsing



Words / lexicon

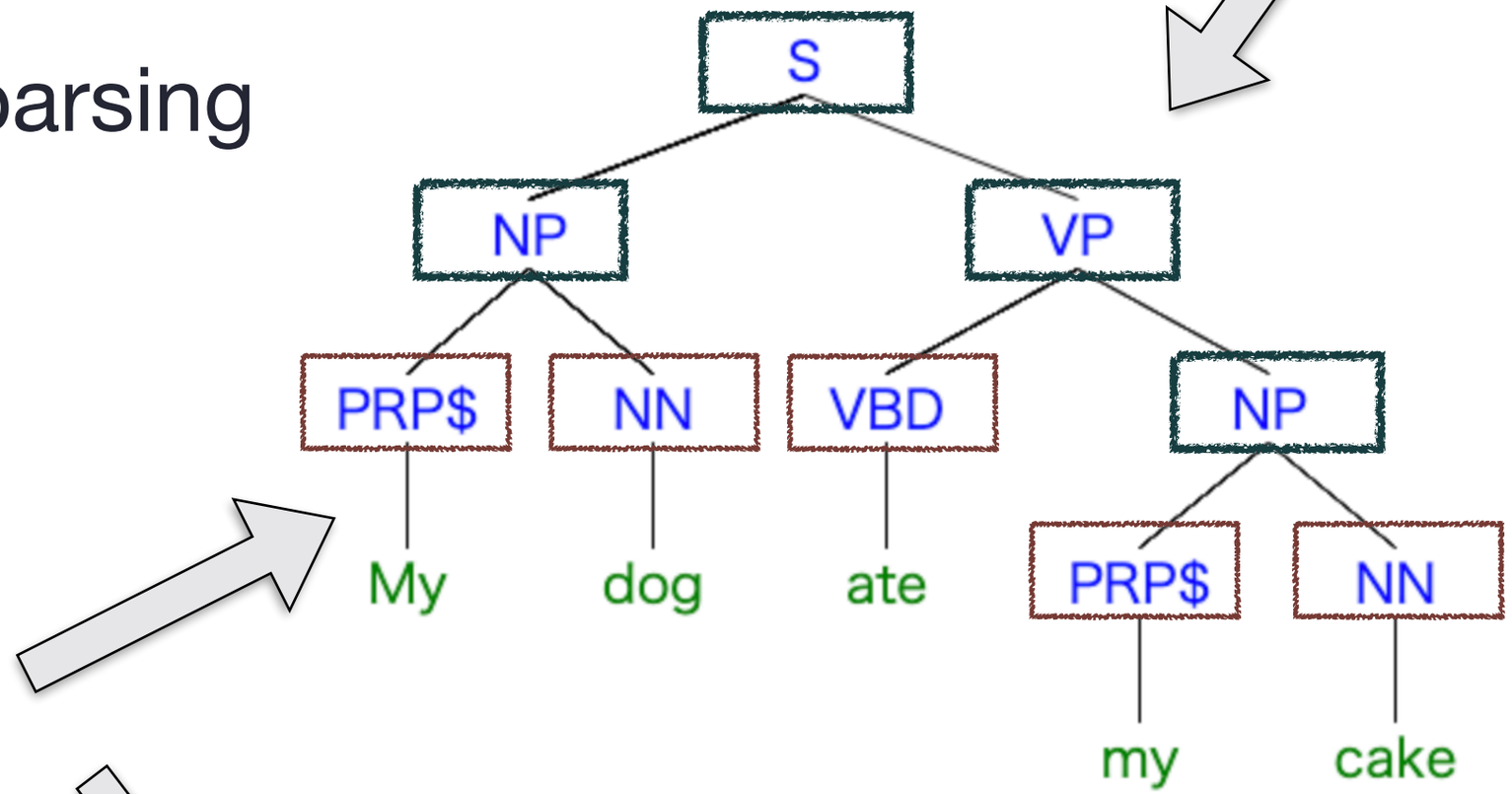
- Dependency parsing



Two Kinds of Structures

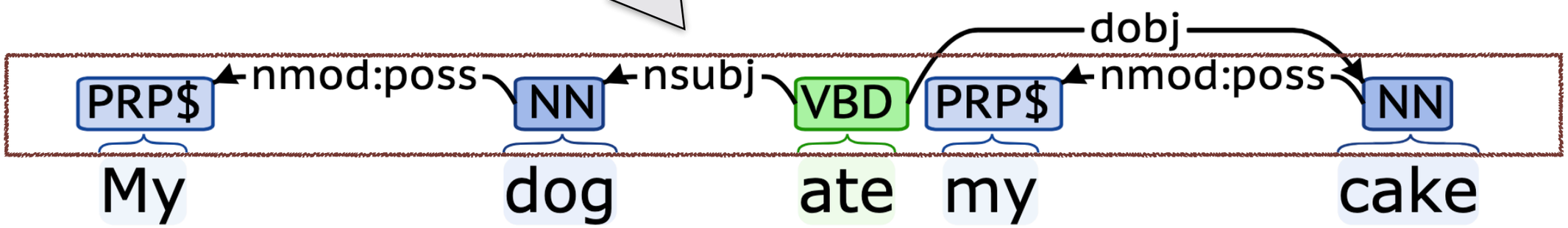
Units larger than POS

- Constituency parsing



Phrase types

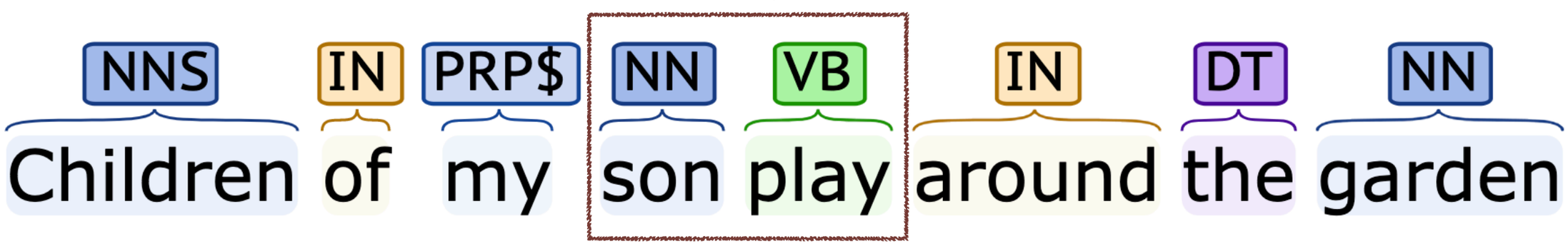
- Dependency parsing



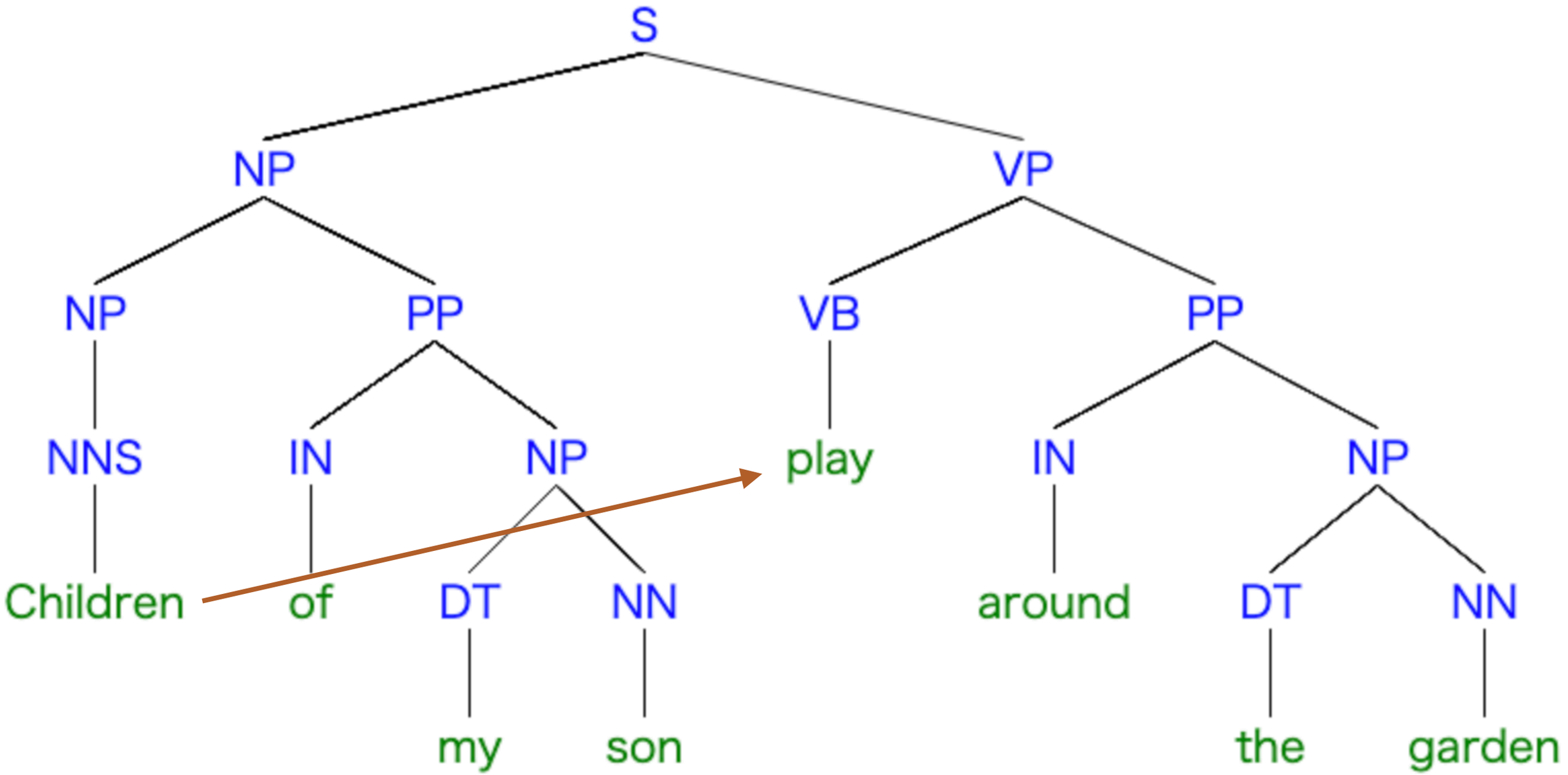
Limitation of the Linear Order

- Children of my son play around the garden
  - children of
  - of my
  - my son
  - **son play**
  - play around
  - around the
  - the garden

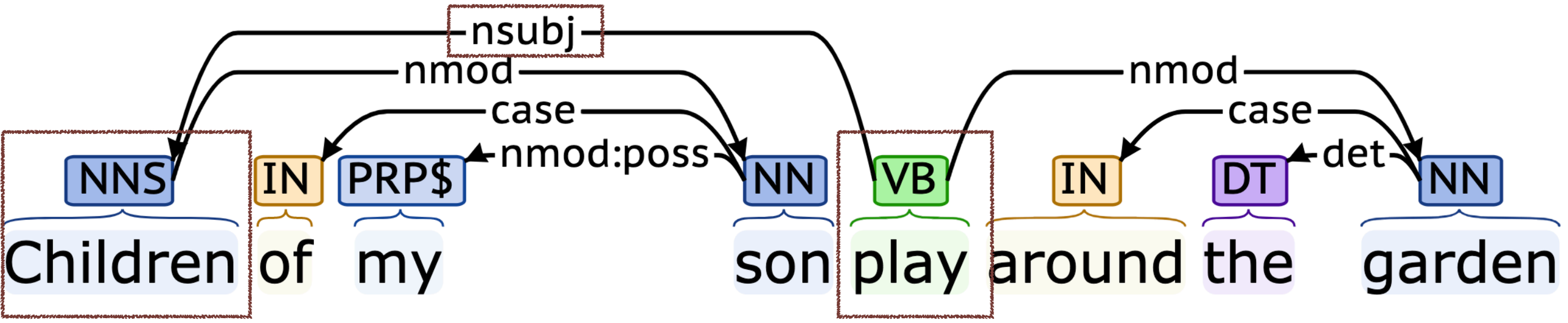
Singular nouns with a verb for non-singular noun



Advantage of Tree Structure

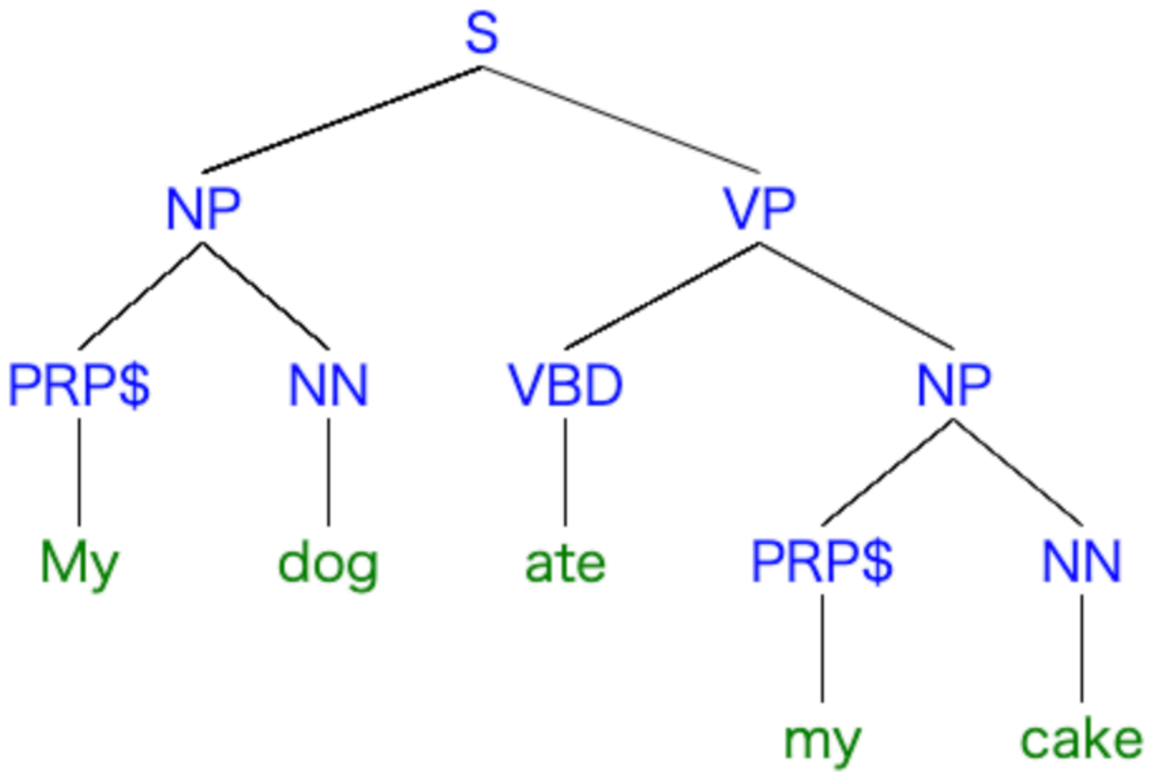


Advantage of Tree Structure



Constituent（結構成份）

- A word or a group of words that functions as a single unit within a hierarchical structure.
- A unit that can appear in different places



S → NP VP

NP → NN

NP → PRP$ NN

VP → VBD NP

Context-Free Grammar (上下文無關文法)

- $G = (T, N, start, R)$
  - T: $\{w^k\}$: A set of terminal symbols (words)
  - N: $\{N^i\}$: A set of non-terminal symbols (NP, VP, etc.)
  - start: N^1, The start symbol (S)
  - R: A set of rules/productions that convert N^i to y
    - y is a number of members of $\{N^i\}$ or $\{w^k\}$

Example of Context Free Grammar

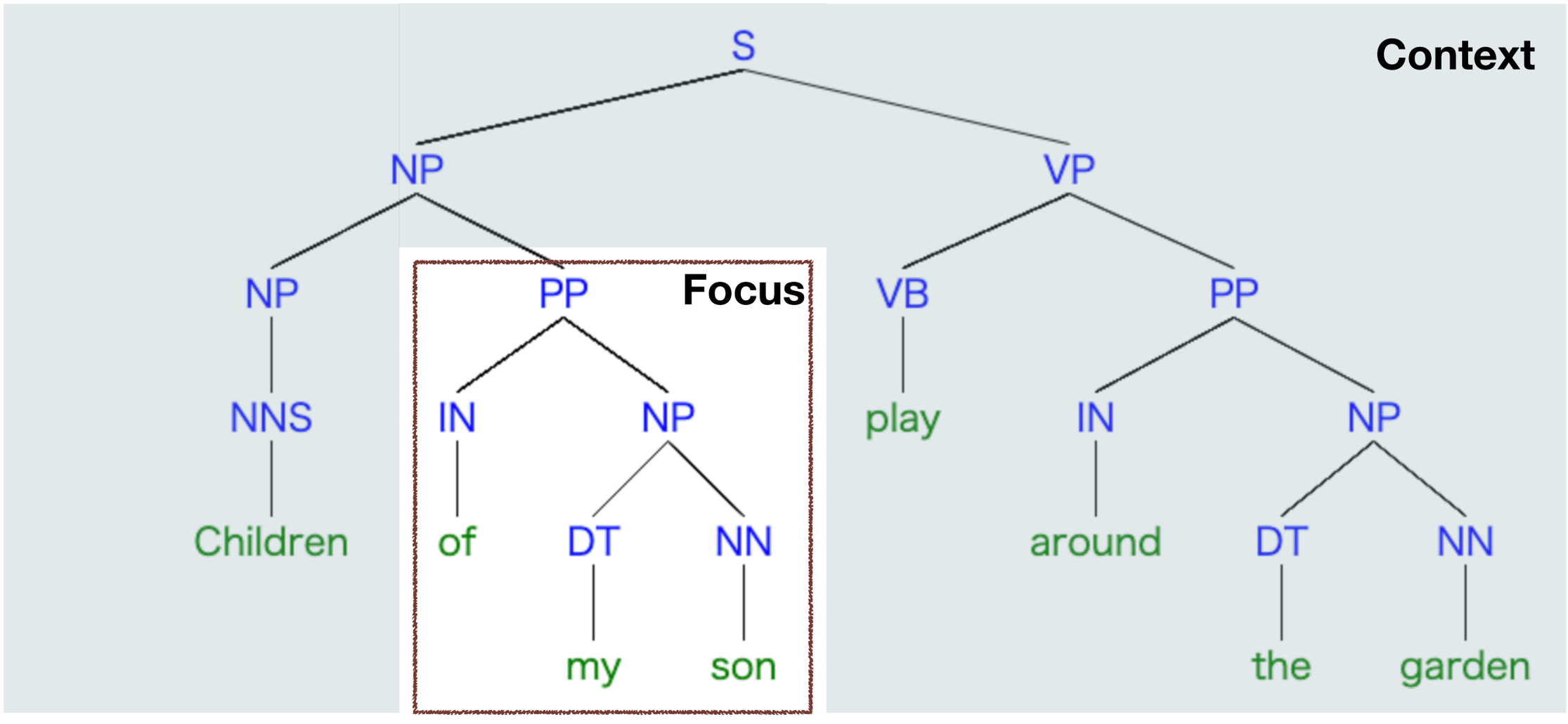
- $G = (\{w^k\}, \{N^i\}, S, R)$
- $\{w^k\}$ = ate | cake | dog | cat | my
- $\{N^i\}$ = VBD | NN | PRP$ | NP | VP | S
- N^1 = S
- R

S	NP VP
VP	VBD NP
NP	PRP$ N

VBD	ate
NN	dog
NN	cat
NN	cake
PRP$	my

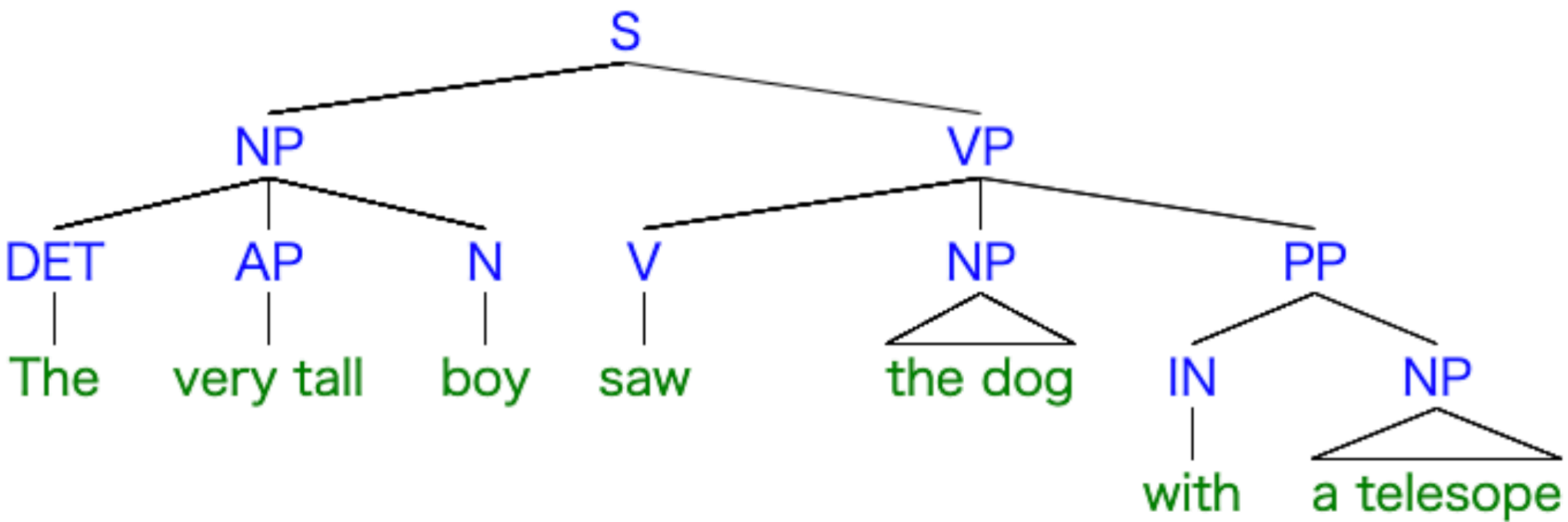
Context Free

- Break down large unit into a number of independent smaller units.
- Dealing with the focused unit without the dependency of its surrounding context.



Major Phrase Types

- S: whole sentence (S → NP VP)
- NP: noun phrase (名詞片語)
- PP: prepositional phrase (介繫詞片語)
- VP: verb phrase (動詞片語)
- AP: adjective phrase (形容詞片語)



Noun Phrases (NPs)

- A syntactic unit of the sentence, in which information about the noun is put together.

- The noun is the **head** of a noun phrase.

- NPs are often arguments of verbs.

  - Subject of an action

  - Object of an action

NP → N
NP → DET N
NP → DET AP N
NP → NP PP
…

Verb Phrases (VPs)

- Organizing all elements of the sentence that grammatically depend on the verb.
- The verb is the head of a verb phrase.
- The core of a sentence.

VP → V
VP → V NP
VP → V NP NP
VP → V NP AP
VP → VP PP
…

Adjective Phrases (APs)

- Complex adjective phrases are rare.

  *He is **pretty sure of himself**.*
  *She seemed a girl who was **quite certain to succeed**.*

AP → JJ (Adjective)
AP → AD (Adverb) JJ
AP → AP AP
…

Prepositional Phrases (PPs)

- Co-occur with NP, VP, and AP.
- PP → IN (Preposition) NP
- Usually used to express spatial and temporal locations and other attributes.

The boy saw the dog ***with a telescope****.*

I joined the conference held ***in the last year****.*

The cup ***on the table*** *is hers.*

Rewrite Rules

- The regularities of word order are formulated by rewrite rules.
- The unit in the left side can be rewritten as the sequence of units in the right side.

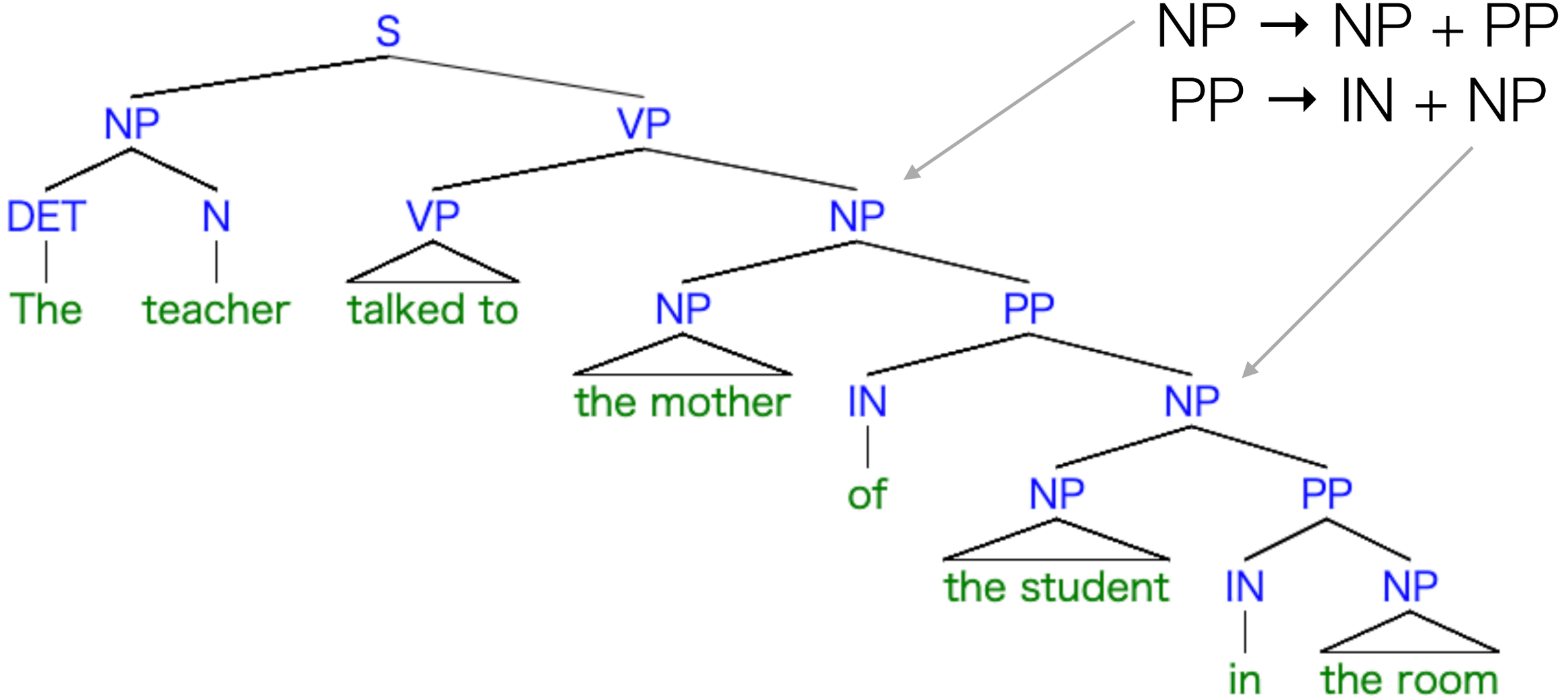
S → NP VP
NP → DET NN
NP → NP PP
VP → VP PP
VP → V NP
PP → IN NP

DET → the | a | an
NN → cat | mouse | egg
V → ate | slept | ran
IN → in | of | on

the mouse ate a cat on an egg **(valid)**
the cat ate a mouse ***and an egg*** **(invalid)**

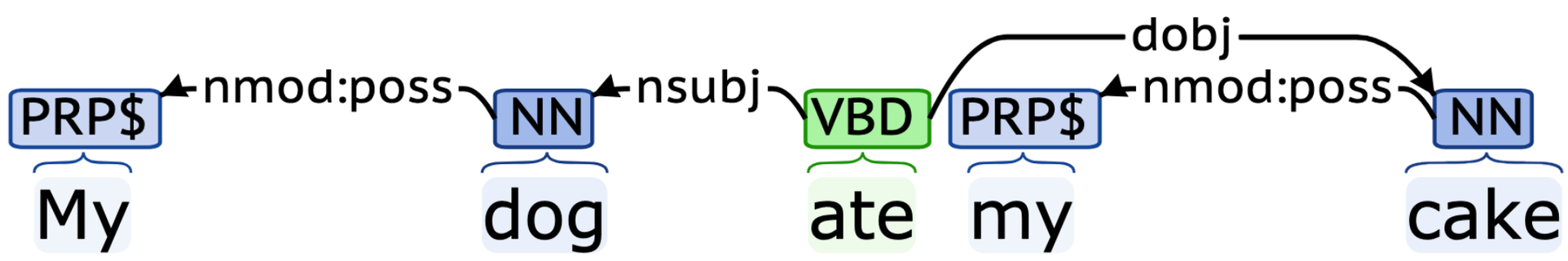
Recursive Constituent Structure

- Rewrite rules can be applied a number of times.
- VP and NP can be expanded to a large number of words.



Dependency Structure

- The dependency between words
- A sentence with *n* words will generate *n* triples:
  - (*relation*, *head*, *dependent*)

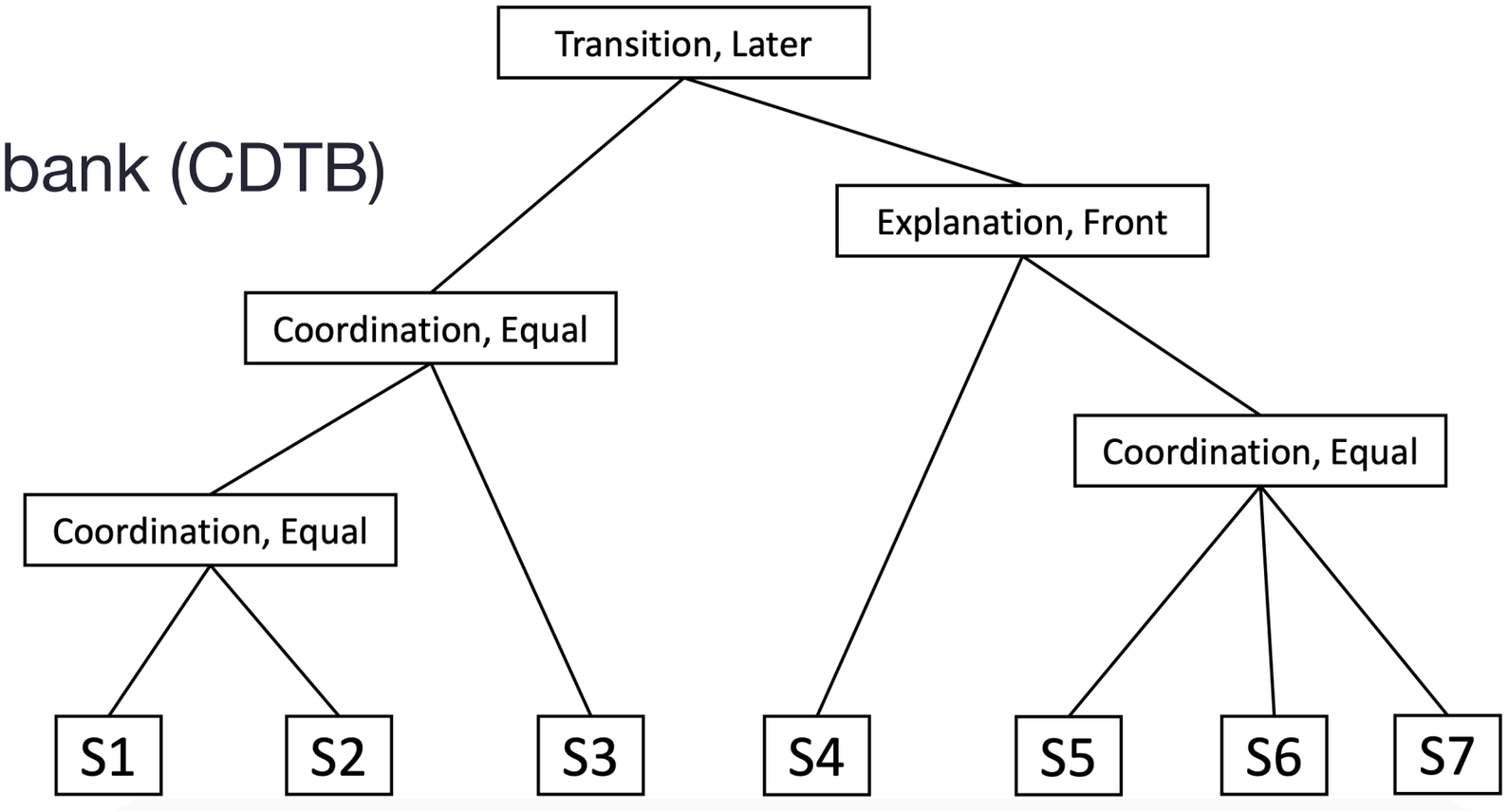


- (root, S, ate) (nmod:poss, dog, my) (nsubj, ate, dog) (dobj, ate, cake) (nmod:poss, cake, my)

Discourse Parsing

Paragraph-level Discourse Parsing

- Chinese Discourse Treebank (CDTB)
  - 4 rhetorical relations
  - Nuclearity labels

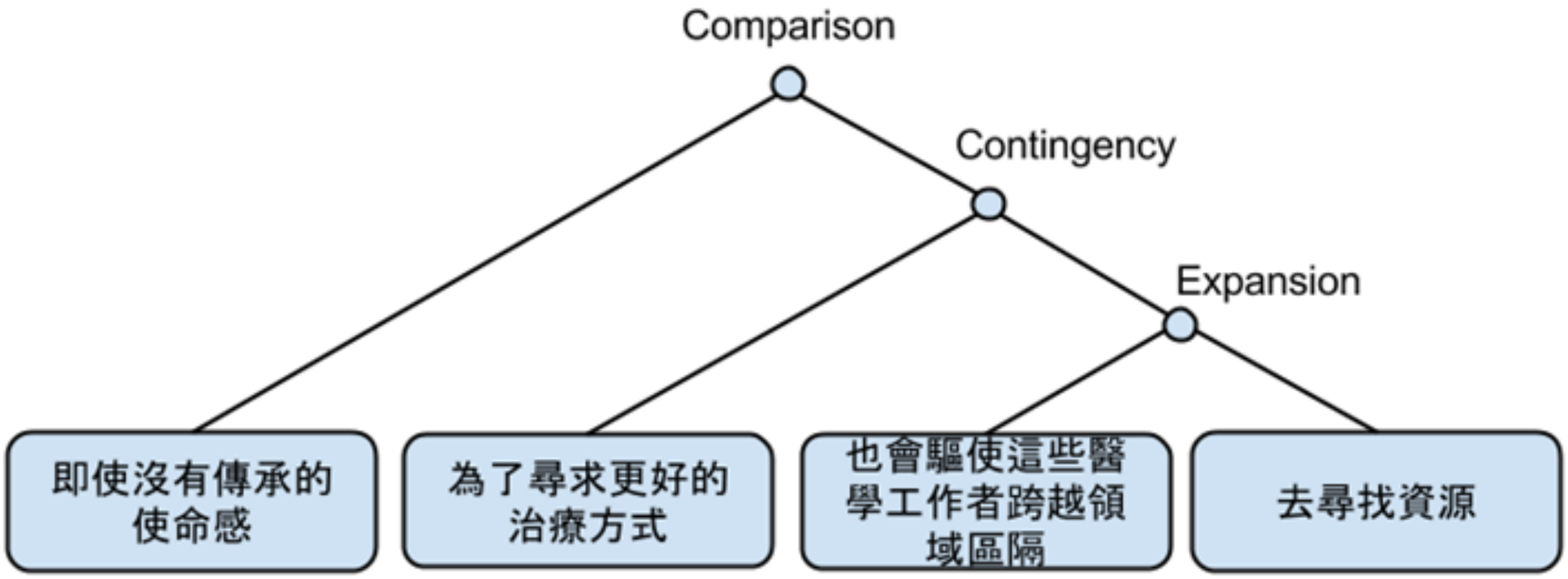


1.儘管浦東新區製定的法規性文件有些比較“粗”
2.有些還只是臨時行規定
3.有待在實踐中逐步完善
4.但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
5.他們認為，到浦東新區投資辦事有章法
6.講規矩
7.利益能得到保障

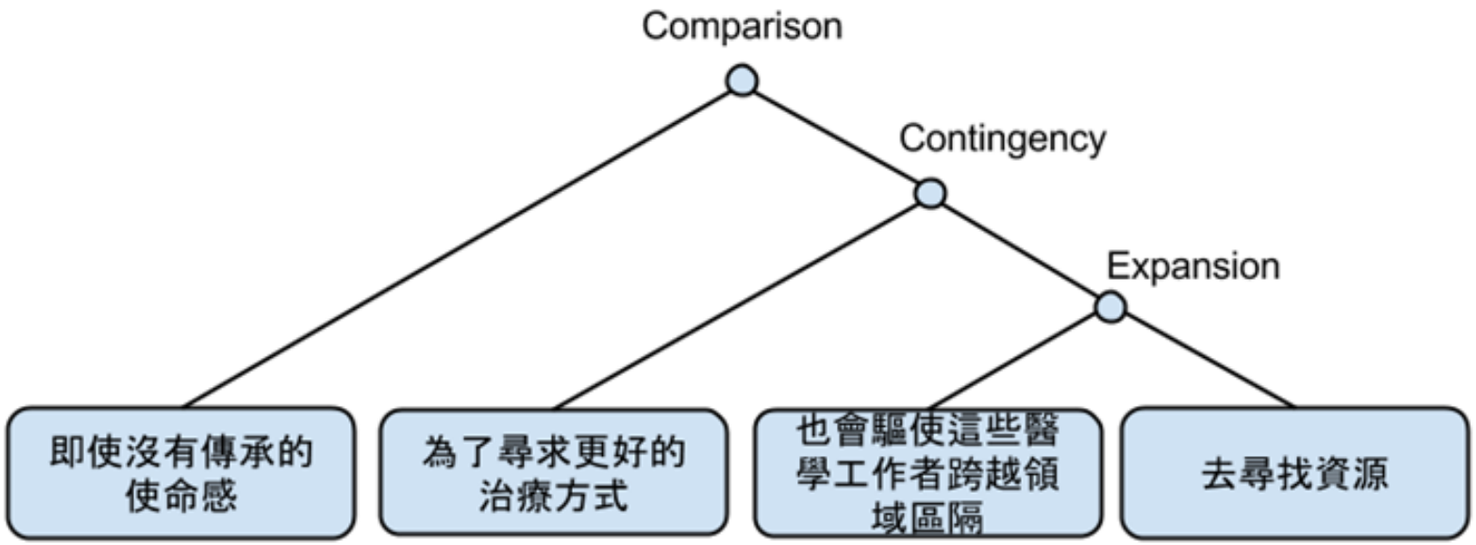
Chinese Discourse Parsing

- Predicting the relation structure of a number of clauses/sentences.

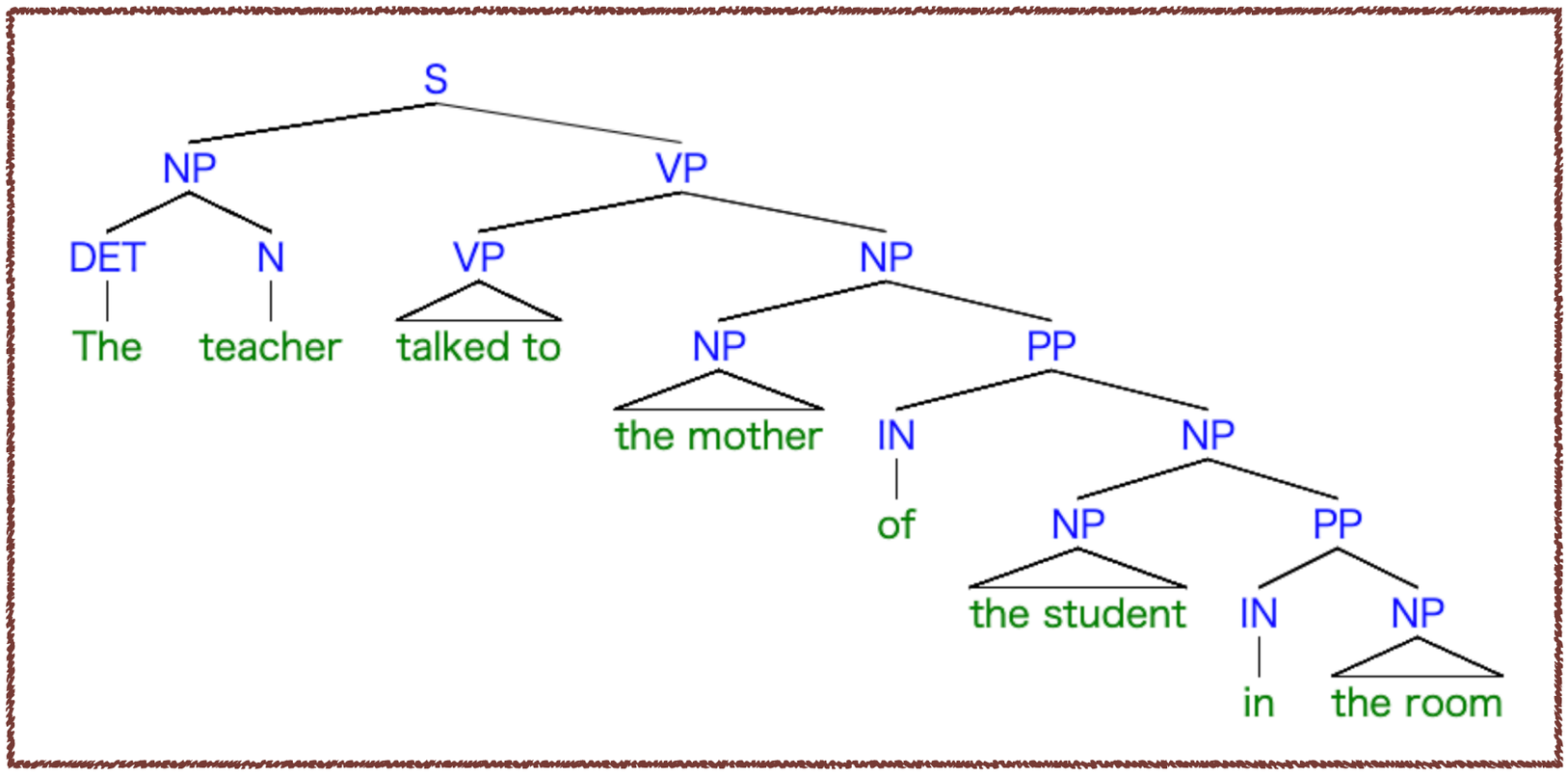
$$\hat{t} = \arg\max_{t \in T} P(t|EDU_1, EDU_2, ..., EDU_n)$$



Discourse Parsing vs Syntactic Parsing



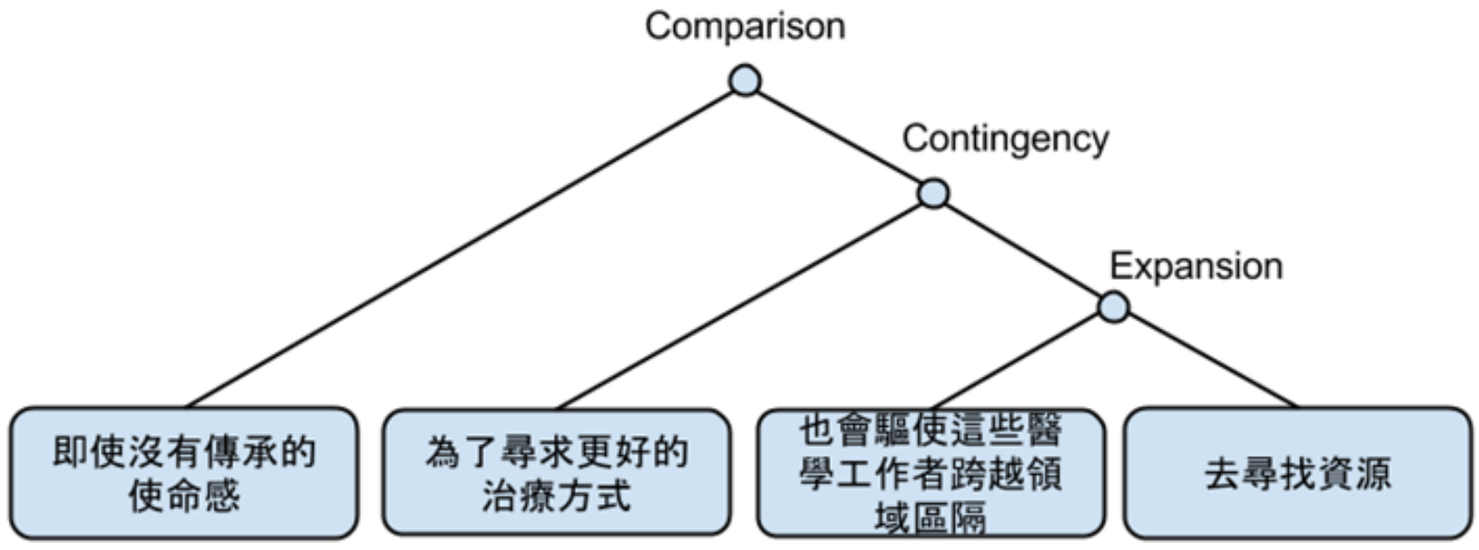
Discourse Parsing



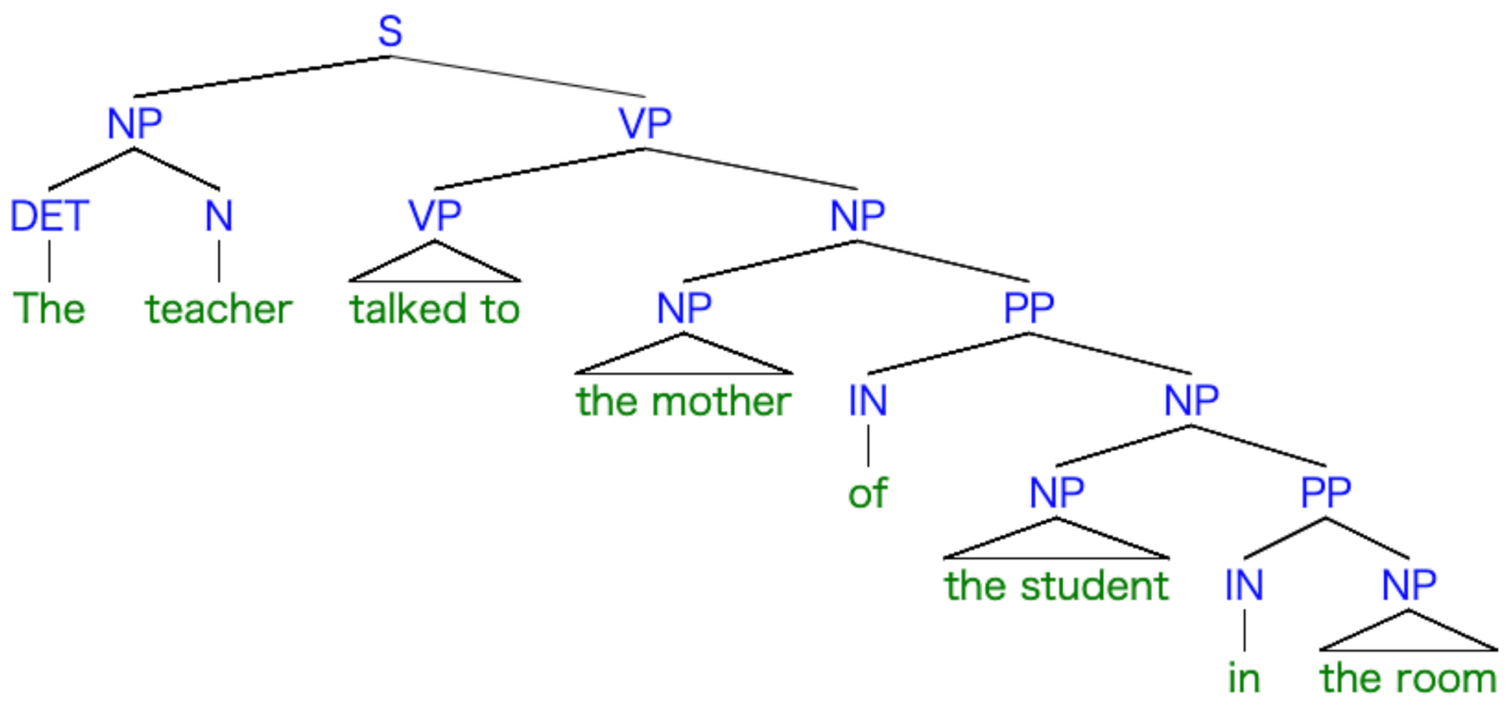
Syntactic Parsing

POS Tagging

Discourse Parsing vs Syntactic Parsing



Discourse Parsing

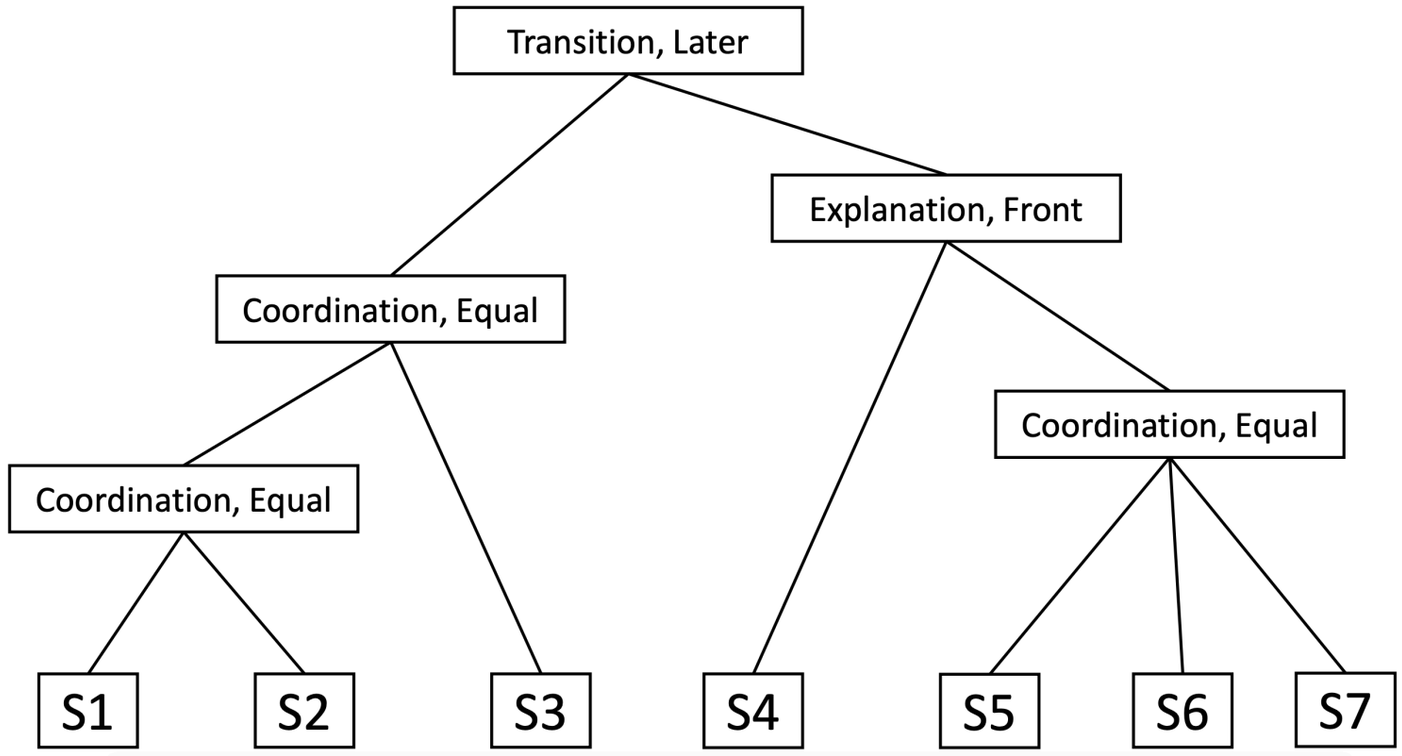


Syntactic Parsing

POS Tagging

Full Discourse Parsing

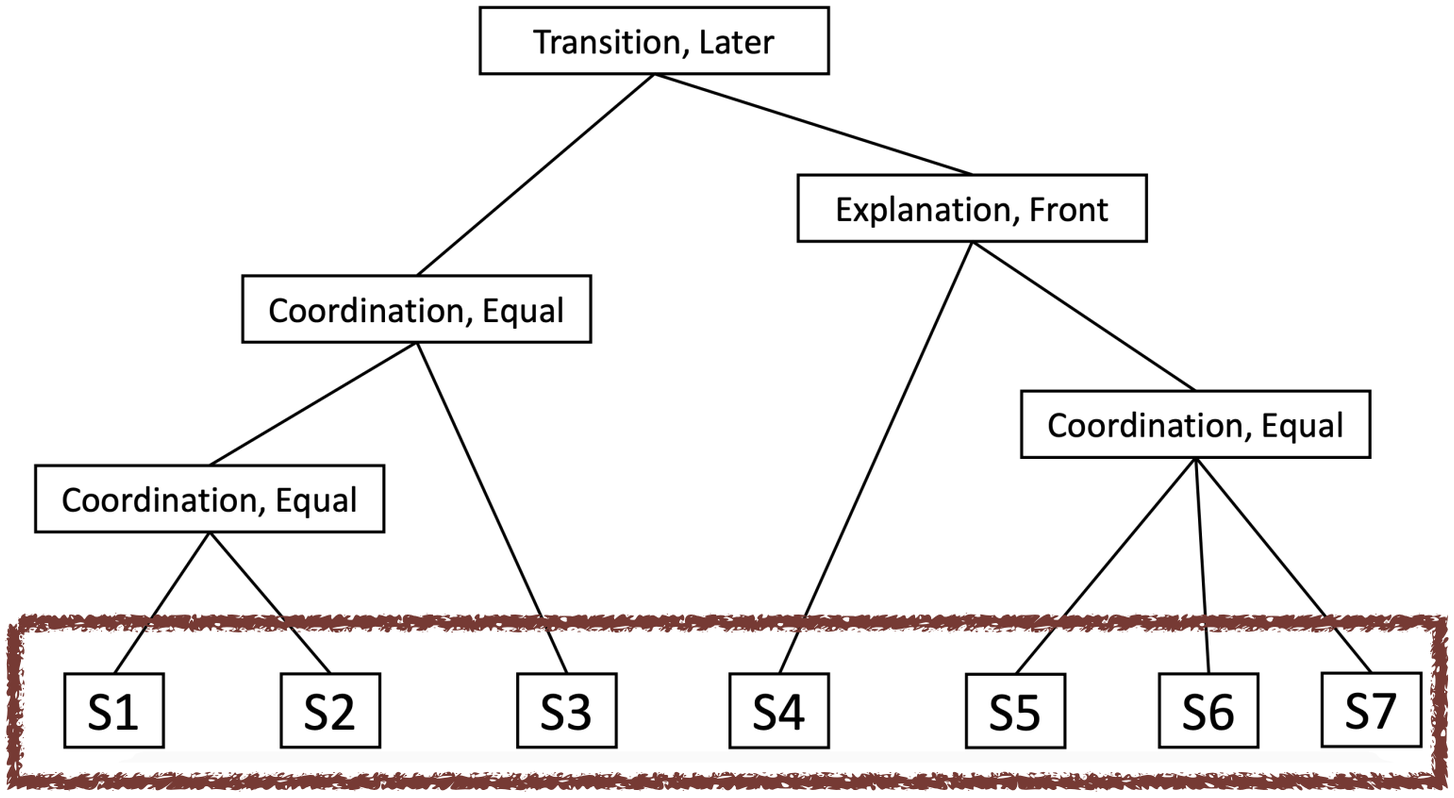
- EDU segmentation
- Tree construction
- Relation recognition
- Nuclearity labeling



- (S1) 儘管浦東新區製定的法規性文件有些比較"粗"
- (S2) 有些還只是暫行規定
- (S3) 有待在實踐中逐步完善
- (S4) 但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
- (S5) 他們認為，到浦東新區投資辦事有章法
- (S6) 講規矩
- (S7) 利益能得到保障

Subtasks in Discourse Parsing

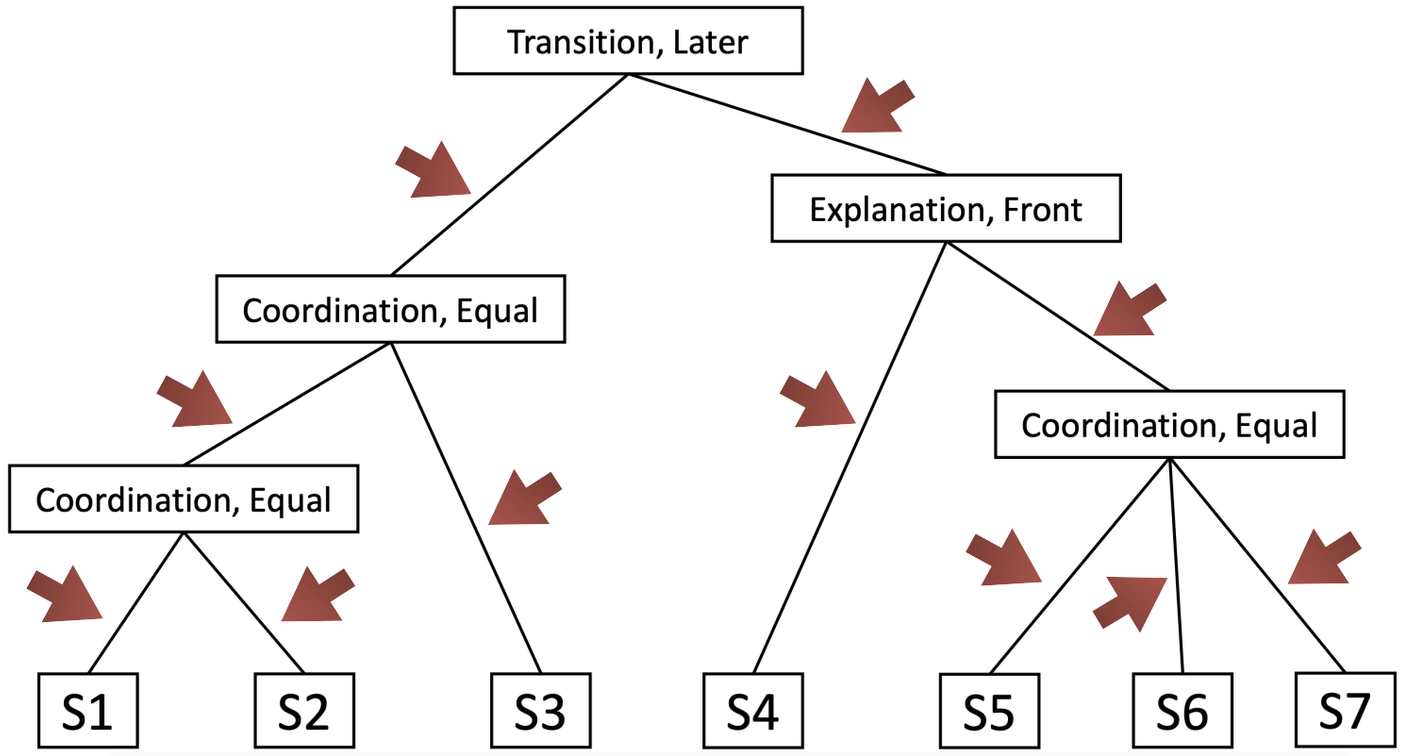
- EDU segmentation
- Tree construction
- Relation recognition
- Nuclearity labeling



- (S1) 儘管浦東新區製定的法規性文件有些比較"粗"
- (S2) 有些還只是暫行規定
- (S3) 有待在實踐中逐步完善
- (S4) 但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
- (S5) 他們認為，到浦東新區投資辦事有章法
- (S6) 講規矩
- (S7) 利益能得到保障

Subtasks in Discourse Parsing

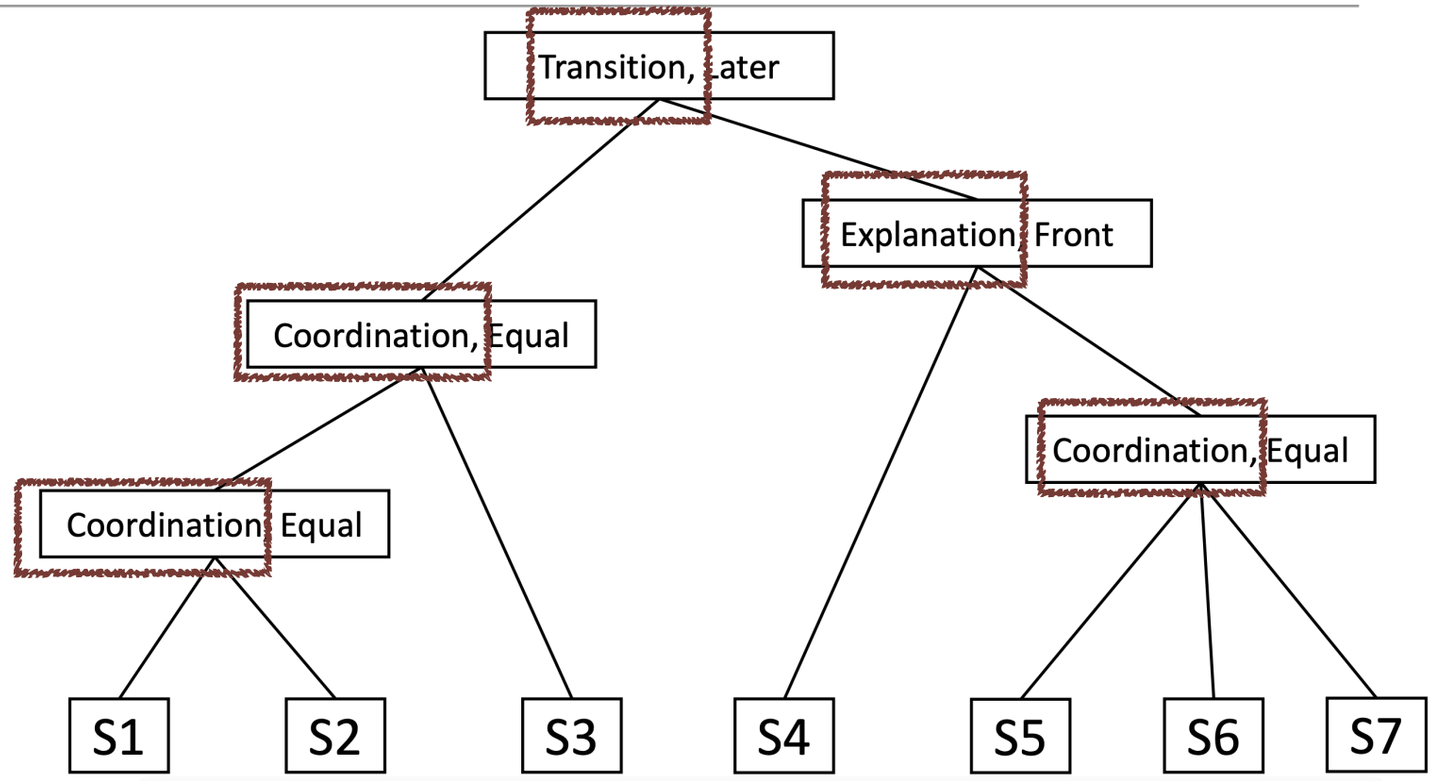
- EDU segmentation
- Tree construction
- Relation recognition
- Nuclearity labeling



- (S1) 儘管浦東新區製定的法規性文件有些比較"粗"
- (S2) 有些還只是暫行規定
- (S3) 有待在實踐中逐步完善
- (S4) 但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
- (S5) 他們認為，到浦東新區投資辦事有章法
- (S6) 講規矩
- (S7) 利益能得到保障

Subtasks in Discourse Parsing

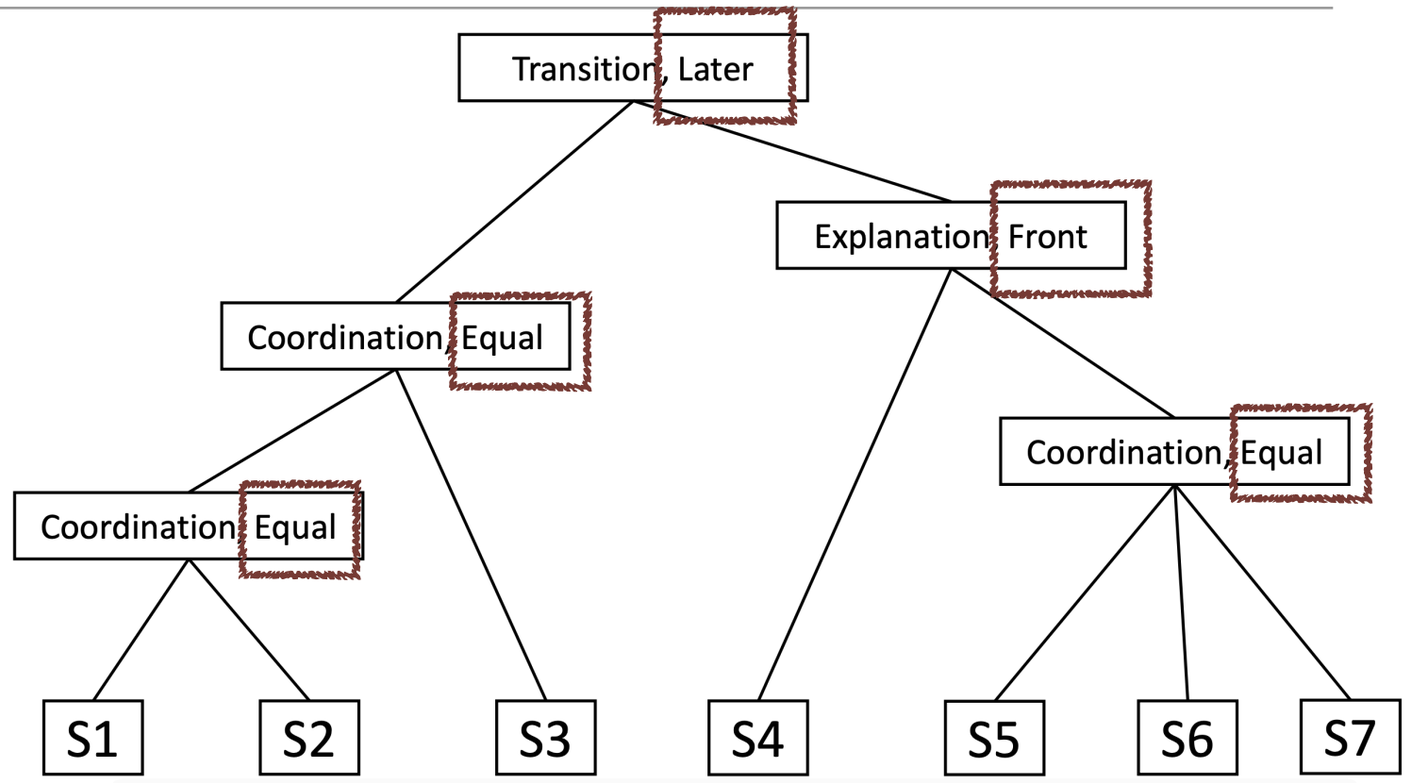
- EDU segmentation
- Tree construction
- Relation recognition
- Nuclearity labeling



- (S1) 儘管浦東新區製定的法規性文件有些比較"粗"
- (S2) 有些還只是暫行規定
- (S3) 有待在實踐中逐步完善
- (S4) 但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
- (S5) 他們認為，到浦東新區投資辦事有章法
- (S6) 講規矩
- (S7) 利益能得到保障

Subtasks in Discourse Parsing

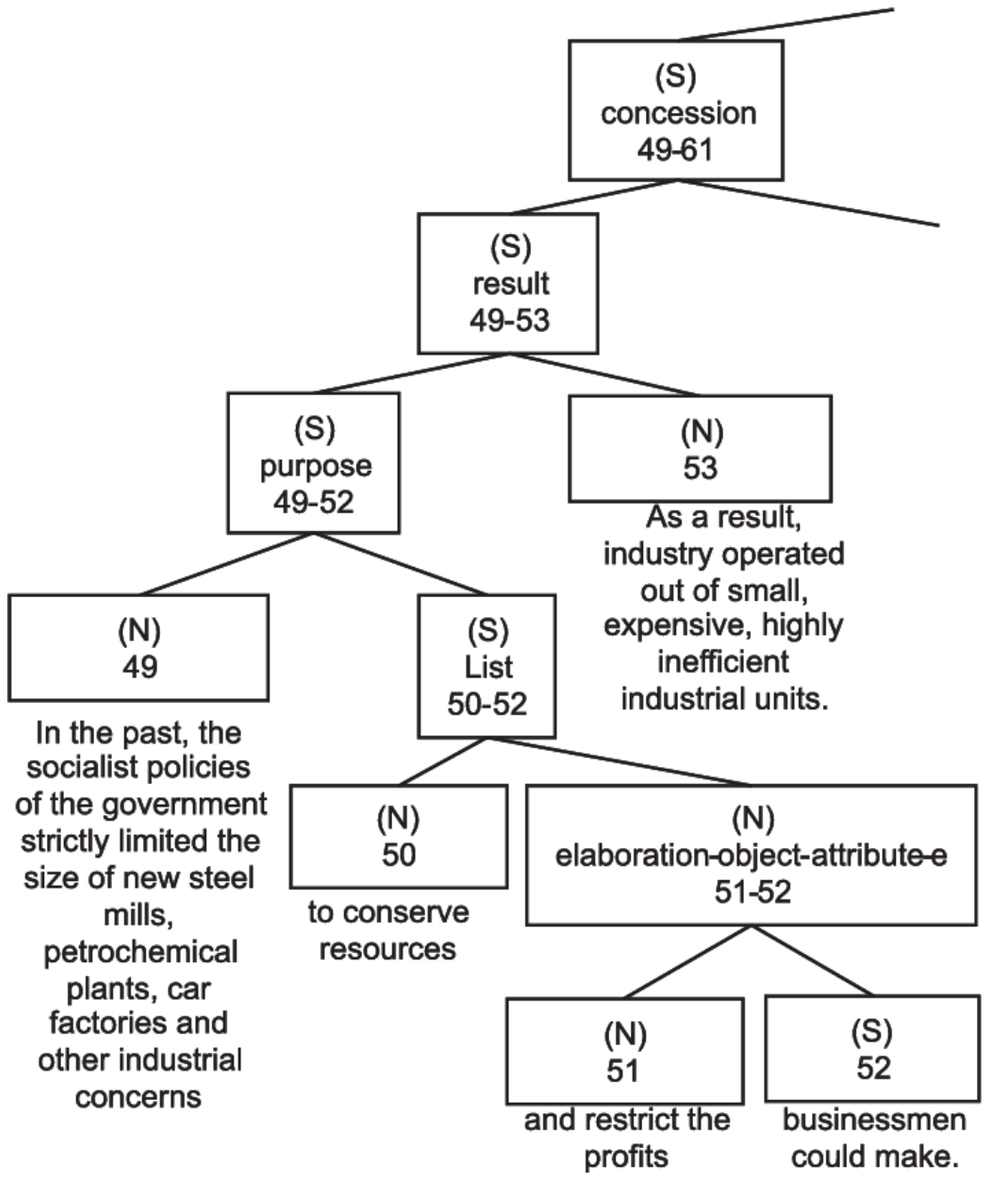
- EDU segmentation
- Tree construction
- Relation recognition
- Nuclearity labeling



- (S1) 儘管浦東新區製定的法規性文件有些比較"粗"
- (S2) 有些還只是暫行規定
- (S3) 有待在實踐中逐步完善
- (S4) 但這種法制緊跟經濟和社會活動的做法，受到了國內外投資者的好評
- (S5) 他們認為，到浦東新區投資辦事有章法
- (S6) 講規矩
- (S7) 利益能得到保障

Document-level Discourse Parsing

- Rhetorical Structure Theory Discourse Treebank (RST-DT)
  - Hierarchical information is extensively labeled.
- 78 relations in 16 classes.



Approaches to Parsing

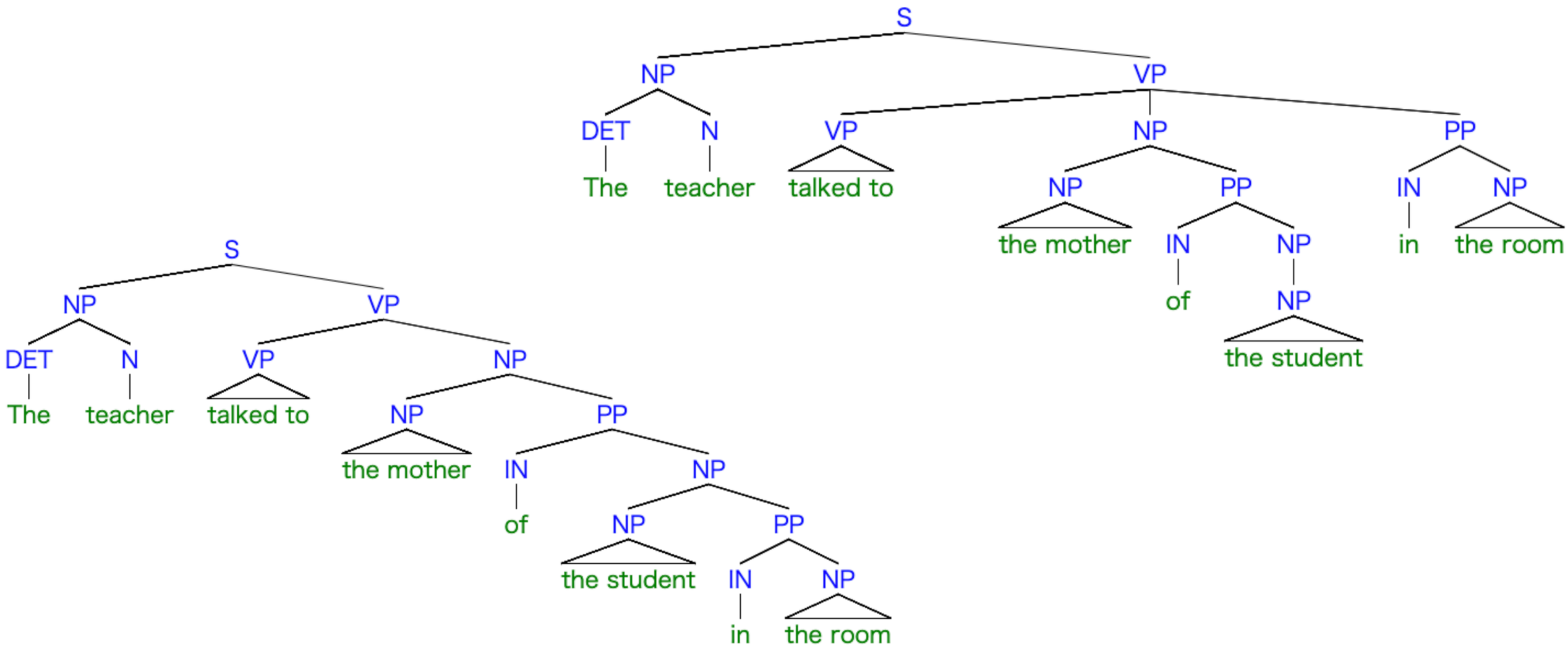
Symbolic Parsing

- The traditional (classical) way to parsing
- Define the CFG rules and lexicon
- Parse the sentences with the grammar
- Poor to deal with ambiguity
  - *Fed raises interest rates 0.5% in effort to control inflation*
  - Up to millions of possible parse trees!

http://spark-public.s3.amazonaws.com/nlp/slides/Parsing-Intro.pdf

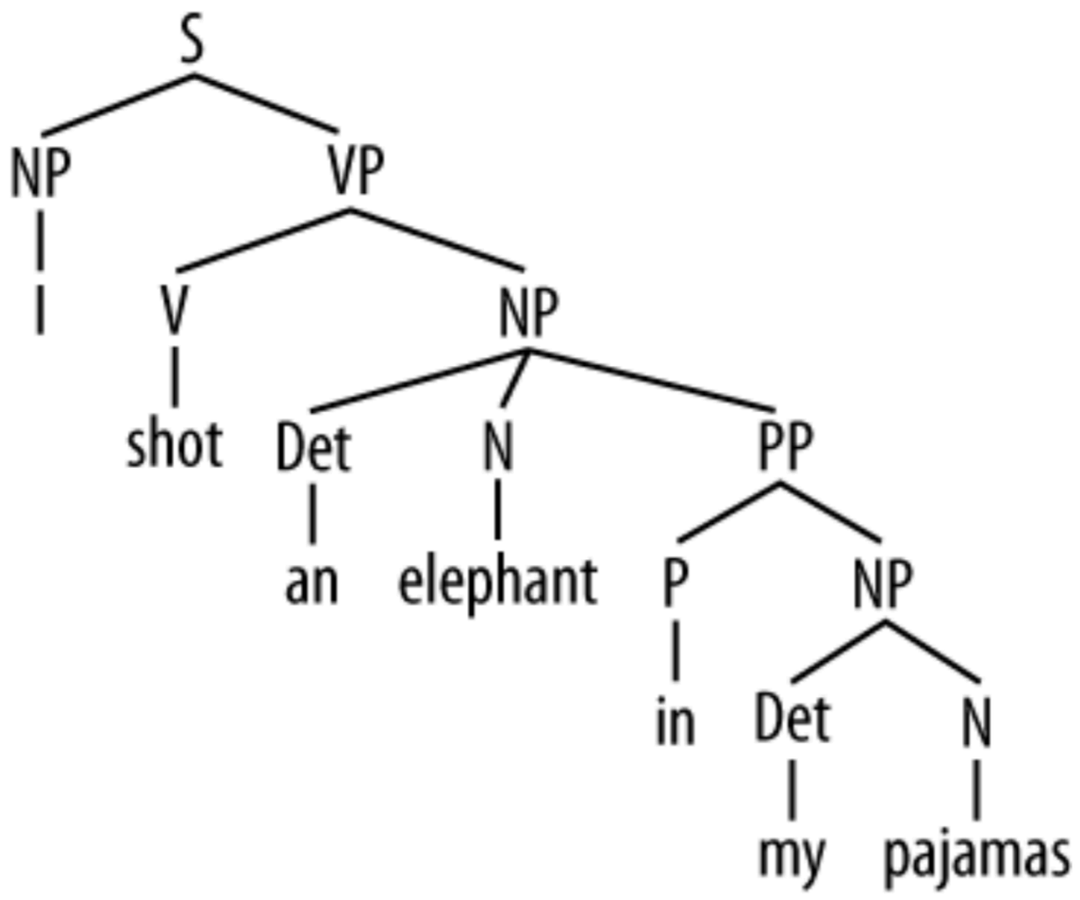
Attachment Ambiguity

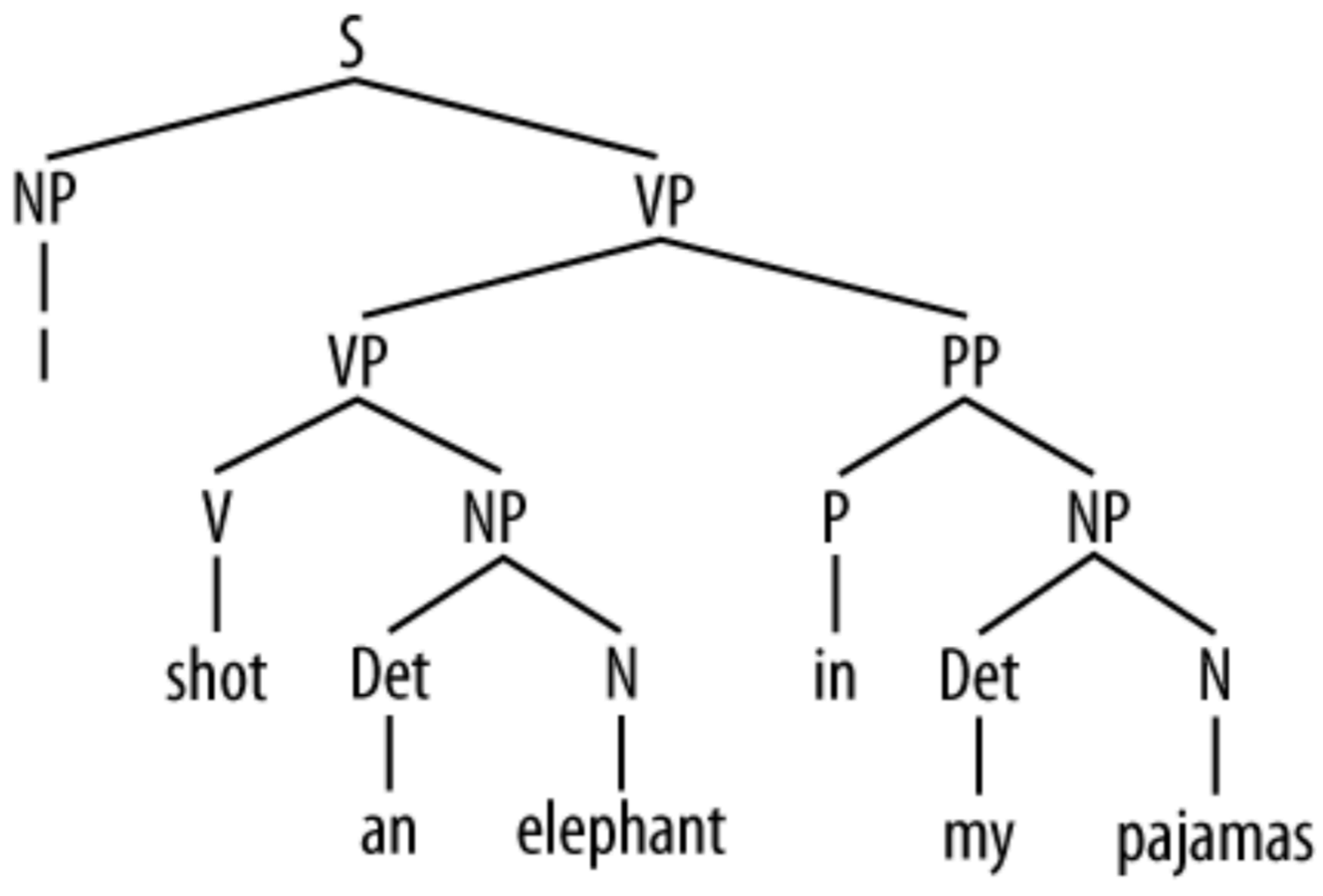
- The key of parsing decision is how to attach a constituent to another constituent



Attachment Ambiguity

- I shot an elephant in my pajamas (睡袍).





Challenges of Symbolic Parsing

- Additional constrains can be applied to restrict the rare parse trees
  - Poor coverage
- Loose grammar leads to more ambiguity
  - A sentence will have more possible parse trees and yet to choose the correct one.
- Choose the best (most probable) parse tree
  - Statistical parsing

Probabilistic Context-Free Grammar (PCFG)

- $G = (T, N, start, R)$
  - $\{w^k\}$: A set of terminal symbols (words)
  - $\{N^i\}$: A set of non-terminal symbols (NP, VP, etc.)
    - N^1: The start symbol (S)
    - R: A set of rules/productions that convert N^i to y
      - y is a number of members of $\{N^i\}$ or $\{w^k\}$
    - **P: Probability of each rule in R.**

Probabilities of Production Rules

Rules	Probability
S -> NP VP	1
VP -> V NP	0.65
VP -> V NP PP	0.35
NP -> NP NP	0.2
NP -> NP PP	0.3
NP -> N	0.5
PP -> IN NP	1

Rules	Probability
V -> ate	0.65
V -> runs	0.35
N -> runs	0.1
N -> dog	0.4
N -> cat	0.5
PRP$ -> my	1
IN -> of	1

Questions of PCFGs

- What is the probability of a sequence *s* according to a grammar *G*?

$$P(s|G)$$

- What is the most likely parse tree *t'* for *s*?

$$\arg\max_{t \in T} P(t|s, G)$$ **Parsing**

- How to choose rule probabilities for the grammar *G* that maximize the probability of *s*

$$\arg\max_{G} P(s|G)$$ **Training**

Probability of Trees and Sentences

- $P(t)$: The probability of a syntax tree t is the **product** the probabilities of the rules that constructs the t

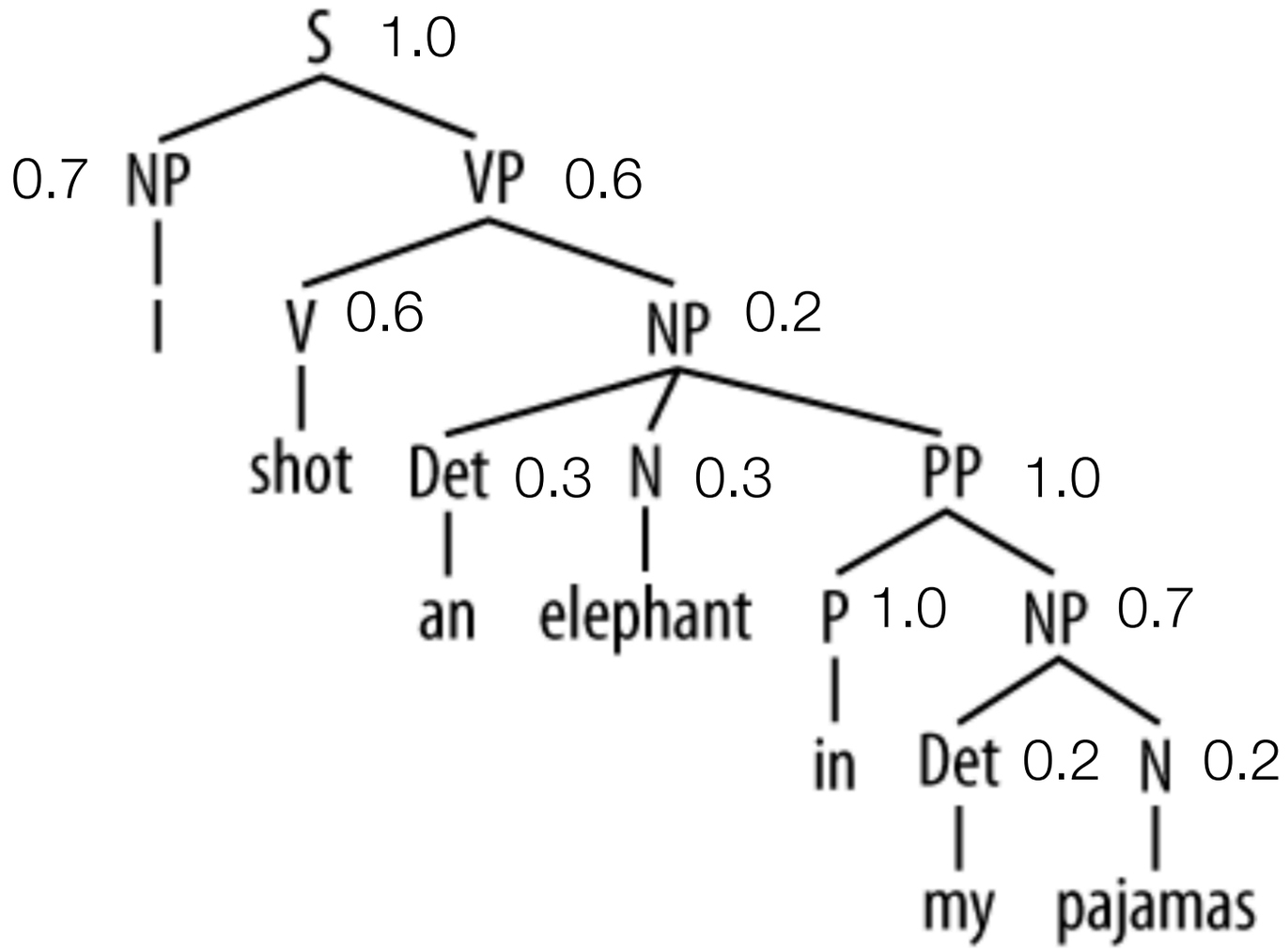
- $P(s)$: The probability of a sentence s is sum of the probabilities of all possible trees of s.

$$P(s) = \sum_{t \in T} P(t)$$

All possible parse tree of s

Probability of a Tree

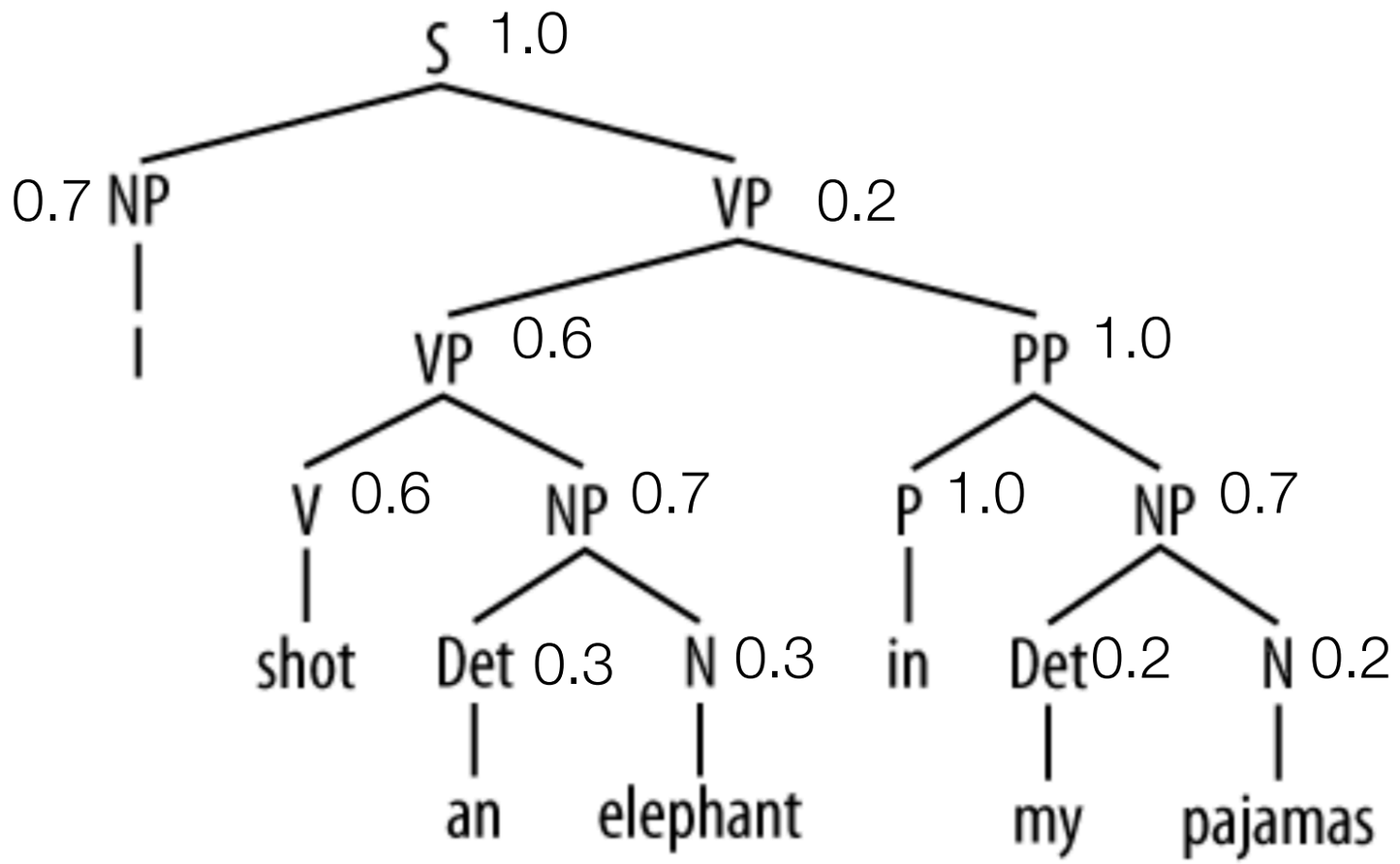
- *s*: I shot an elephant in my pajamas.



$P(t_1)$=1.0 x 0.7 x 0.5 x 0.6 x 0.6 x 0.2 x 0.3 x 0.3 x 1.0 x 1.0 x 0.7 x 0.2 x 0.2 = 6.3504×10^{-5}

Probability of a Tree

- *s*: I shot an elephant in my pajamas.



$P(t_2)$=1.0 x 0.7 x 0.5 x 0.2 x 0.6 x 0.6 x 0.7 x 0.3 x 0.3 x 1.0 x 1.0 x 0.7 x 0.2 x 0.2= 4.44528×10^{-5}

$P(s) = P(t_1) + P(t_2) = 6.3504 \times 10^{-5} + 4.44528 \times 10^{-5} = 1.079568 \times 10^{-4}$

Parsing

- Finding the most likely parse tree for a sentence

$$\arg\max_{t \in T} P(t|s, G)$$

- It is time-consuming to exploit *T*, all possible parse trees, given a sentence s.
- Dynamic programming: CKY parsing
- Greedy: Shift-reduce parsing

Divide and Conquer

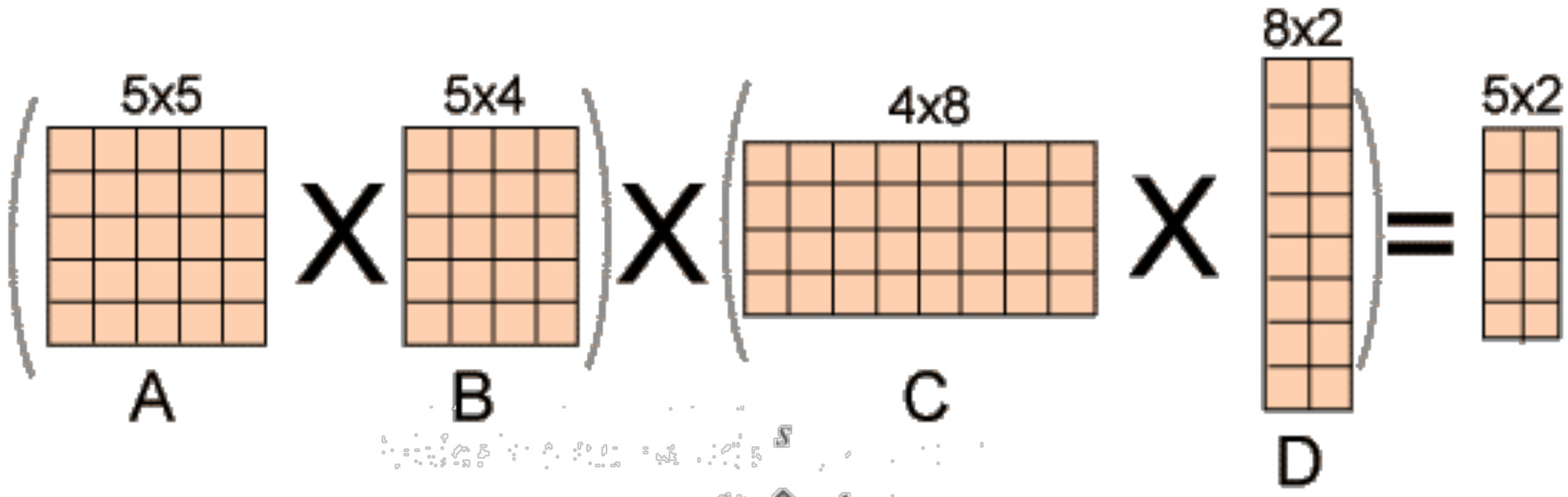
- With the independent assumption of context-free grammar, we can find the most likely parse tree by using a **divide and conquer** algorithm.
  - *s*: *I shot an elephant in my pajamas*

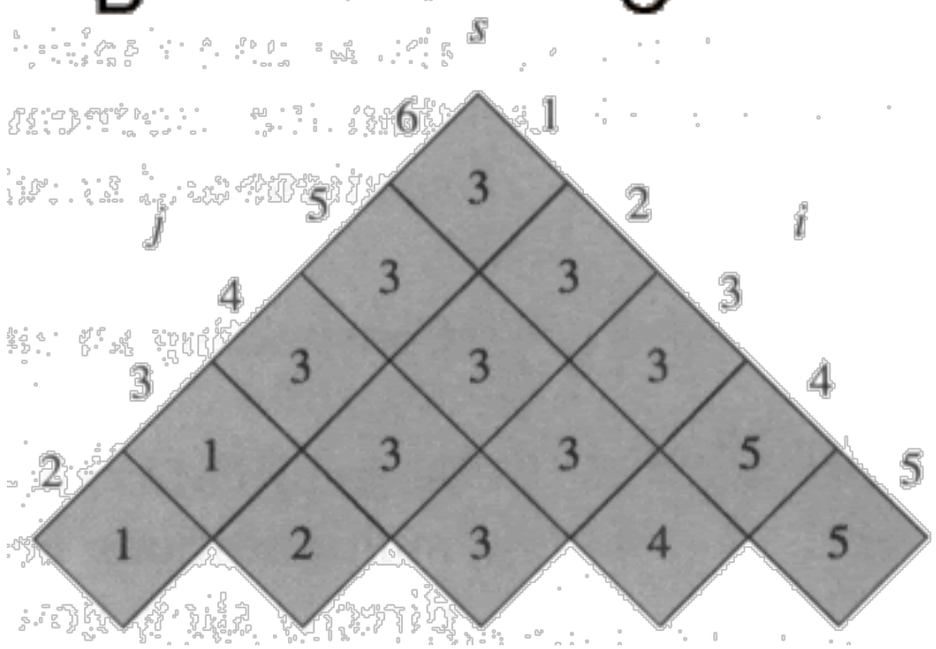
$$P_{i,j} = \arg \max_{i \leq k < j} P_{i,k} P_{k+1,j}$$

	P_1	P_2
$P_{1,1} P_{2,7}$	I	shot an elephant in my pajamas
$P_{1,2} P_{3,7}$	I shot	an elephant in my pajamas
$P_{1,3} P_{4,7}$	I shot an	elephant in my pajamas
$P_{1,4} P_{5,7}$	I shot an elephant	in my pajamas
$P_{1,5} P_{6,7}$	I shot an elephant in	my pajamas
$P_{1,6} P_{7,7}$	I shot an elephant in my	pajamas

Speed up with Dynamic Programming

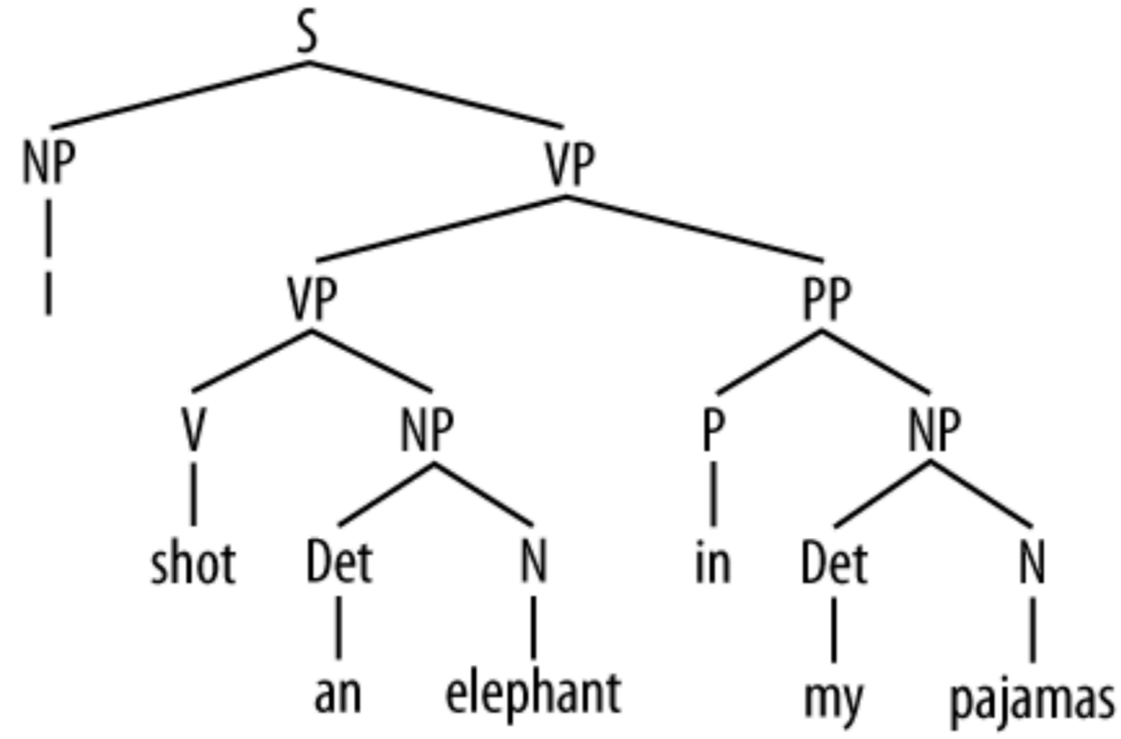
- Similar to the task that finds the best order of matrix multiplication



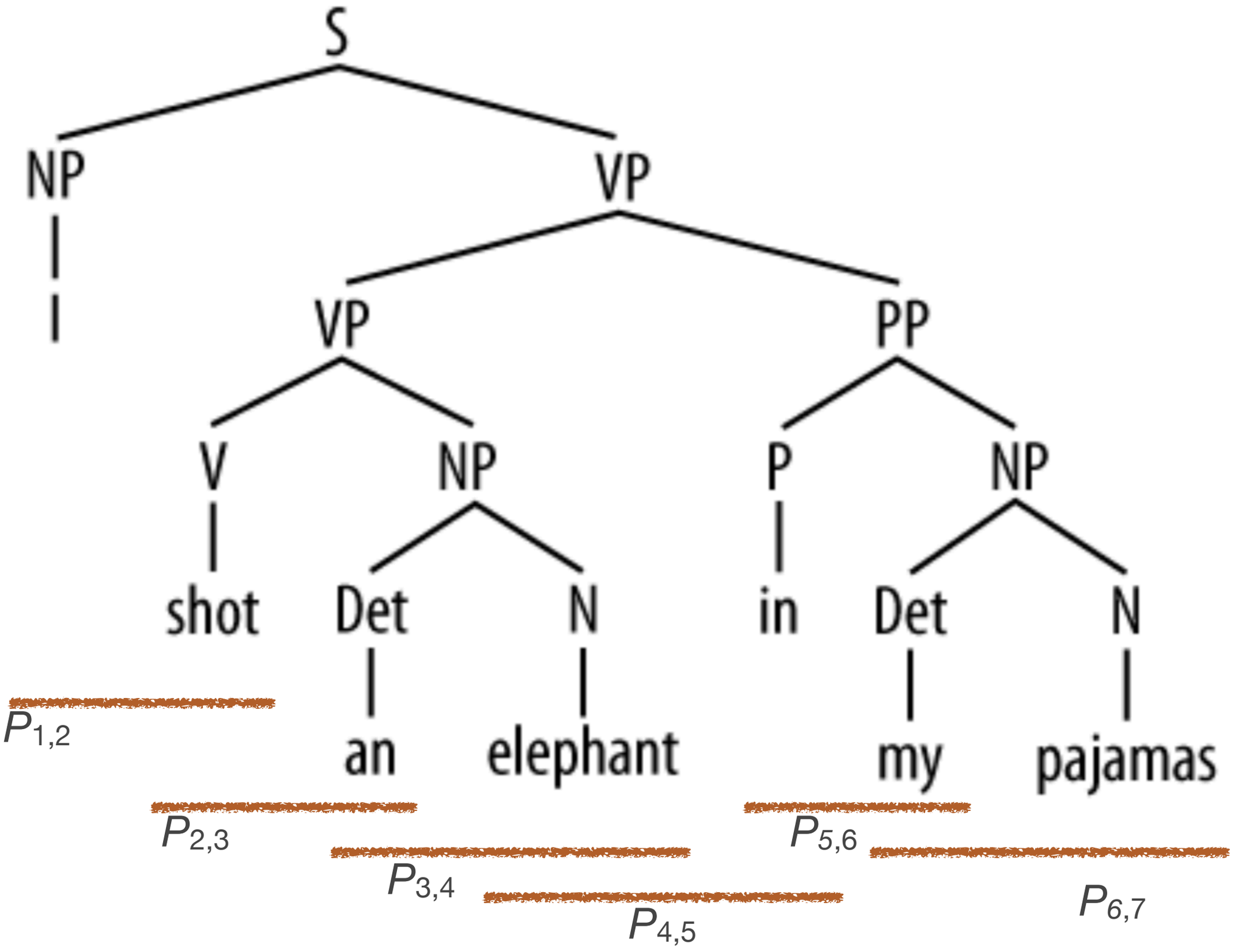


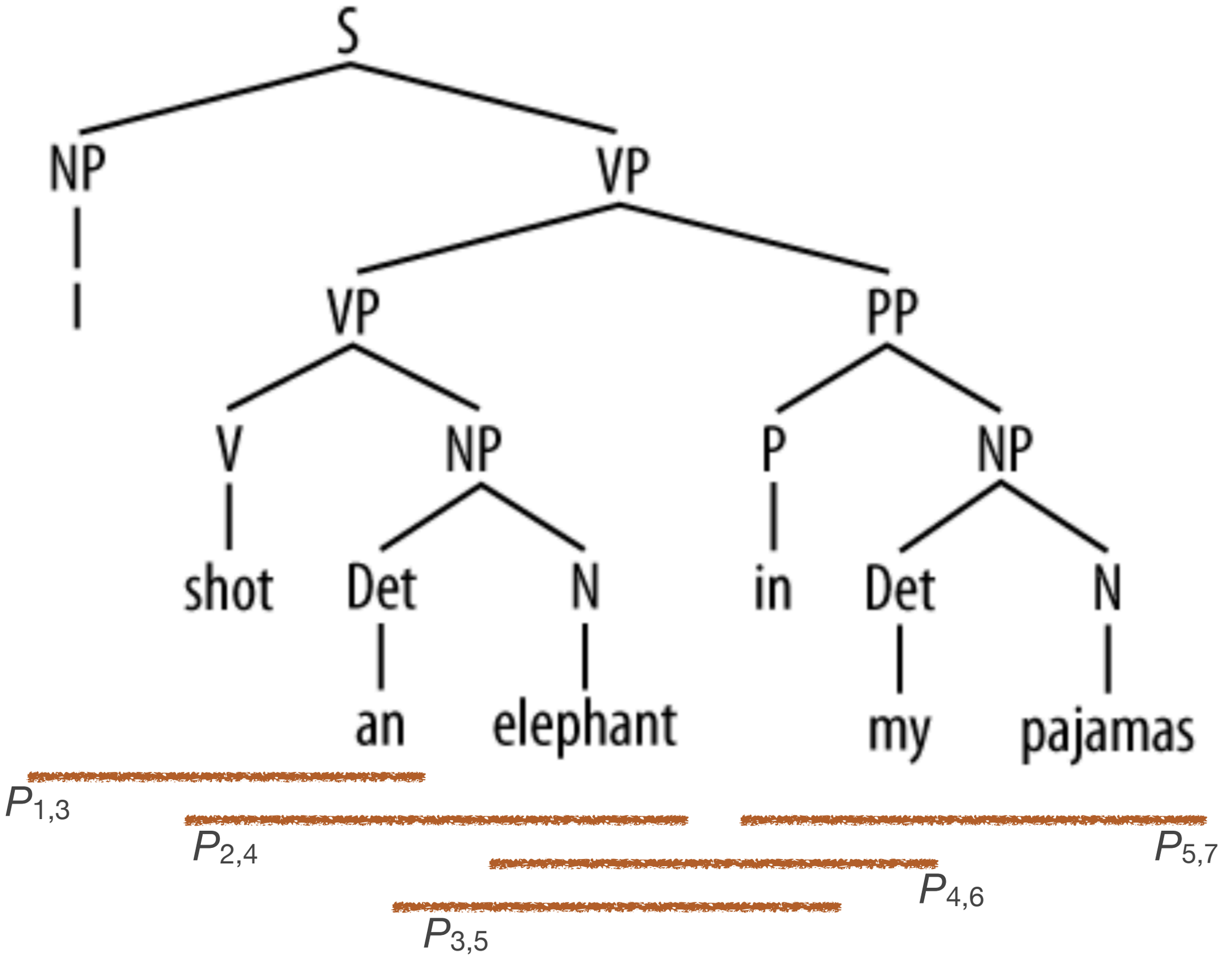
CKY Parsing

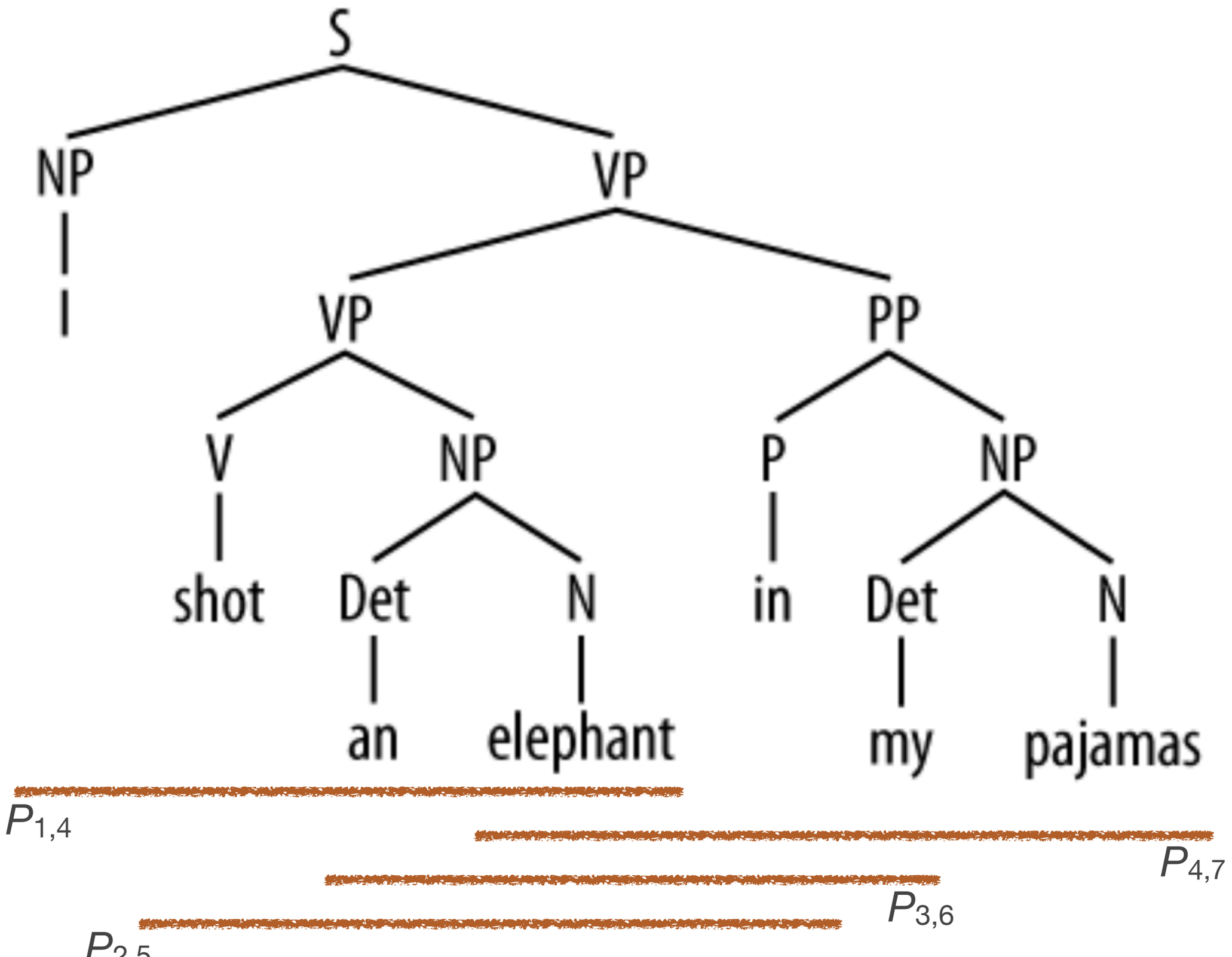
- Merge successive elements in the bottom-up fashion

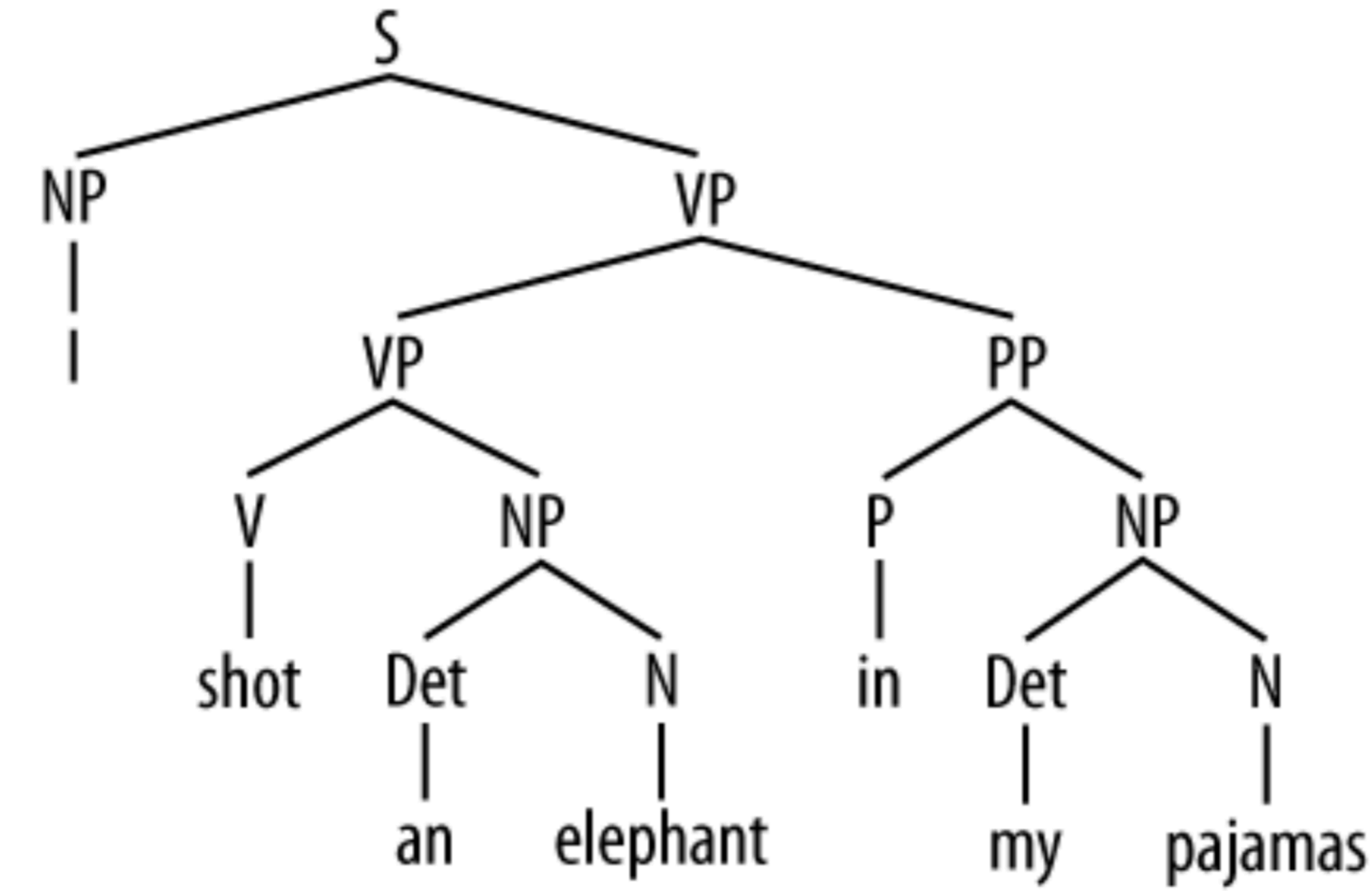


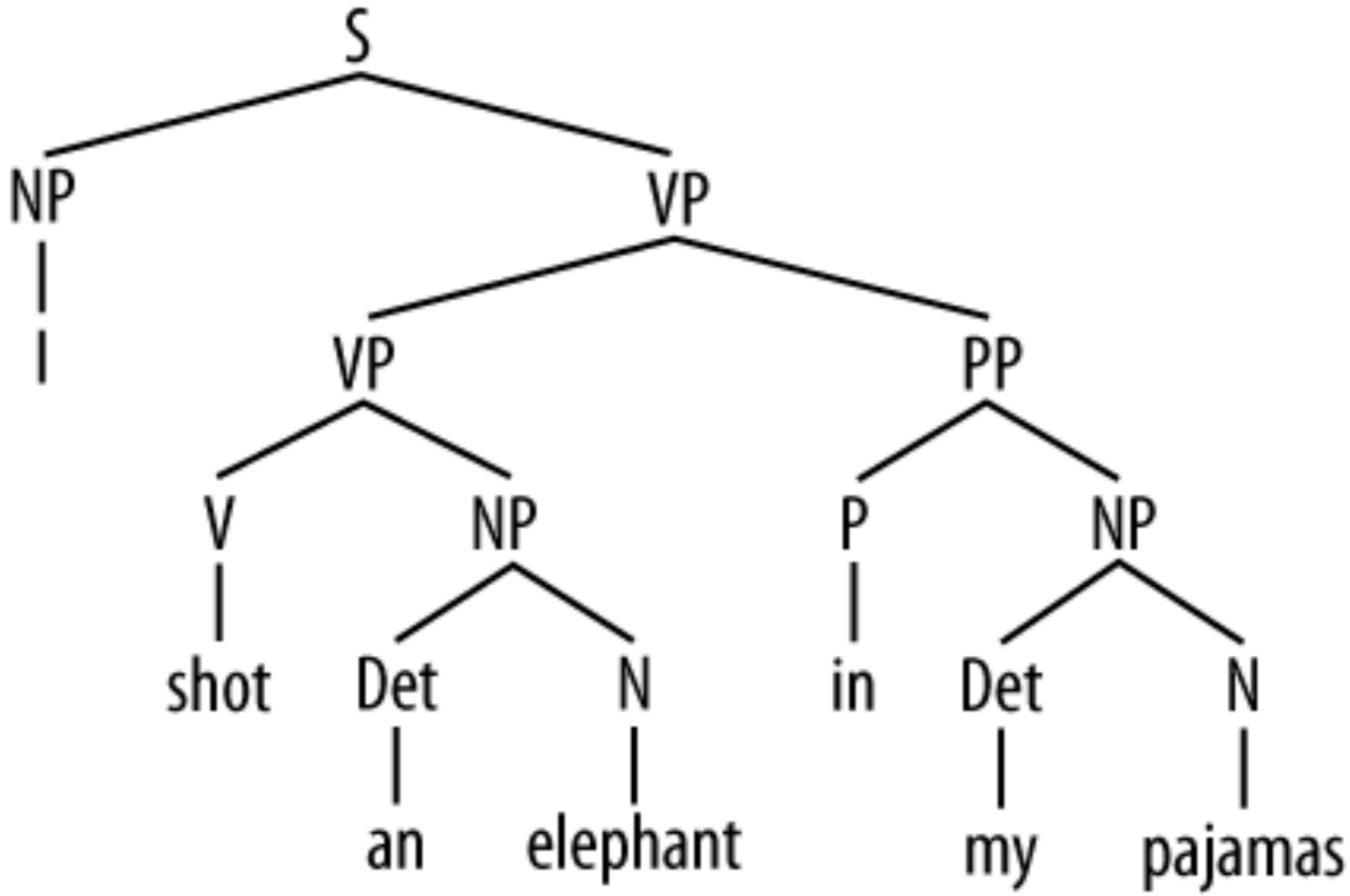
(I, …, pajamas)					
(I, …, my)	(shot, …, pajamas)				
(I, …, in)	(shot, …, my)	(an, …, pajamas)			
(I, …, elephant)	(shot, …, in)	(an, …, my)	(elephant, …, pajamas)		
(I, shot, an)	(shot, an, elephant)	(an, elephant, in)	(elephant, in, my)	(in, my, pajamas)	
(I, shot)	(shot an)	(an elephant)	(elephant, in)	(in, my)	(my, pajamas)


S
NP
VP
I
VP
PP
V
NP
P
NP
shot
Det
N
in
Det
N
an
elephant
my
pajamas
P1,2
P2,3
P3,4
P4,5
P5,6
P6,7


S
NP
VP
I
VP
PP
V
NP
P
NP
shot
Det
N
in
Det
N
an
elephant
my
pajamas
$P_{1,3}$
$P_{2,4}$
$P_{5,7}$
$P_{4,6}$
$P_{3,5}$

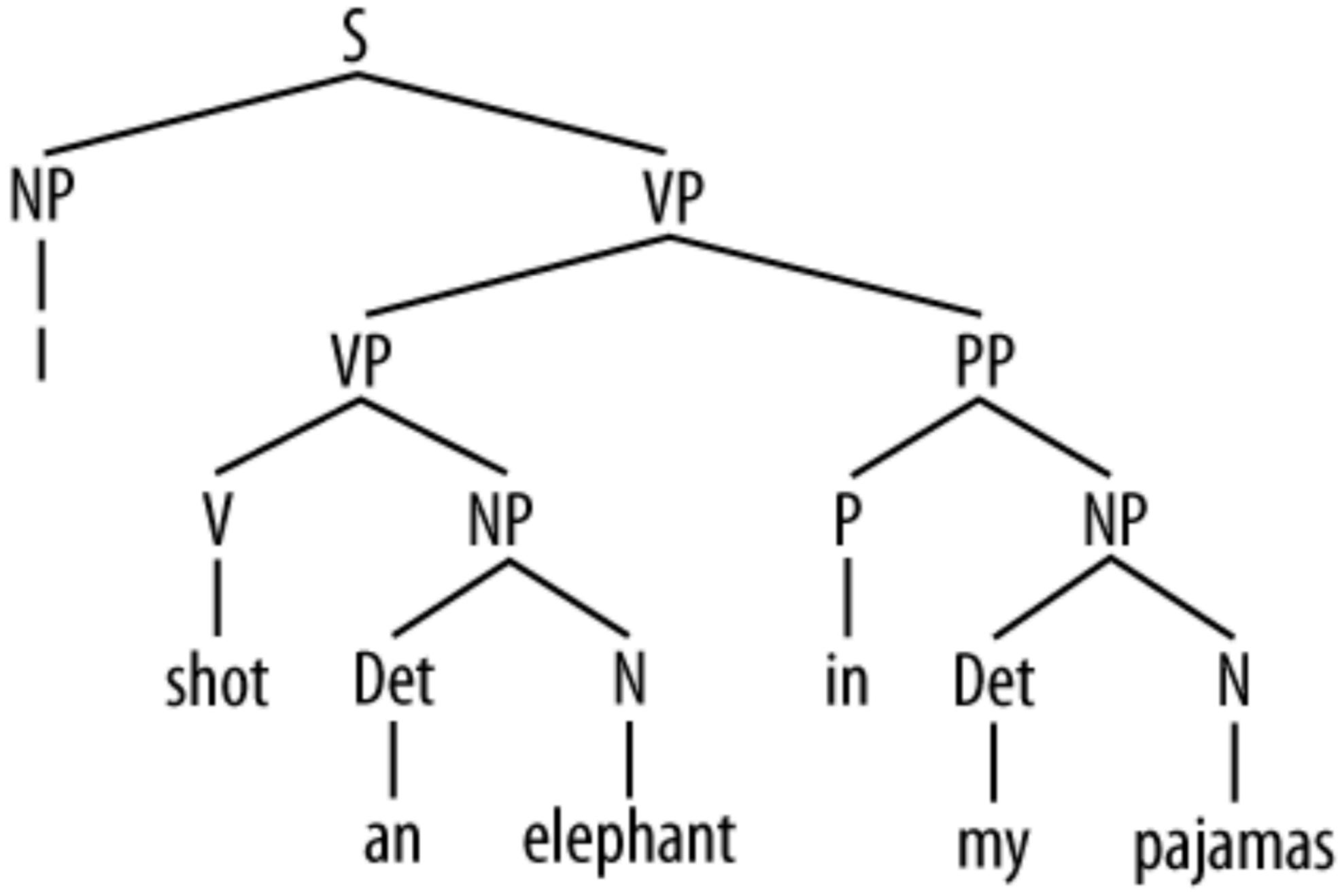

S
NP
VP
I
VP
PP
V
NP
P
NP
shot
Det
N
in
Det
N
an
elephant
my
pajamas
$P_{1,4}$
$P_{4,7}$
$P_{3,6}$
$P_{2,5}$


S
NP
VP
I
VP
PP
V
NP
P
NP
shot
Det
N
in
Det
N
an
elephant
my
pajamas
$P_{1,5}$
$P_{2,6}$
$P_{3,7}$

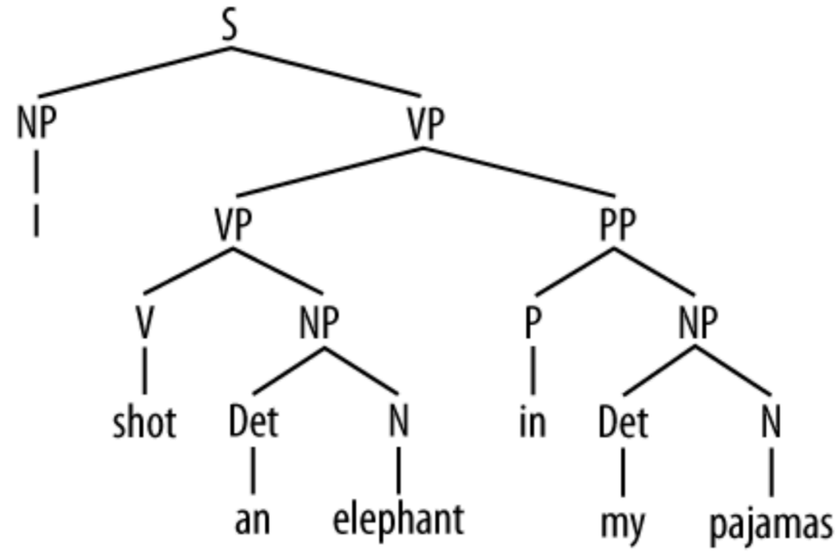


$P_{1,6}$

$P_{2,7}$



$P_{1,7}$



(I, …, pajamas)

(I, …, my)

(shot, …, pajamas)

(I, …, in)

(shot, …, my)

(an, …, pajamas)

(I, …, elephant)

(shot, …, in)

(an, …, my)

(elephant, …, pajamas)

(I, shot, an)

(shot, an, elephant)

(an, elephant, in)

(elephant, in, my)

(in, my, pajamas)

(I, shot)

(shot an)

(an elephant)

(elephant, in)

(in, my)

(my, pajamas)

I

shot

an

elephant

in

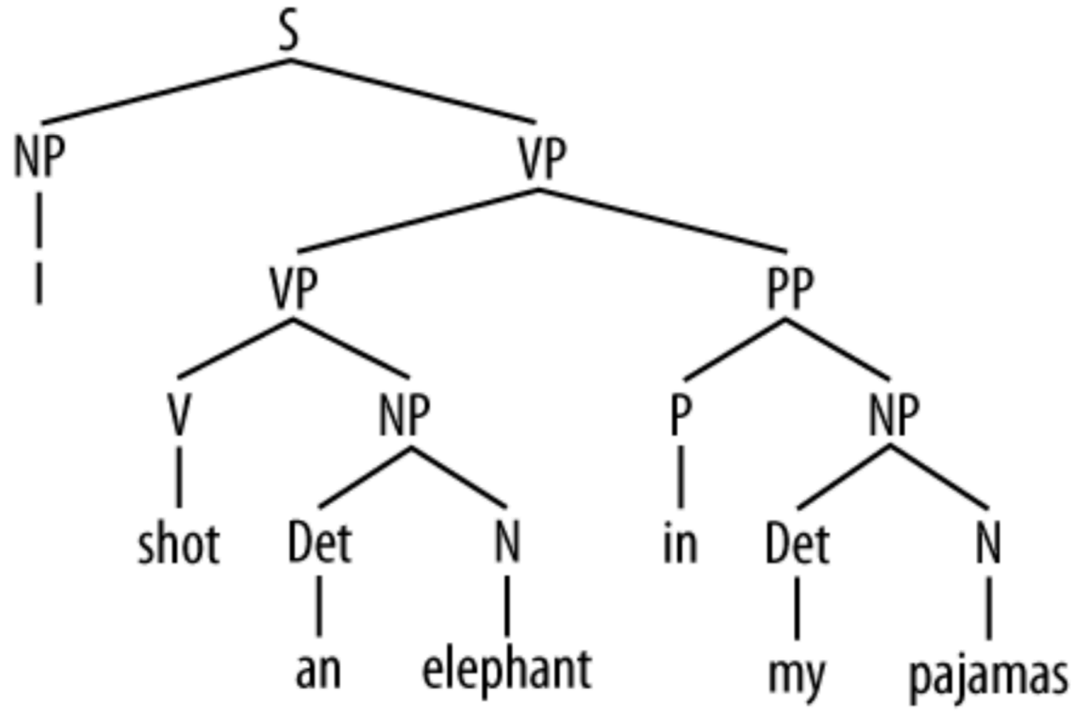
my

pajamas

Shift-Reduce Parsing

- A classifier trained to determine the action in a state
    - **Shift:** Advancing in the input stream by one symbol. That shifted symbol becomes a new single-node parse tree.
    - **Reduce**: Applying a completed grammar rule to some of the recent parse trees, joining them together as one tree with a new root symbol.
        - Merge two elements
        - Predict their parent phrase type

Example

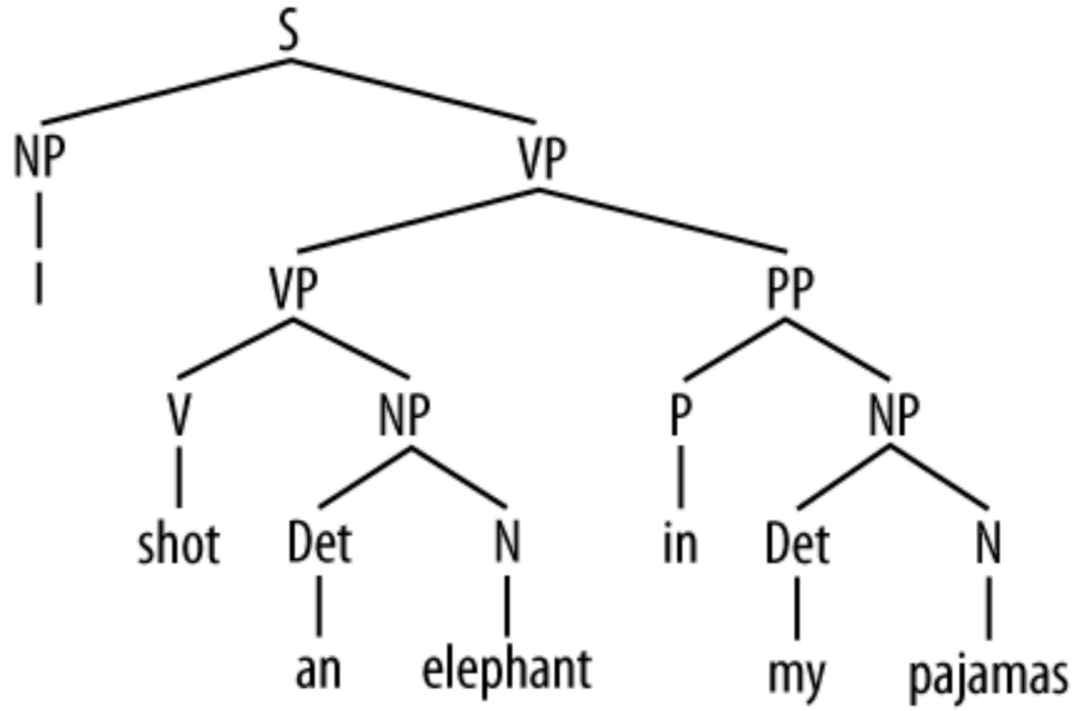


Stack	Queue
(VP (V Shot) (NP (Det an) (N elephant))) (NP I)	in my pajamas

Given Stack the Queue, predict to

- Shift ***in*** to Stack
- Reduce the top two elements of the Stack

Example of Reduce



Stack

(**S**
  (NP I)
  (VP
    (V Shot)
    (NP (Det an)
      (N elephant))

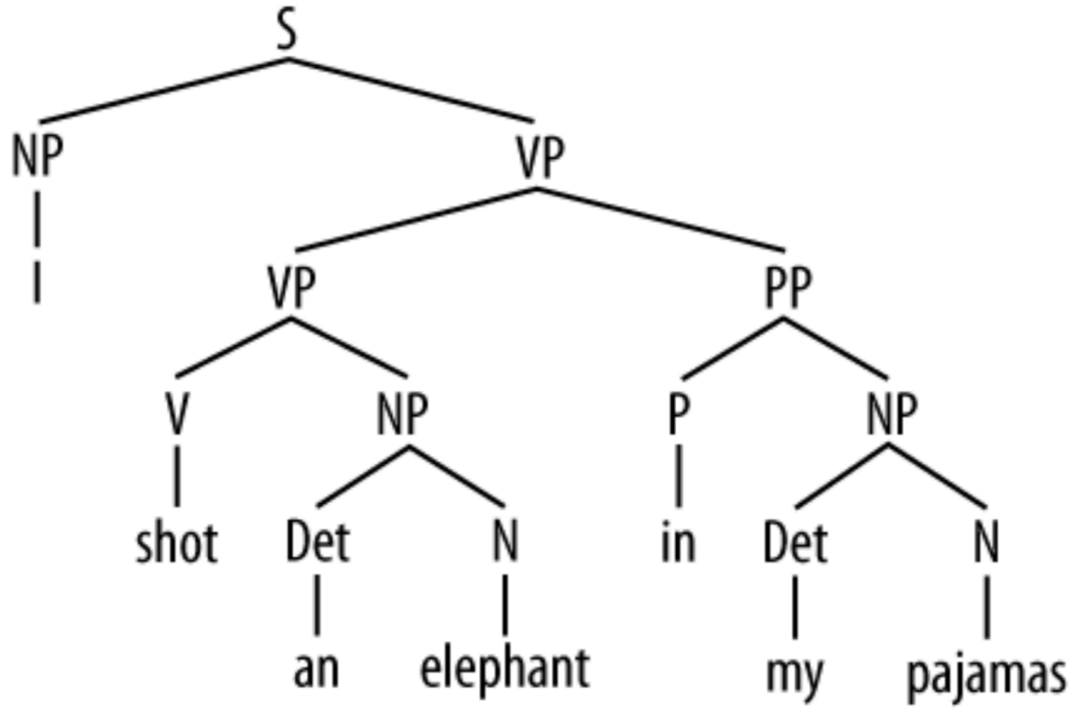
Queue

my pajamas

Given Stack the Queue, predict to

- Shift ***in*** to Stack
- Reduce the top two elements of the Stack

Example of Shift



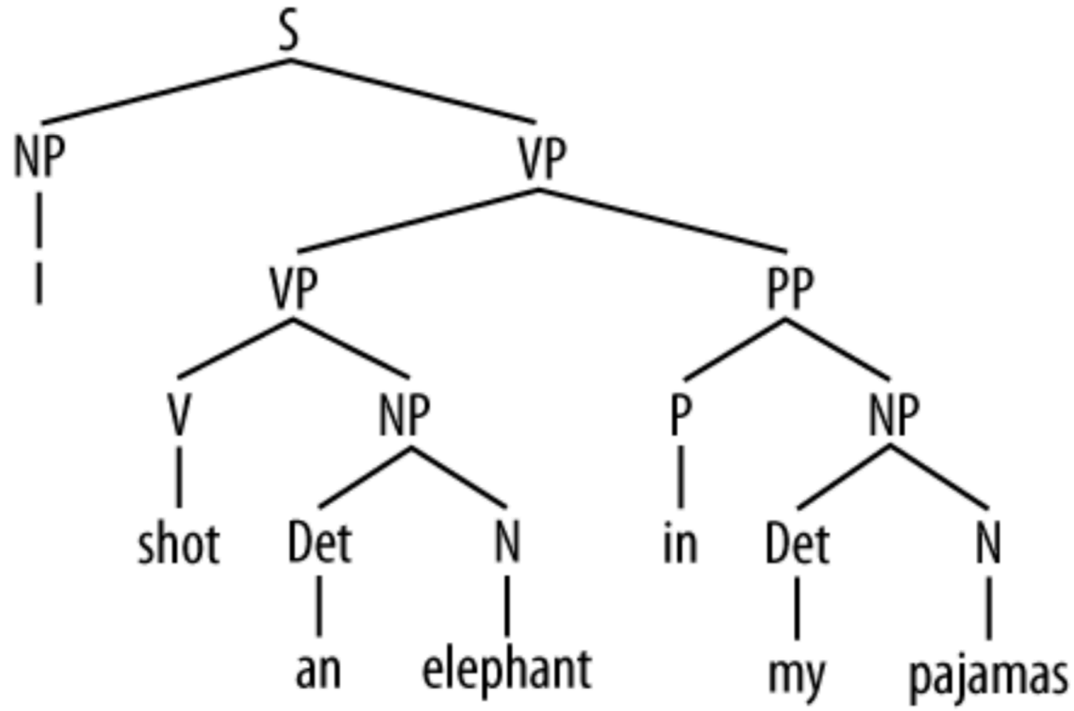
Stack
(P in) (S (NP I) (VP (V Shot) (NP (Det an) (N elephant))

Queue	
my	pajamas

Given Stack the Queue, predict to

- Shift ***my*** to Stack
- Reduce the top two elements of the Stack

Example of Shift



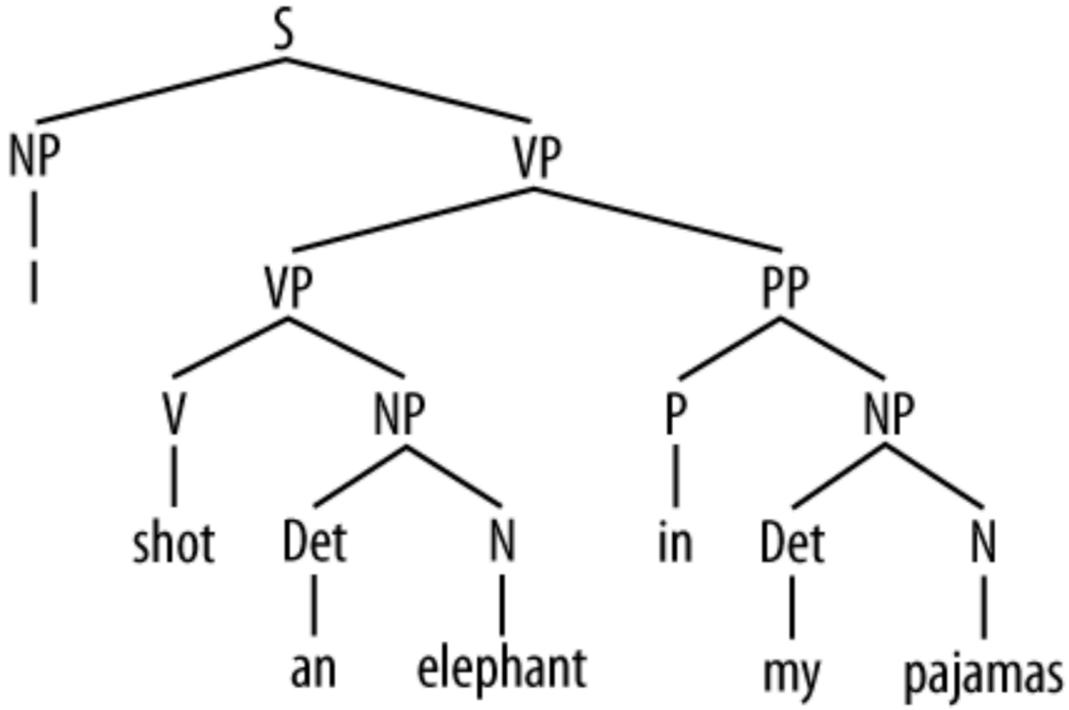
Stack
(Det my)
(P in)
(S (NP I) (VP (V Shot) (NP (Det an) (N elephant))

Queue
pajamas

Given Stack the Queue, predict to

- Shift ***pajamas*** to Stack
- Reduce the top two elements of the Stack

Example of Reduce



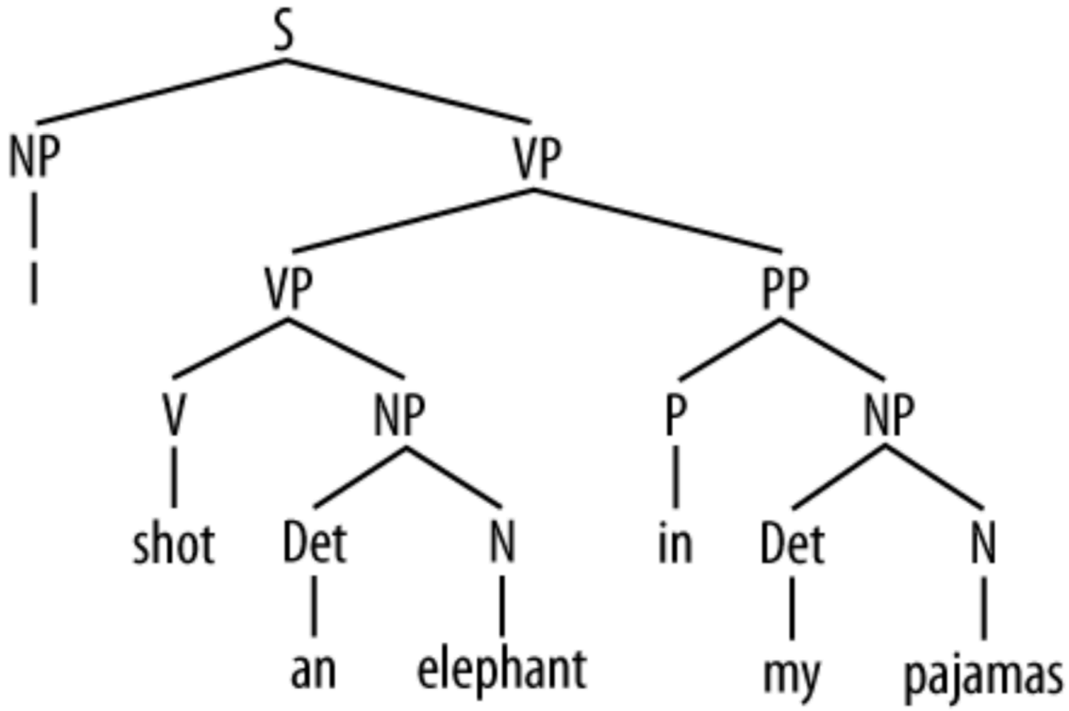
Stack
(N pajamas)
(Det my)
(P in)
(S (NP I) (VP (V Shot) (NP (Det an) (N elephant))

Queue

Given Stack the Queue, predict to

- Shift ? to Stack
- Reduce the top two elements of the Stack

Example of Reduce



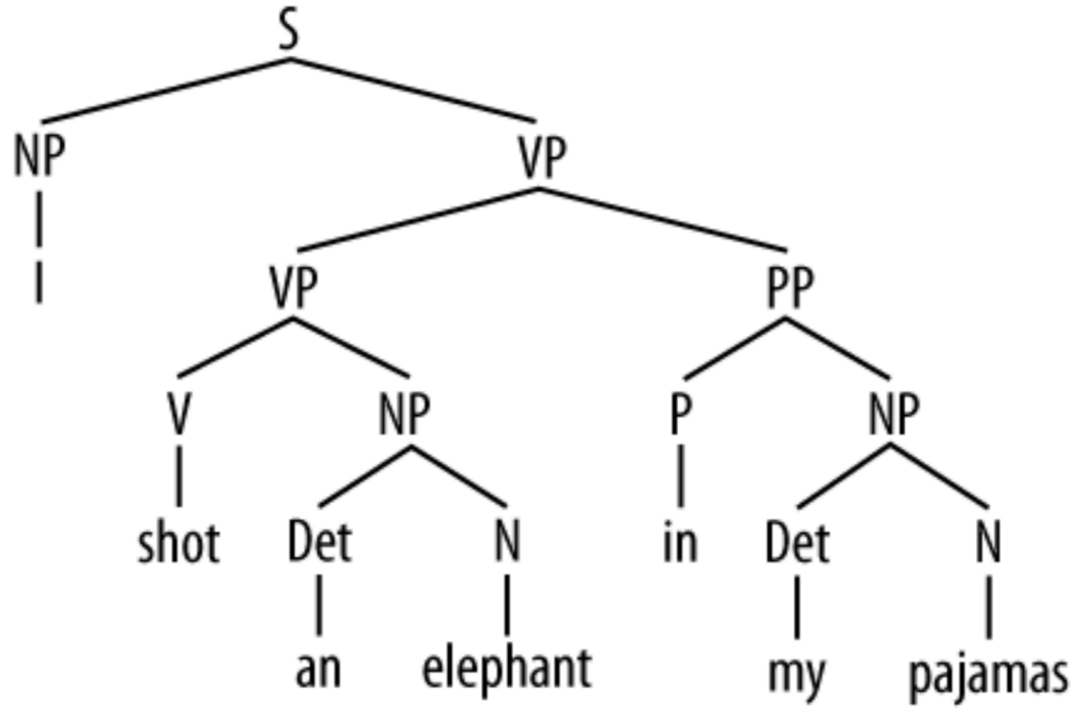
Stack
(**NP** (Det my) (N pajams))
(P in)
(S (NP I) (VP (V Shot) (NP (Det an) (N elephant))

Queue

Given Stack the Queue, predict to

- Shift ? to Stack
- Reduce the top two elements of the Stack

Example of Reduce



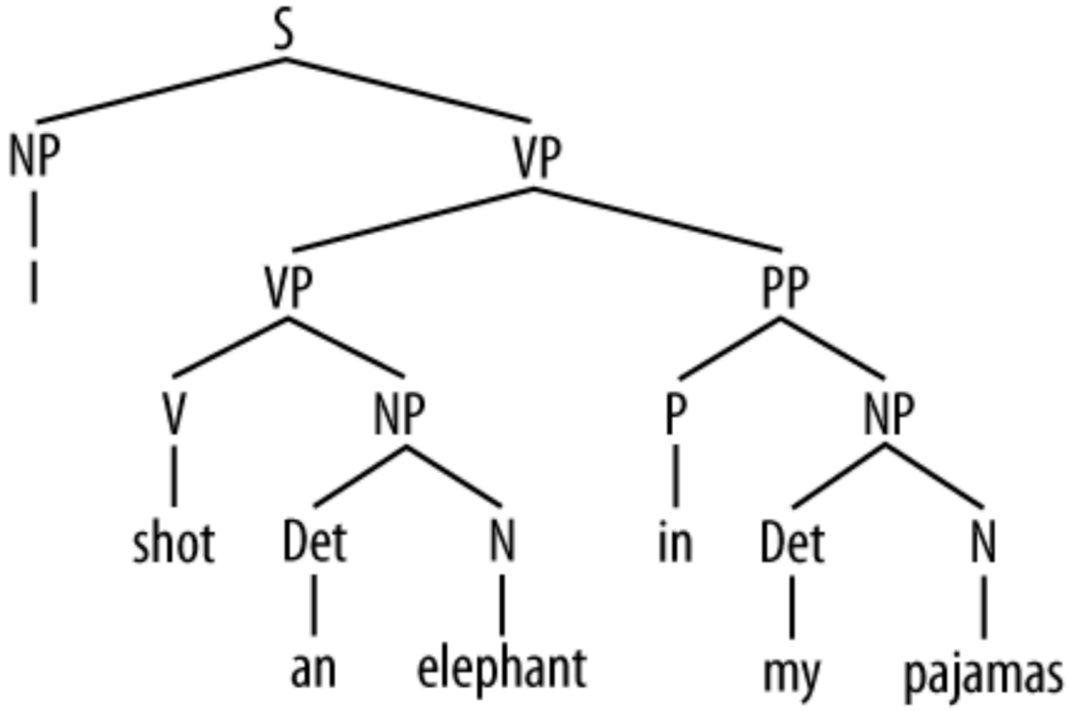
Stack
(**PP** (P in) (NP (Det my) (N pajams))
(S (NP I) (VP (V Shot) (NP (Det an) (N elephant))

Queue

Given Stack the Queue, predict to

- Shift ? to Stack
- Reduce the top two elements of the Stack

Example of Reduce



Stack

(S
  (NP I)
  (**VP**
    (VP
      (V Shot)
      (NP (Det an)
      (N elephant)
    (PP (P in)
      (NP (Det my)
        (N pajams))))

Queue

Given Stack the Queue, predict to

- Shift ? to Stack
- Reduce the top two elements of the Stack

CKY vs Shift-Reduce

- CKY
  - High computational cost. Slow.
  - Considering global information to find the optimal parse tree.
- Shift-Reduce
  - Faster than CKY
  - Mainstream. Why?

Resources

Treebanks

- Treebank: a collection of parsed sentences for building statistical parsers.
  - Penn Treebank
  - Penn Chinese Treebank
  - CKIP
- Treebank is the important resource for training statistical parsers.

Annotation of the Penn Treebank

- Constituent parse of the sentence
  - *factory inventories fell 0.1% in September the first decline since February 1987.*

```
(S (NP-SBJ factory inventories)
   (VP (VP fell
           (NP-EXT 0.1 %)
           (PP-TMP in
                   (NP September)))
       ,
       (NP (NP the first decline)
           (PP-TMP since
                   (NP February 1987)))))
```

Phrase Types in the Penn Treebank

Tag	Type
S	Sentence
SBAR	Clause with complementizer
SBARQ	Wh-question clauses
NP	Noun phrases
VP	Verb phrases
PP	Prepositional phrases
QP	Quantifier phrases

Tag	Type
CONJP	Multiword conjunctional phrases
ADJP	Adjective phrases
ADVP	Adverbial phrases
WHNP	Wh-noun phrases
WHPP	Wh-prepositional phrases
WH-ADJP	Wh-adjective phrases
WH-ADVP	Wh-adverb phrases

Chinese Treebank

```
(IP (NP-SBJ (NN 中资))
    (VP (ADVP (AD 已))
        (VP (VV 成为)
            (NP-OBJ (NP-PN (NR 澳门))
                    (CP (WHNP-1 (-NONE- *OP*))
                        (CP (IP (NP-SBJ (-NONE- *T*-1))
                                (VP (ADVP (AD 最))
                                    (VP (VA 大))))
                            (DEC 的) ))
                    (ADJP (JJ 外来))
                    (NP (NN 投资者)))))
(PU 。))
```

Chinese Treebank 9.0 (2016)

Data Unit	Size
Documents	3,726
Sentences	132,076
Words	2,084,387
Characters	3,246,331

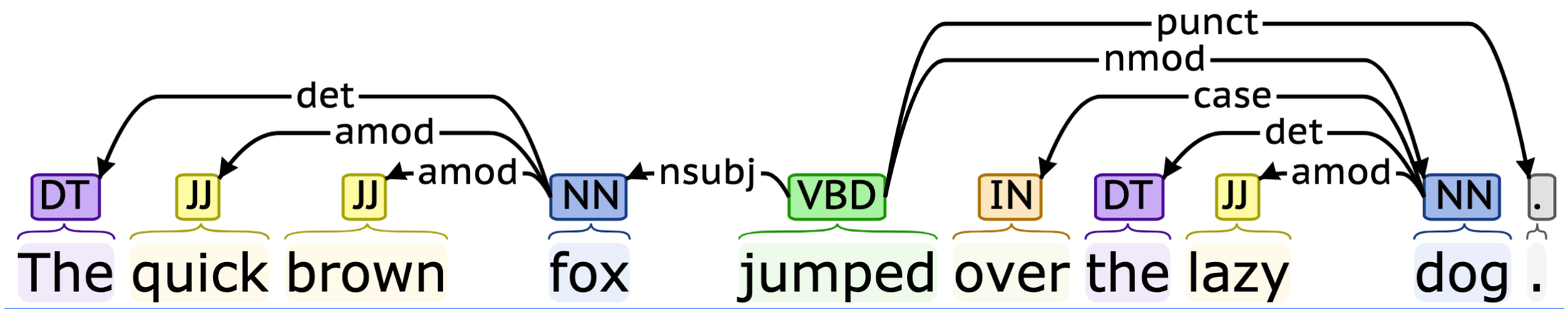
Annotated with word boundary, POS tags, and constituent

Pre-trained Parser

- Stanford parser
  - CoreNLP
- CKIP CoreNLP
- NLTK

Stanford CoreNLP

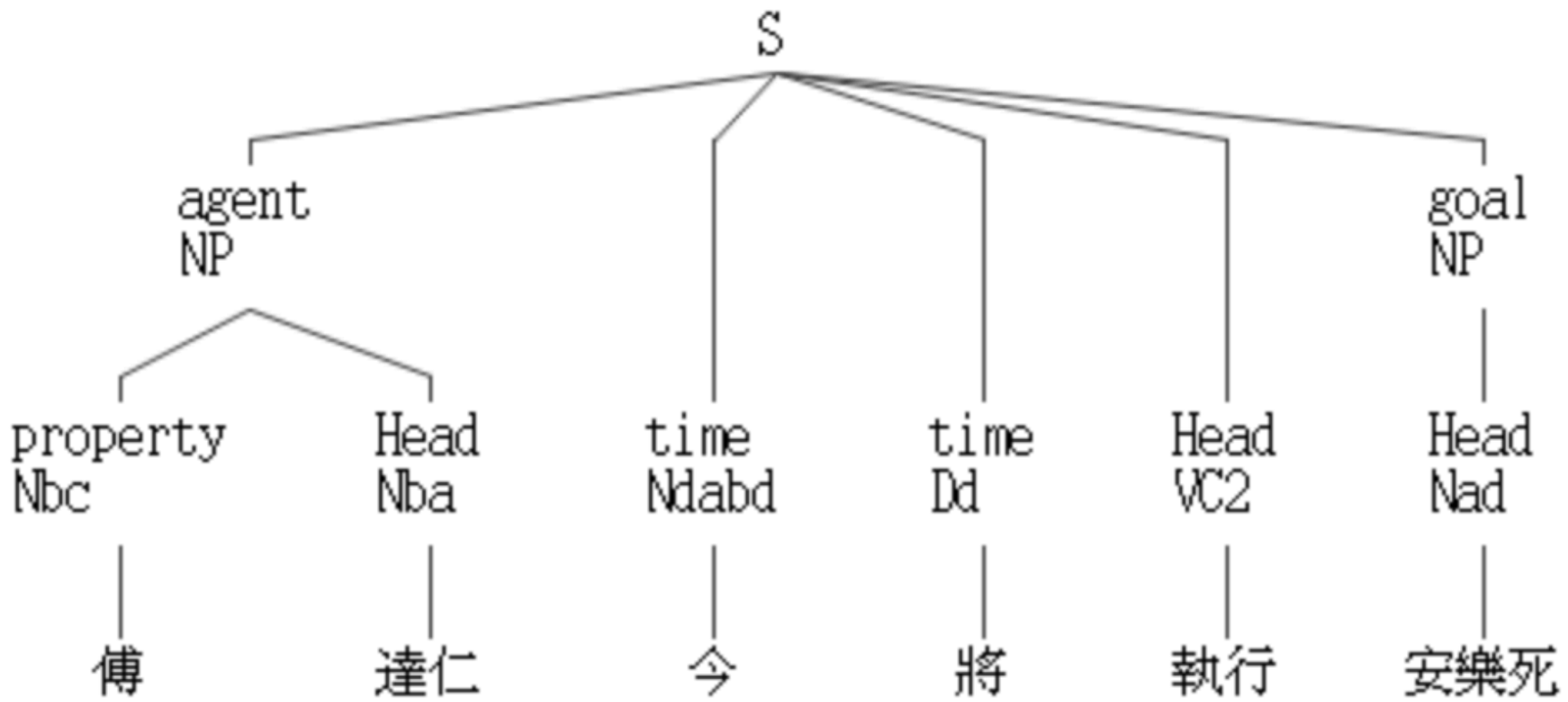
- Both constituent parsing and dependency parsing
- Arabic, Chinese, English, French, German, Spanish



https://corenlp.run/

CKIP CoreNLP

- Maintained by Academia Sinica 詞庫小組
- Clause-based constituent parsing



http://ckip.iis.sinica.edu.tw/service/corenlp/