CS 4650/7650:
Natural Language Processing

Constituency Parsing

Diyi Yang

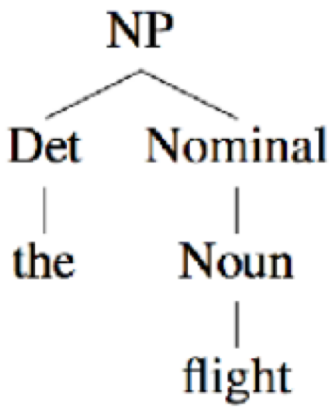
Many slides from Yulia Tsvetkov, Greg Durrett, David Bamman, and others

Logistics

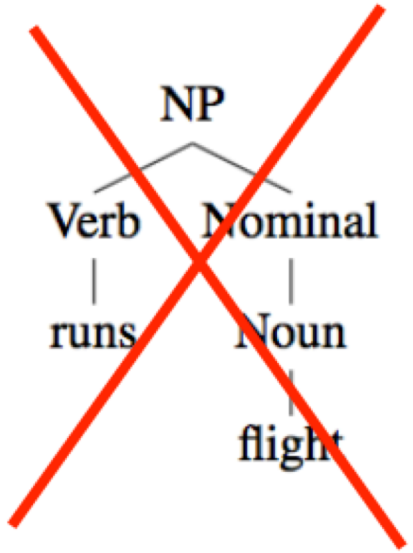
- Midterm review: Feb 24th
- Midterm: Feb 26th, 3:00-4:15pm
  - One-page cheat sheet

Context-free Grammar

- A CFG gives a formal way to define what meaningful constituents are and exactly how a constituent is formed out of other constituents (or words). It defines valid structure in a language.



NP → Det Nominal



NP → Verb Nominal

Context-free Grammar

- A context-free grammar defines how symbols in a language combine to form valid structures

NP	→	Det Nominal
NP	→	ProperNoun
Nominal	→	Noun | Nominal Noun
Det	→	a | the
Noun	→	flight

non-terminals

lexicon/
terminals

Context-free Grammar

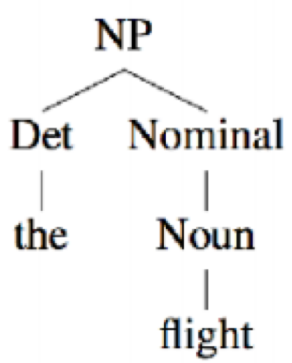
N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ $\beta \in (\Sigma, N)$	$S \rightarrow$ NP VP Noun $\rightarrow$ dog
S	Start symbol	

Infinite Strings with Finite Productions

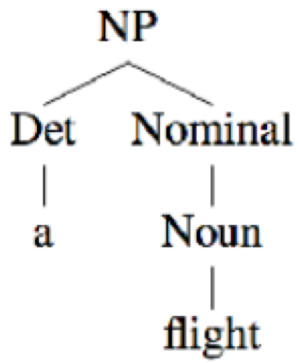
- This is the house
- This is the house that Jack built
- This is the cat that lives in the house that Jack built
- This is the dog that chased the cat that lives in the house that Jack built
- This is the flea that bit the dog that chased the cat that lives in the house the Jack built
- This is the virus that infected the flea that bit the dog that chased the cat that lives in the house that Jack built

Derivation

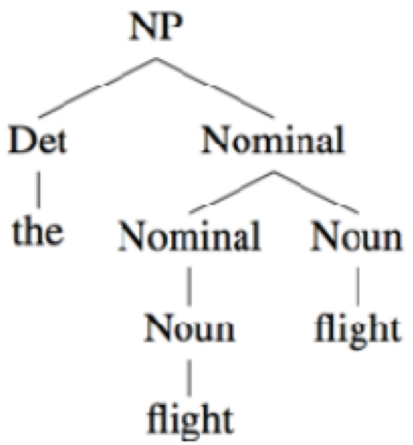
Given a CFG, a derivation is the sequence of productions used to generate a string of words (e.g., a sentence), often visualized as a parse tree



the flight

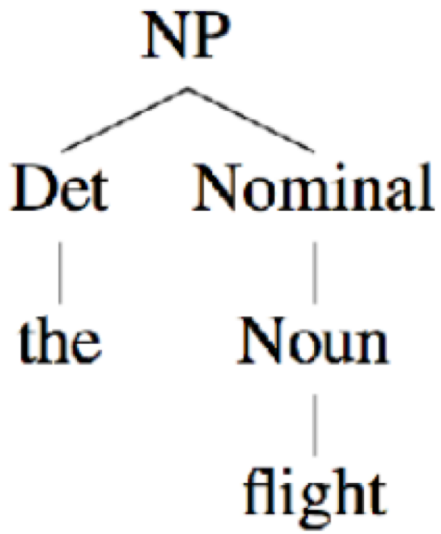


a flight



the flight flight

Bracketed Notation



[NP [Det the] [Nominal [Noun flight]]]

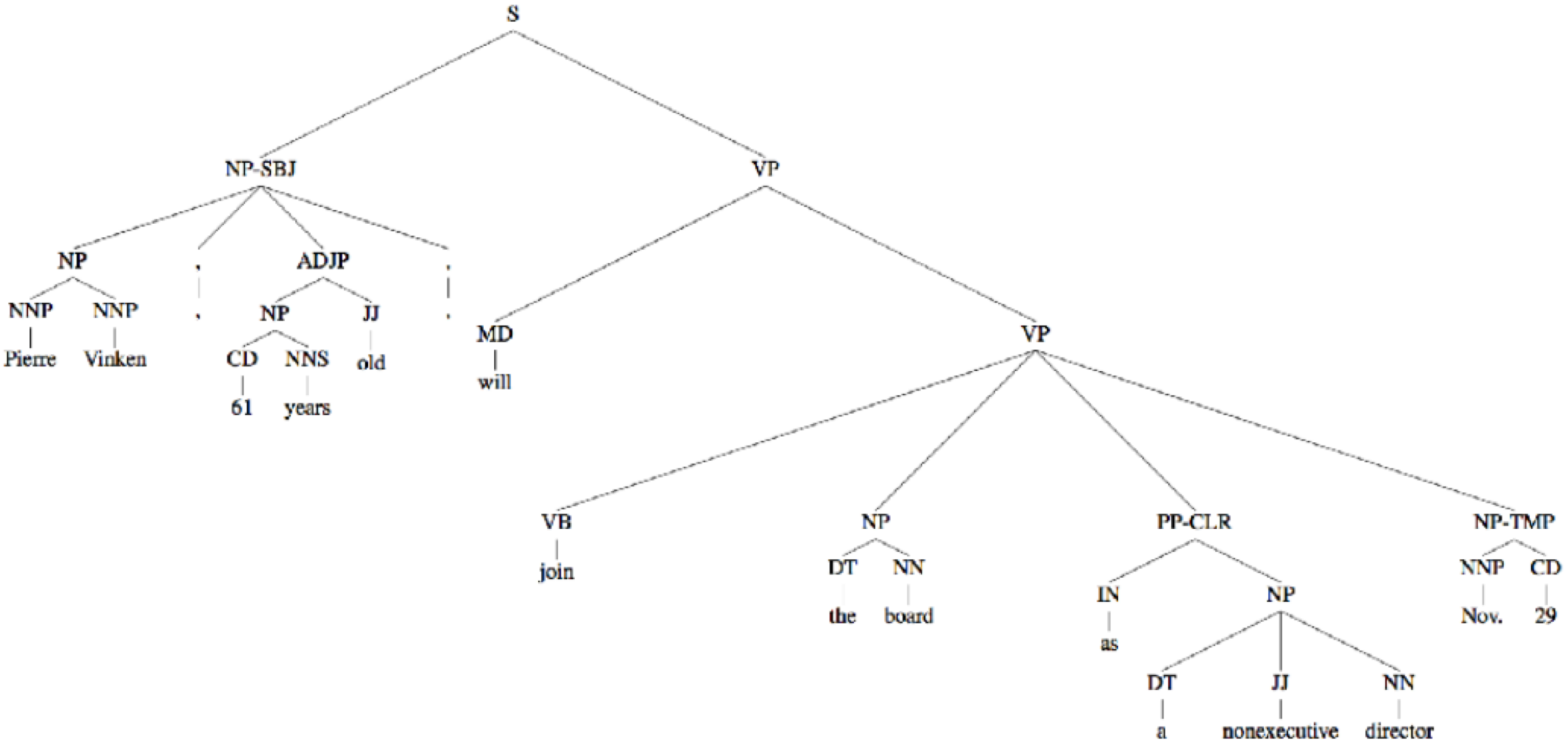
Treebanks

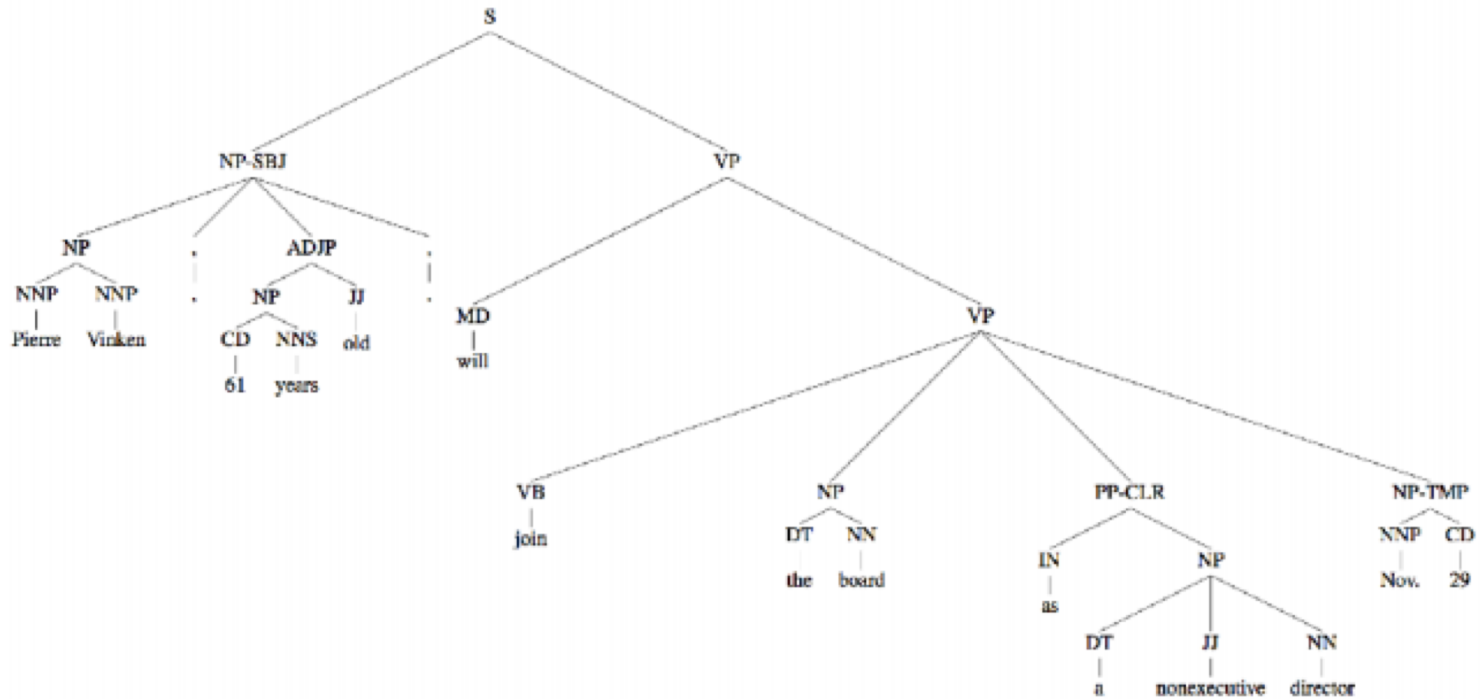
- Rather than create the rules by hand, we can annotate sentences with their syntactic structure and then extract the rules from the annotations

- Treebanks:

Collections of sentences annotated with syntactic structure

Penn Treebank





NP	→	NNP NNP
NP-SBJ	→	NP , ADJP ,
S	→	NP-SBJ VP
VP	→	VB NP PP-CLR NP-TMP

Summary: Syntax Through The Use Of Context-free Grammar

- Groups of consecutive words act as a group or a constituent
- A context-free grammar consists of a set of rules or productions, expressed over a set of non-terminal symbols and a set of terminal symbols.
- A particular context-free language is the set of strings that can be derived from a particular context-free grammar
- Verbs can be subcategorized by the types of complements they expect. Simple subcategories are transitive and intransitive
- Treebanks of parsed sentences exist for many genres of English and for many languages.

Summary: Syntax Through The Use Of Context-free Grammar

- Groups of consecutive words act as a group or a **constituent**
- A **context-free grammar** consists of a set of **rules** or **productions**, expressed over a set of **non-terminal** symbols and a set of **terminal** symbols.
- A particular **context-free language** is the set of strings that can be derived from a particular **context-free grammar**
- Verbs can be **subcategorized** by the types of **complements** they expect. Simple subcategories are **transitive** and **intransitive**
- **Treebanks** of parsed sentences exist for many genres of English and for many languages.

CFG

- A basic CFG allows us to check whether a sentence is grammatical in the language it defines
- Binary decision: a sentence is either in the language (a series of productions yields the words we see) or it is not
- Where would this be useful?

PCFG

- Probabilistic context-free grammar: each production is also associated with a probability
- This lets us calculate the probability of a parse for a given sentence; for a given parse tree T for sentence S comprised of n rules from R (each $A \rightarrow \beta$):

$$\mathrm{P}(T, S) = \prod_{i=1}^{n} P(\beta | A)$$

PCFG

N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ [p] $p = P(\beta \mid A)$	S → NP VP Noun → dog
S	Start symbol	

Estimating PCFGs

How do we calculate $P(A \rightarrow \beta)$?

Maximum likelihood estimates

Estimating PCFGs

$$\sum_{\beta} P(\beta \mid A) = \frac{C(A \rightarrow \beta)}{\sum_{\gamma} C(A \rightarrow \gamma)}$$

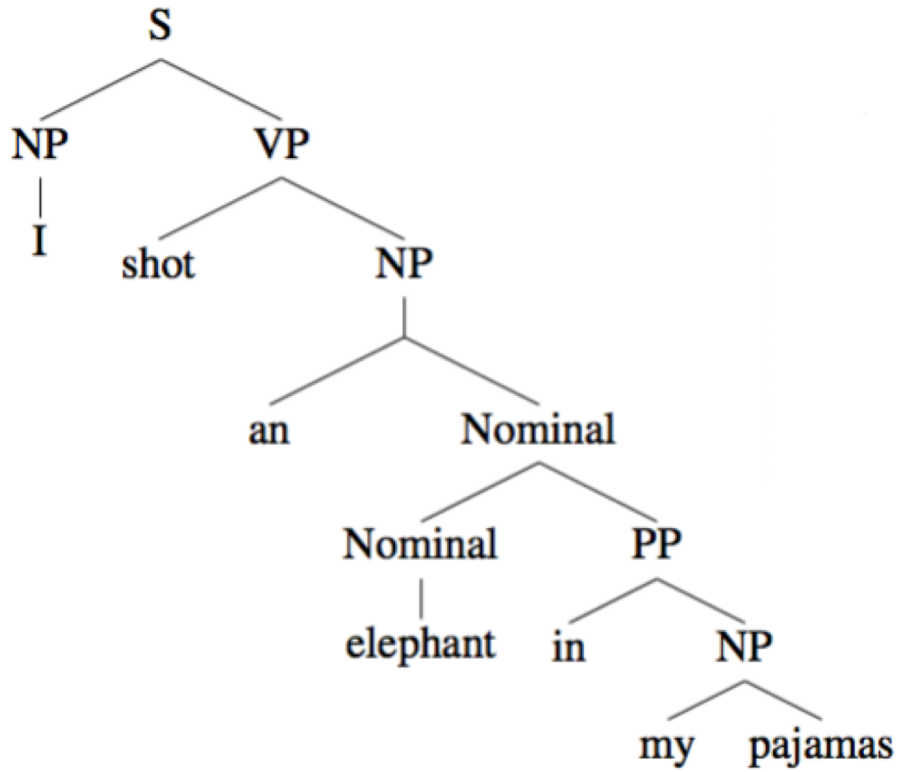
(equivalently)

$$\sum_{\beta} P(\beta \mid A) = \frac{C(A \rightarrow \beta)}{C(A)}$$

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

- CNF and CKY Parsing

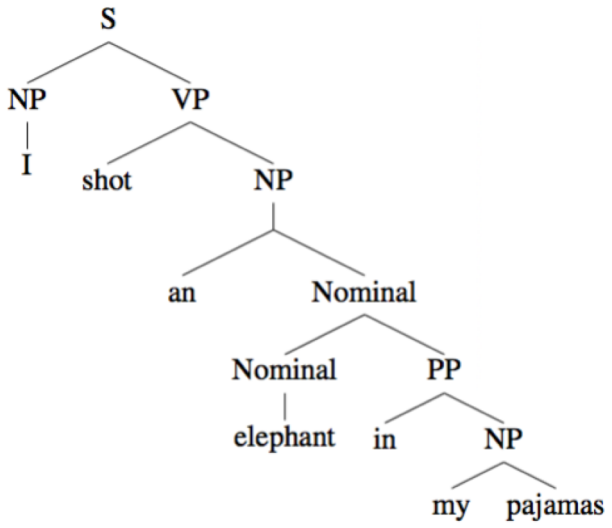
Constituents



Every internal node is a phrase

- My pajamas
- In my pajamas
- Elephant in my pajamas
- an elephant in my pajamas
- Shot an elephant in my pajamas
- I shot an elephant in my pajamas

Constituents



Each phrase could be replaced by another of the same type of constituent

Every internal node is a phrase

- My pajamas
- In my pajamas
- Elephant in my pajamas
- an elephant in my pajamas
- Shot an elephant in my pajamas
- I shot an elephant in my pajamas

Context-Free Grammar

N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ $\beta \in (\Sigma, N)$	NP → DT JJ NN Noun → dog
S	Start symbol	

Chomsky Normal Form (CNF)

N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ β = single terminal (from Σ) or two non-terminals (from N)	S → NP VP Noun → dog
S	Start symbol	

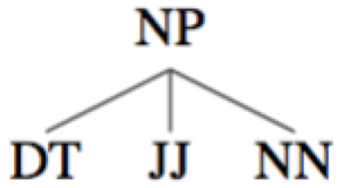
Context-free Grammar vs. Chomsky Normal Form

- A context-free grammar is in Chomsky normal form (CNF) if
  - each production is either $A \rightarrow B\ C$, $A \rightarrow a$, or $S \rightarrow \varepsilon$

Chomsky Normal Form (CNF)

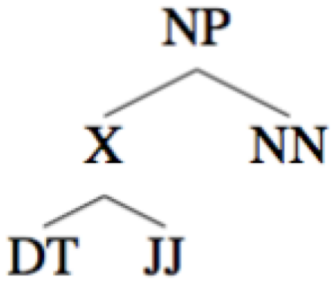
- Any CFG can be converted into weakly equivalent CNF (recognizing the same set of sentences as existing in the grammar but differing in their derivation)

$$NP \rightarrow DT\ JJ\ NN$$



$$NP \rightarrow X\ NN$$

$$X \rightarrow DT\ JJ$$



S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	NN
Nominal	→	NNS
Nominal	→	PRP
PP	→	IN NP
NP	→	DT NN
NP	→	Nominal
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
NN	→	elephant
NNS	→	pajamas
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

CKY

- Cocke-Kasami-Younger algorithm (also CYK) for parsing for a grammar expressed in CNF.
- Bottom-up dynamic programming

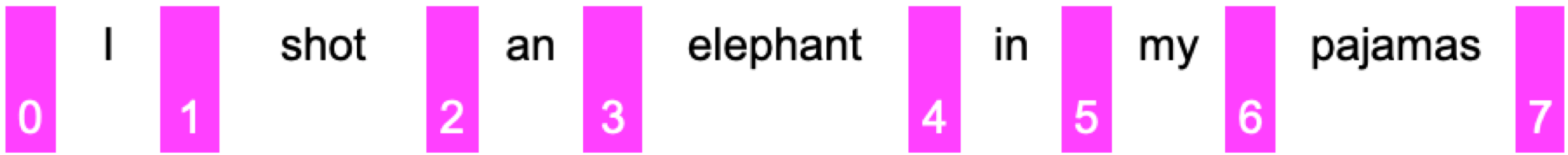
CKY

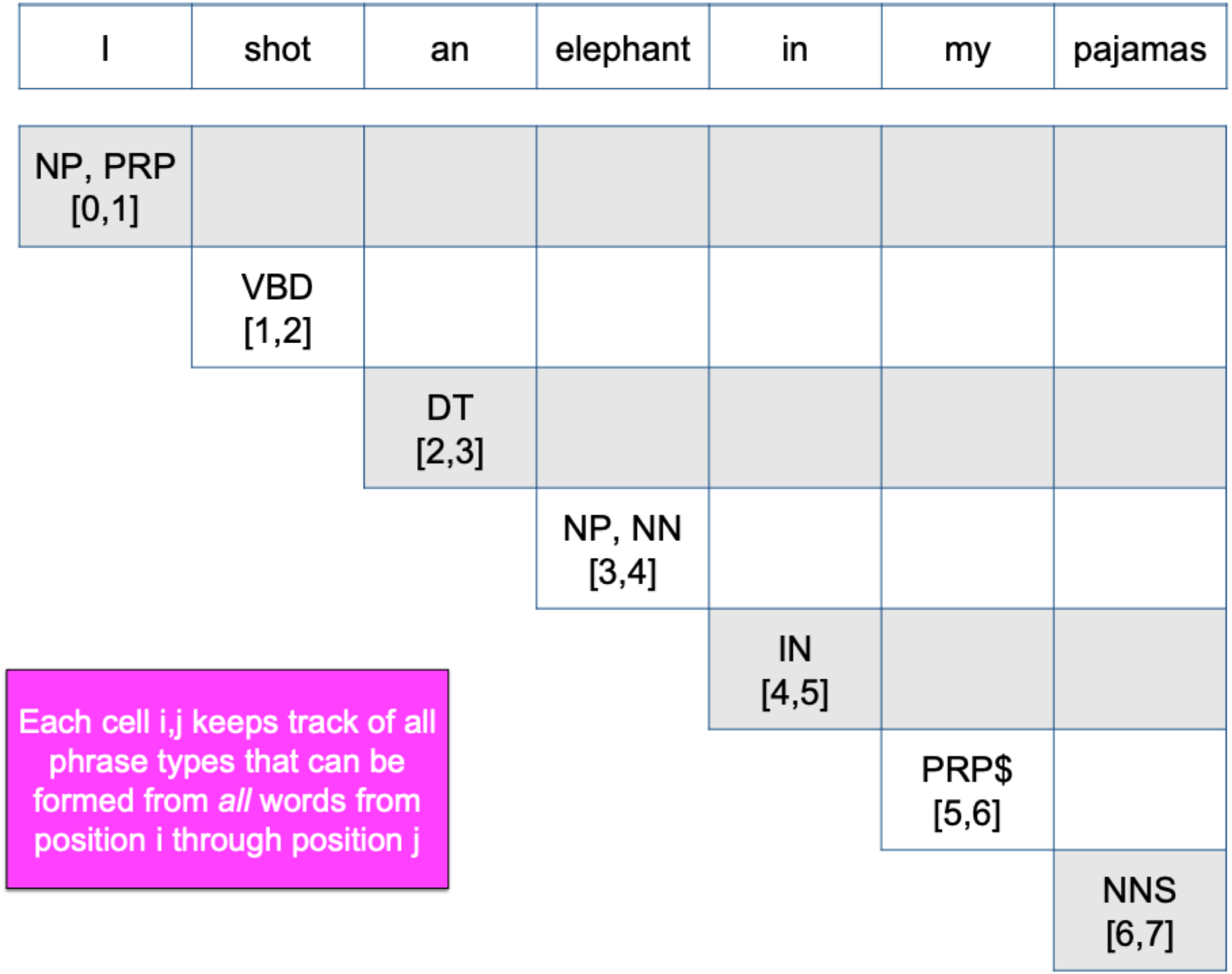
- The subproblems for our dynamic program are all possible spans of contiguous words in the sentence.

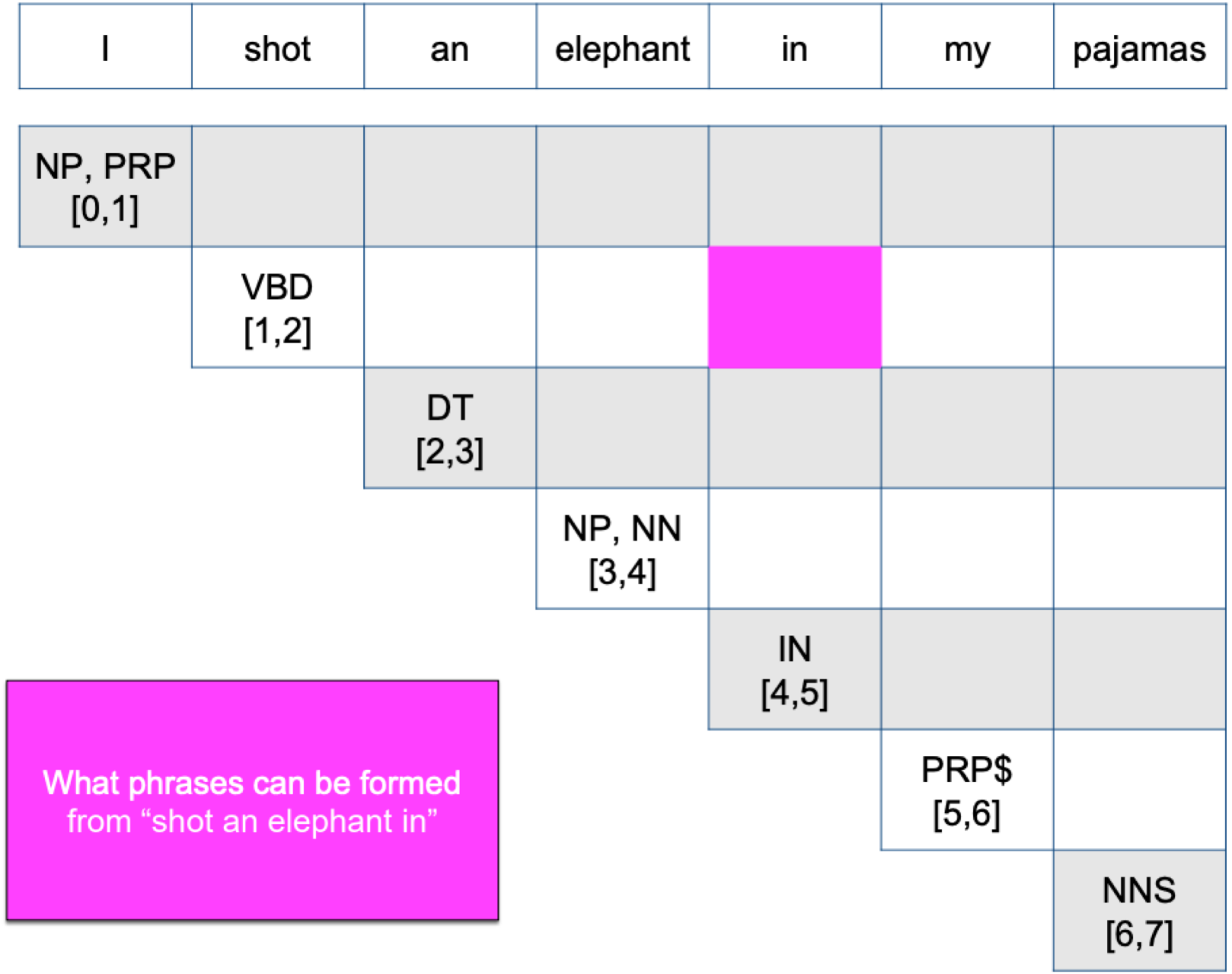
S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

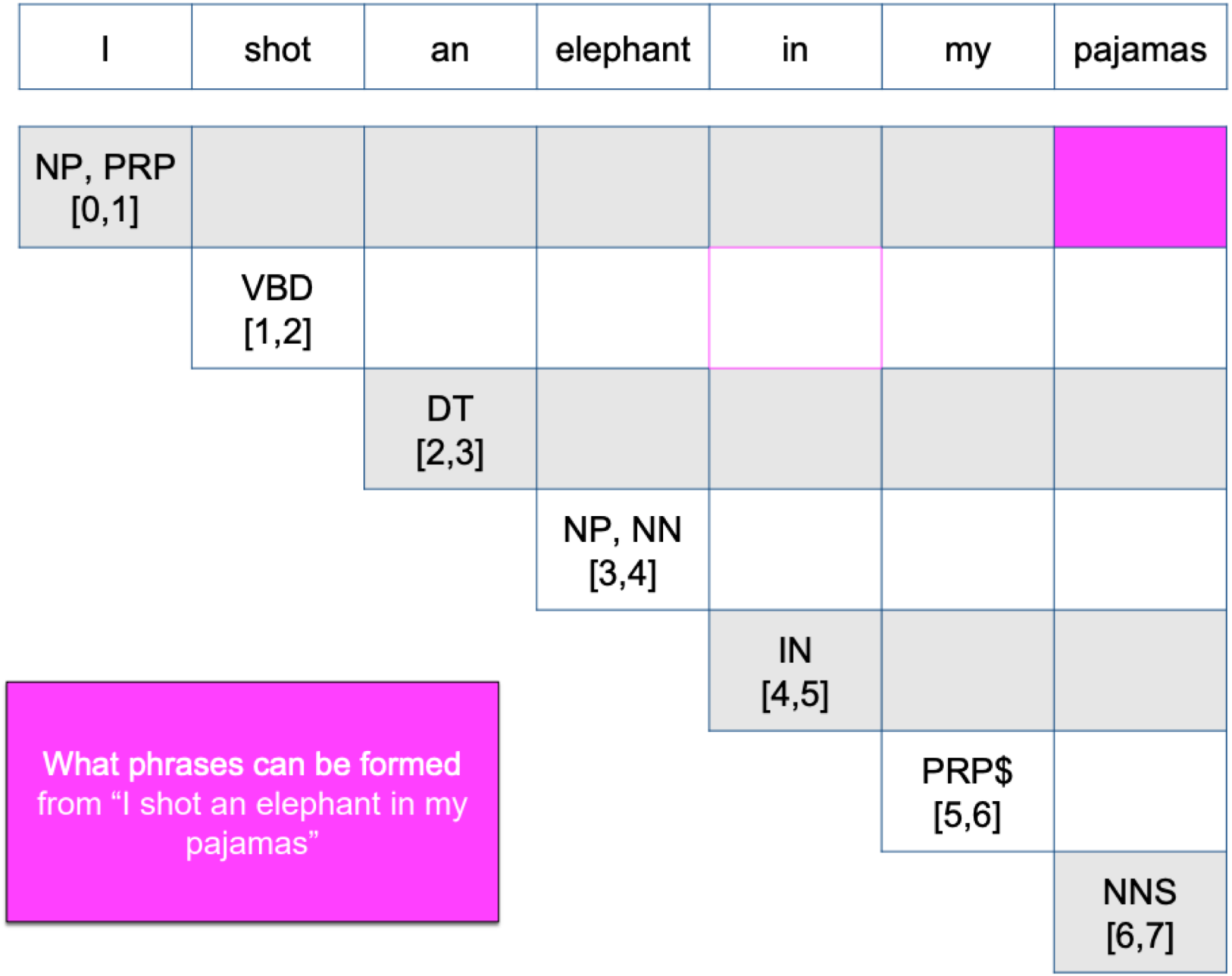
VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas


0
I
1
shot
2
an
3
elephant
4
in
5
my
6
pajamas
7


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
Each cell i,j keeps track of all phrase types that can be formed from *all* words from position i through position j


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
What phrases can be formed
from “shot an elephant in”


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
What phrases can be formed from “I shot an elephant in my pajamas”

CKY

- In CNF, each non-terminal generates two non-terminals

$$S \rightarrow NP\ VP$$

[S [NP I] [VP shot an elephant in my pajamas]]

- The recursive step of our dynamic program needs to select a split point and rule
- If the parent spans tokens i-j, then there is a k where the children span i-k and k-j.

Completed Parse Table

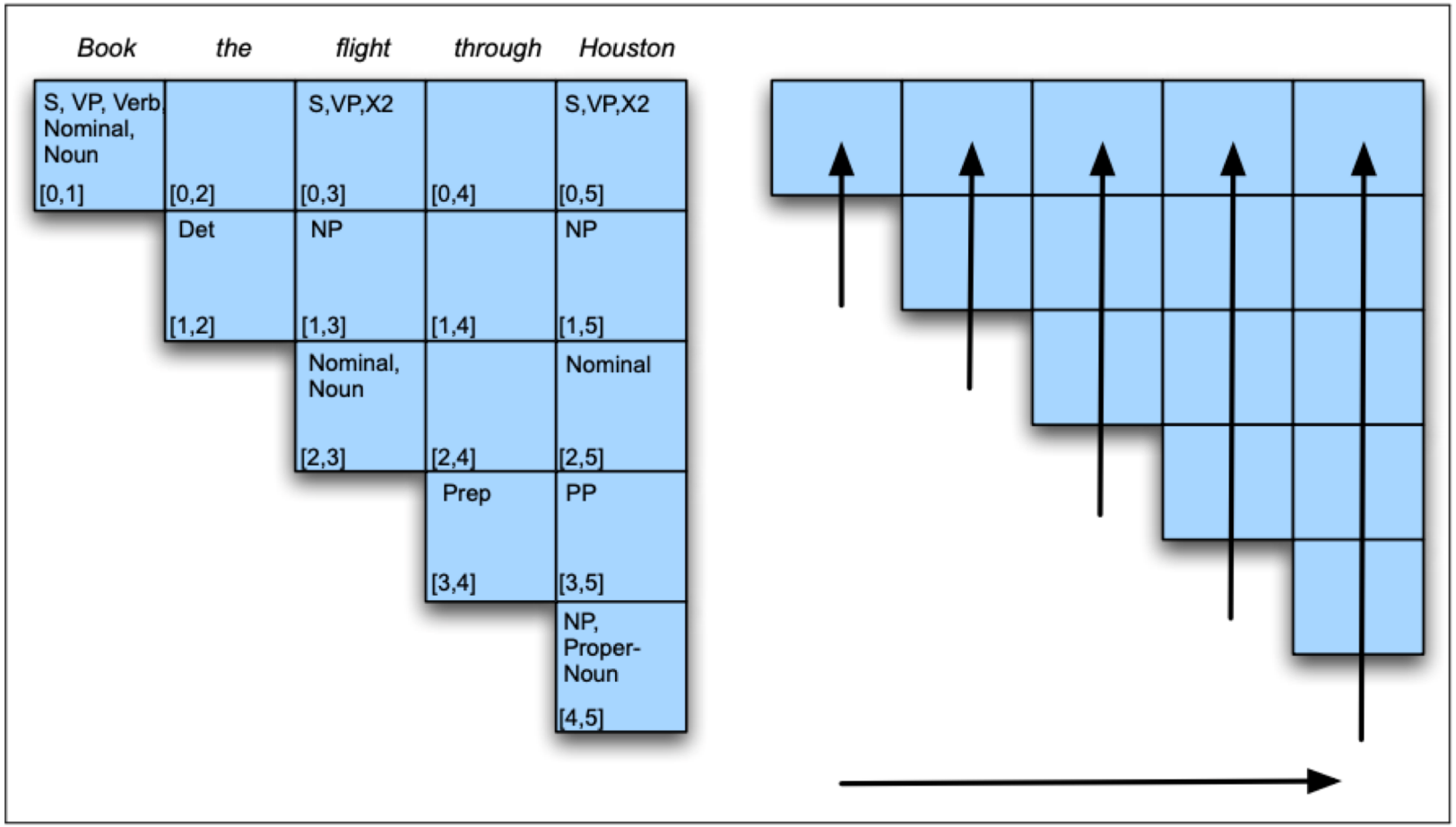


Figure 13.4 Completed parse table for *Book the flight through Houston.*

CKY Algorithm

```
function CKY-PARSE(words, grammar) returns table

  for j ← from 1 to LENGTH(words) do
      for all {A | A → words[j] ∈ grammar}
              table[j − 1, j] ← table[j − 1, j] ∪ A
      for i ← from j − 2 downto 0 do
          for k ← i + 1 to j − 1 do
              for all {A | A → BC ∈ grammar and B ∈ table[i, k] and C ∈ table[k, j]}
                      table[i,j] ← table[i,j] ∪ A
```

Figure 13.5 The CKY algorithm.

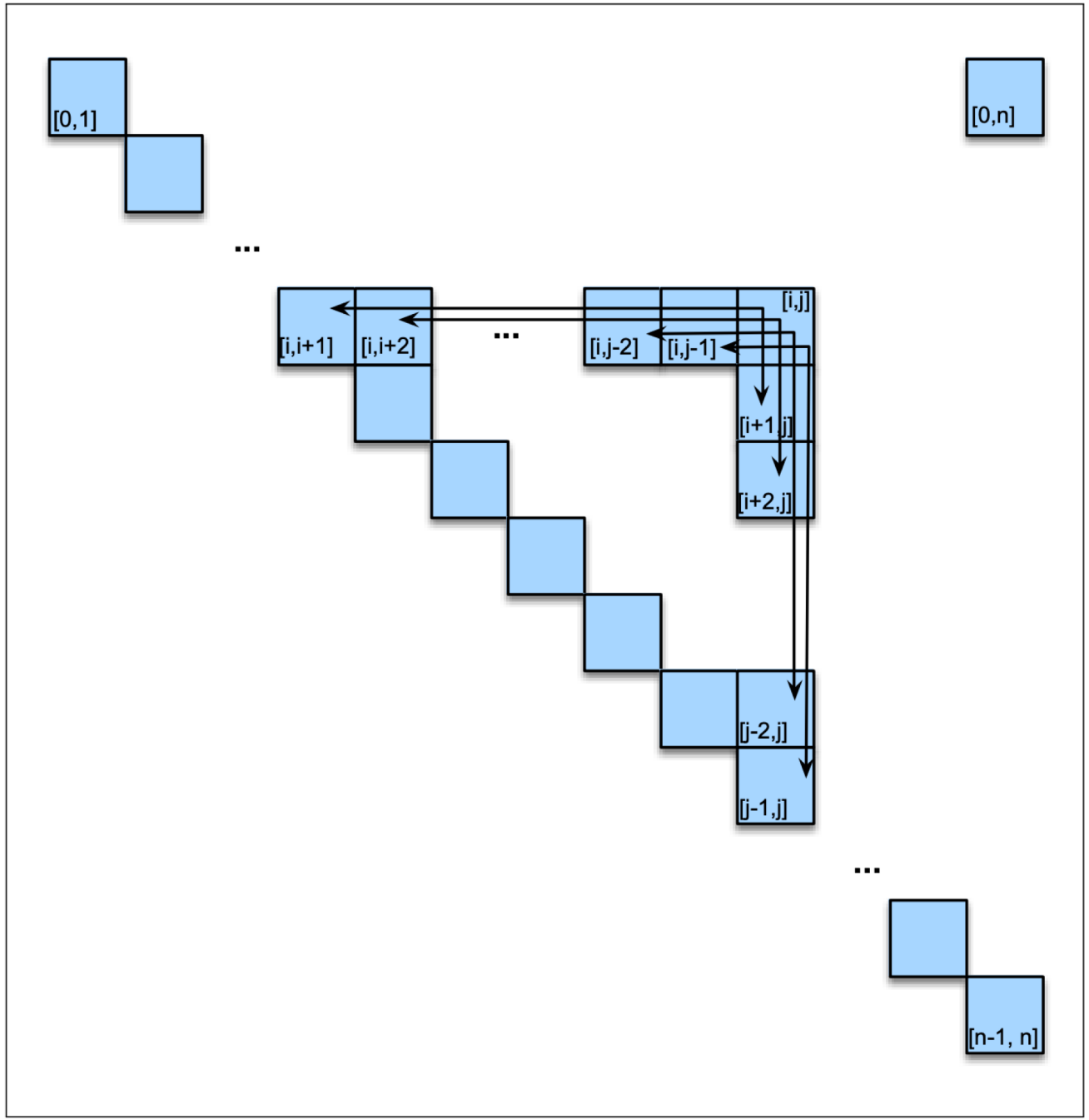
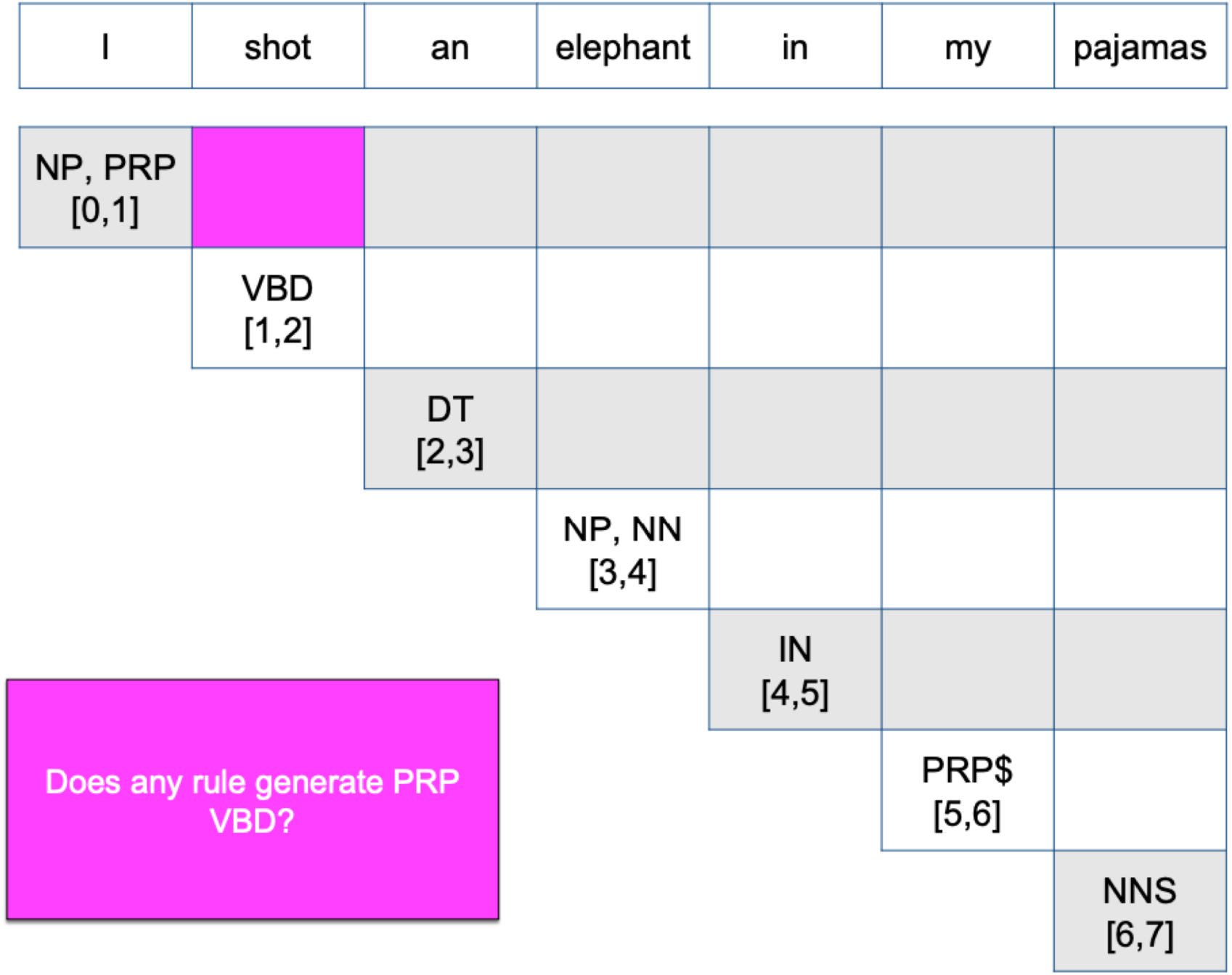
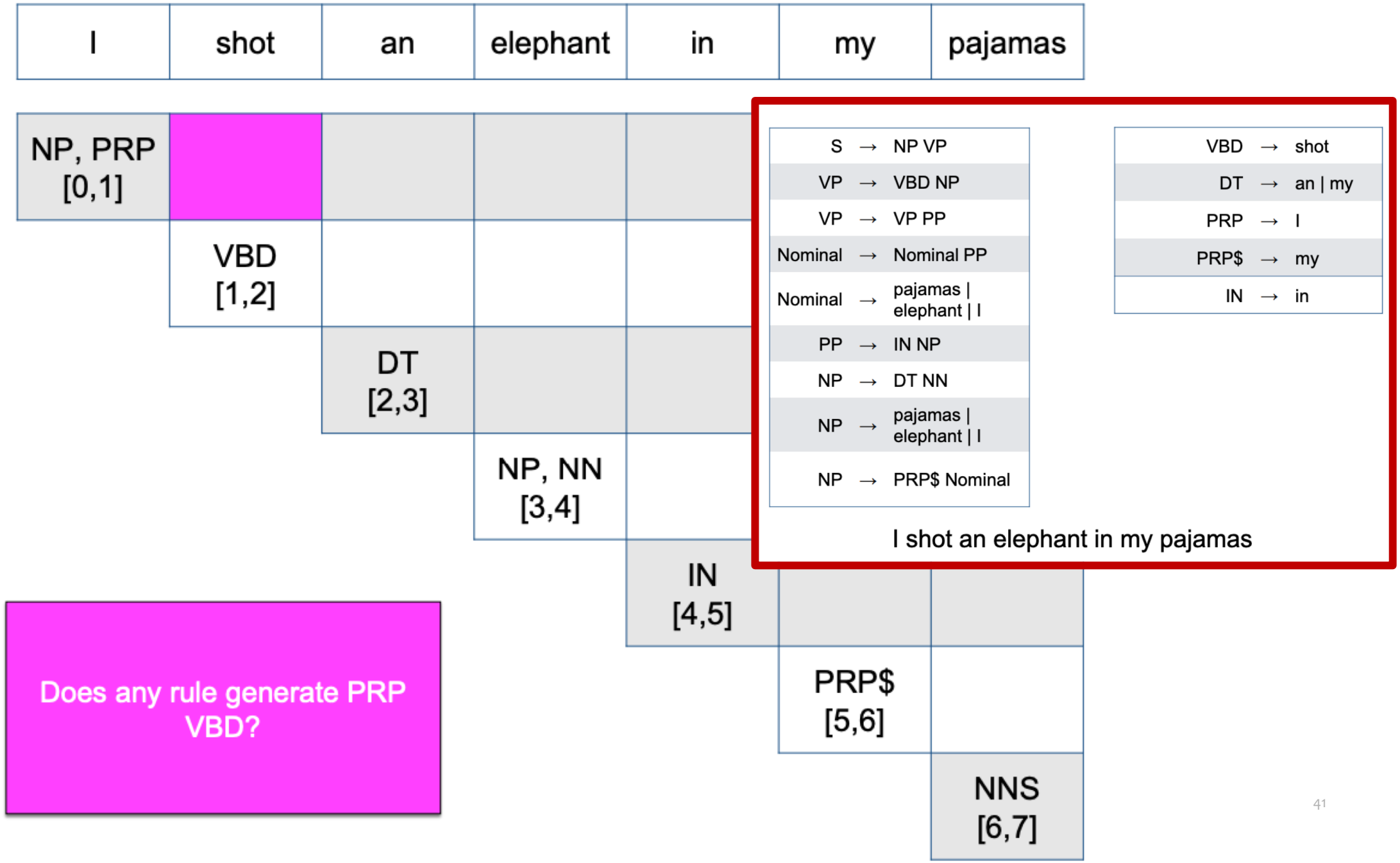
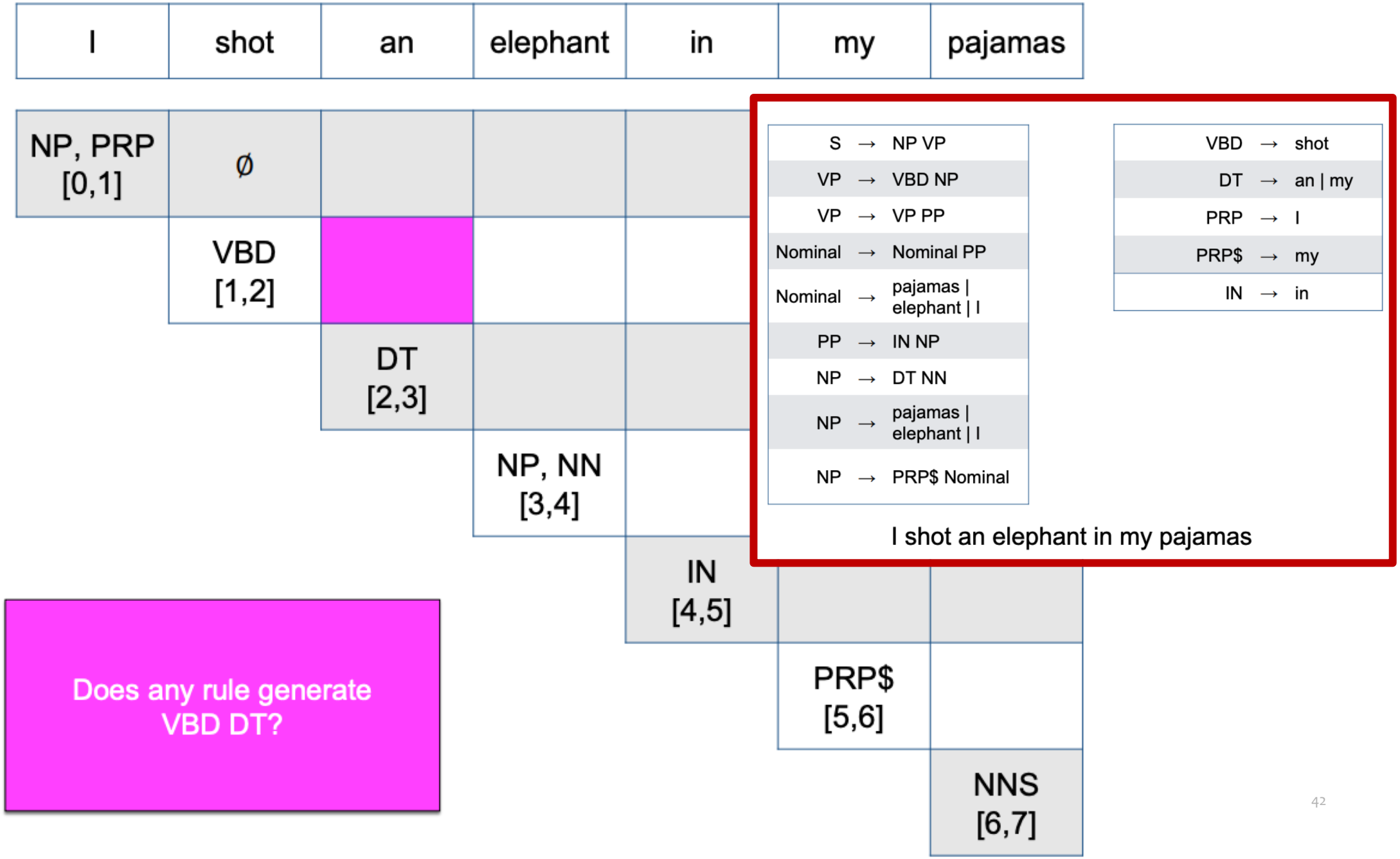
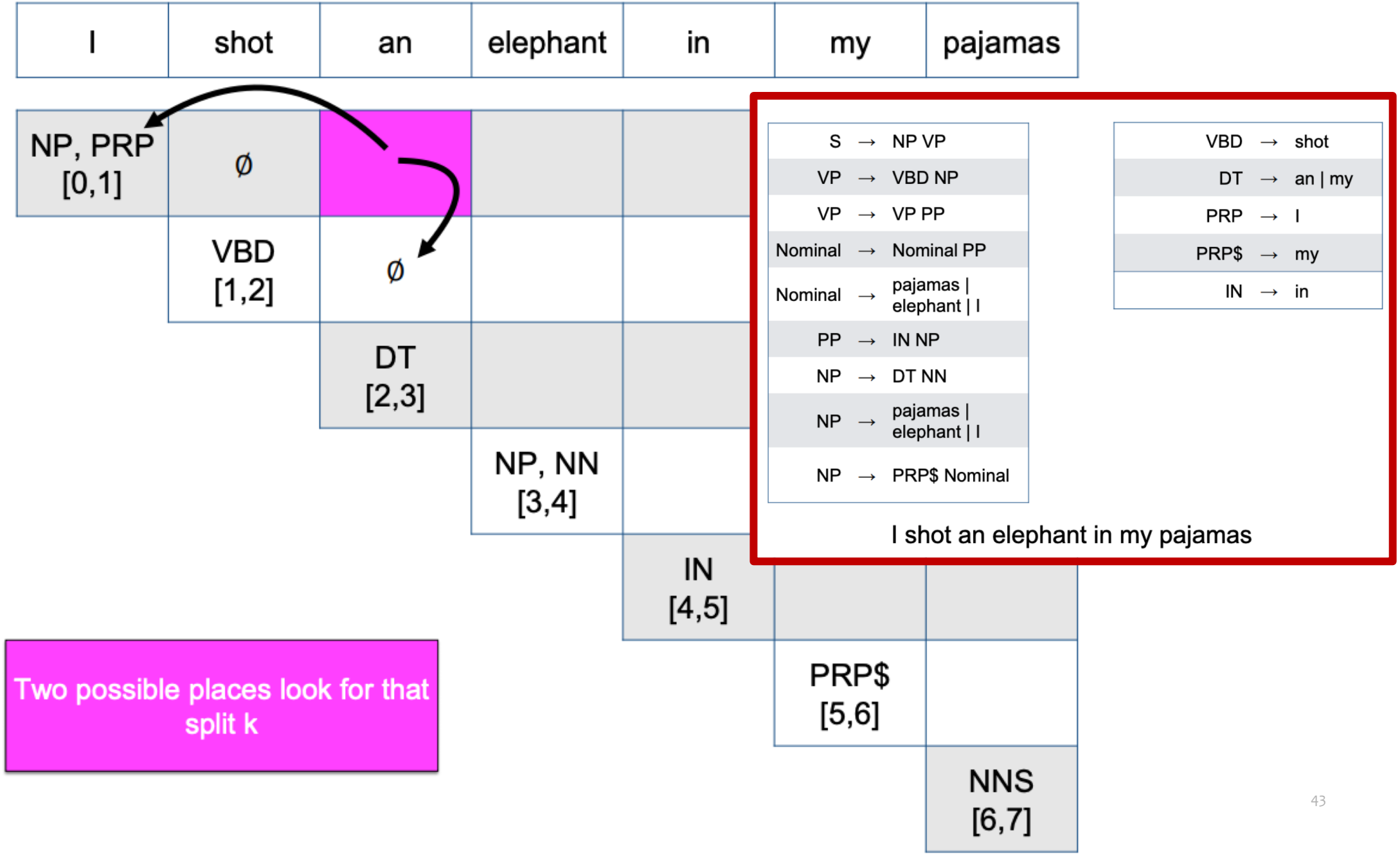


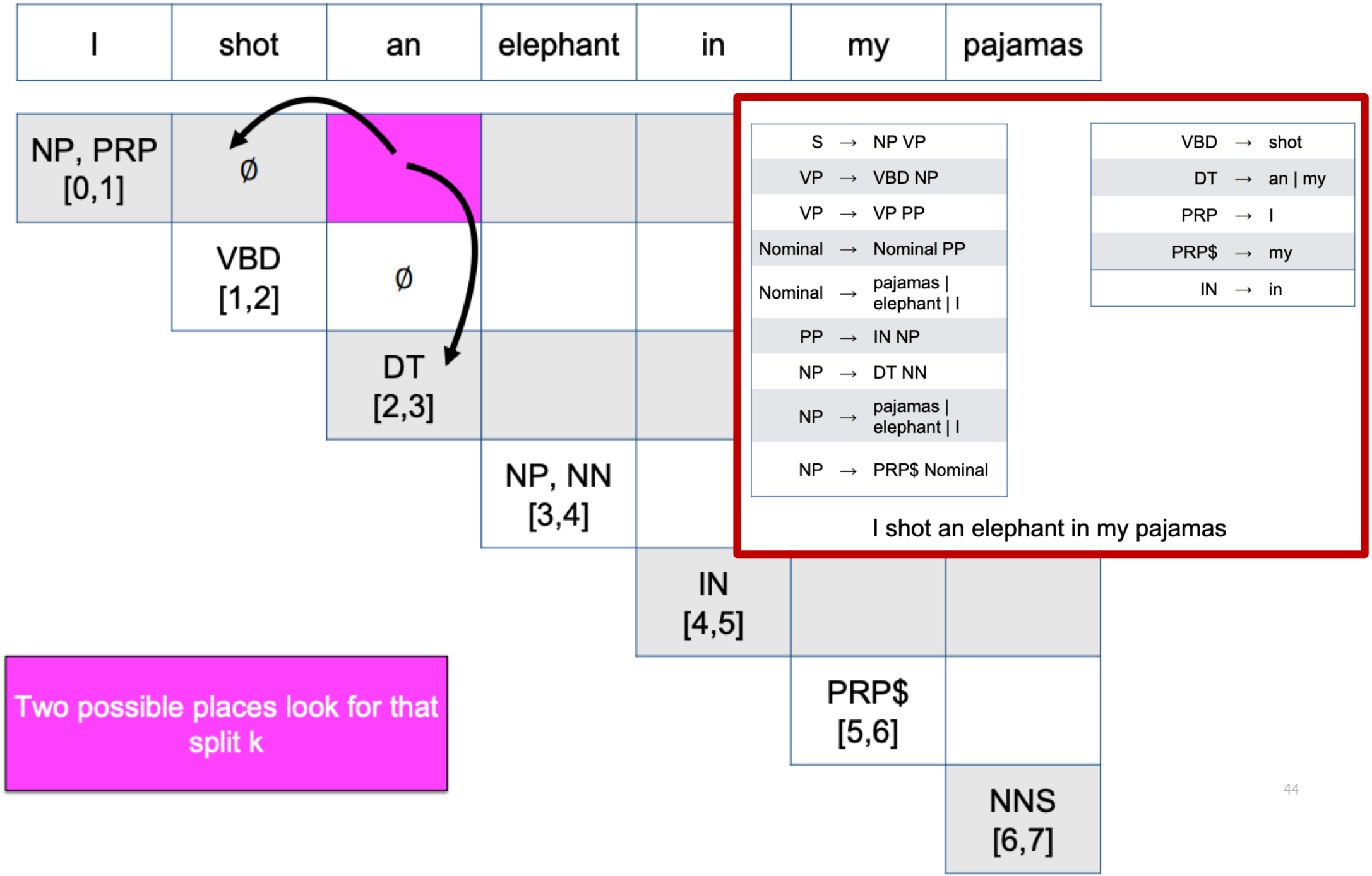
Figure 13.6 All the ways to fill the $[i, j]$th cell in the CKY table.


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
Does any rule generate PRP VBD?


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Does any rule generate PRP VBD?


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
VBD
[1,2]
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Does any rule generate VBD DT?


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
VBD
[1,2]
Ø
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Two possible places look for that split k


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
VBD
[1,2]
Ø
DT
[2,3]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Two possible places look for that split k

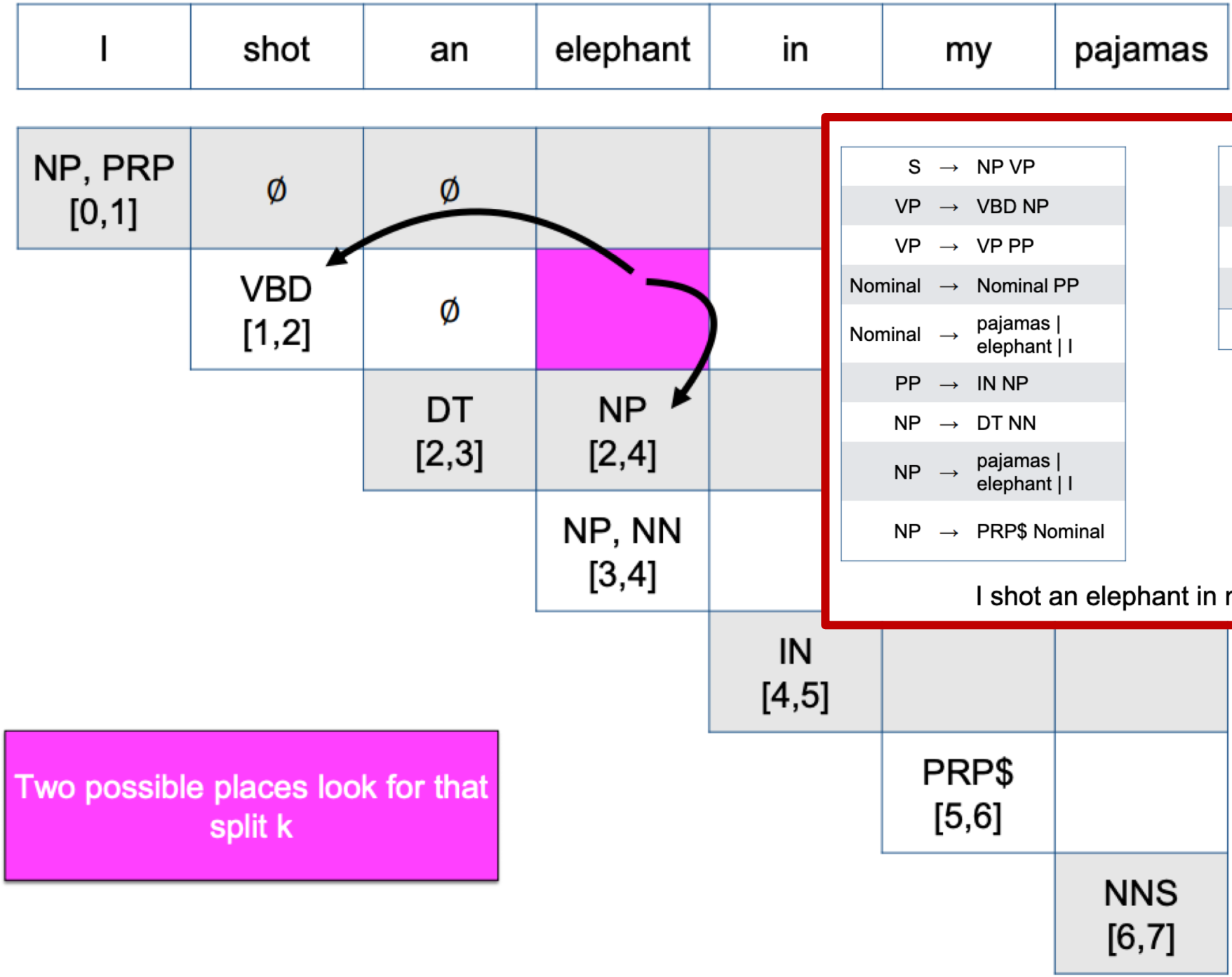
I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø				
	VBD [1,2]	Ø				
		DT [2,3]				
			NP, NN [3,4]			
				IN [4,5]		
					PRP$ [5,6]	
						NNS [6,7]

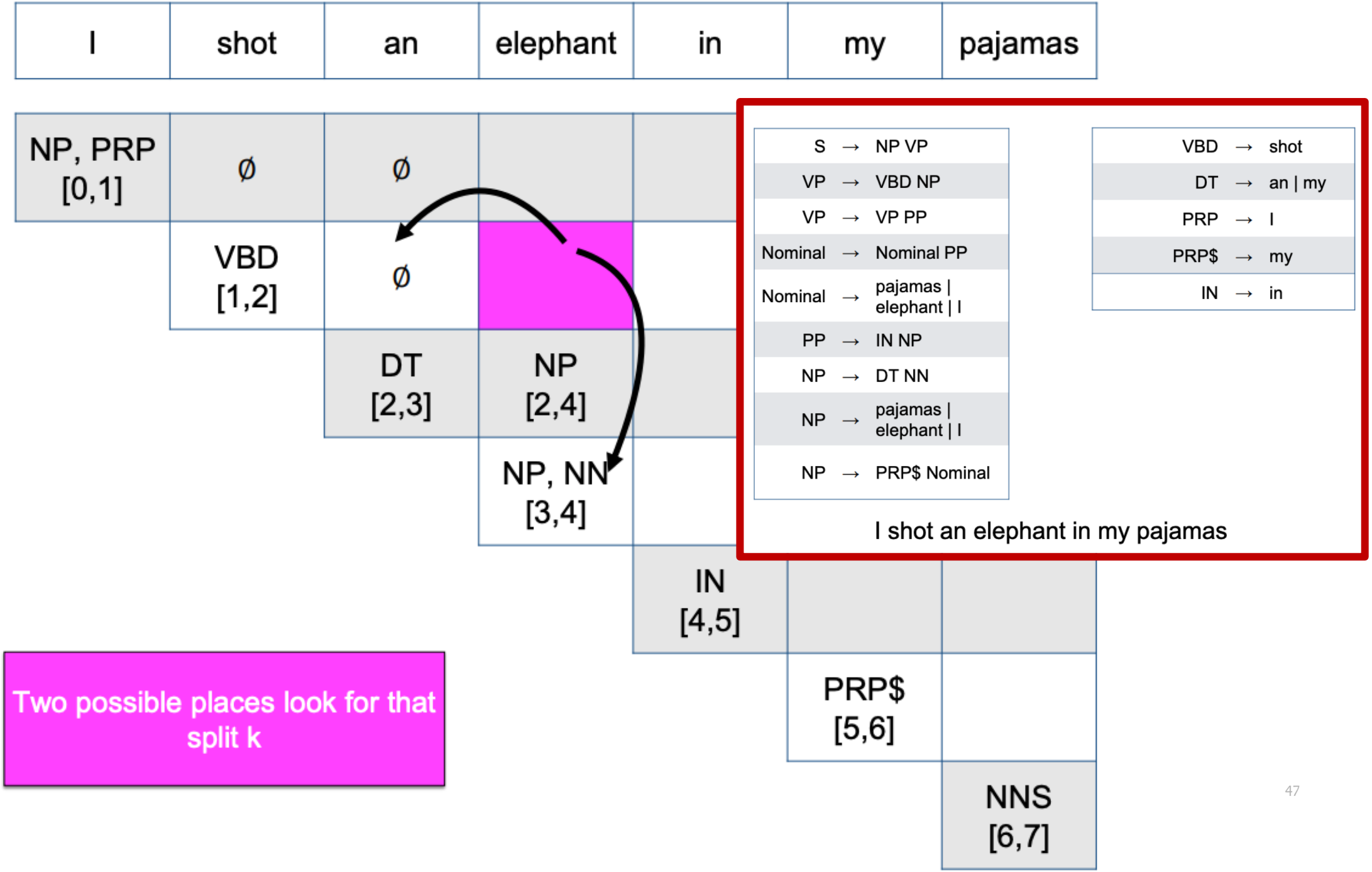
S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

Does any rule generate DT NN?


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
Ø
VBD
[1,2]
Ø
DT
[2,3]
NP
[2,4]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Two possible places look for that split k


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
Ø
VBD
[1,2]
Ø
DT
[2,3]
NP
[2,4]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Two possible places look for that split k

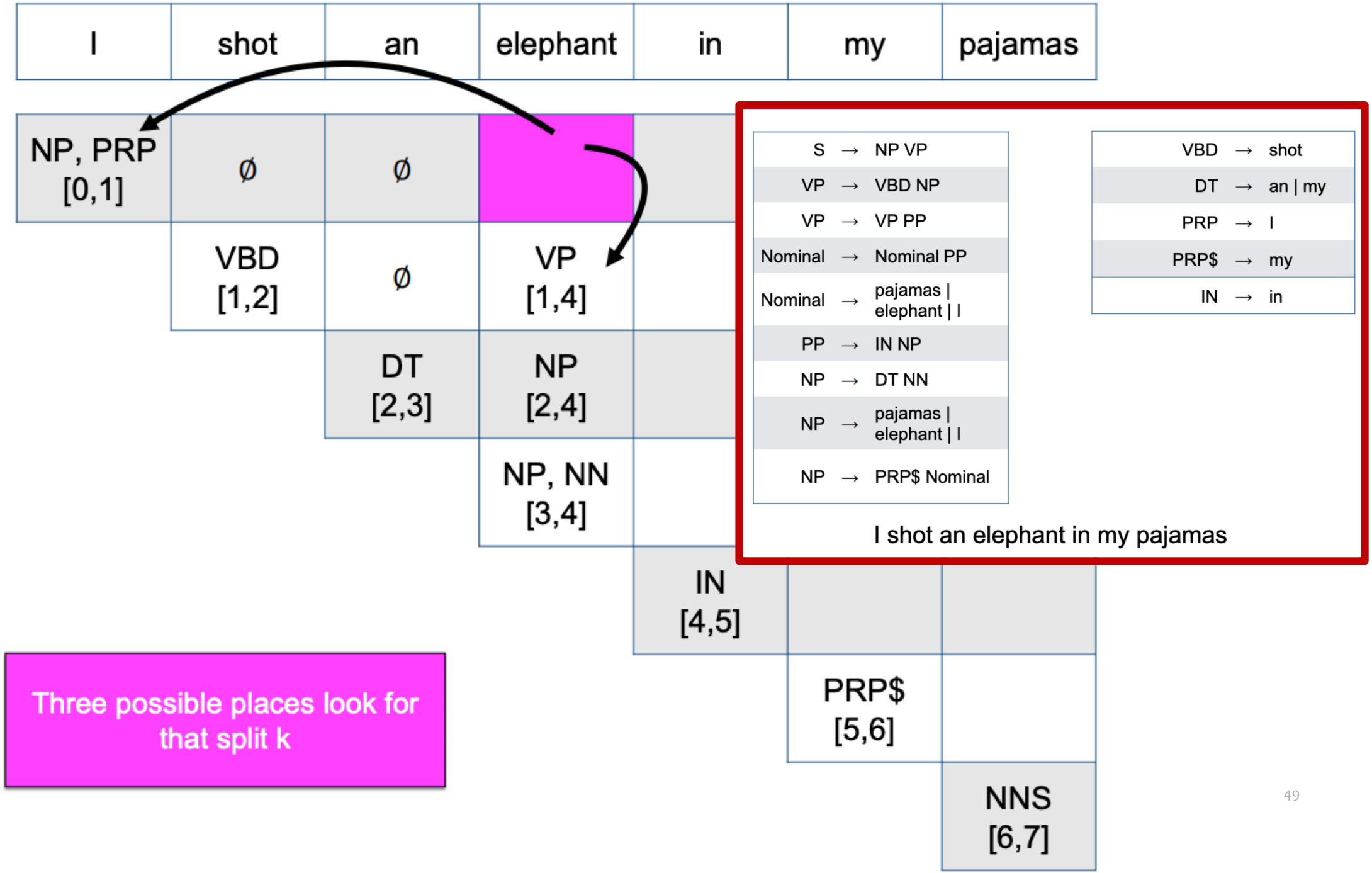
I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø				
	VBD [1,2]	Ø	VP [1,4]			
		DT [2,3]	NP [2,4]			
			NP, NN [3,4]			
				IN [4,5]		
					PRP$ [5,6]	
						NNS [6,7]

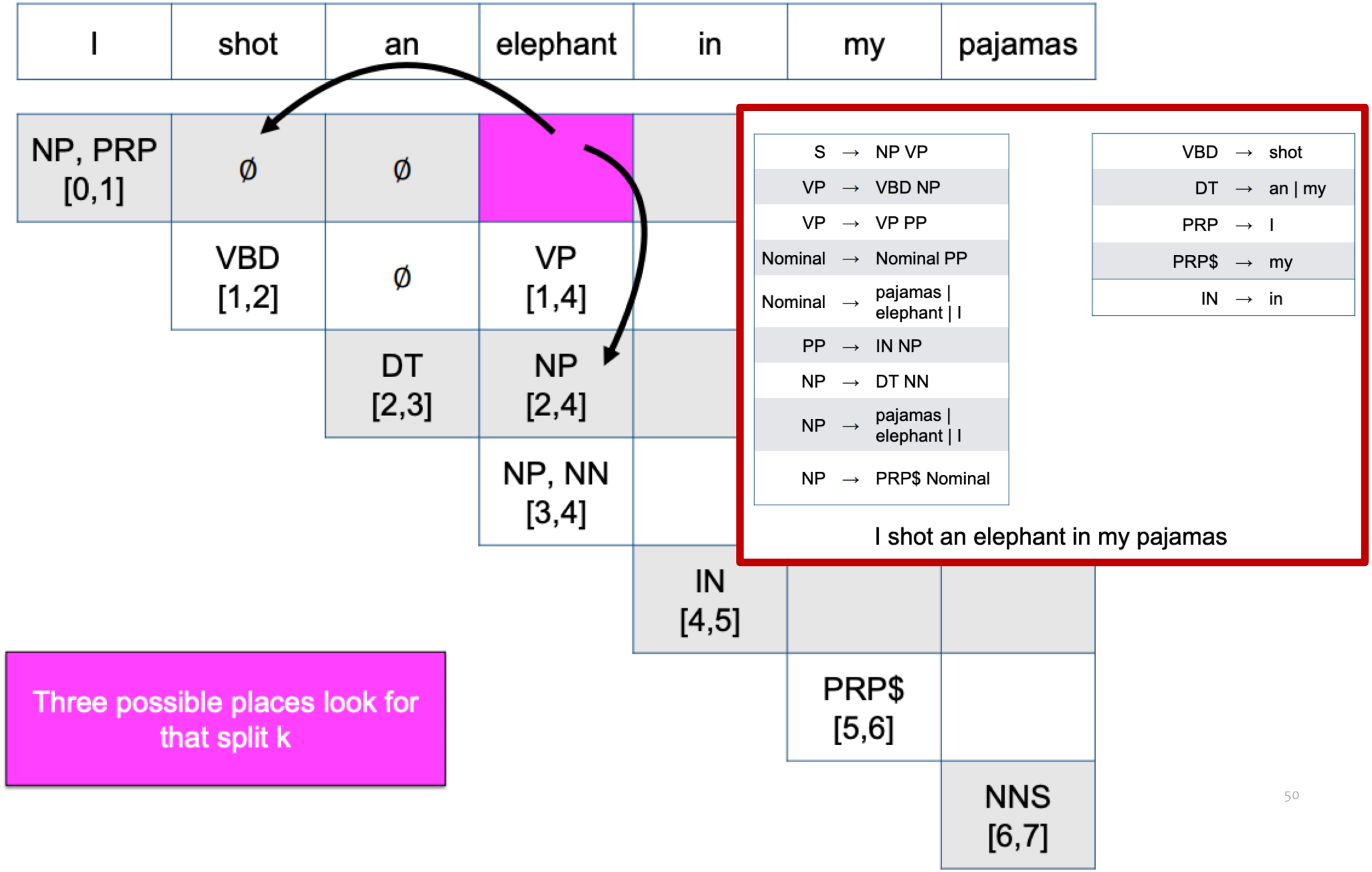
S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

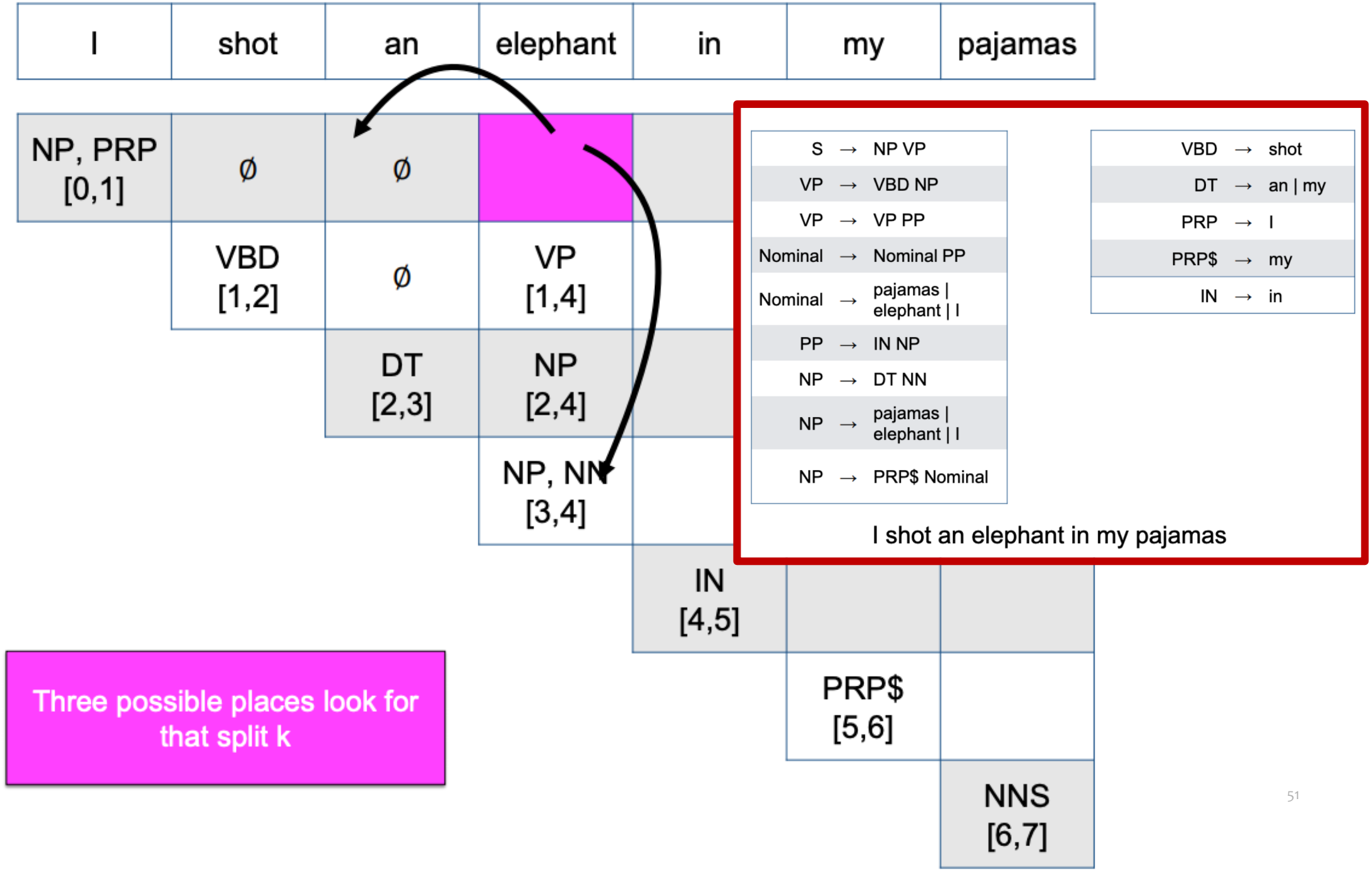
VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

Three possible places look for that split k


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
Ø
VBD
[1,2]
Ø
VP
[1,4]
DT
[2,3]
NP
[2,4]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Three possible places look for that split k


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
Ø
VBD
[1,2]
Ø
VP
[1,4]
DT
[2,3]
NP
[2,4]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Three possible places look for that split k


I
shot
an
elephant
in
my
pajamas
NP, PRP
[0,1]
Ø
Ø
VBD
[1,2]
Ø
VP
[1,4]
DT
[2,3]
NP
[2,4]
NP, NN
[3,4]
IN
[4,5]
PRP$
[5,6]
NNS
[6,7]
S → NP VP
VP → VBD NP
VP → VP PP
Nominal → Nominal PP
Nominal → pajamas | elephant | I
PP → IN NP
NP → DT NN
NP → pajamas | elephant | I
NP → PRP$ Nominal
VBD → shot
DT → an | my
PRP → I
PRP$ → my
IN → in
I shot an elephant in my pajamas
Three possible places look for that split k

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]			
	VBD [1,2]	Ø	VP [1,4]			
		DT [2,3]	NP [2,4]			
			NP, NN [3,4]			
				IN [4,5]		
					PRP$ [5,6]	
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	∅	∅	S [0,4]	∅	∅	
	VBD [1,2]	∅	VP [1,4]	∅	∅	
		DT [2,3]	NP [2,4]	∅	∅	
			NP, NN [3,4]	∅	∅	
				IN [4,5]	∅	
					PRP$ [5,6]	
						NNS [6,7]

*elephant in
*an elephant in
*shot an elephant in
*I shot an elephant in

*in my
*elephant in my
*an elephant in my
*shot an elephant in my
*I shot an elephant in my

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	
			NP, NN [3,4]	Ø	Ø	
				IN [4,5]	Ø	
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	
			NP, NN [3,4]	Ø	Ø	
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	∅	∅	S [0,4]	∅	∅	
	VBD [1,2]	∅	VP [1,4]	∅	∅	
		DT [2,3]	NP [2,4]	∅	∅	NP [3,7]
			NP, NN [3,4]	∅	∅	NP [3,7]
				IN [4,5]	∅	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	
		DT [2,3]	NP [2,4]	Ø	Ø	NP [3,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

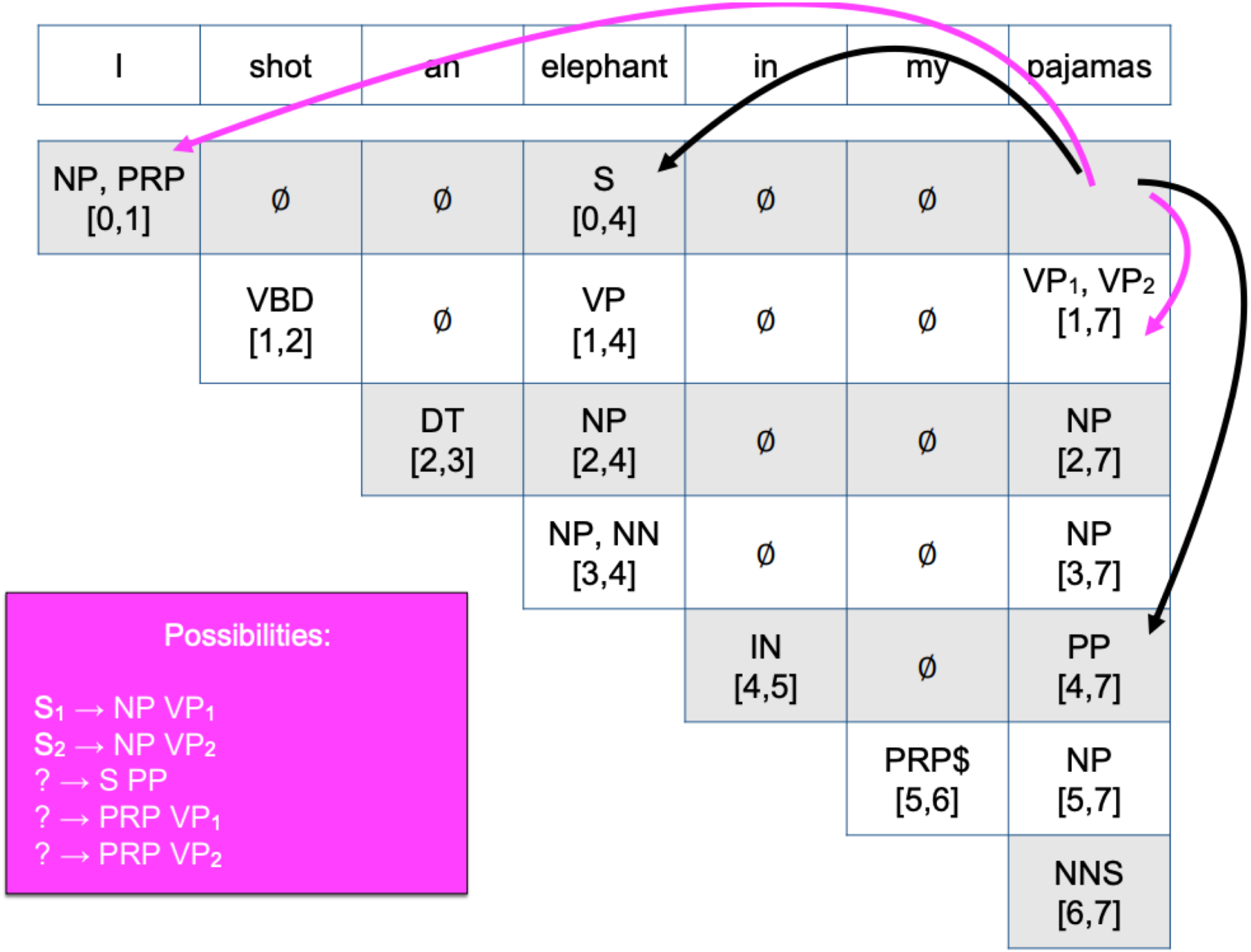
I shot an elephant in my pajamas

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	VP_1, VP_2 [1,7]
		DT [2,3]	NP [2,4]	Ø	Ø	NP [2,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

S	→	NP VP
VP	→	VBD NP
VP	→	VP PP
Nominal	→	Nominal PP
Nominal	→	pajamas | elephant | I
PP	→	IN NP
NP	→	DT NN
NP	→	pajamas | elephant | I
NP	→	PRP$ Nominal

VBD	→	shot
DT	→	an | my
PRP	→	I
PRP$	→	my
IN	→	in

I shot an elephant in my pajamas


I
shot
an
elephant
in
my
pajamas
NP, PRP [0,1]
Ø
Ø
S [0,4]
Ø
Ø
VBD [1,2]
Ø
VP [1,4]
Ø
Ø
VP1, VP2 [1,7]
DT [2,3]
NP [2,4]
Ø
Ø
NP [2,7]
NP, NN [3,4]
Ø
Ø
NP [3,7]
IN [4,5]
Ø
PP [4,7]
PRP$ [5,6]
NP [5,7]
NNS [6,7]
Possibilities:
S1 → NP VP1
S2 → NP VP2
? → S PP
? → PRP VP1
? → PRP VP2

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	S_1, S_2 [0,7]
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	VP_1, VP_2 [1,7]
		DT [2,3]	NP [2,4]	Ø	Ø	NP [2,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

Success! We've recognized a total of two valid parses

CKY algorithm

```
function CKY-PARSE(words, grammar) returns table

  for j ← from 1 to LENGTH(words) do
      for all {A | A → words[j] ∈ grammar}
              table[j − 1, j] ← table[j − 1, j] ∪ A
      for i ← from j − 2 downto 0 do
          for k ← i + 1 to j − 1 do
              for all {A | A → BC ∈ grammar and B ∈ table[i, k] and C ∈ table[k, j]}
                      table[i,j] ← table[i,j] ∪ A
```

Figure 12.5 The CKY algorithm.

I	shot	an	elephant	in	my	pajamas
NP, PRP [0,1]	Ø	Ø	S [0,4]	Ø	Ø	S_1, S_2 [0,7]
	VBD [1,2]	Ø	VP [1,4]	Ø	Ø	VP_1, VP_2 [1,7]
		DT [2,3]	NP [2,4]	Ø	Ø	NP [2,7]
			NP, NN [3,4]	Ø	Ø	NP [3,7]
				IN [4,5]	Ø	PP [4,7]
					PRP$ [5,6]	NP [5,7]
						NNS [6,7]

Runtime complexity?

CKY

- This use of CKY allows us to:
  - **Recognize** whether a sentence in grammatical in the language defined by the CFG
  - Enumerate all possible parses for a sentence
- But it does not tell us on its one which of those possible parses is most likely

- Under a full grammar, how many valid parses does an average-length sentence have?
  a) 1-10
  b) 10-100
  c) 100-1000
  d) 1000+

CKY Summary & In Practice

- Dynamic programming parsing algorithms, such as CKY, use a table of partial parses to efficiently parse ambiguous sentences.
- CKY restricts the form of the grammar to Chomsky normal form (CNF)
- In Practice
  - Does not return trees that are consistently with given grammar
  - Conversion to CNF complicates any syntax-driven approach to semantic analysis

PCFG

- Probabilistic context-free grammar: each production is also associated with a probability
- This lets us calculate the probability of a parse for a given sentence; for a given parse tree T for sentence S comprised of n rules from R (each $A \rightarrow \beta$):

$$\mathrm{P}(T, S) = \prod_i^n P(\beta|A)$$

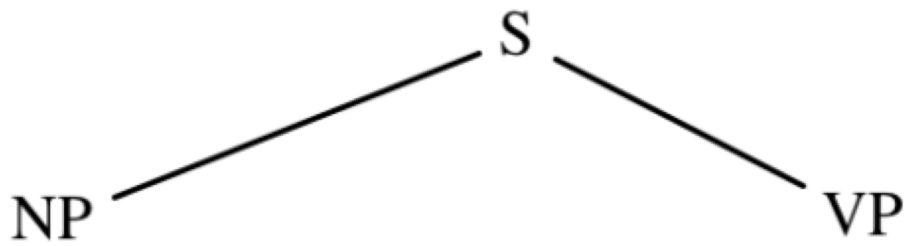
PCFG

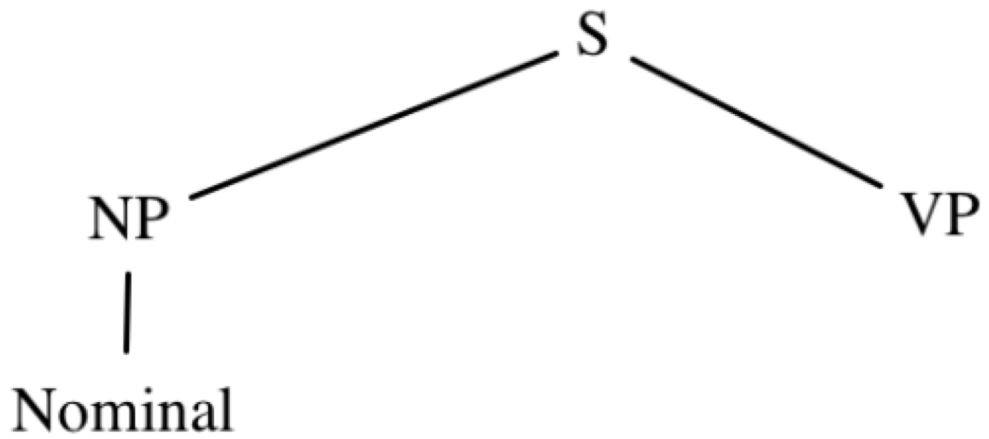
$$P(T,S) = \prod_i^n P(\beta|A)$$

N	Finite set of non-terminal symbols	NP, VP, S
Σ	Finite alphabet of terminal symbols	the, dog, a
R	Set of production rules, each $A \rightarrow \beta$ [p] p = P(β | A)	S → NP VP Noun → dog
S	Start symbol	

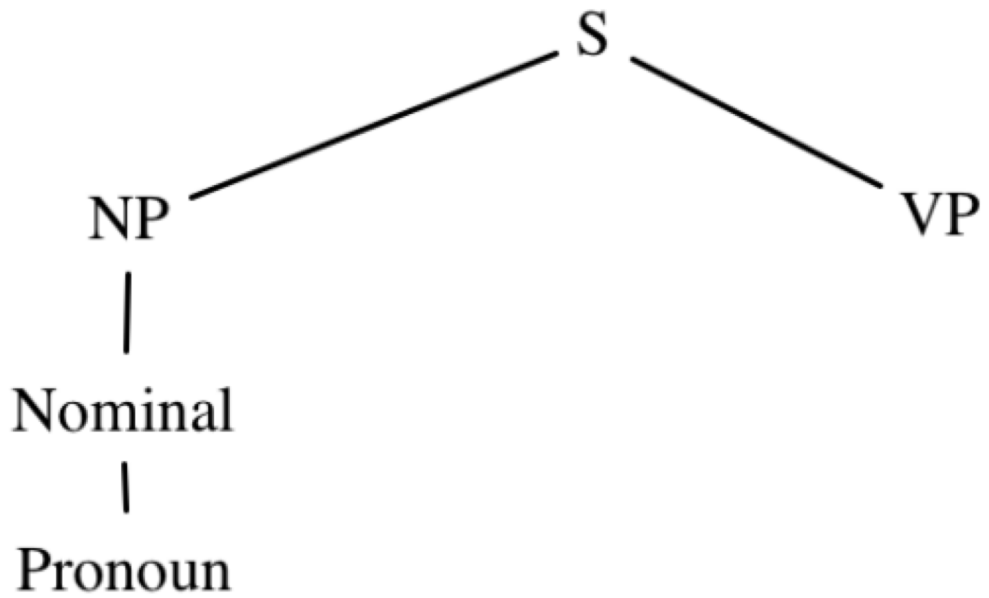
S

$$P(\text{NP VP} \mid \text{S})$$

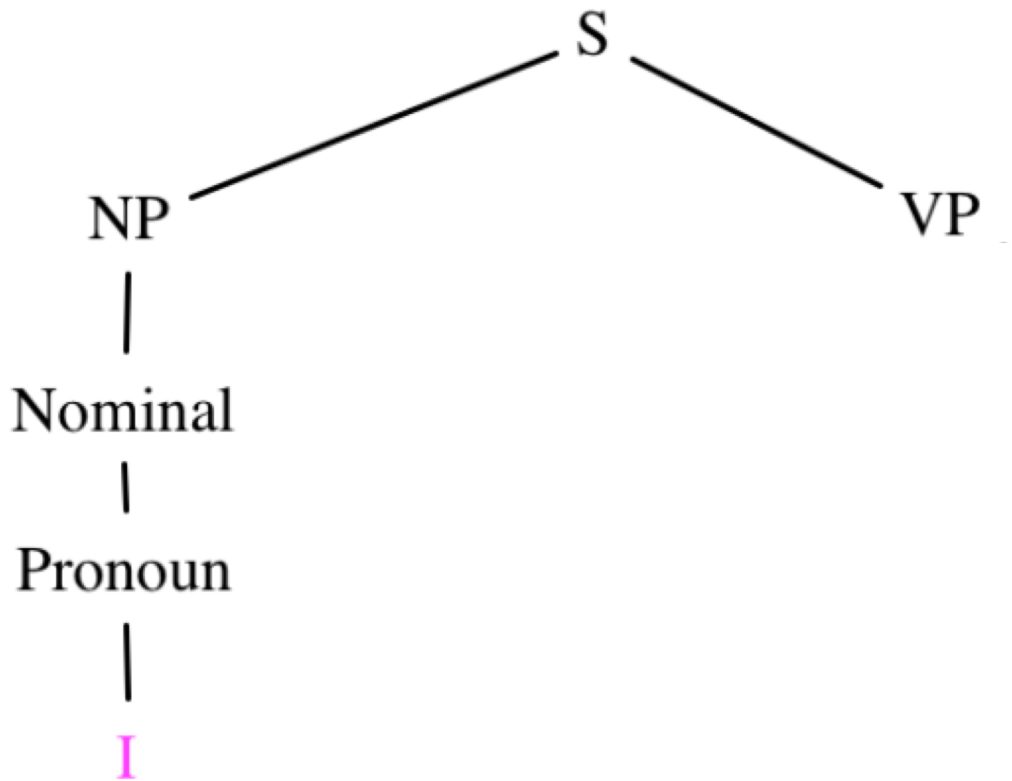




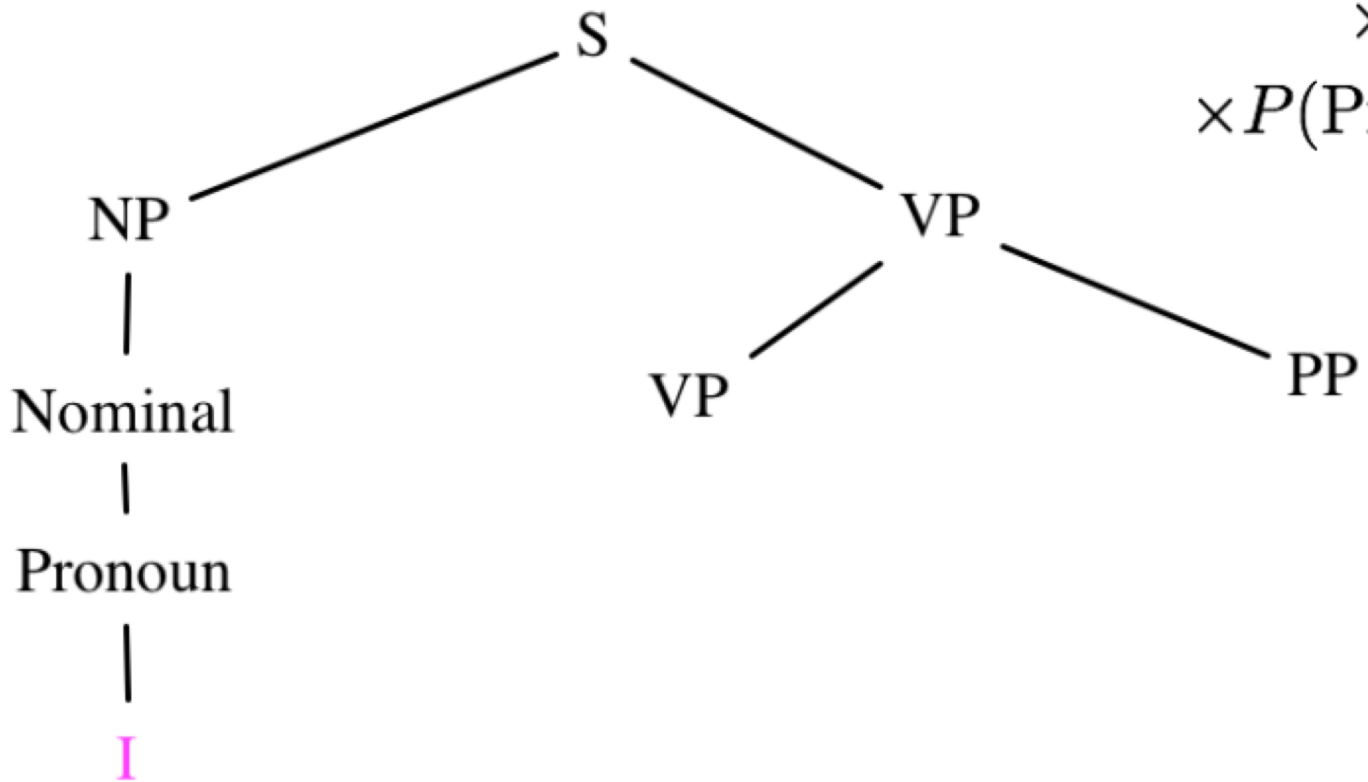
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$



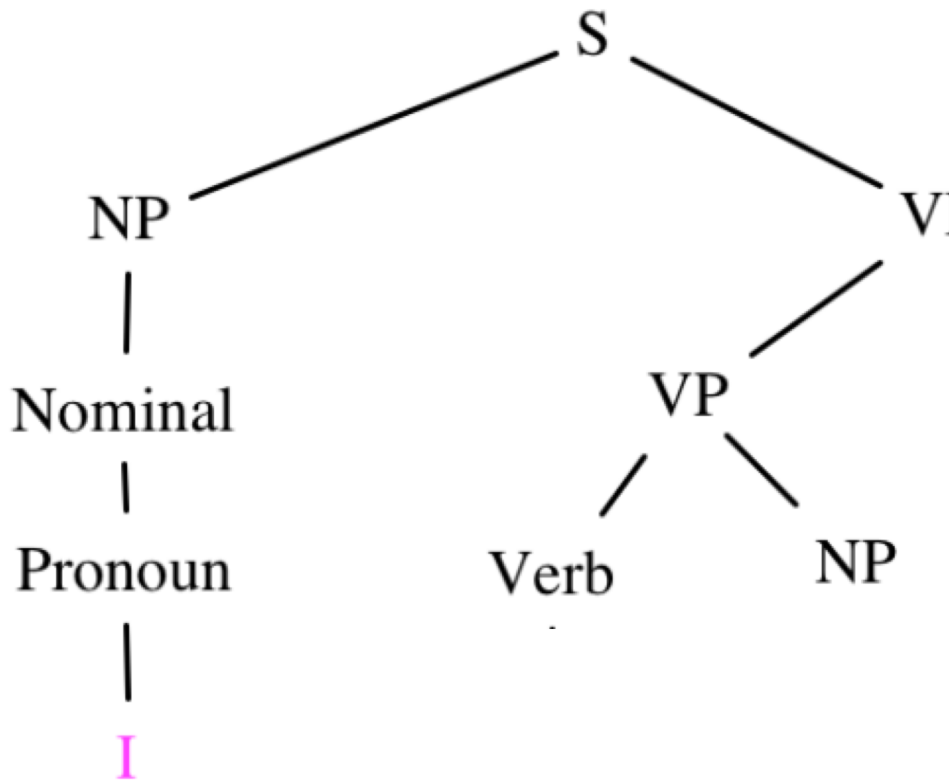
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$



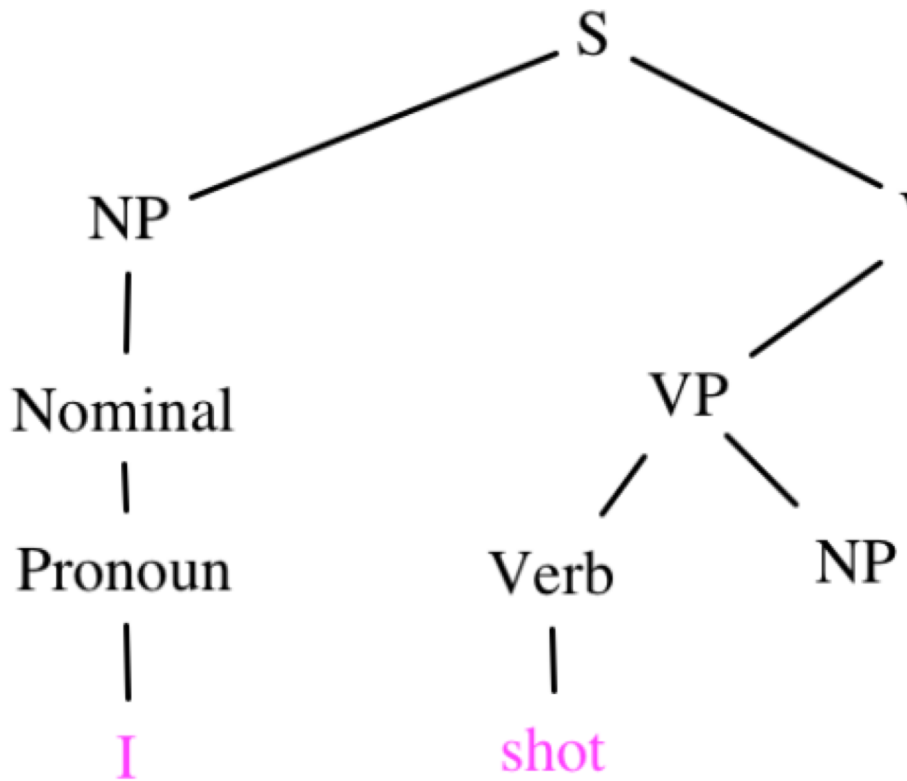
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$



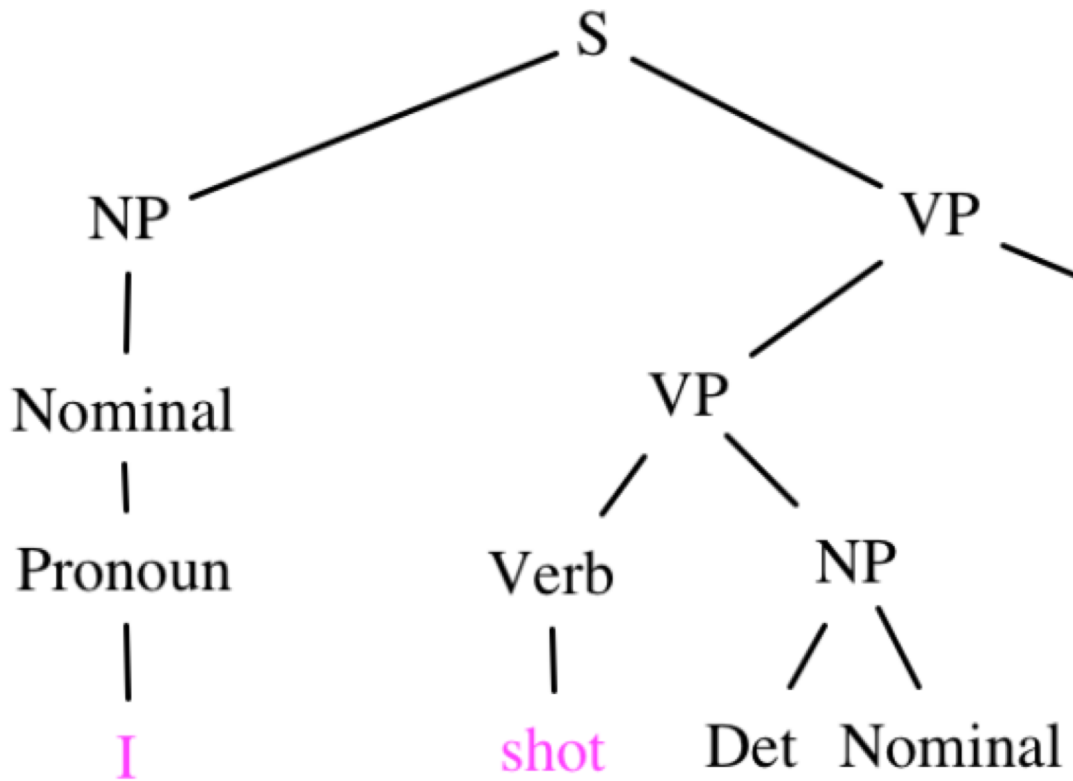
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$



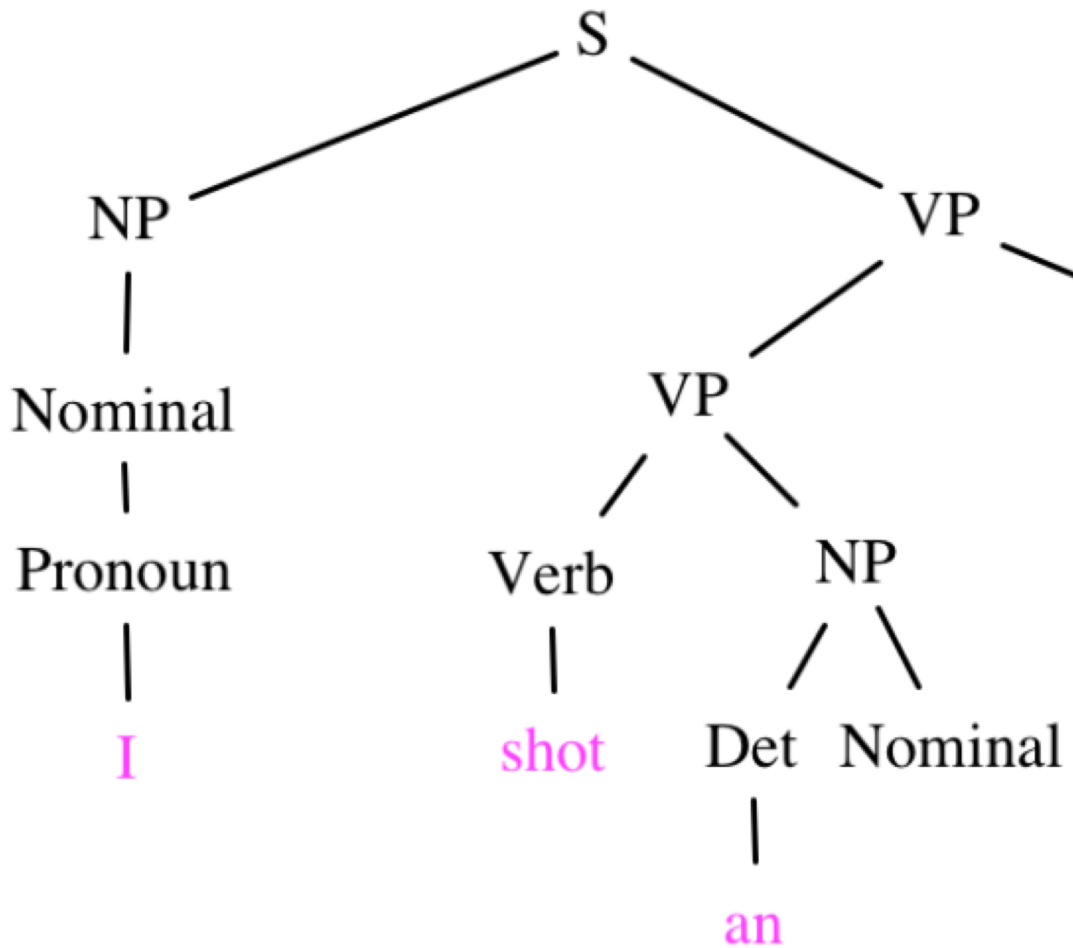
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$



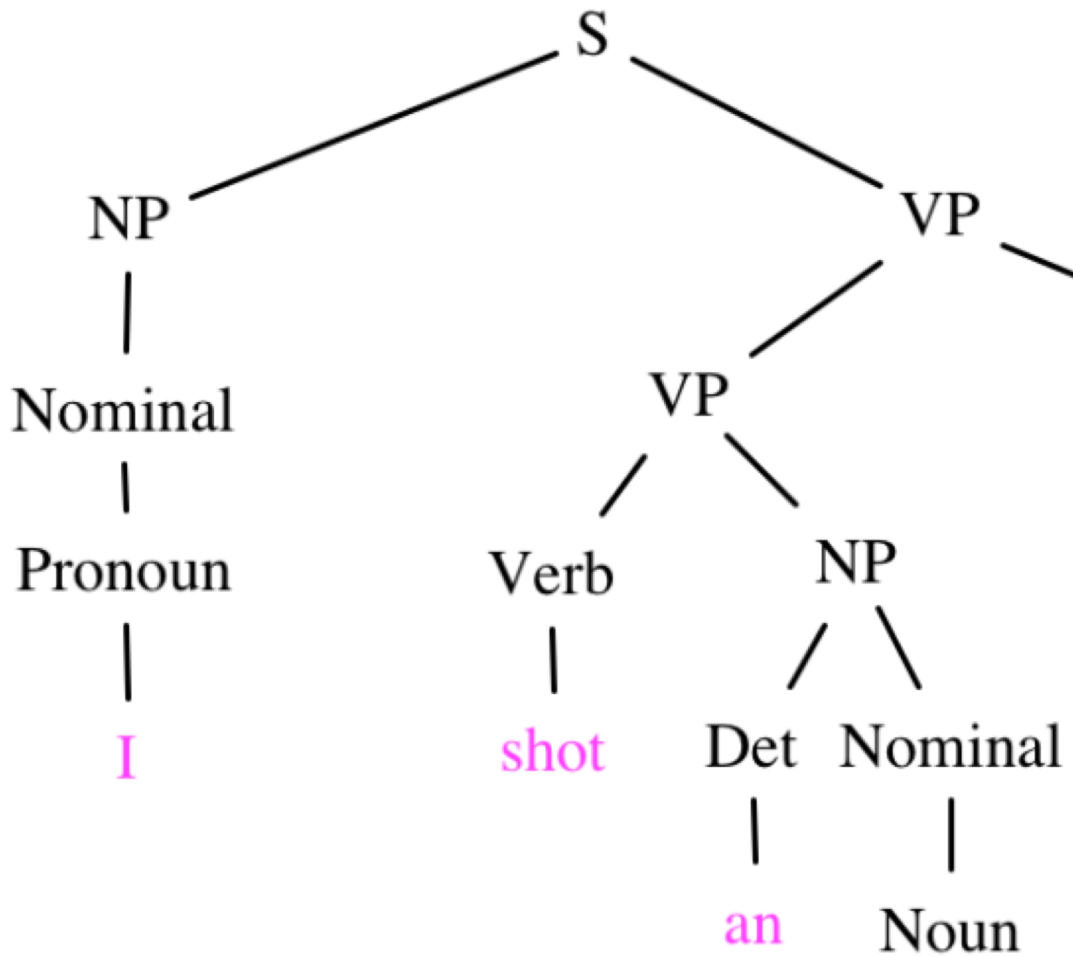
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$
$$\times P(\text{shot} \mid \text{Verb})$$



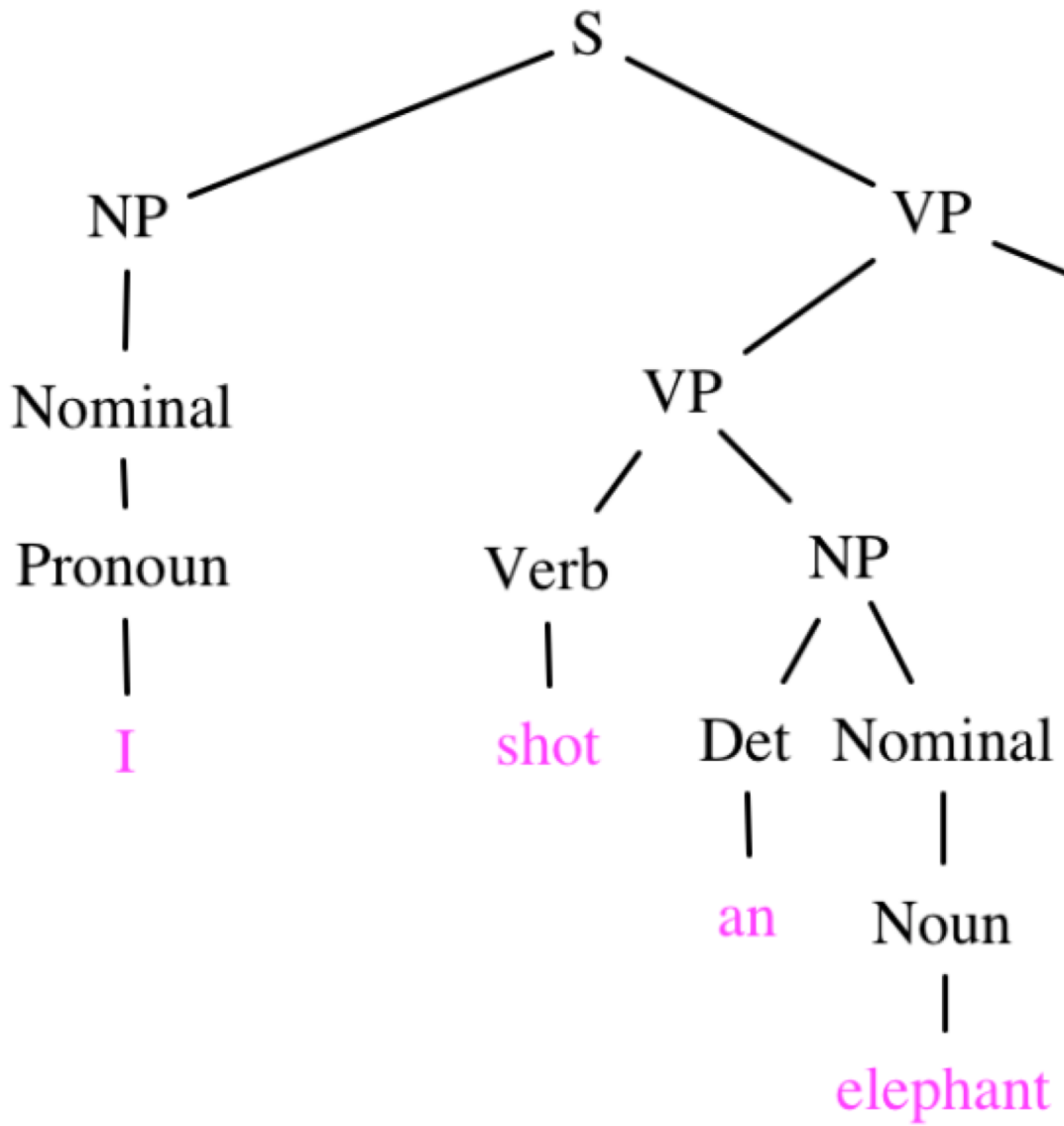
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$
$$\times P(\text{shot} \mid \text{Verb})$$
$$\times P(\text{Det Nominal} \mid \text{NP})$$



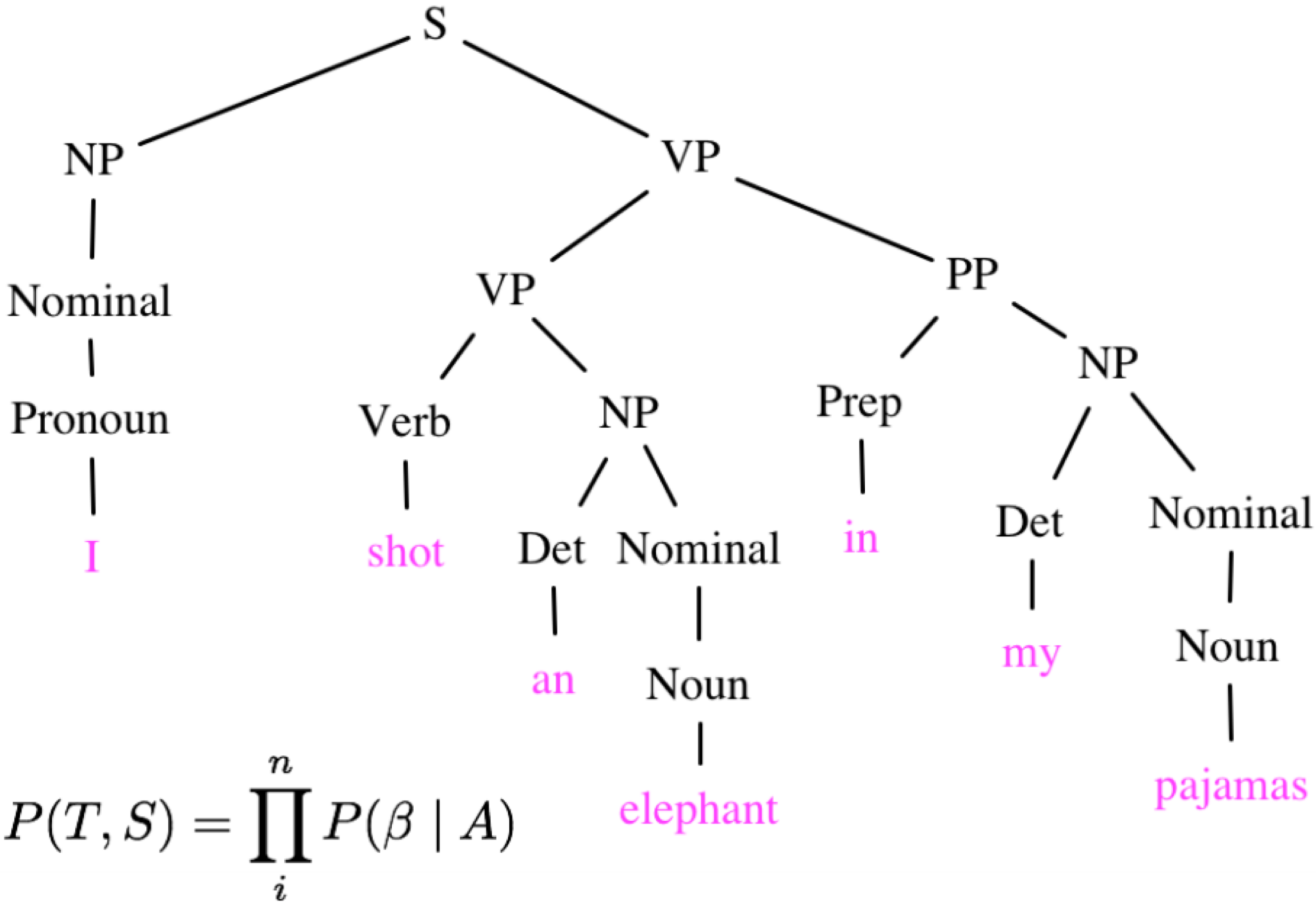
$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$
$$\times P(\text{shot} \mid \text{Verb})$$
$$\times P(\text{Det Nominal} \mid \text{NP})$$
$$\times P(\text{an} \mid \text{Det})$$



$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$
$$\times P(\text{shot} \mid \text{Verb})$$
$$\times P(\text{Det Nominal} \mid \text{NP})$$
$$\times P(\text{an} \mid \text{Det})$$
$$\times P(\text{Noun} \mid \text{Nominal})$$



$$P(\text{NP VP} \mid \text{S})$$
$$\times P(\text{Nominal} \mid \text{NP})$$
$$\times P(\text{Pronoun} \mid \text{Nominal})$$
$$\times P(\text{I} \mid \text{Pronoun})$$
$$\times P(\text{VP PP} \mid \text{VP})$$
$$\times P(\text{Verb NP} \mid \text{VP})$$
$$\times P(\text{shot} \mid \text{Verb})$$
$$\times P(\text{Det Nominal} \mid \text{NP})$$
$$\times P(\text{an} \mid \text{Det})$$
$$\times P(\text{Noun} \mid \text{Nominal})$$
$$\times P(\text{elephant} \mid \text{Noun})$$



$$P(T,S) = \prod_{i}^{n} P(\beta \mid A)$$

CKY PCFG

- A PCFG gives us a mechanism for assigning scores (here, probabilities) to different parses for the same sentence
- But we often care about is finding **the single best parse** with the highest probability
- In general, when people talk about CKY, they mean the PCFG version, since the CFG version isn't very useful

CKY PCFG

- We calculate the max probability parse using CKY by storing the probability of each phrase within each cell as we built it up.
- In particular, we fill the cell for span i-j and label A with the maximum over splits and rules of

$$table(i, j, A) = P(A \rightarrow BC) \times table(i, k, B) \times table(k, j, C)$$

```
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]
```

VS. CKY

```
function CKY-PARSE(words, grammar) returns table

  for j ← from 1 to LENGTH(words) do
      for all {A | A → words[j] ∈ grammar}
              table[j − 1, j] ← table[j − 1, j] ∪ A
      for i ← from j − 2 downto 0 do
          for k ← i + 1 to j − 1 do
              for all {A | A → BC ∈ grammar and B ∈ table[i, k] and C ∈ table[k, j]}
                      table[i,j] ← table[i,j] ∪ A
```

Figure 13.5 The CKY algorithm.

I	shot	an	elephant	in	my	pajamas
PRP:0.04 [0,1]						
	VBD:0.04 [1,2]					
		DT:0.05 [2,3]				
			NN:0.03 [3,4]			
				IN:0.10 [4,5]		
					PRP$:0.12 [5,6]	
						NNS:0.01 [6,7]

Probaiblity of a terminal (word) given its tag

$$P(A \rightarrow \beta)$$

I	shot	an	elephant	in	my	pajamas
PRP:0.04 [0,1]	Ø	Ø				
	VBD:0.04 [1,2]	Ø				
		DT:0.05 [2,3]	NP:0.0001 5 [2,4]			
			NN:0.03 [3,4]			
				IN:0.10 [4,5]		
					PRP$:0.12 [5,6]	
						NNS:0.01 [6,7]

$$table(2,4,NP) = P(\text{NP} \rightarrow \text{DT NN}) \times table(2,3,DT) \times table(3,4,NN)$$

I	shot	an	elephant	in	my	pajamas
PRP:0.04 [0,1]	Ø	Ø				
	VBD:0.04 [1,2]	Ø	VP:0.0000006 [1,4]			
		DT:0.05 [2,3]	NP:0.00015 [2,4]			
			NN:0.03 [3,4]			
				IN:0.10 [4,5]		
					PRP$:0.12 [5,6]	
						NNS:0.01 [6,7]

We just calculated this value and can use it now

$$table(1,4,VP) = P(\text{VP} \rightarrow \text{VBD NP}) \times table(1,2,VBD) \times table(2,4,NP)$$

I	shot	an	elephant	in	my	pajamas
PRP:0.04 [0,1]	ø	ø	S: 0.0000000048 [0,4]			
	VBD:0.04 [1,2]	ø	VP:0.0000006 [1,4]			
		DT:0.05 [2,3]	NP:0.00015 [2,4]			
			NN:0.03 [3,4]			
				IN:0.10 [4,5]		
					PRP$:0.12 [5,6]	
						NNS:0.01 [6,7]

We just calculated this value and can use it now

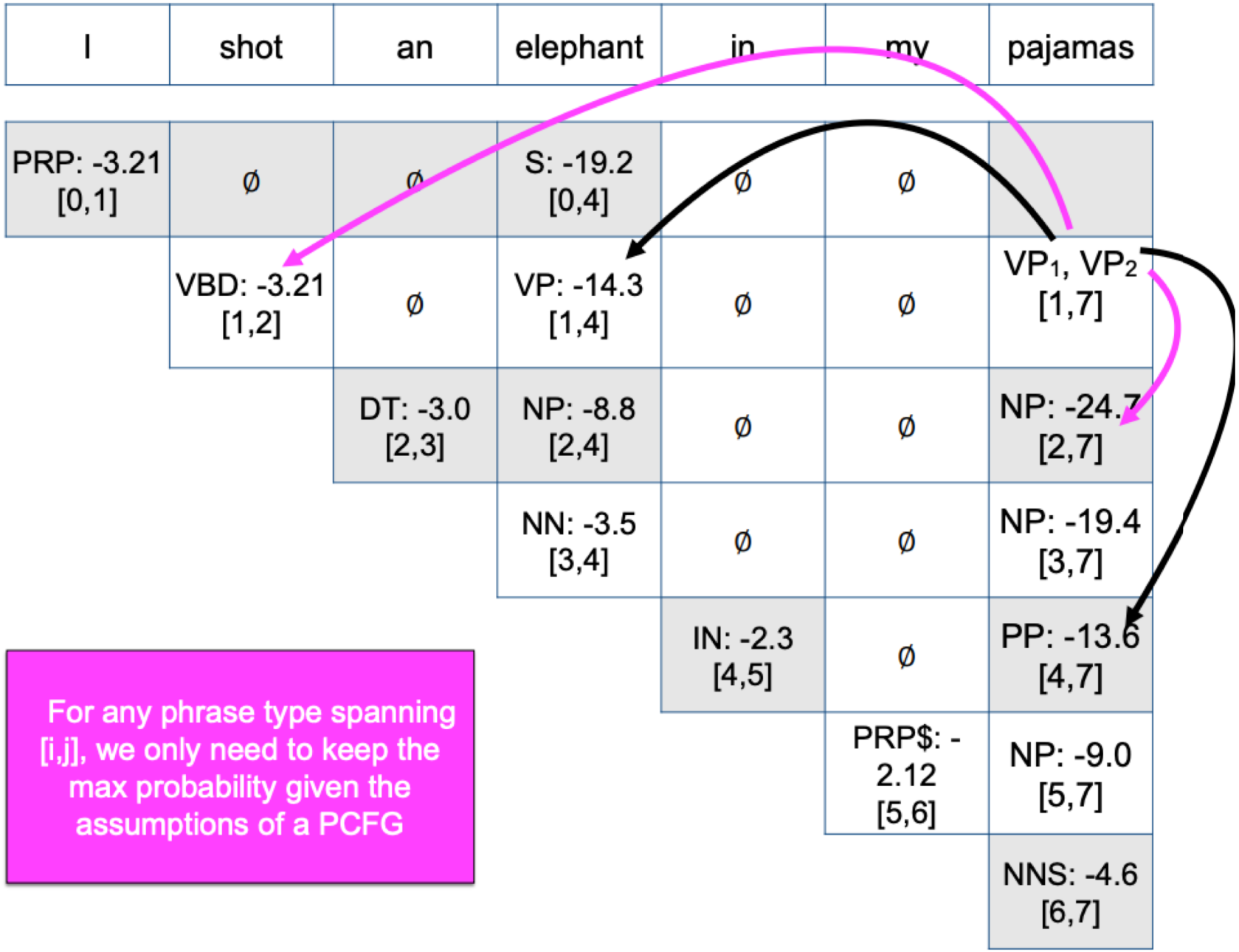
$$table(0,4,S) = P(\text{S} \rightarrow \text{NP VP}) \times table(0,1,NP) \times table(1,4,VP)$$

I	shot	an	elephant	in	my	pajamas
PRP:0.04 [0,1]	ø	ø	S: 0.0000000048 [0,4]			
	VBD:0.04 [1,2]	ø	VP:0.0000006 [1,4]			
		DT:0.05 [2,3]	NP:0.00015 [2,4]			
			NN:0.03 [3,4]			
				IN:0.10 [4,5]		
					PRP$:0.12 [5,6]	
						NNS:0.01 [6,7]

Note these values are getting very small! Better to add in log space

I	shot	an	elephant	in	my	pajamas
PRP: -3.21 [0,1]	Ø	Ø	S: -19.2 [0,4]			
	VBD: -3.21 [1,2]	Ø	VP: -14.3 [1,4]			
		DT: -3.0 [2,3]	NP: -8.8 [2,4]			
			NN: -3.5 [3,4]			
				IN: -2.3 [4,5]		
					PRP$: -2.12 [5,6]	
						NNS: -4.6 [6,7]

Note these values are getting very small! Better to add in log space

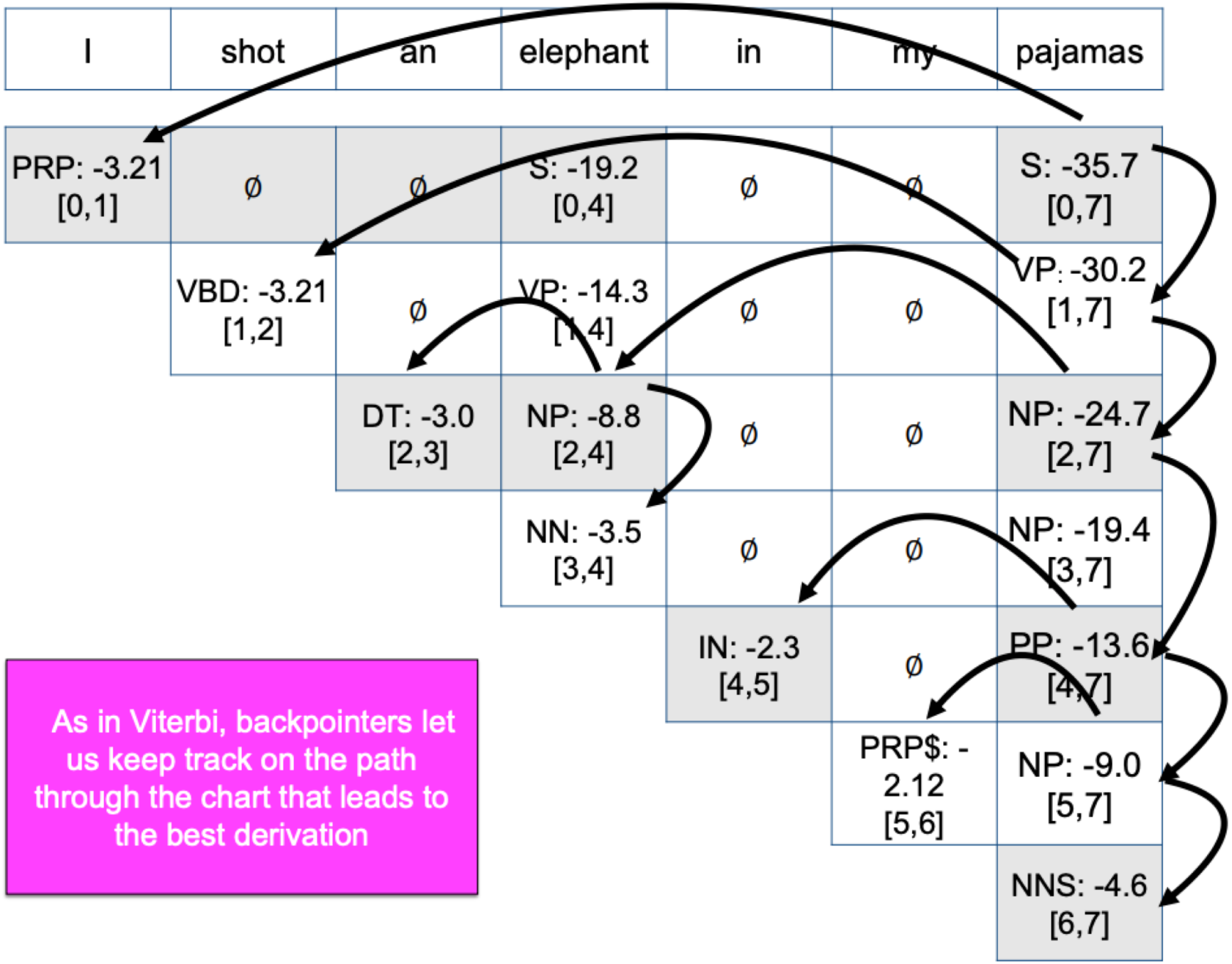

I
shot
an
elephant
in
my
pajamas
PRP: -3.21 [0,1]
Ø
Ø
S: -19.2 [0,4]
Ø
Ø
VBD: -3.21 [1,2]
Ø
VP: -14.3 [1,4]
Ø
Ø
VP1, VP2 [1,7]
DT: -3.0 [2,3]
NP: -8.8 [2,4]
Ø
Ø
NP: -24.7 [2,7]
NN: -3.5 [3,4]
Ø
Ø
NP: -19.4 [3,7]
IN: -2.3 [4,5]
Ø
PP: -13.6 [4,7]
PRP$: -2.12 [5,6]
NP: -9.0 [5,7]
NNS: -4.6 [6,7]
For any phrase type spanning [i,j], we only need to keep the max probability given the assumptions of a PCFG

I	shot	an	elephant	in	my	pajamas
PRP: -3.21 [0,1]	Ø	Ø	S: -19.2 [0,4]	Ø	Ø	
	VBD: -3.21 [1,2]	Ø	VP: -14.3 [1,4]	Ø	Ø	VP: -30.2 [1,7]
		DT: -3.0 [2,3]	NP: -8.8 [2,4]	Ø	Ø	NP: -24.7 [2,7]
			NN: -3.5 [3,4]	Ø	Ø	NP: -19.4 [3,7]
				IN: -2.3 [4,5]	Ø	PP: -13.6 [4,7]
					PRP$: -2.12 [5,6]	NP: -9.0 [5,7]
						NNS: -4.6 [6,7]

For any phrase type spanning [i,j], we only need to keep the max probability given the assumptions of a PCFG

I	shot	an	elephant	in	my	pajamas
PRP: -3.21 [0,1]	Ø	Ø	S: -19.2 [0,4]	Ø	Ø	S: -35.7 [0,7]
	VBD: -3.21 [1,2]	Ø	VP: -14.3 [1,4]	Ø	Ø	VP: -30.2 [1,7]
		DT: -3.0 [2,3]	NP: -8.8 [2,4]	Ø	Ø	NP: -24.7 [2,7]
			NN: -3.5 [3,4]	Ø	Ø	NP: -19.4 [3,7]
				IN: -2.3 [4,5]	Ø	PP: -13.6 [4,7]
					PRP$: -2.12 [5,6]	NP: -9.0 [5,7]
						NNS: -4.6 [6,7]

For any phrase type spanning [i,j], we only need to keep the max probability given the assumptions of a PCFG


I
shot
an
elephant
in
my
pajamas
PRP: -3.21 [0,1]
Ø
Ø
S: -19.2 [0,4]
Ø
Ø
S: -35.7 [0,7]
VBD: -3.21 [1,2]
Ø
VP: -14.3 [1,4]
Ø
Ø
VP: -30.2 [1,7]
DT: -3.0 [2,3]
NP: -8.8 [2,4]
Ø
Ø
NP: -24.7 [2,7]
NN: -3.5 [3,4]
Ø
Ø
NP: -19.4 [3,7]
IN: -2.3 [4,5]
Ø
PP: -13.6 [4,7]
PRP$: -2.12 [5,6]
NP: -9.0 [5,7]
NNS: -4.6 [6,7]
As in Viterbi, backpointers let us keep track on the path through the chart that leads to the best derivation