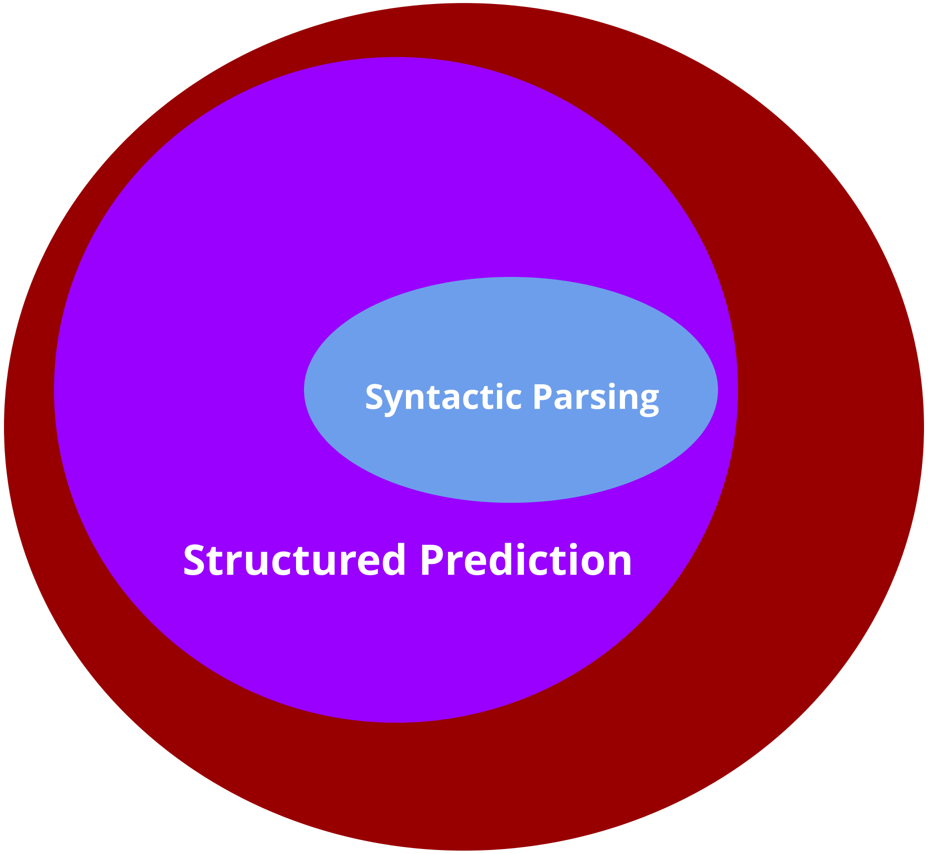
50.040
Natural Language Processing

Lu, Wei

SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

Tasks in NLP

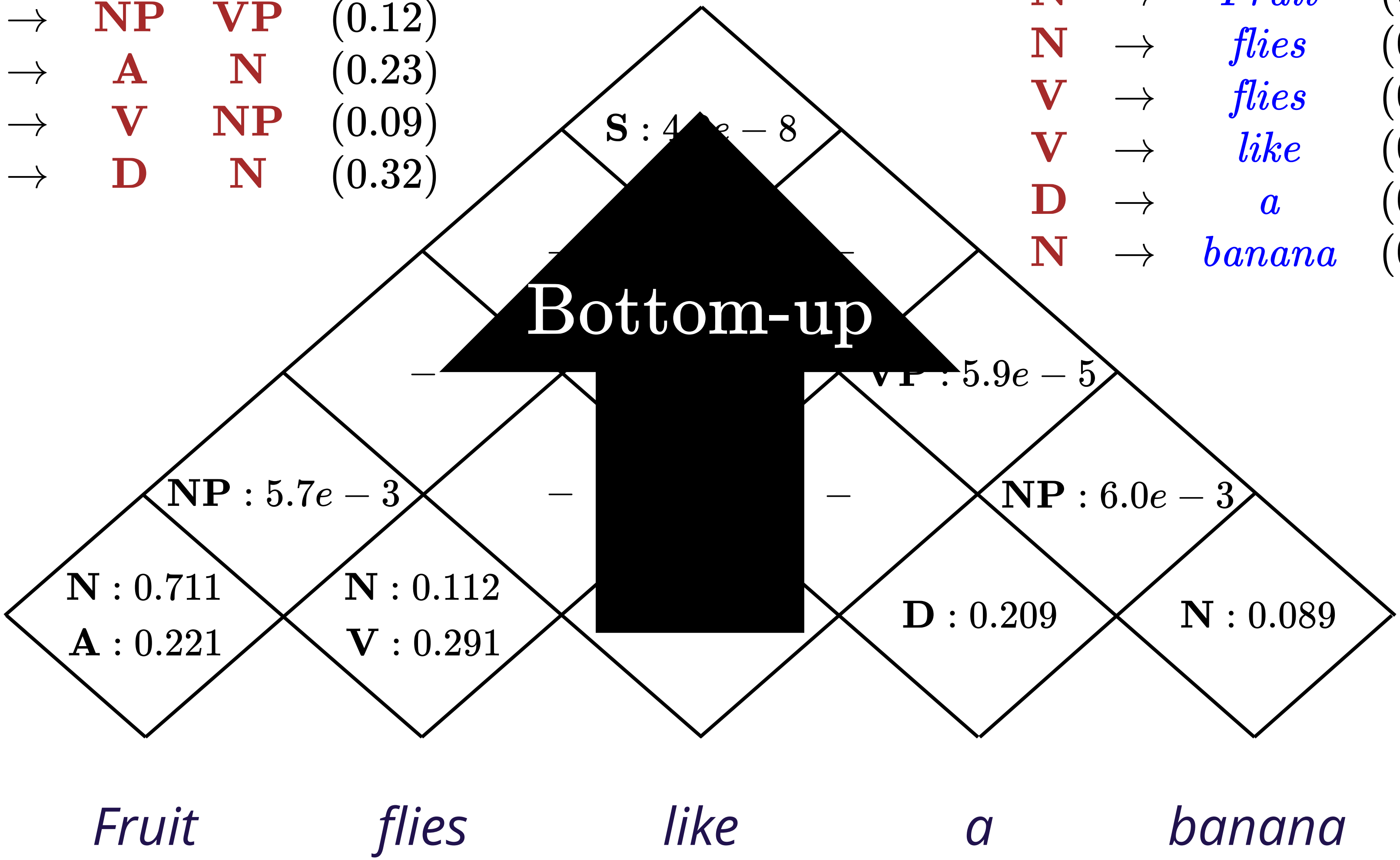


Supervised

CKY Algorithm

S	$\rightarrow$	**NP**	**VP**	(0.12)
NP	$\rightarrow$	**A**	**N**	(0.23)
VP	$\rightarrow$	**V**	**NP**	(0.09)
NP	$\rightarrow$	**D**	**N**	(0.32)

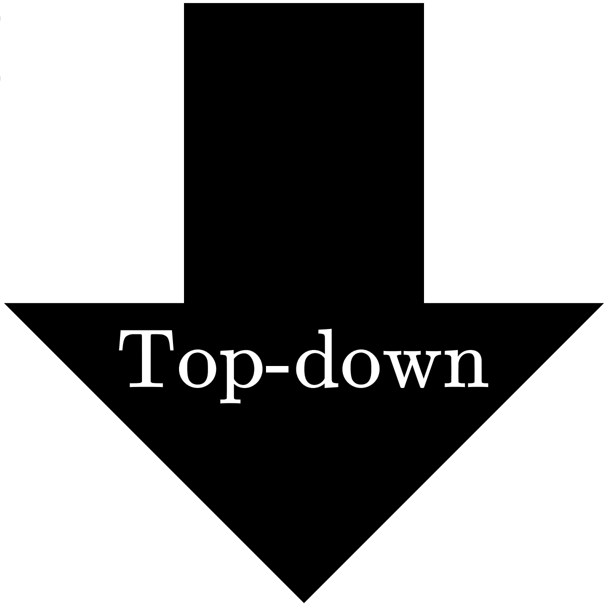
A	$\rightarrow$	*Fruit*	(0.221)
N	$\rightarrow$	*Fruit*	(0.711)
N	$\rightarrow$	*flies*	(0.112)
V	$\rightarrow$	*flies*	(0.291)
V	$\rightarrow$	*like*	(0.108)
D	$\rightarrow$	*a*	(0.209)
N	$\rightarrow$	*banana*	(0.089)



A Top-down Algorithm?

$$
\begin{array}{lclll}
\mathbf{S} & \rightarrow & \mathbf{NP} & \mathbf{VP} & (0.12) \\
\mathbf{NP} & \rightarrow & \mathbf{A} & \mathbf{N} & (0.23) \\
\mathbf{VP} & \rightarrow & \mathbf{V} & \mathbf{NP} & (0.09) \\
\mathbf{NP} & \rightarrow & \mathbf{D} & \mathbf{N} & (0.32)
\end{array}
$$

$$
\begin{array}{lccl}
\mathbf{A} & \rightarrow & \mathit{Fruit} & (0.221) \\
\mathbf{N} & \rightarrow & \mathit{Fruit} & (0.711) \\
\mathbf{N} & \rightarrow & \mathit{flies} & (0.112) \\
\mathbf{V} & \rightarrow & \mathit{flies} & (0.291) \\
\mathbf{V} & \rightarrow & \mathit{like} & (0.108) \\
\mathbf{D} & \rightarrow & \mathit{a} & (0.209) \\
\mathbf{N} & \rightarrow & \mathit{banana} & (0.089)
\end{array}
$$



Fruit *flies* *like* *a* *banana*

Optional

Earley Algorithm
(Earley, 1970)

$$\mathbf{S} \rightarrow \mathbf{NP} \bullet \mathbf{VP}\ [3, 5]$$

The above *dotted rule* tells me where I am in the parsing process based on which I can decide which of the following actions to take.

Predictor allows me to grow the subtree

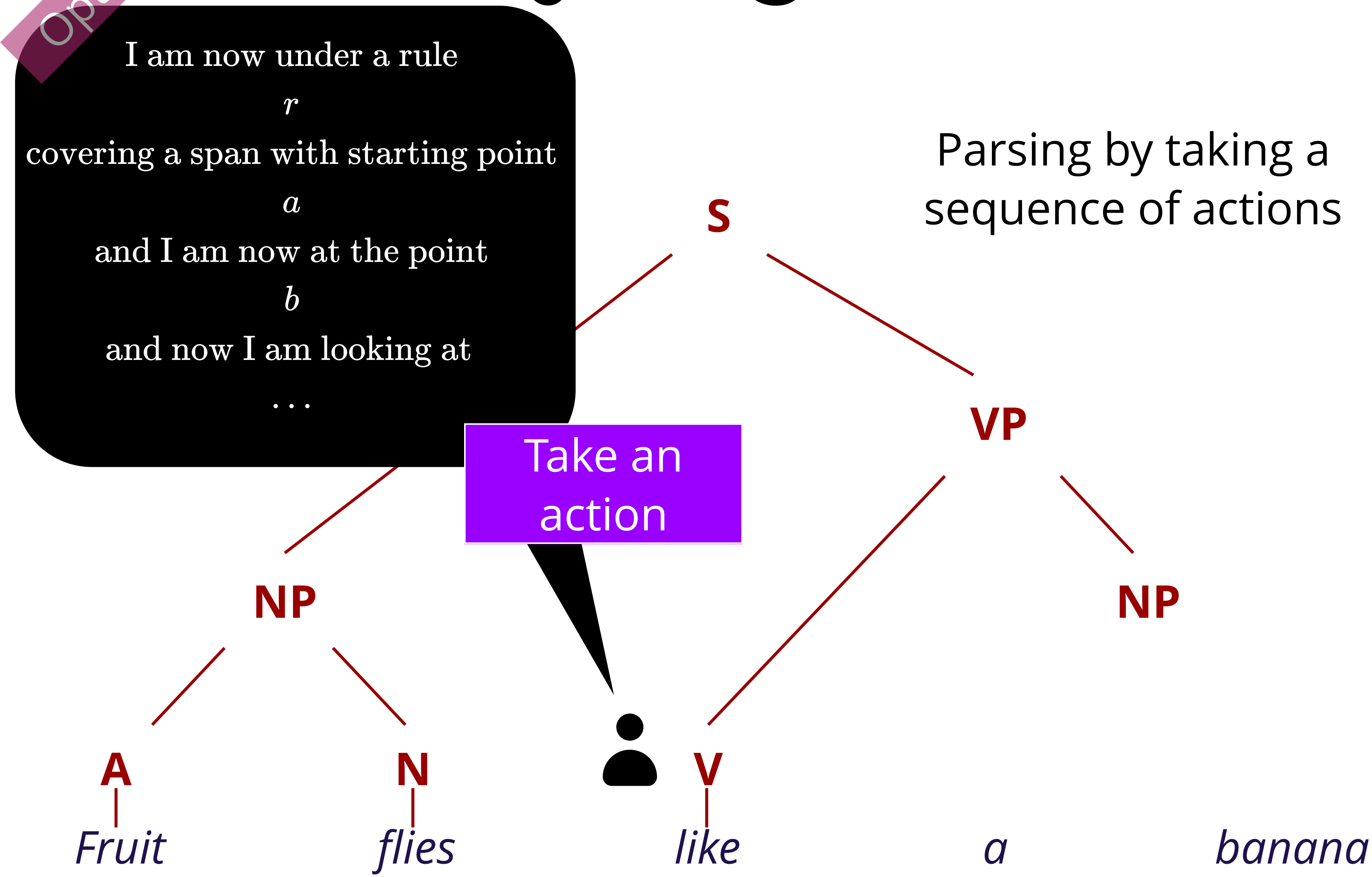
Scanner allows me to scan the terminals

Completer allows me to complete the construction of a subtree

Optional

Earley Algorithm

Parsing by taking a sequence of actions



Optional

Earley Algorithm

$\bullet\, \mathbf{S}\ [0, 0]$

S

I am now under a rule
$\gamma \to \mathbf{S}$
covering a span with starting point
0
and I am now at the point
0
and now I am looking at ways to *expand*
$\mathbf{S}$

One of the rules I can use is
$\mathbf{S} \to \mathbf{NP\ VP}$

Call "Predictor"

Fruit *flies* *like* *a* *banana*

Optional

Earley Algorithm

I am now under a rule

$\mathbf{S} \rightarrow \mathbf{NP\ VP}$

covering a span with starting point

0

and I am now at the point

0

and now I am looking at ways to *expand*

NP

$$\mathbf{S} \rightarrow \bullet \mathbf{NP\ VP}\ [0, 0]$$

S

VP

Call "Predictor"

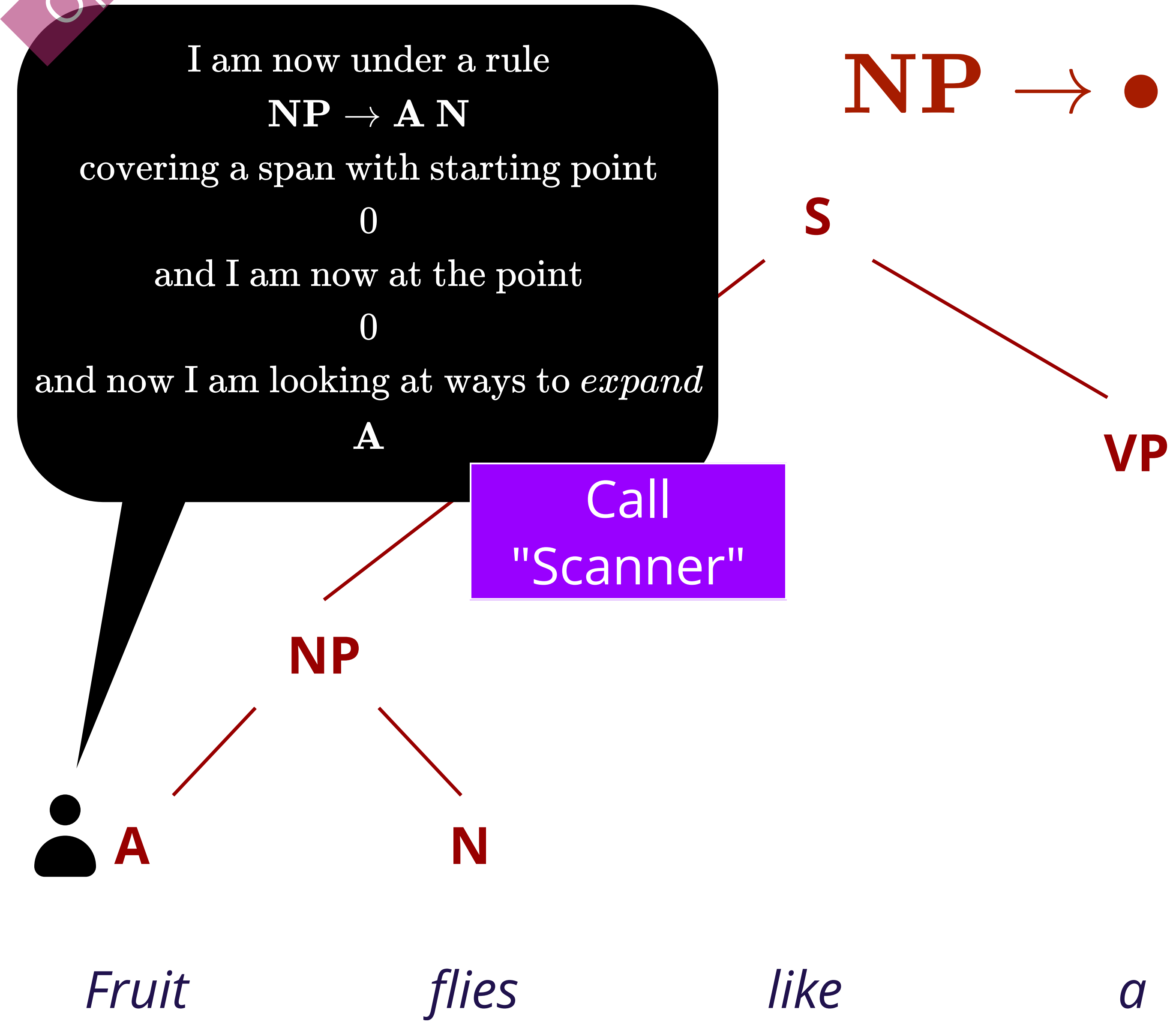
NP

Fruit *flies* *like* *a* *banana*

Earley Algorithm

Optional

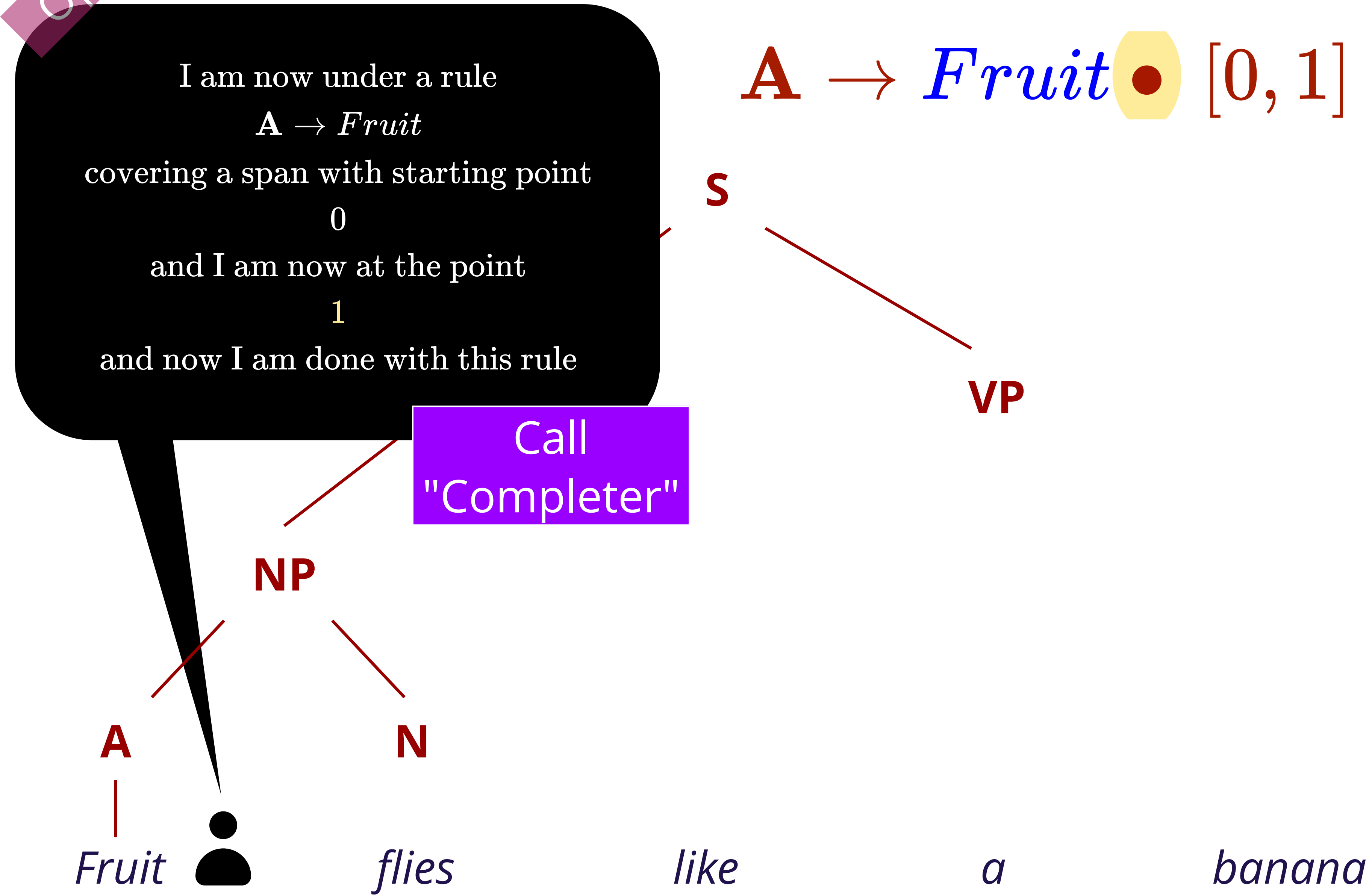
$$\mathbf{NP} \rightarrow \bullet\ \mathbf{A}\ \mathbf{N}\ [0,0]$$



Optional

Earley Algorithm

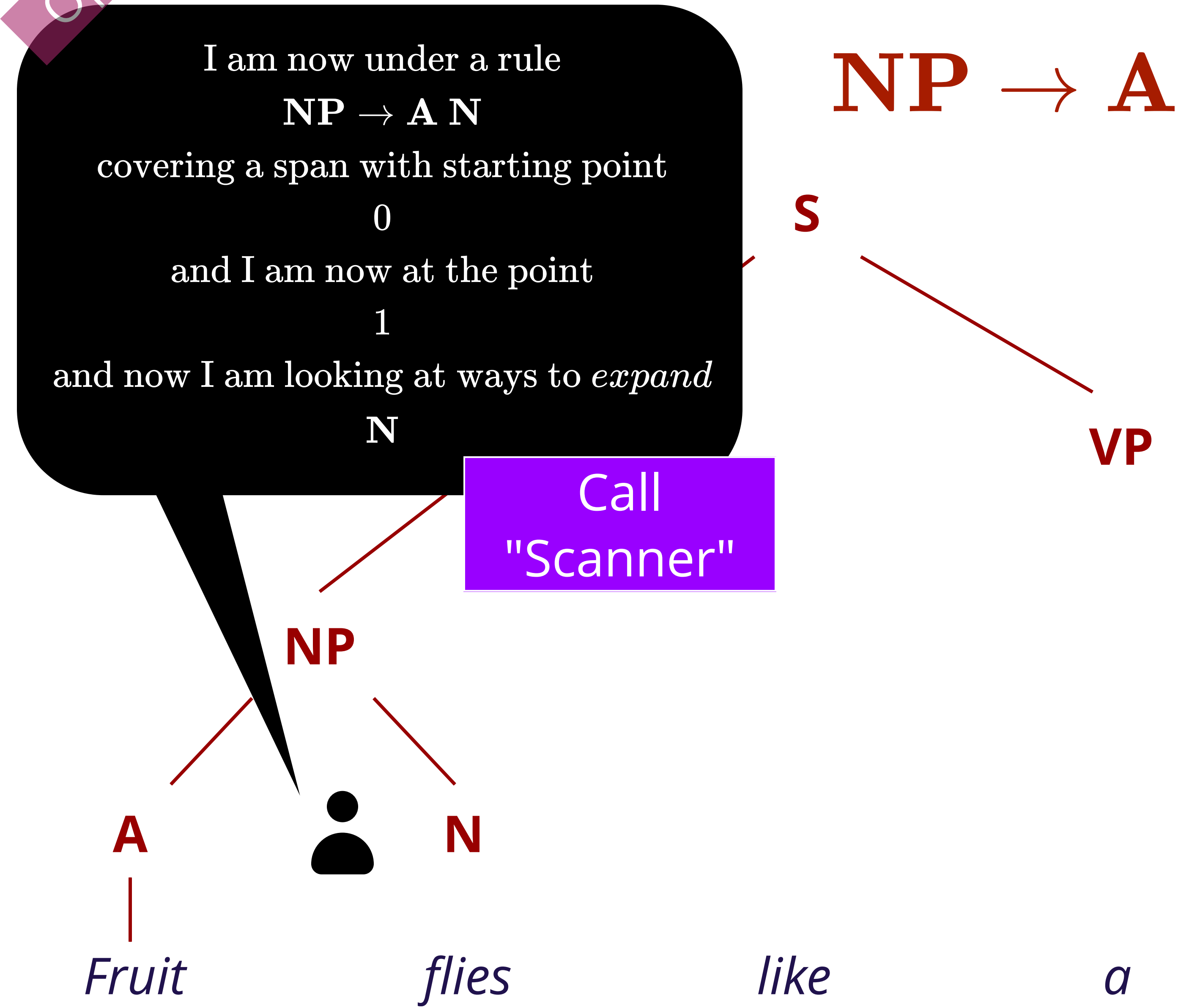
$$\mathbf{A} \rightarrow \mathit{Fruit} \bullet \; [0, 1]$$



Earley Algorithm

Optional

$$\mathbf{NP} \rightarrow \mathbf{A} \bullet \mathbf{N}\ [0, 1]$$



Earley Algorithm

Optional

I am now under a rule

$\mathbf{N} \rightarrow \mathit{flies}$

covering a span with starting point

1

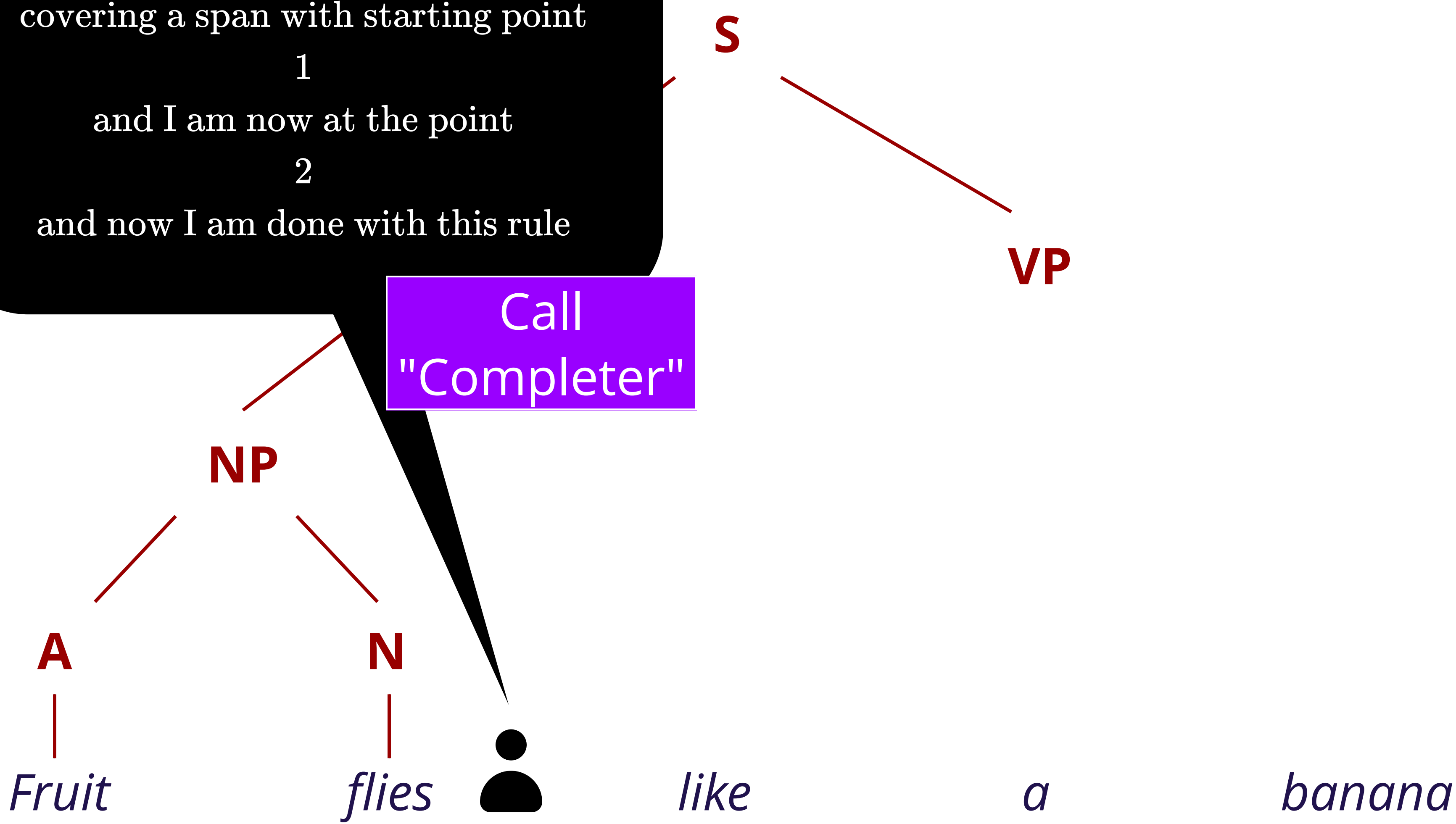
and I am now at the point

2

and now I am done with this rule

$\mathbf{N} \rightarrow \mathit{flies} \bullet \;\; [1, 2]$

Call "Completer"



Optional

Earley Algorithm

$$\mathbf{NP} \rightarrow \mathbf{A}\ \mathbf{N} \bullet \quad [0, 2]$$

I am now under a rule

$\mathbf{NP} \rightarrow \mathbf{A}\ \mathbf{N}$

covering a span with starting point

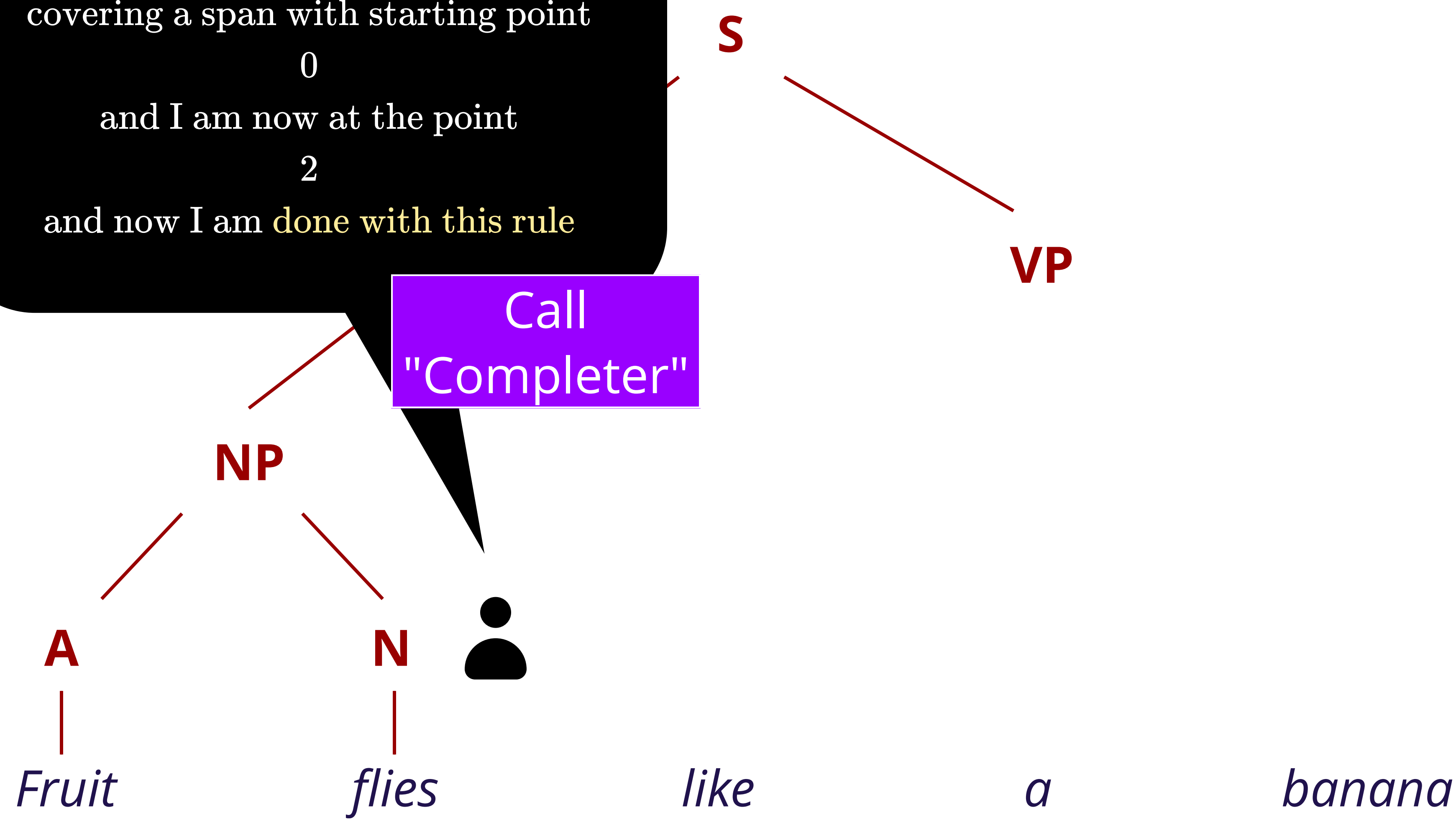
0

and I am now at the point

2

and now I am done with this rule

Call "Completer"



Optional

Earley Algorithm

I am now under a rule

$\mathbf{S} \rightarrow \mathbf{NP}\ \mathbf{VP}$

covering a span with starting point

0

and I am now at the point

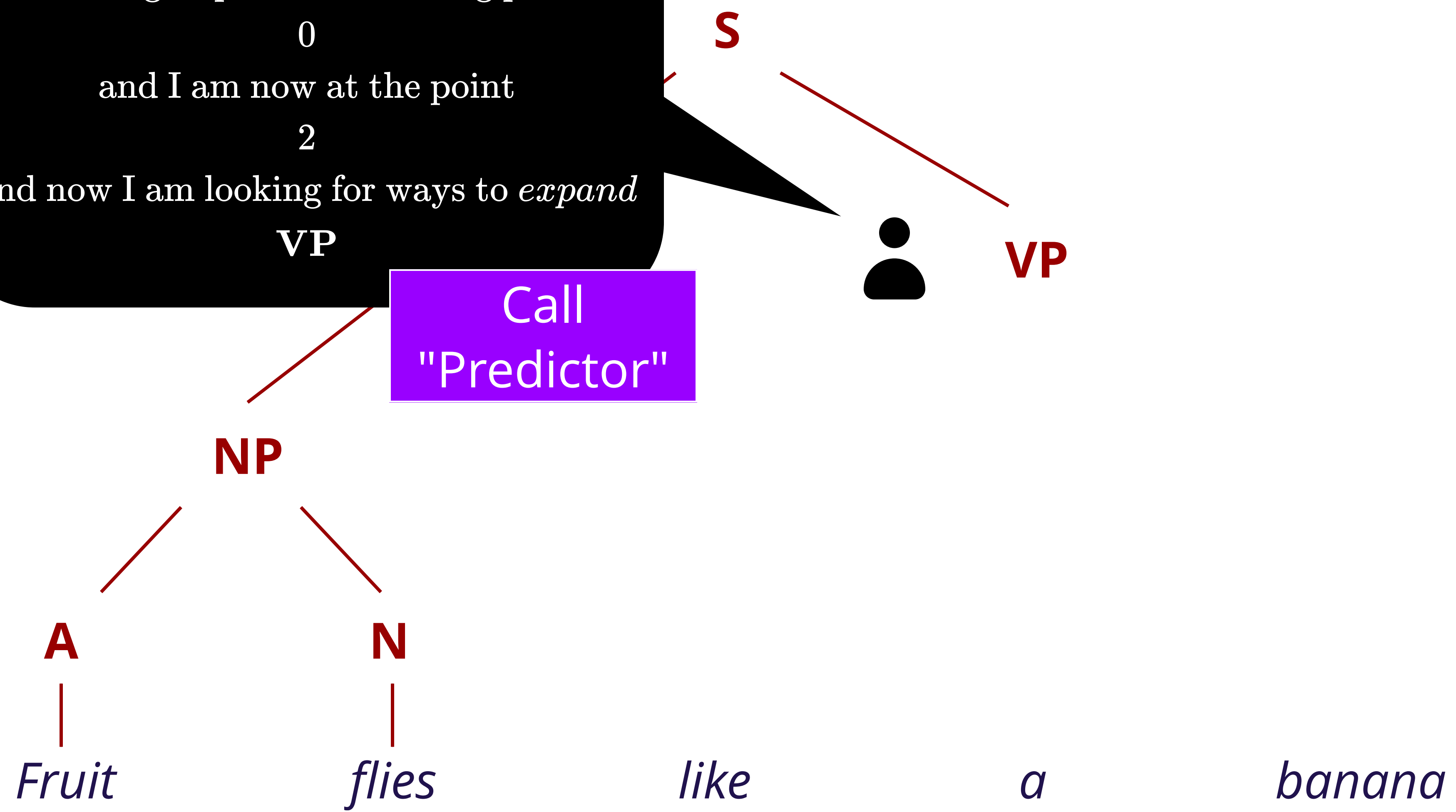
2

and now I am looking for ways to *expand*

$\mathbf{VP}$

$$\mathbf{S} \rightarrow \mathbf{NP} \bullet \mathbf{VP}\ [0, 2]$$

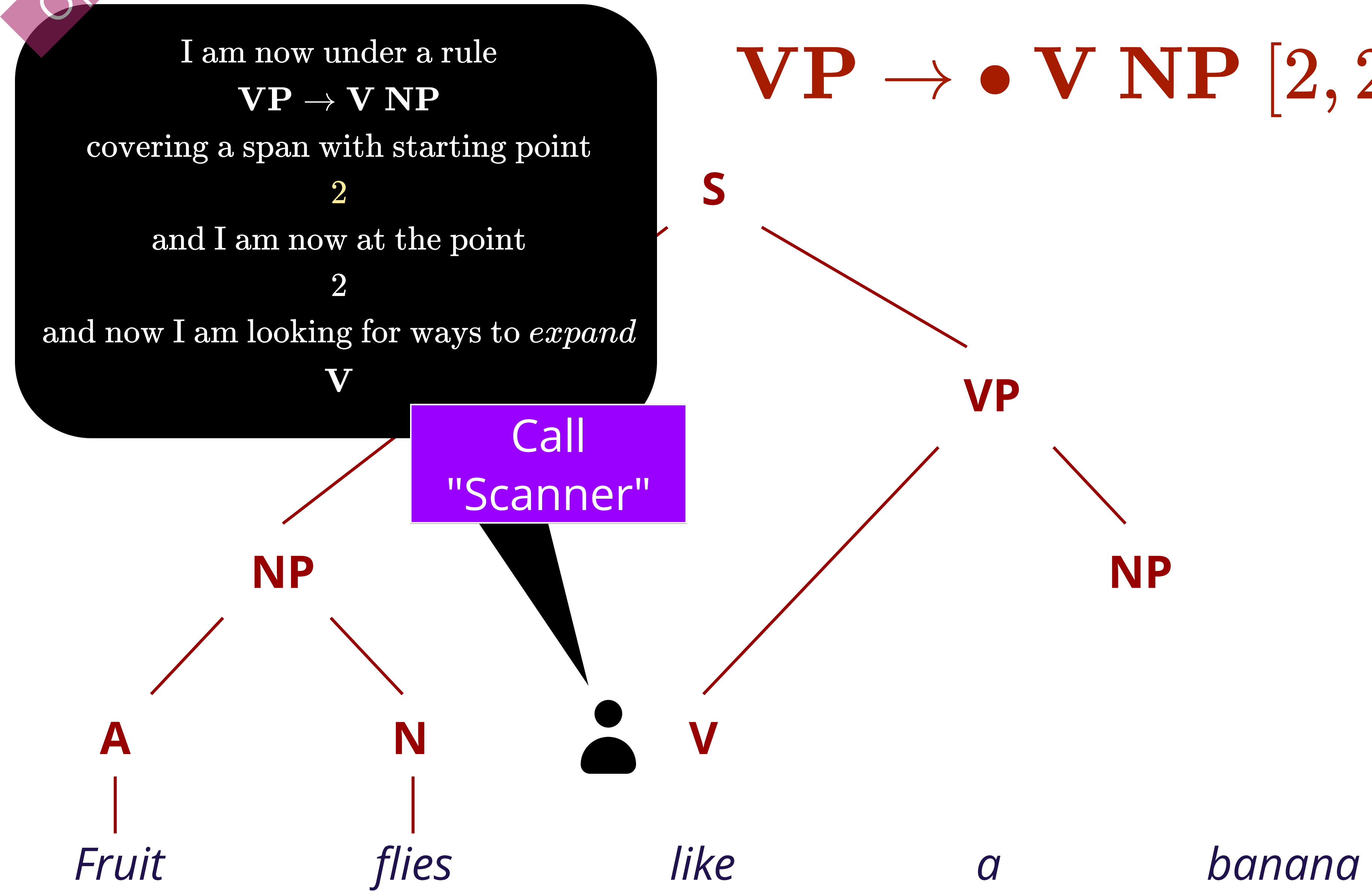
Call "Predictor"



Earley Algorithm

Optional

$$\mathbf{VP} \rightarrow \bullet\, \mathbf{V}\ \mathbf{NP}\ [2, 2]$$



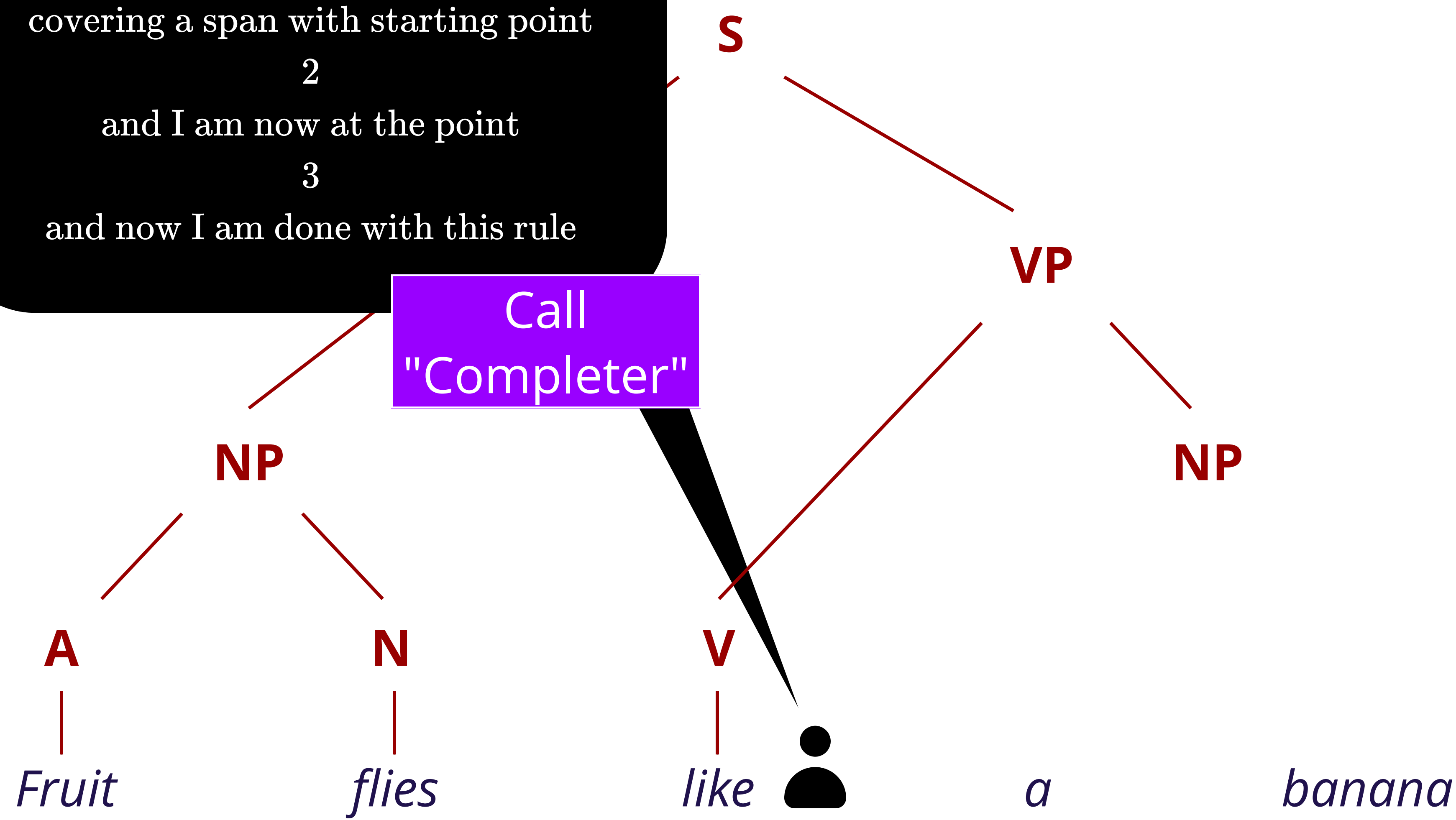
Earley Algorithm

Optional

$$\mathbf{V} \rightarrow \textit{like} \bullet \quad [2, 3]$$

I am now under a rule
$\mathbf{V} \rightarrow \textit{like}$
covering a span with starting point
2
and I am now at the point
3
and now I am done with this rule

Call "Completer"



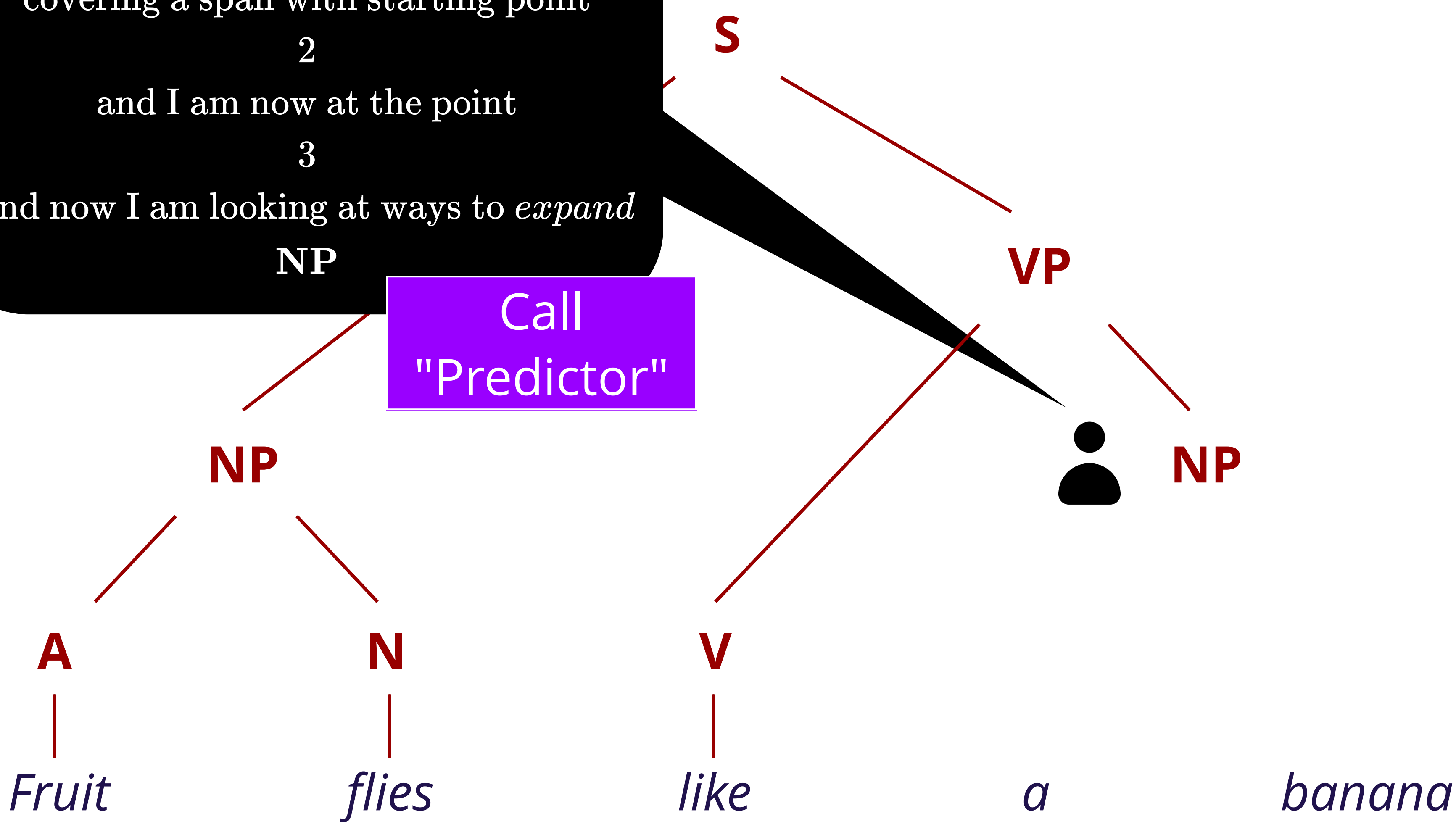
Earley Algorithm

Optional

$$\mathbf{VP} \rightarrow \mathbf{V} \bullet \mathbf{NP}\ [2, 3]$$

I am now under a rule
$\mathbf{VP} \rightarrow \mathbf{V}\ \mathbf{NP}$
covering a span with starting point
2
and I am now at the point
3
and now I am looking at ways to *expand*
NP

Call "Predictor"



Optional

Earley Algorithm

I am now under a rule

NP → D N

covering a span with starting point

3

and I am now at the point

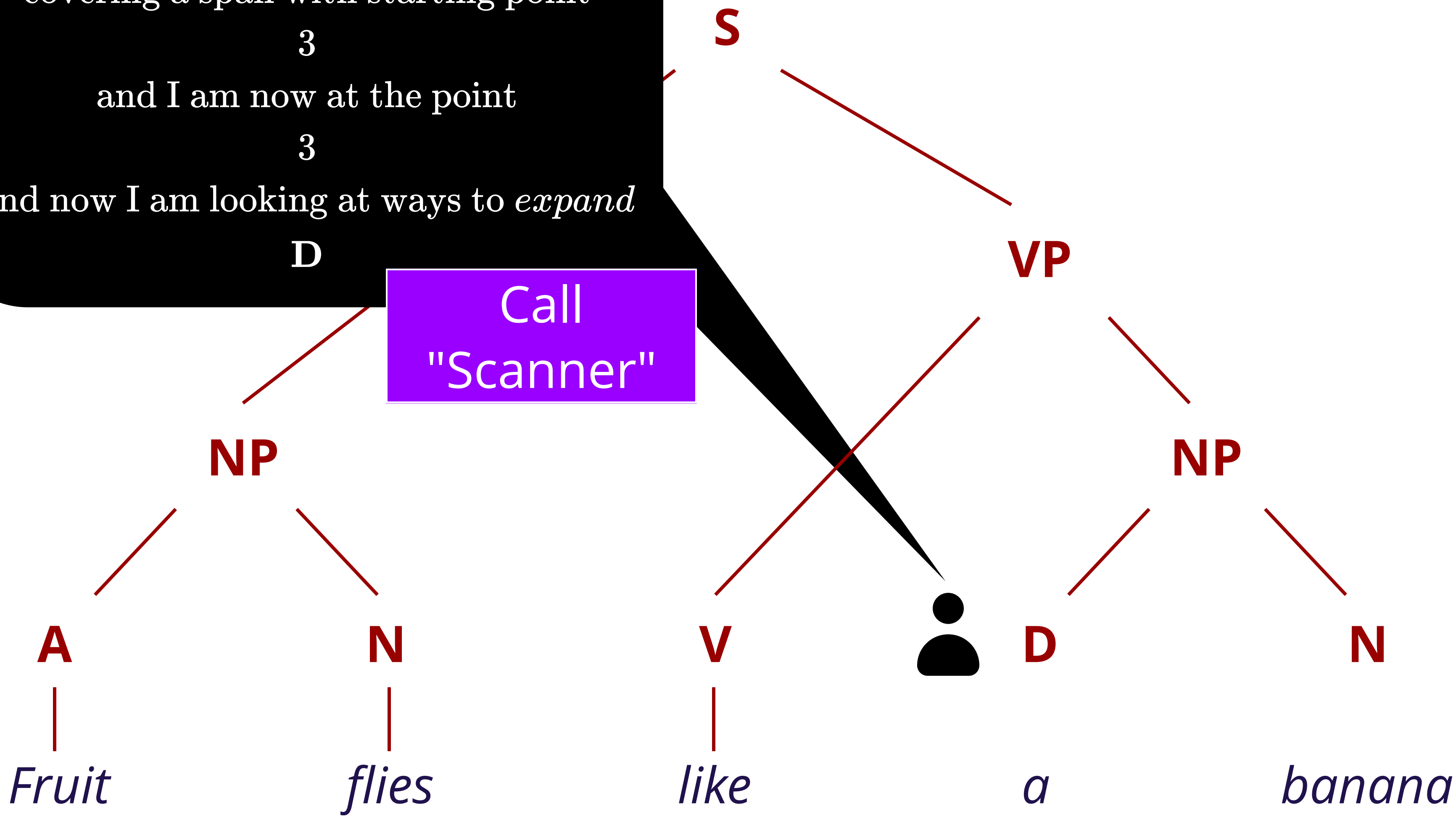
3

and now I am looking at ways to *expand*

D

$$\mathbf{NP} \rightarrow \bullet\, \mathbf{D}\ \mathbf{N}\ [3,3]$$

Call "Scanner"



Earley Algorithm

Optional

I am now under a rule

$\mathbf{D} \to a$

covering a span with starting point

3

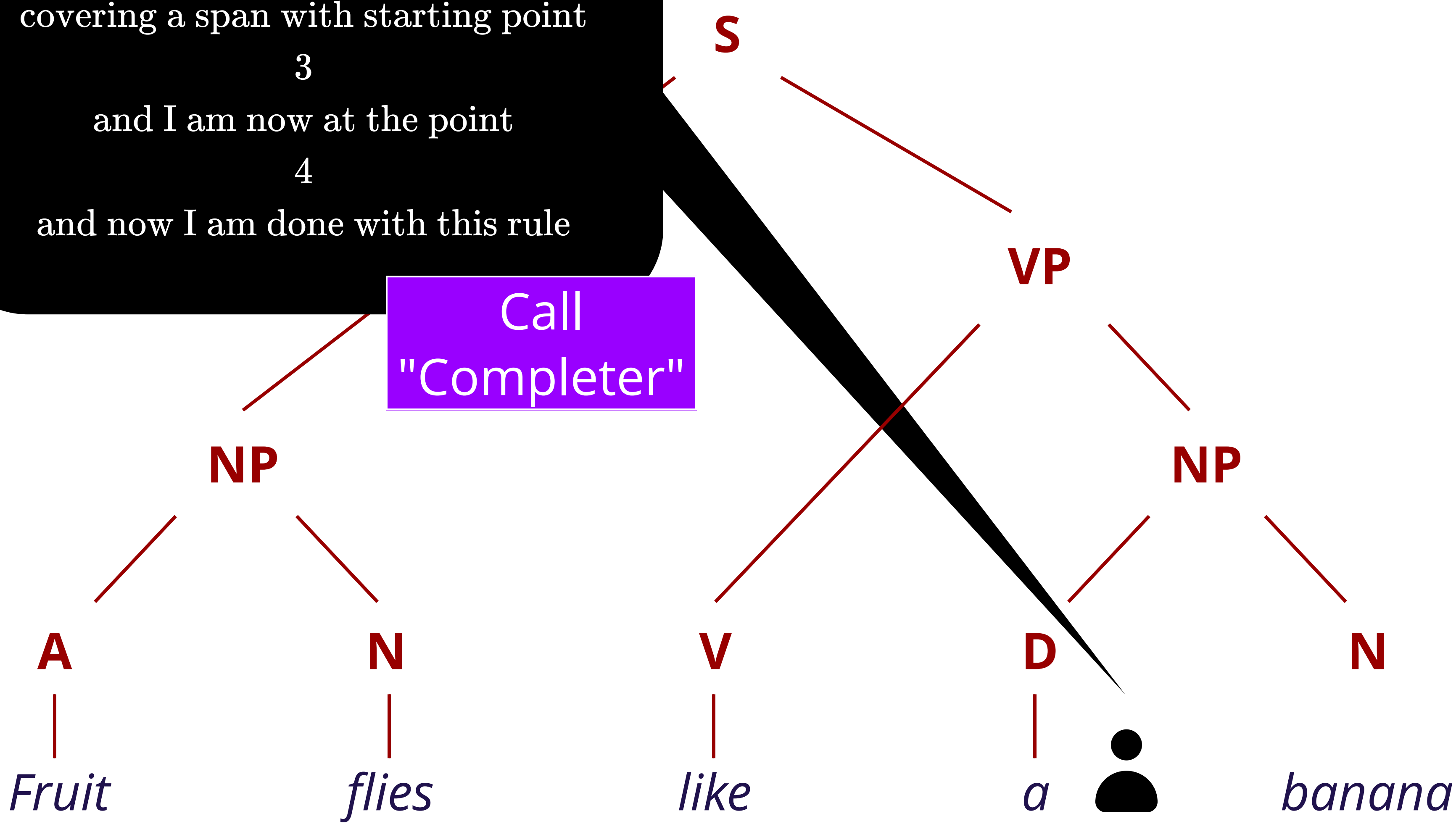
and I am now at the point

4

and now I am done with this rule

$$\mathbf{D} \to a \bullet [3, 4]$$

Call "Completer"



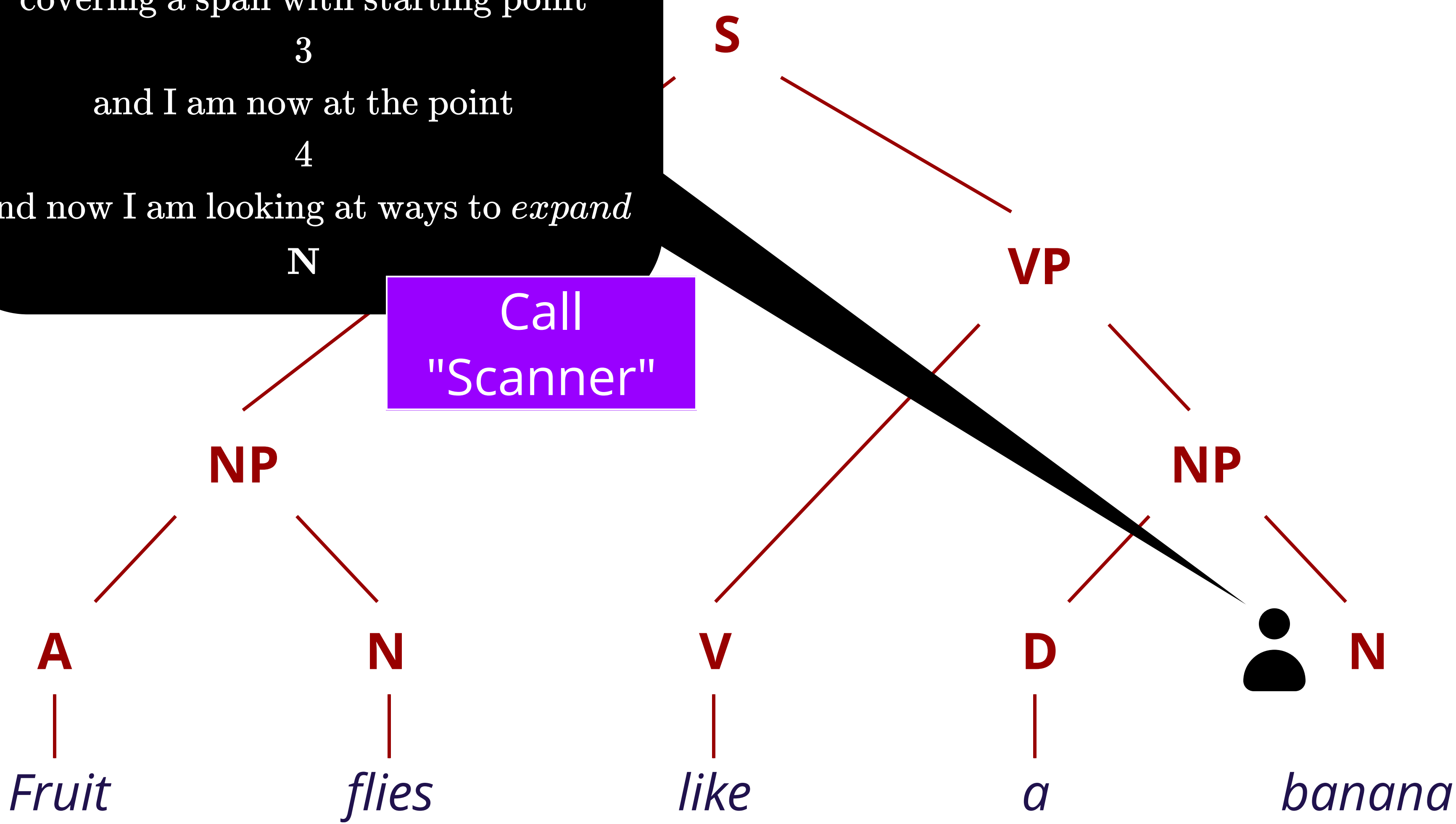
Earley Algorithm

Optional

$$\mathbf{NP} \rightarrow \mathbf{D} \bullet \mathbf{N}\ [3,4]$$

I am now under a rule
NP $\rightarrow$ **D N**
covering a span with starting point
3
and I am now at the point
4
and now I am looking at ways to *expand*
N

Call "Scanner"



Optional

Earley Algorithm

I am now under a rule

$\mathbf{N} \to \mathit{banana}$

covering a span with starting point

4

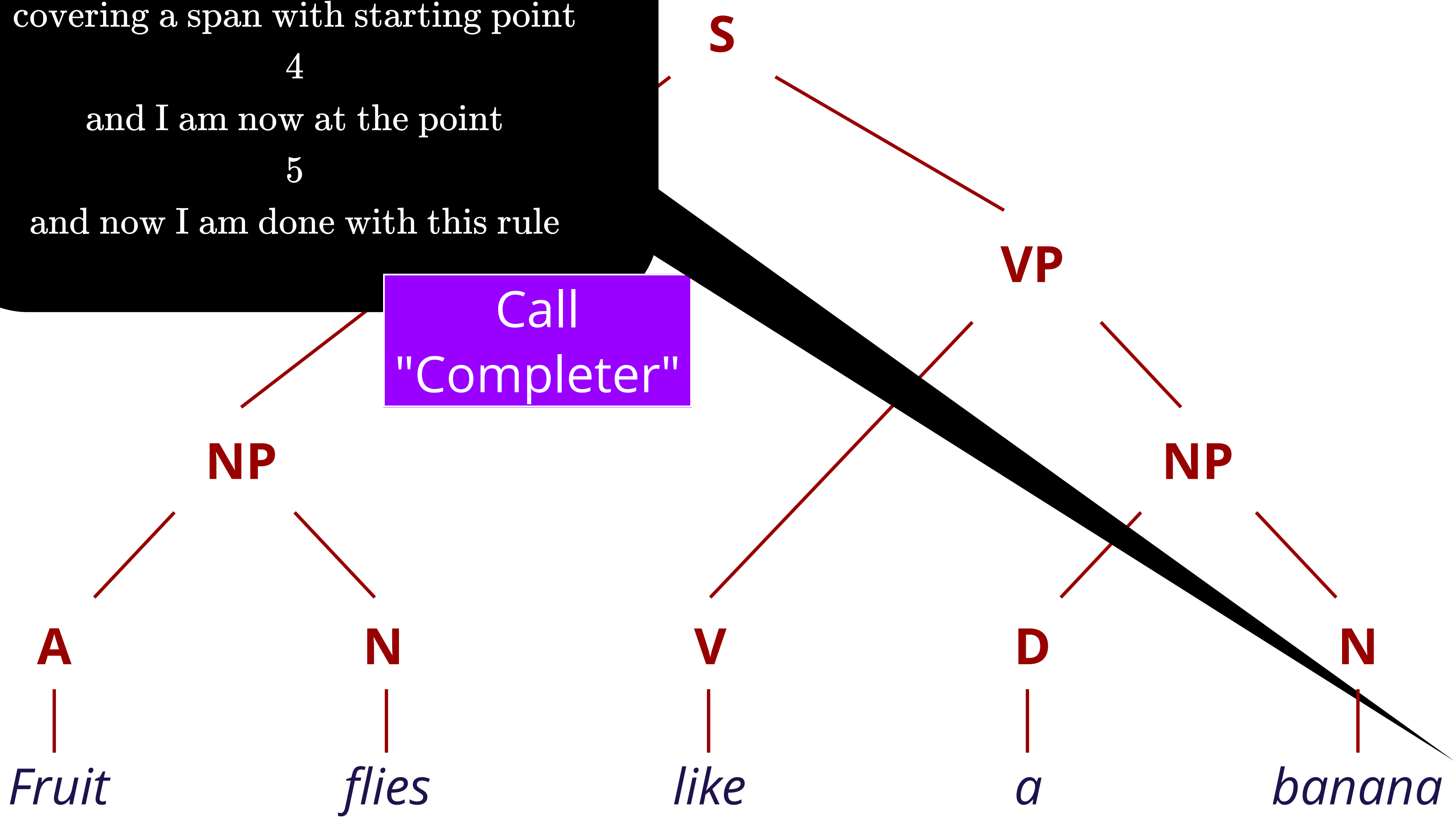
and I am now at the point

5

and now I am done with this rule

$$\mathbf{N} \to \mathit{banana} \bullet \ [4, 5]$$

Call "Completer"



Optional

Earley Algorithm

I am now under a rule

$\mathbf{NP} \rightarrow \mathbf{D\ N}$

covering a span with starting point

3

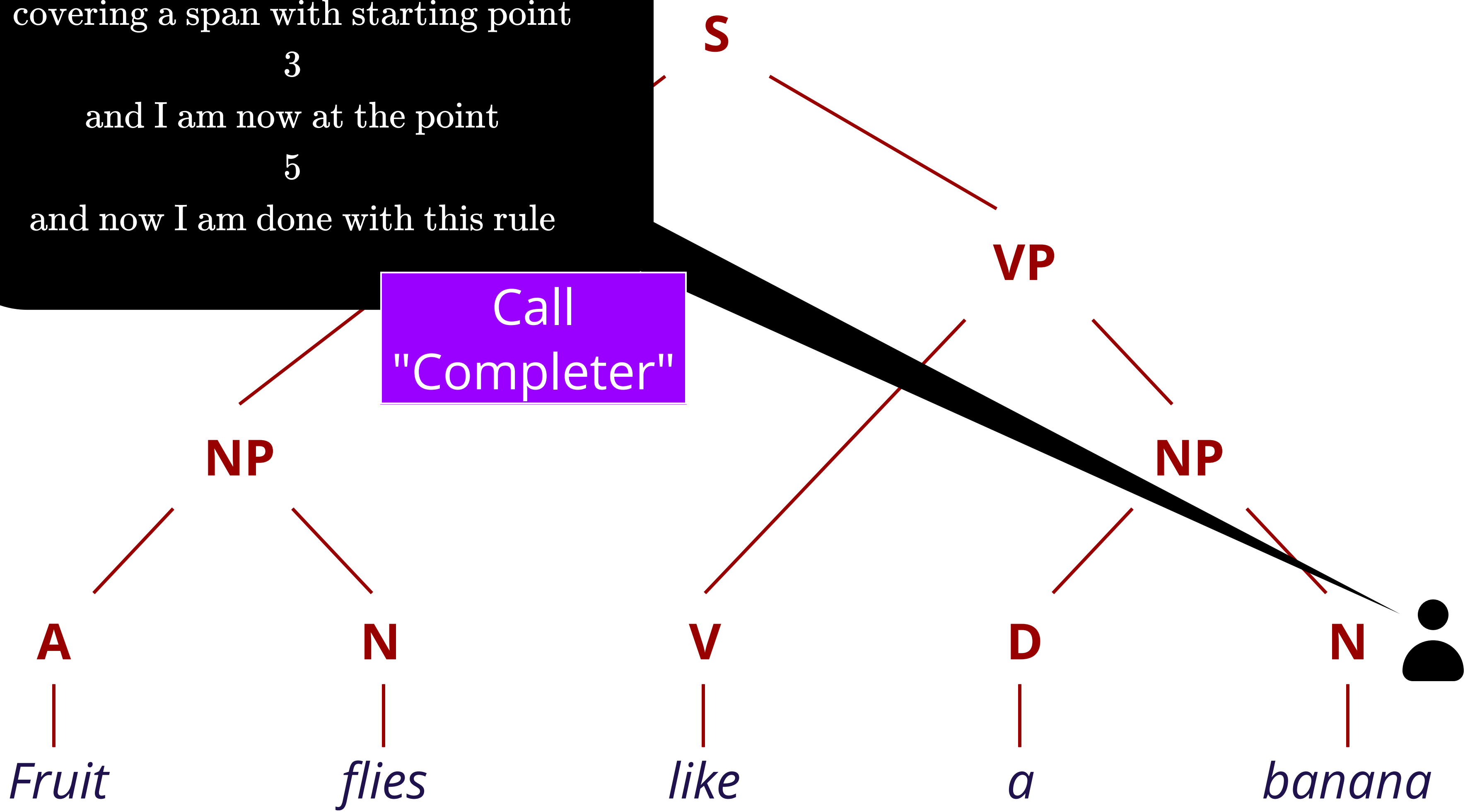
and I am now at the point

5

and now I am done with this rule

$$\mathbf{NP} \rightarrow \mathbf{D\ N} \bullet \ [3, 5]$$

Call "Completer"



Optional

Earley Algorithm

$$\mathbf{VP} \rightarrow \mathbf{V\ NP} \bullet \ [2, 5]$$

I am now under a rule

$\mathbf{VP} \rightarrow \mathbf{V\ NP}$

covering a span with starting point

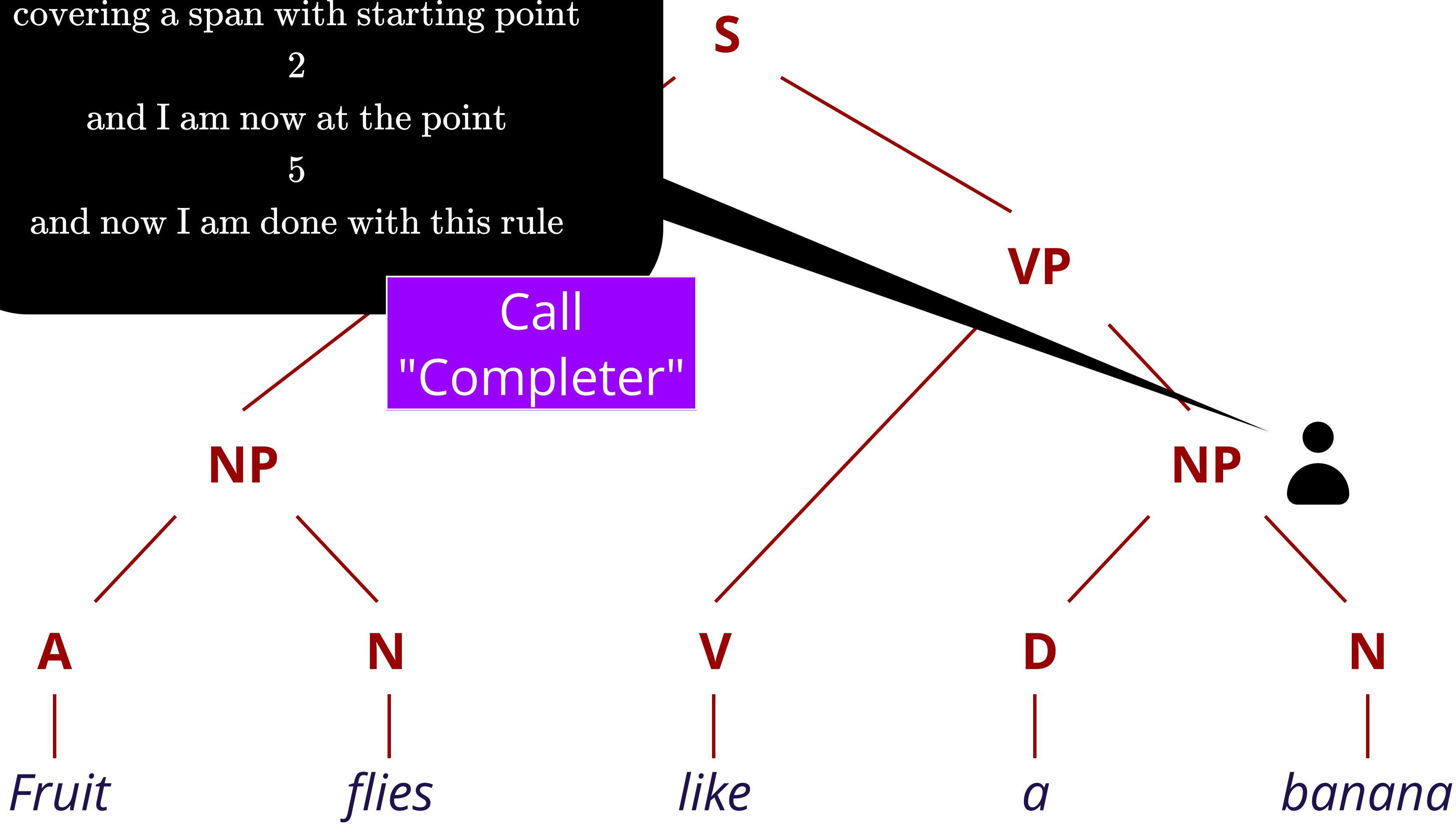
2

and I am now at the point

5

and now I am done with this rule

Call "Completer"



Optional

Earley Algorithm

I am now under a rule

$\mathbf{S \rightarrow NP\ VP}$

covering a span with starting point

0

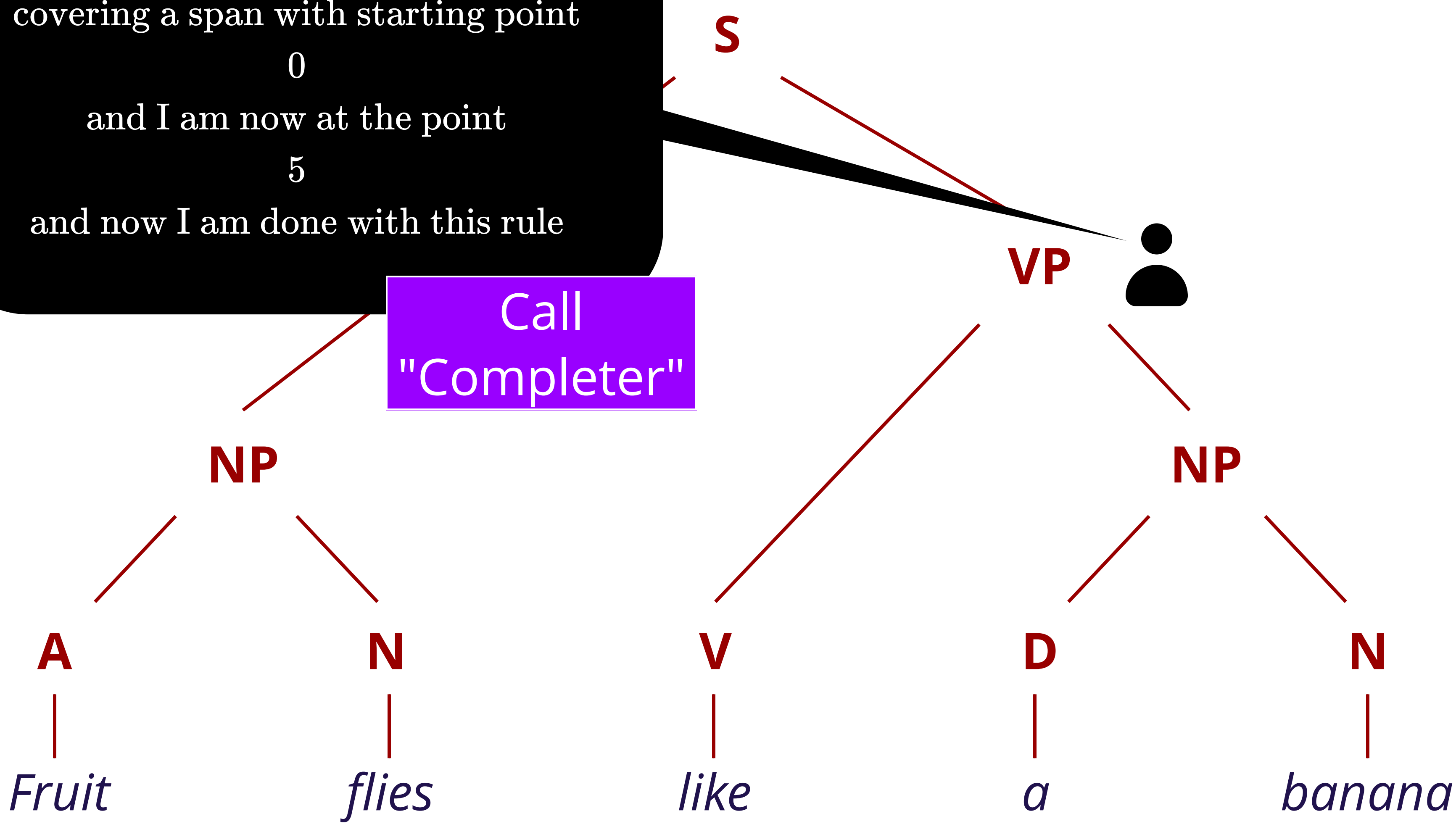
and I am now at the point

5

and now I am done with this rule

$$\mathbf{S \rightarrow NP\ VP} \bullet \ [0,5]$$

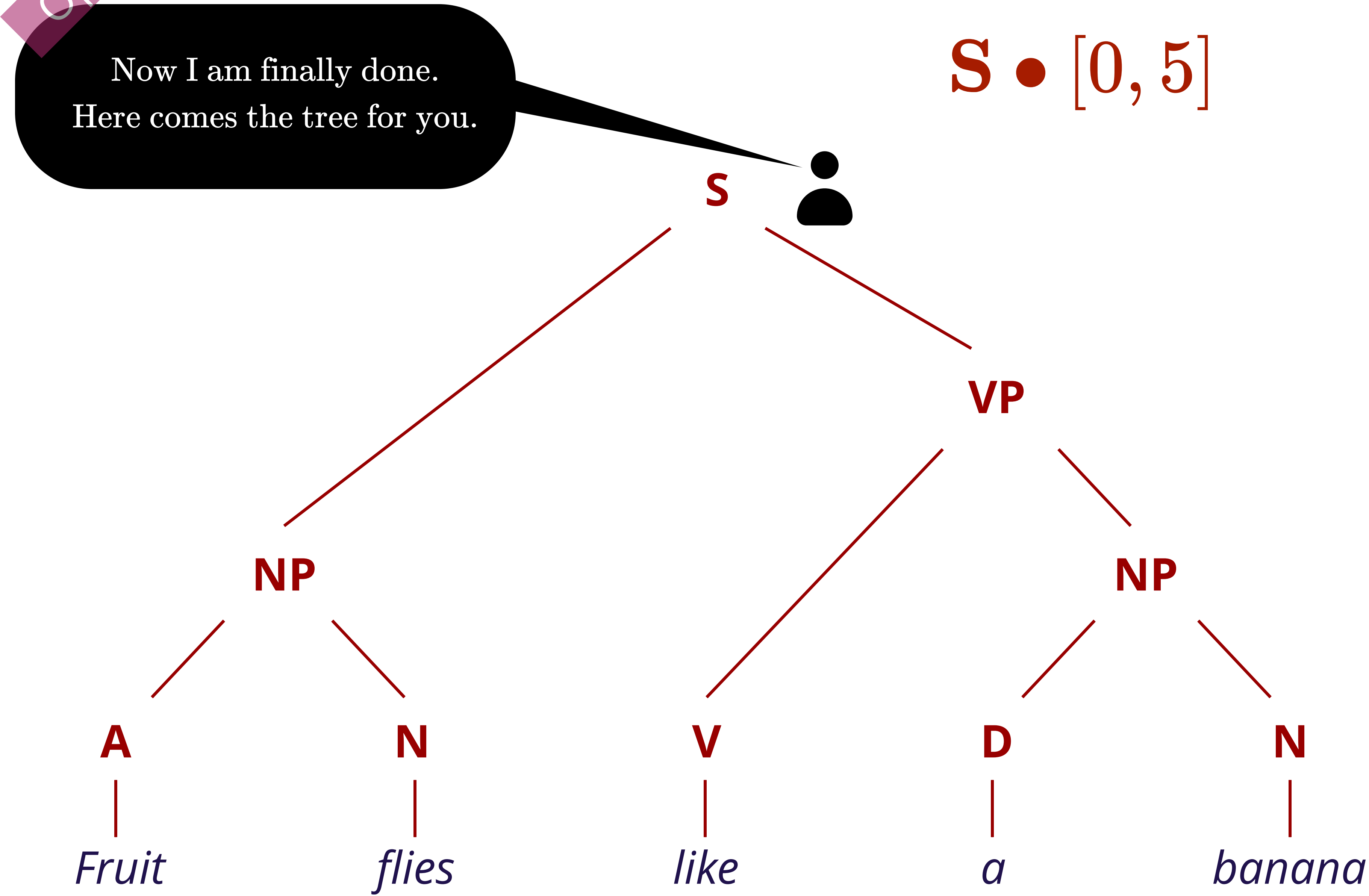
Call "Completer"



Earley Algorithm

Optional

$$\mathbf{S} \bullet [0, 5]$$



Optional

Questions

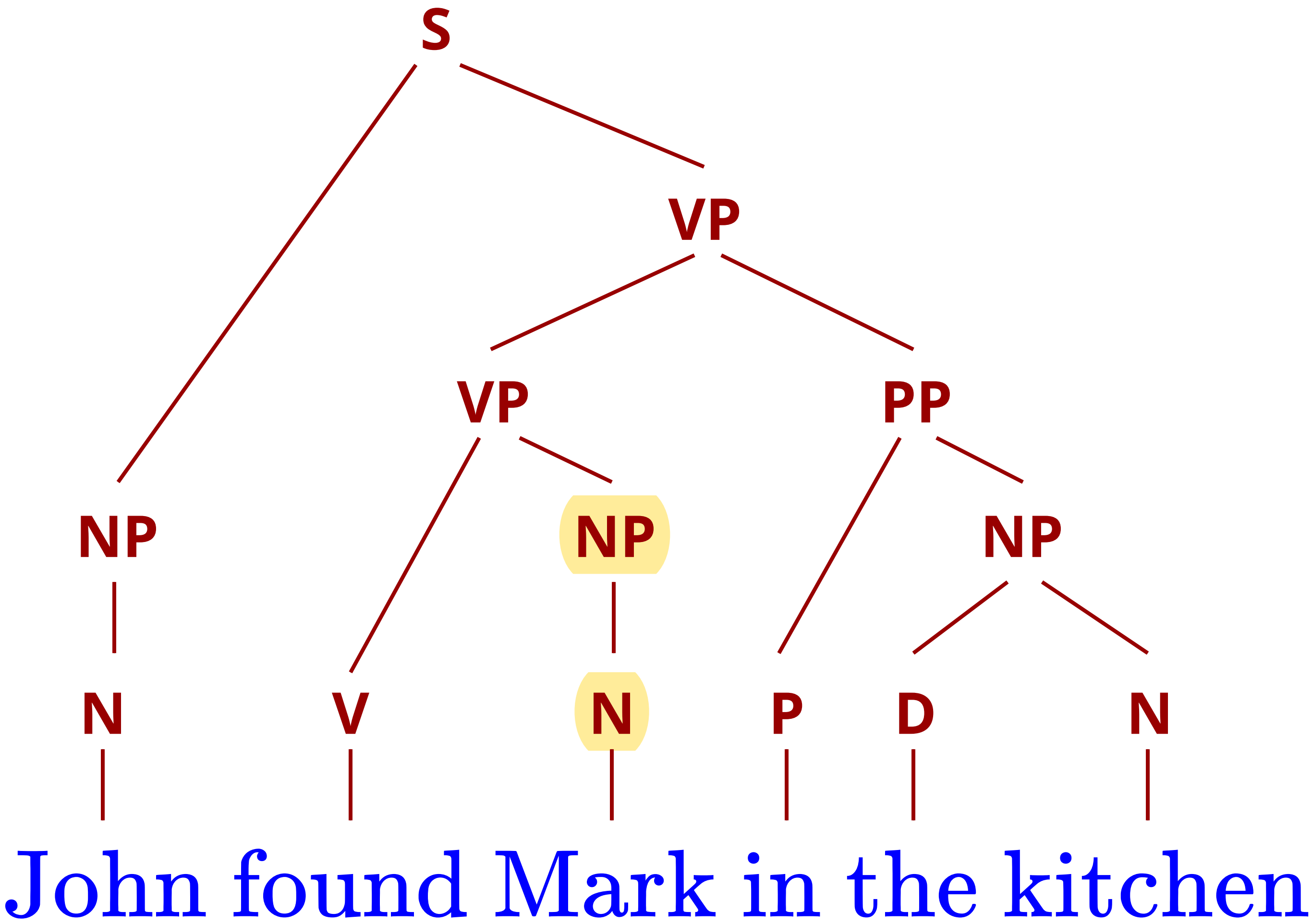
What if we have multiple valid parse trees? How to handle probabilistic rules?

CKY vs Earley
Comparison

CKY	Earley
Bottom-up	Top-down (It builds a chart in a left-to-right, bottom-up manner, but performs prediction in a top-down manner)
More efficient if the grammar is converted into CNF	Does not require conversion of the grammar into CNF

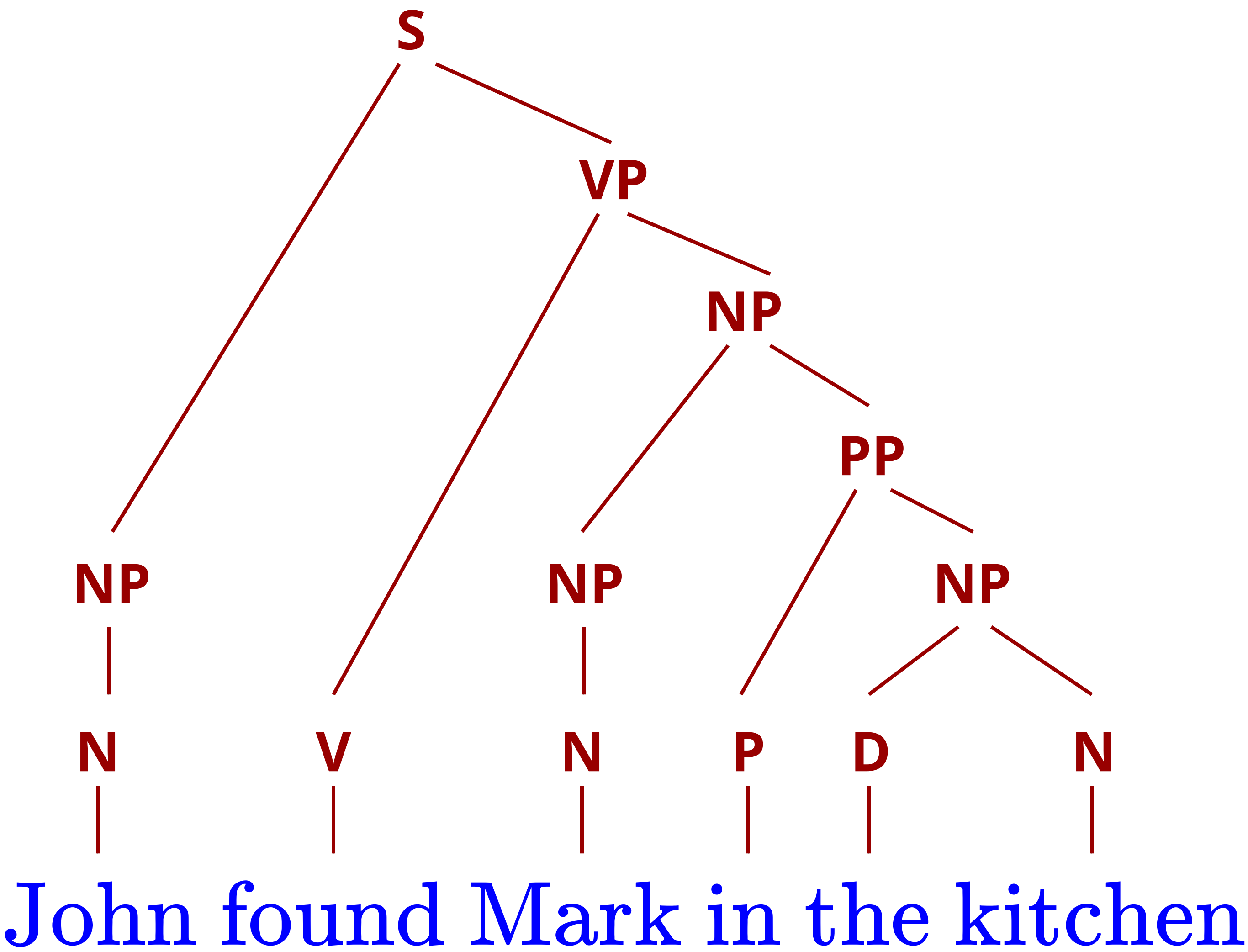
😃 For Earley algorithm, understanding the general procedure is sufficient, no implementation or algorithm questions will be asked in this course.

Problems with PCFG



What are the grammar rules involved?

Problems with PCFG



What are the grammar rules involved?

Problems with PCFG

$$
\begin{array}{ccl}
\mathbf{S} & \rightarrow & \mathbf{NP}\ \ \mathbf{VP} \\
\mathbf{NP} & \rightarrow & \mathbf{N} \\
\mathbf{VP} & \rightarrow & \mathbf{VP}\ \ \mathbf{PP} \\
\mathbf{VP} & \rightarrow & \mathbf{V}\ \ \mathbf{NP} \\
\mathbf{NP} & \rightarrow & \mathbf{N} \\
\mathbf{PP} & \rightarrow & \mathbf{P}\ \ \mathbf{NP} \\
\mathbf{NP} & \rightarrow & \mathbf{D}\ \ \mathbf{N} \\
\mathbf{N} & \rightarrow & John \\
\mathbf{V} & \rightarrow & found \\
\mathbf{N} & \rightarrow & Mark \\
\mathbf{P} & \rightarrow & in \\
\mathbf{D} & \rightarrow & the \\
\mathbf{N} & \rightarrow & kitchen
\end{array}
\qquad\qquad
\begin{array}{ccl}
\mathbf{S} & \rightarrow & \mathbf{NP}\ \ \mathbf{VP} \\
\mathbf{NP} & \rightarrow & \mathbf{N} \\
\mathbf{NP} & \rightarrow & \mathbf{NP}\ \ \mathbf{PP} \\
\mathbf{VP} & \rightarrow & \mathbf{V}\ \ \mathbf{NP} \\
\mathbf{NP} & \rightarrow & \mathbf{N} \\
\mathbf{PP} & \rightarrow & \mathbf{P}\ \ \mathbf{NP} \\
\mathbf{NP} & \rightarrow & \mathbf{D}\ \ \mathbf{N} \\
\mathbf{N} & \rightarrow & John \\
\mathbf{V} & \rightarrow & found \\
\mathbf{N} & \rightarrow & Mark \\
\mathbf{P} & \rightarrow & in \\
\mathbf{D} & \rightarrow & the \\
\mathbf{N} & \rightarrow & kitchen
\end{array}
$$

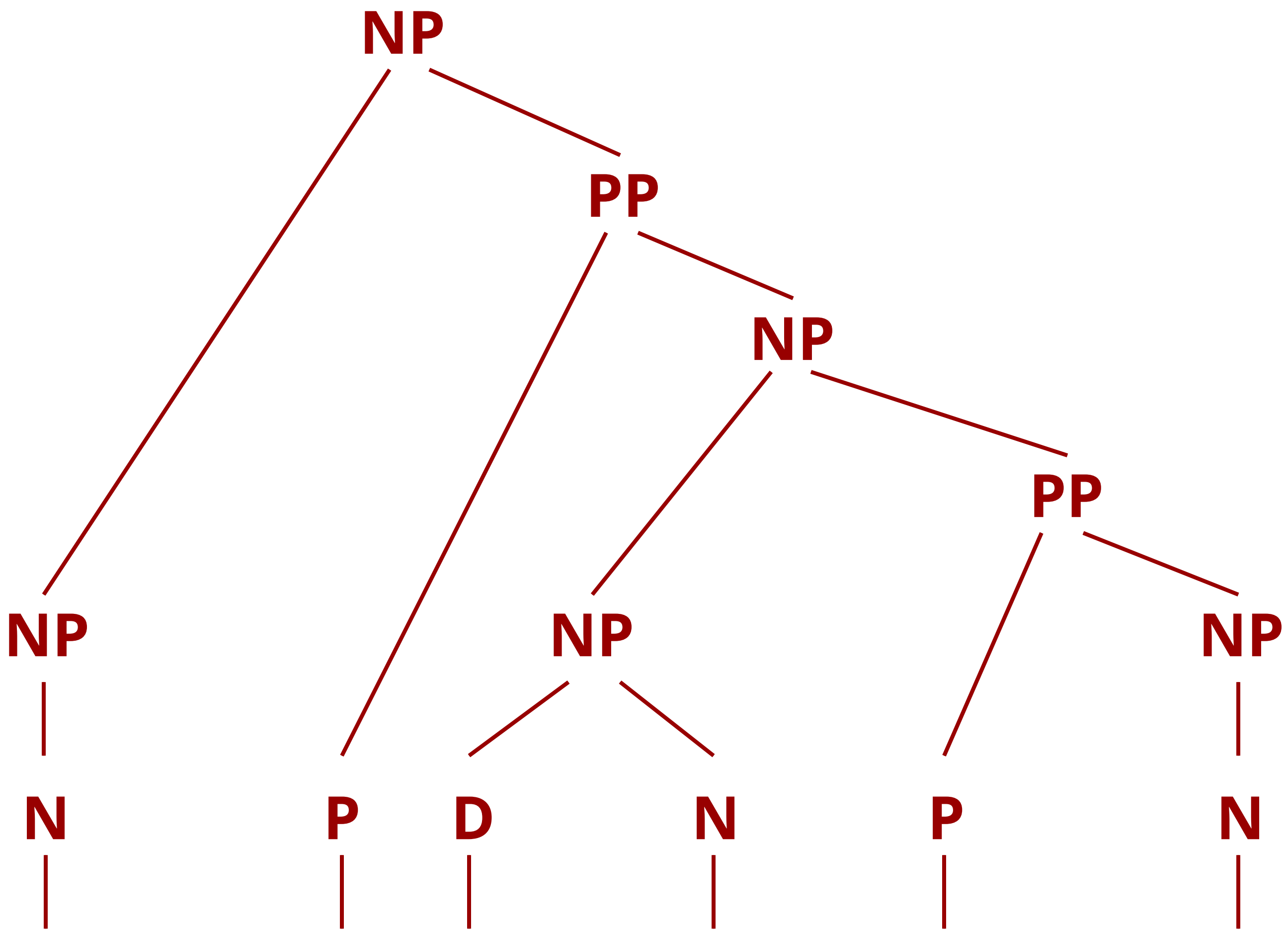
Only one rule is different, which has nothing to do with the lexical information of the sentence.

Problems with PCFG

Limitation 1

Lack of Sensitivity to Lexical Information

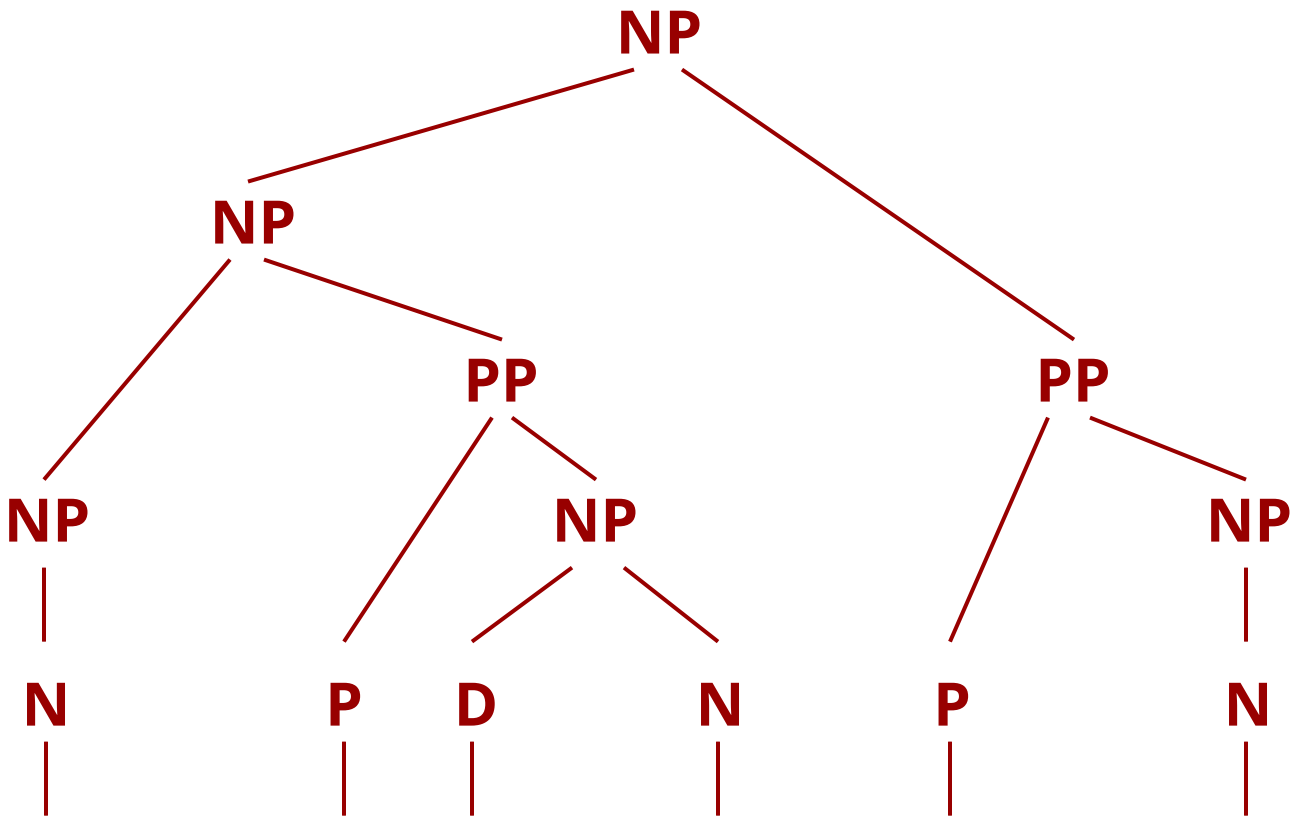
Problems with PCFG



co-founder of the startup in Singapore

What are the grammar rules involved?

Problems with PCFG



co-founder of the startup in Singapore

What about this tree? What are the rules?

Problems with PCFG

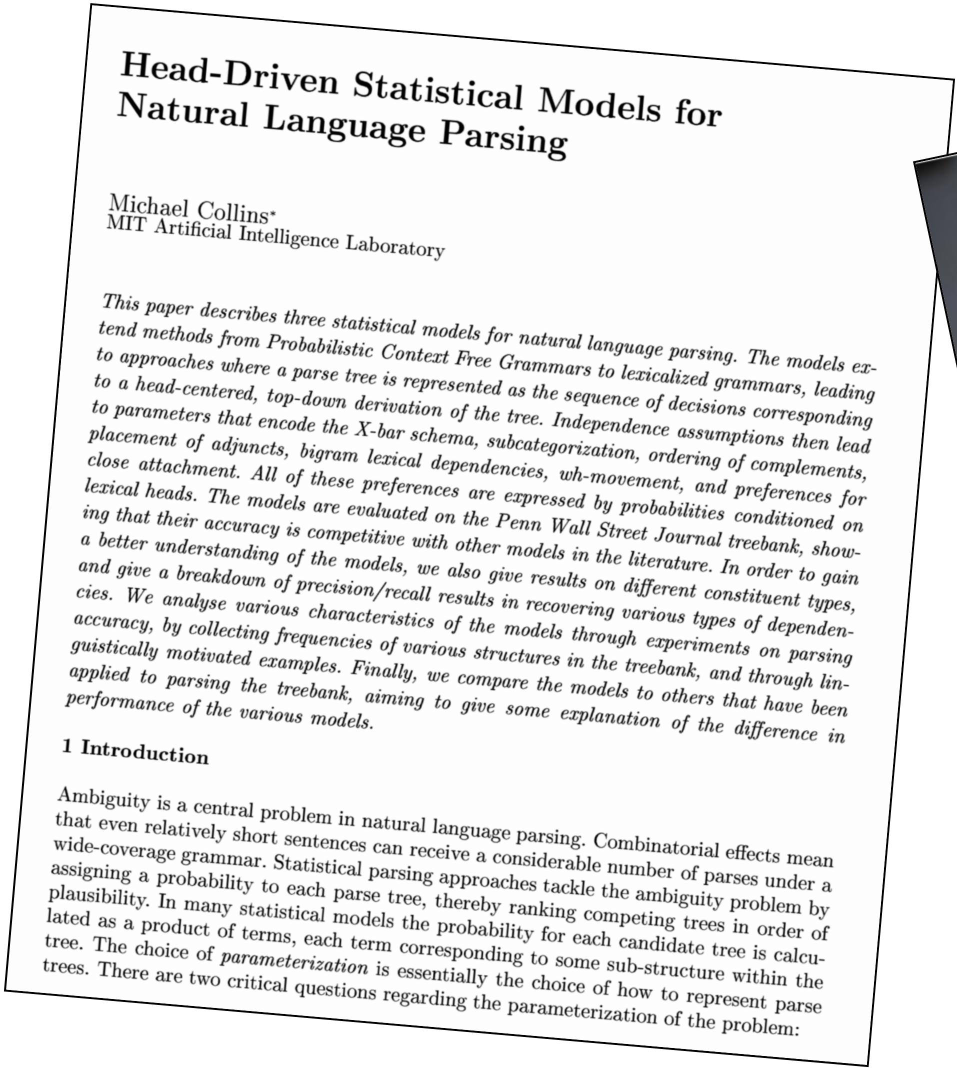
<u>Limitation 1</u>

Lack of Sensitivity to Lexical Information

<u>Limitation 2</u>

Lack of Sensitivity to Structural Preference

Lexicalized Parser
(Collins, 1999)



Head-Driven Statistical Models for Natural Language Parsing

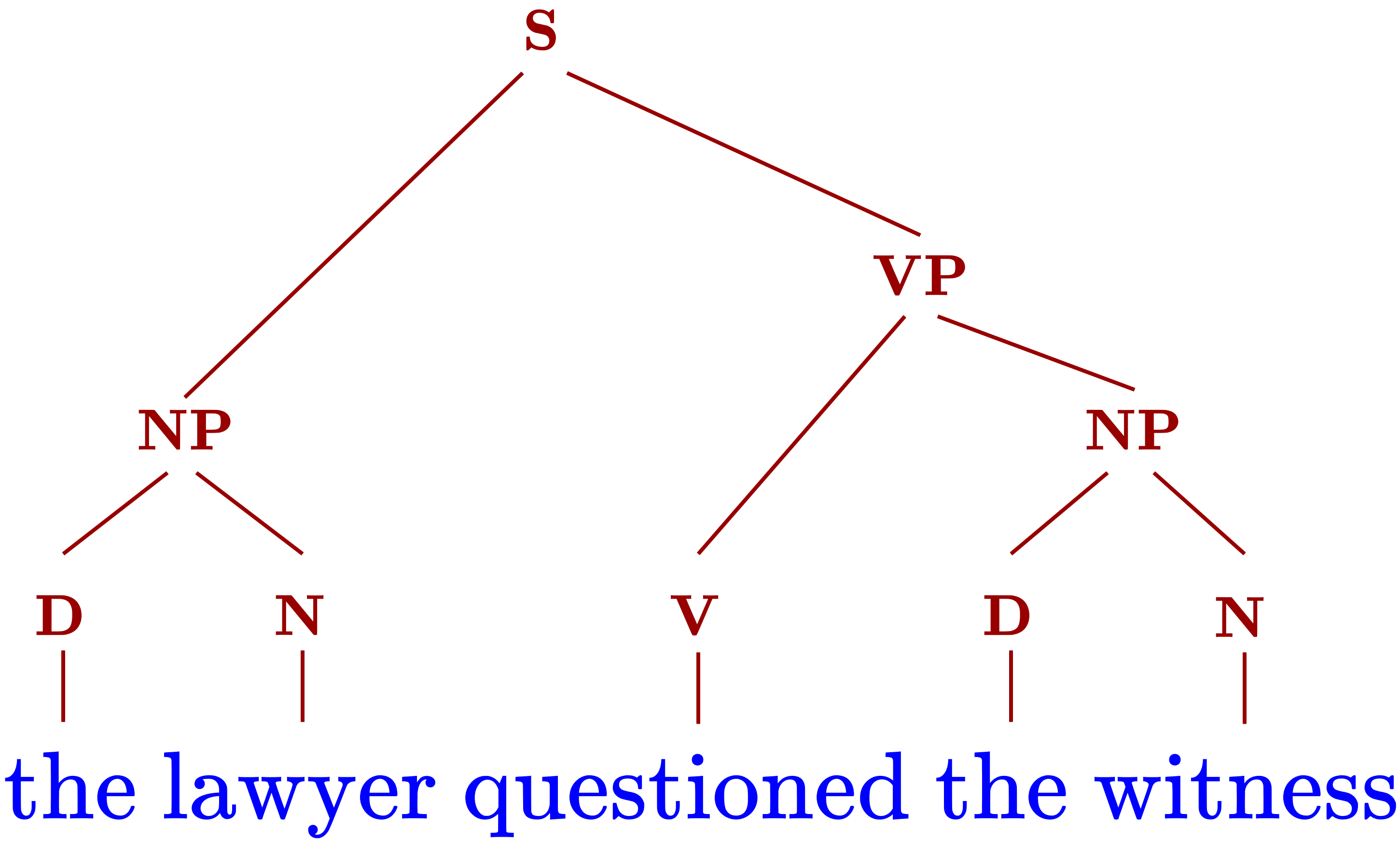
Michael Collins*
MIT Artificial Intelligence Laboratory

This paper describes three statistical models for natural language parsing. The models extend methods from Probabilistic Context Free Grammars to lexicalized grammars, leading to approaches where a parse tree is represented as the sequence of decisions corresponding to a head-centered, top-down derivation of the tree. Independence assumptions then lead to parameters that encode the X-bar schema, subcategorization, ordering of complements, placement of adjuncts, bigram lexical dependencies, wh-movement, and preferences for close attachment. All of these preferences are expressed by probabilities conditioned on lexical heads. The models are evaluated on the Penn Wall Street Journal treebank, showing that their accuracy is competitive with other models in the literature. In order to gain a better understanding of the models, we also give results on different constituent types, and give a breakdown of precision/recall results in recovering various types of dependencies. We analyse various characteristics of the models through experiments on parsing accuracy, by collecting frequencies of various structures in the treebank, and through linguistically motivated examples. Finally, we compare the models to others that have been applied to parsing the treebank, aiming to give some explanation of the difference in performance of the various models.

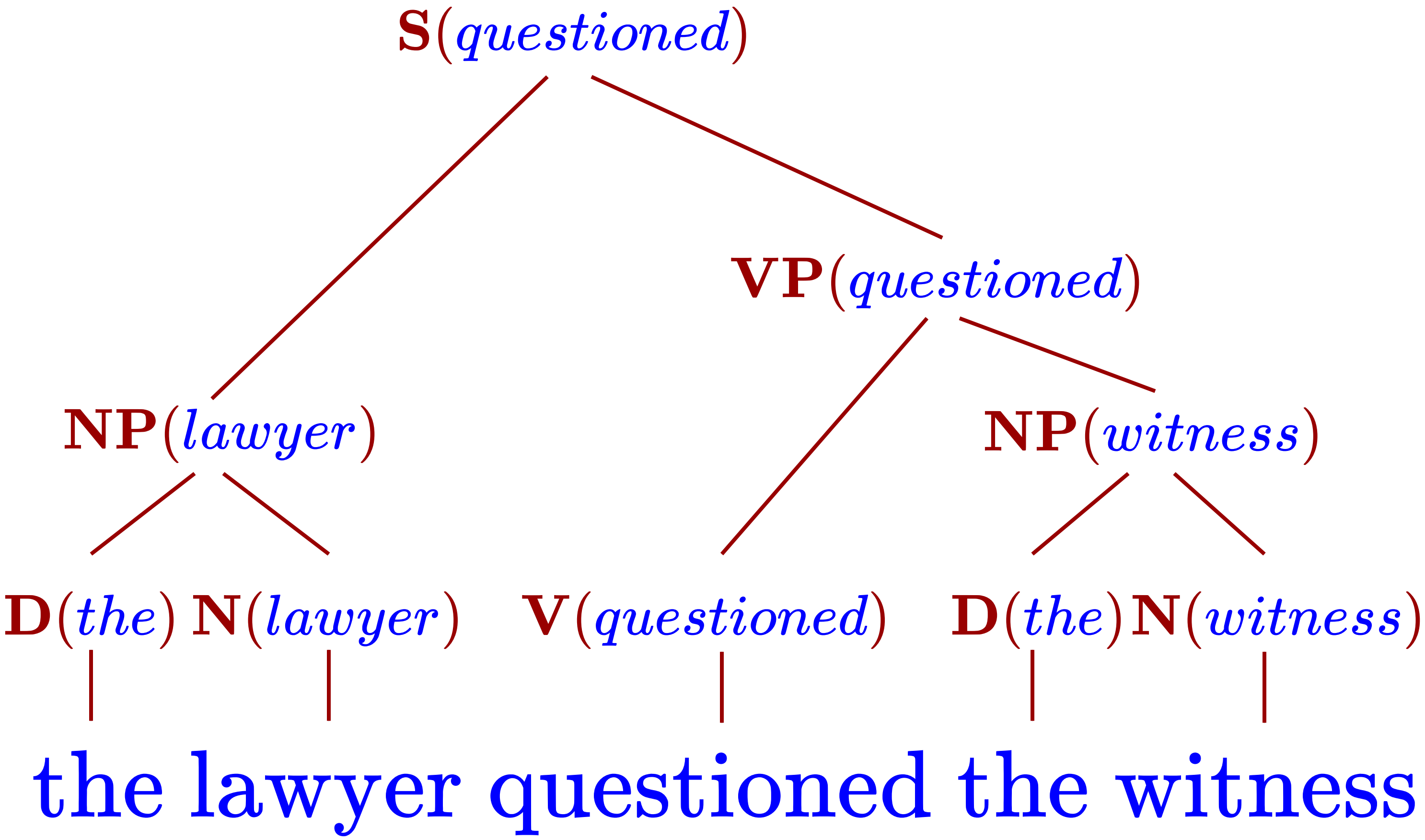
1 Introduction

Ambiguity is a central problem in natural language parsing. Combinatorial effects mean that even relatively short sentences can receive a considerable number of parses under a wide-coverage grammar. Statistical parsing approaches tackle the ambiguity problem by assigning a probability to each parse tree, thereby ranking competing trees in order of plausibility. In many statistical models the probability for each candidate tree is calculated as a product of terms, each term corresponding to some sub-structure within the tree. The choice of *parameterization* is essentially the choice of how to represent parse trees. There are two critical questions regarding the parameterization of the problem:

Conventional Parse Tree



Lexicalized Parse Tree



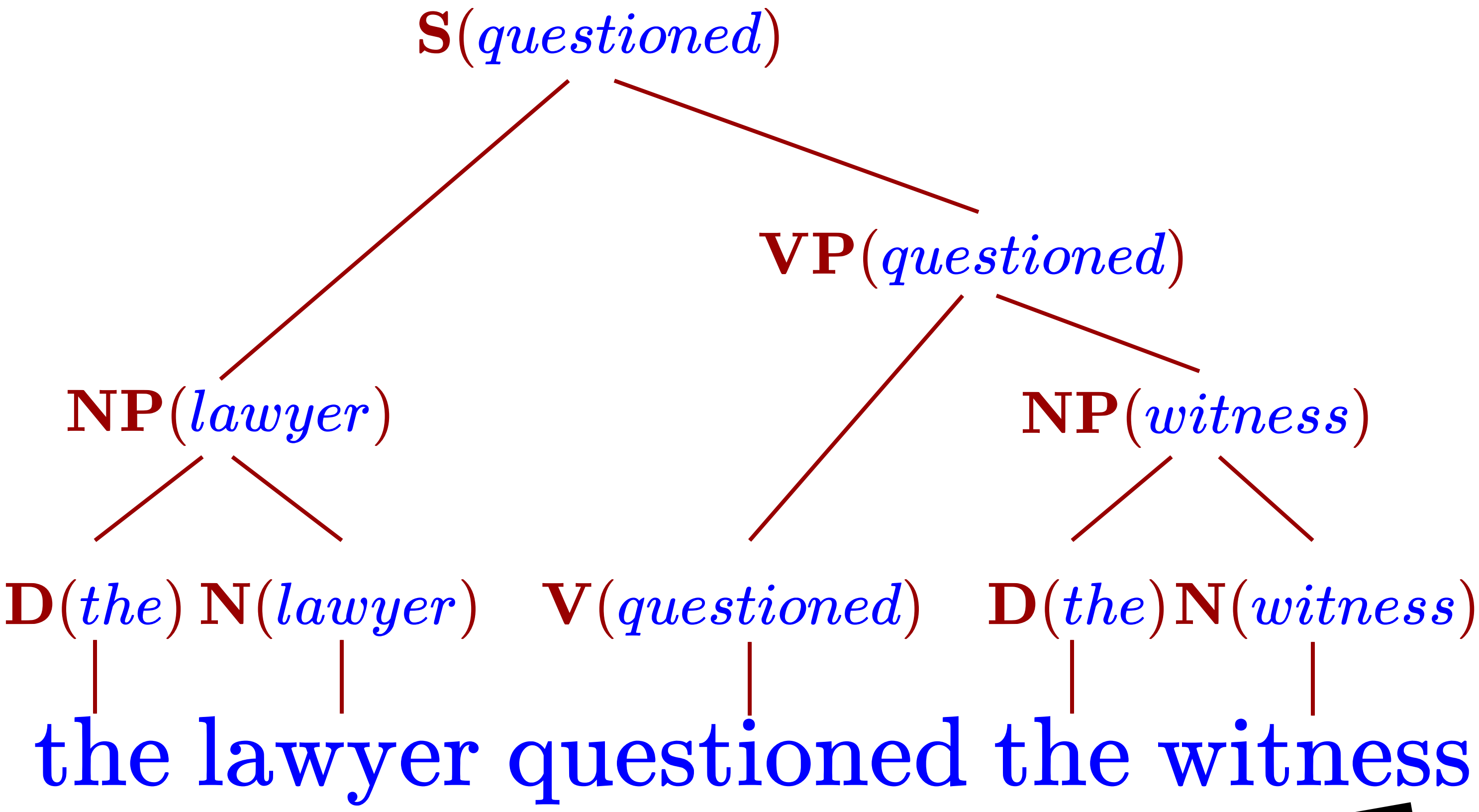
Lexicalized Parse Tree

How to choose the head?

If the rule contains **N**
    choose the rightmost **N**
Else if the rule contains an **NP**
    choose the leftmost **NP**
Else if the rule contains an **A**
    choose the rightmost **A**
...

Example of a set of rules that identifies the head of any rule whose left-hand side is an NP

Lexicalized Parse Tree



$\mathbf{S}(\mathit{questioned}) \rightarrow_2 \mathbf{NP}(\mathit{lawyer})\ \ \mathbf{VP}(\mathit{questioned})$

$\mathbf{NP}(\mathit{lawyer}) \rightarrow_2 \mathbf{D}(\mathit{the})\ \ \mathbf{N}(\mathit{lawyer})$

$\mathbf{D}(\mathit{the}) \rightarrow \mathit{the}$

$\mathbf{N}(\mathit{lawyer}) \rightarrow \mathit{lawyer}$

indicates where the head is from

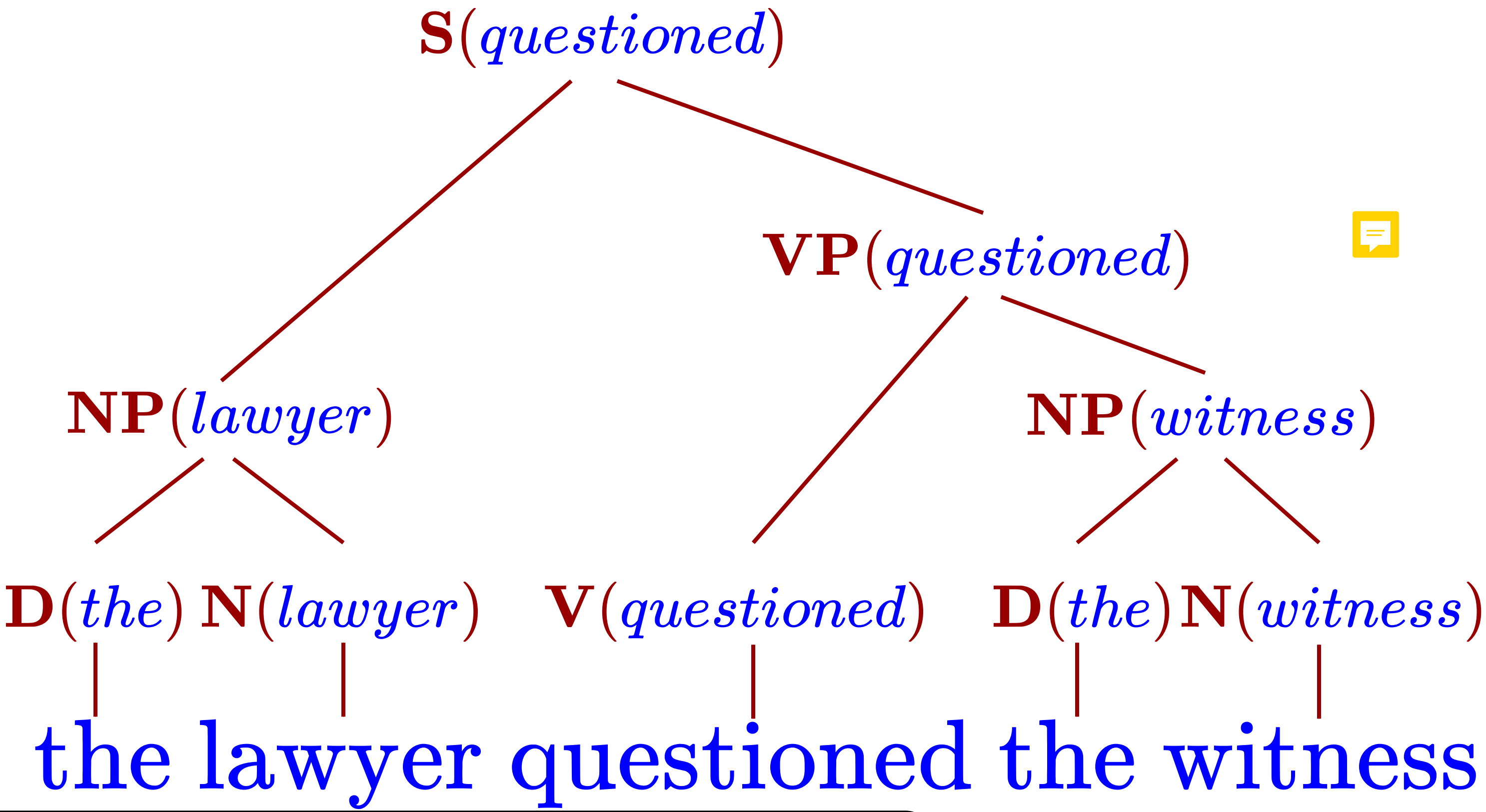
$\mathbf{VP}(\mathit{questioned}) \rightarrow_1 \mathbf{V}(\mathit{questioned})\ \ \mathbf{NP}(\mathit{witness})$

$\mathbf{NP}(\mathit{witness}) \rightarrow_2 \mathbf{D}(\mathit{the})\ \ \mathbf{N}(\mathit{witness})$

$\mathbf{D}(\mathit{the}) \rightarrow \mathit{the}$

$\mathbf{N}(\mathit{witness}) \rightarrow \mathit{witness}$

Lexicalized Parse Tree



$$
\begin{aligned}
& p(\mathbf{S}, \textit{questioned}) \\
& \times p(\mathbf{S}(\textit{questioned}) \rightarrow_2 \mathbf{NP}(\textit{lawyer}) \;\; \mathbf{VP}(\textit{questioned})) \\
& \times p(\mathbf{NP}(\textit{lawyer}) \rightarrow_2 \mathbf{D}(\textit{the}) \;\; \mathbf{N}(\textit{lawyer})) \\
& \times p(\mathbf{D}(\textit{the}) \rightarrow \textit{the}) \times p(\mathbf{N}(\textit{lawyer}) \rightarrow \textit{lawyer}) \\
& \times p(\mathbf{VP}(\textit{questioned}) \rightarrow_1 \mathbf{V}(\textit{questioned}) \;\; \mathbf{NP}(\textit{witness})) \\
& \times p(\mathbf{V}(\textit{questioned}) \rightarrow \textit{questioned}) \\
& \times p(\mathbf{NP}(\textit{witness}) \rightarrow_2 \mathbf{D}(\textit{the}) \;\; \mathbf{N}(\textit{witness})) \\
& \times p(\mathbf{D}(\textit{the}) \rightarrow \textit{the}) \times p(\mathbf{N}(\textit{witness}) \rightarrow \textit{witness})
\end{aligned}
$$

Learning of the parameters can be done similar to that of conventional PCFG. Some smoothing may be helpful.

Question

When you say a lexicalized parser can be more effective, what do you mean by "effective"?

Need a benchmark dataset and evaluation metrics!

Penn Treebank
(Marcus et al, 1993)

Building a Large Annotated Corpus of English: The Penn Treebank

Mitchell P. Marcus[*]
University of Pennsylvania

Beatrice Santorini[†]
Northwestern University

Mary Ann Marcinkiewicz[‡]
University of Pennsylvania

1. Introduction

There is a growing consensus that significant, rapid progress can be made in both text understanding and spoken language understanding by investigating those phenomena that occur most centrally in naturally occurring unconstrained materials and by attempting to automatically extract information about language from very large corpora. Such corpora are beginning to serve as important research tools for investigators in natural language processing, speech recognition, and integrated spoken language systems, as well as in theoretical linguistics. Annotated corpora promise to be valuable for enterprises as diverse as the automatic construction of statistical models for the grammar of the written and the colloquial spoken language, the development of explicit formal theories of the differing grammars of writing and speech, the investigation of prosodic phenomena in speech, and the evaluation and comparison of the adequacy of parsing models.

In this paper, we review our experience with constructing one such large annotated corpus—the Penn Treebank, a corpus[1] consisting of over 4.5 million words of American English. During the first three-year phase of the Penn Treebank Project (1989–1992), this corpus has been annotated for part-of-speech (POS) information. In addition, over half of it has been annotated for skeletal syntactic structure. These materials are available to members of the Linguistic Data Consortium; for details, see Section 5.1.

The standard benchmark used for parsing. A remarkable contribution to NLP.

Penn Treebank

Student of Marcus

1. CC	Coordinating conjunction	25. TO	*to*
2. CD	Cardinal number	26. UH	Interjection
3. DT	Determiner	27. VB	Verb, base form
4. EX	Existential *there*	28. VBD	Verb, past tense
5. FW	Foreign word	29. VBG	Verb, gerund/present participle
6. IN	Preposition/subordinating conjunction	30. VBN	Verb, past participle
7. JJ	Adjective	31. VBP	Verb, non-3rd ps. sing. present
8. JJR	Adjective, comparative	32. VBZ	Verb, 3rd ps. sing. present
9. JJS	Adjective, superlative	33. WDT	*wh*-determiner
10. LS	List item marker	34. WP	*wh*-pronoun
11. MD	Modal	35. WP$	Possessive *wh*-pronoun
12. NN	Noun, singular or mass	36. WRB	*wh*-adverb
13. NNS	Noun, plural	37. #	Pound sign
14. NNP	Proper noun, singular	38. $	Dollar sign
15. NNPS	Proper noun, plural	39. .	Sentence-final punctuation
16. PDT	Predeterminer	40. ,	Comma
17. POS	Possessive ending	41. :	Colon, semi-colon
18. PRP	Personal pronoun	42. (	Left bracket character
19. PP$	Possessive pronoun	43.)	Right bracket character
20. RB	Adverb	44. "	Straight double quote
21. RBR	Adverb, comparative	45. ‘	Left open single quote
22. RBS	Adverb, superlative	46. “	Left open double quote
23. RP	Particle	47. ’	Right close single quote
24. SYM	Symbol (mathematical or scientific)	48. ”	Right close double quote

The POS Tagset

Battle-tested/NNP*/JJ industrial/JJ managers/NNS here/RB always/RB buck/VB*/VBP up/IN*/RP nervous/JJ newcomers/NNS with/IN the/DT tale/NN of/IN the/DT first/JJ of/IN their/PP$ countrymen/NNS to/TO visit/VB Mexico/NNP ,/, a/DT boatload/NN of/IN samurai/NNS*/FW warriors/NNS blown/VBN ashore/RB 375/CD years/NNS ago/RB ./.

“/“ From/IN the/DT beginning/NN ,/, it/PRP took/VBD a/DT man/NN with/IN extraordinary/JJ qualities/NNS to/TO succeed/VB in/IN Mexico/NNP ,/, ”/” says/VBZ Kimihide/NNP Takimura/NNP ,/, president/NN of/IN Mitsui/NNS*/NNP group/NN ’s/POS Kensetsu/NNP Engineering/NNP Inc./NNP unit/NN ./.

Sample POS-Tagged Text

Penn Treebank

Syntactic Tagset

1.	ADJP	Adjective phrase
2.	ADVP	Adverb phrase
3.	NP	Noun phrase
4.	PP	Prepositional phrase
5.	S	Simple declarative clause
6.	SBAR	Clause introduced by subordinating conjunction or *0* (see below)
7.	SBARQ	Direct question introduced by *wh*-word or *wh*-phrase
8.	SINV	Declarative sentence with subject-aux inversion
9.	SQ	Subconstituent of SBARQ excluding *wh*-word or *wh*-phrase
10.	VP	Verb phrase
11.	WHADVP	*wh*-adverb phrase
12.	WHNP	*wh*-noun phrase
13.	WHPP	*wh*-prepositional phrase
14.	X	Constituent of unknown or uncertain category

Null elements

1.	*	"Understood" subject of infinitive or imperative
2.	0	Zero variant of *that* in subordinate clauses
3.	T	Trace—marks position where moved *wh*-constituent is interpreted
4.	NIL	Marks position where preposition is interpreted in pied-piping contexts

Student of Marcus

Example Parse Tree

```
( (S
   (NP Battle-tested industrial managers
       here)
    always
   (VP buck
       up
       (NP nervous newcomers)
       (PP with
           (NP the tale
               (PP of
                   (NP (NP the
                           (ADJP first
                                 (PP of
                                     (NP their countrymen)))
                           (S (NP *)
                              to
                              (VP visit
                                  (NP Mexico))))
                       ,
                       (NP (NP a boatload
                               (PP of
                                   (NP (NP warriors)
                                       (VP-1 blown
                                             ashore
                                             (ADVP (NP 375 years)
                                                   ago)))))
                           (VP-1 *pseudo-attach*))))))))
   .)
```

Penn Treebank

Mitchell P. Marcus et al. — Building a Large Annotated Corpus of English

Table 4
Penn Treebank (as of 11/92).

Description	Tagged for Part-of-Speech (Tokens)	Skeletal Parsing (Tokens)
Dept. of Energy abstracts	231,404	231,404
Dow Jones Newswire stories	3,065,776	1,061,166
Dept. of Agriculture bulletins	78,555	78,555
Library of America texts	105,652	105,652
MUC-3 messages	111,828	111,828
IBM Manual sentences	89,121	89,121
WBUR radio transcripts	11,589	11,589
ATIS sentences	19,832	19,832
Brown Corpus, retagged	1,172,041	1,172,041
Total:	4,885,798	2,881,188

Statistics for the initial version released in 1992.

Penn Treebank version 3.0 was released in 1999.

Evaluation Metrics

How shall we understand the amount of mistakes a parser makes?

Remember what we did in the ML project?

Evaluation Metrics

Recall

The percentage of desired predictions recovered

$$\mathrm{R} = \frac{\#\ \text{of correct constituents in the predicted parse tree}}{\#\ \text{of correct constituents in the gold parse tree}}$$

A constituent is regarded as correct if and only if its non-terminal, starting point and ending point are exactly the same as those of the gold constituent.

Evaluation Metrics

Precision

The percentage of predictions that are good

$$\mathrm{P} = \frac{\#\ \text{of correct constituents in the predicted parse tree}}{\#\ \text{of total constituents in the predicted parse tree}}$$

Evaluation Metrics

F-measure

A metric that cares about both Precision and Recall

$$\mathrm{F}_\beta = \frac{(\beta^2+1)PR}{\beta^2 P+R}$$

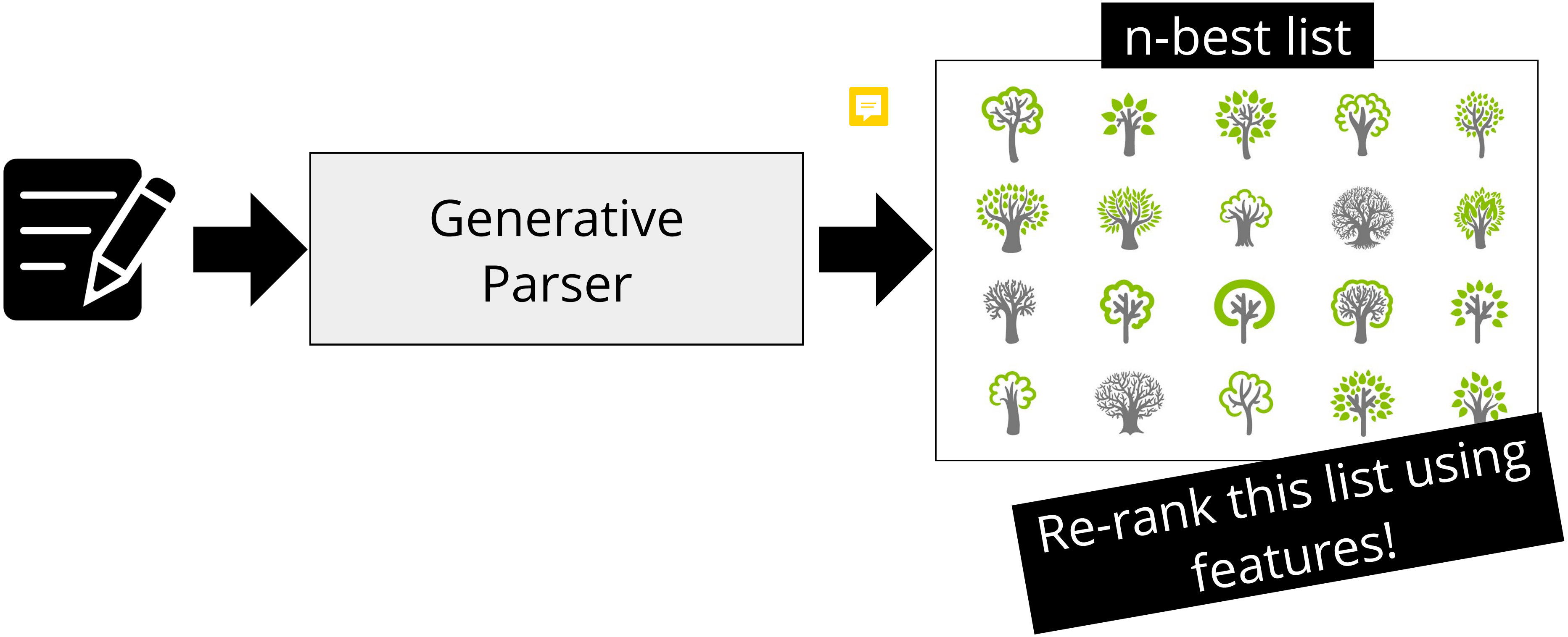
$$\mathbf{F_1} = \frac{2}{\frac{1}{\mathrm{P}}+\frac{1}{\mathrm{R}}}$$

F_1 measure is commonly used

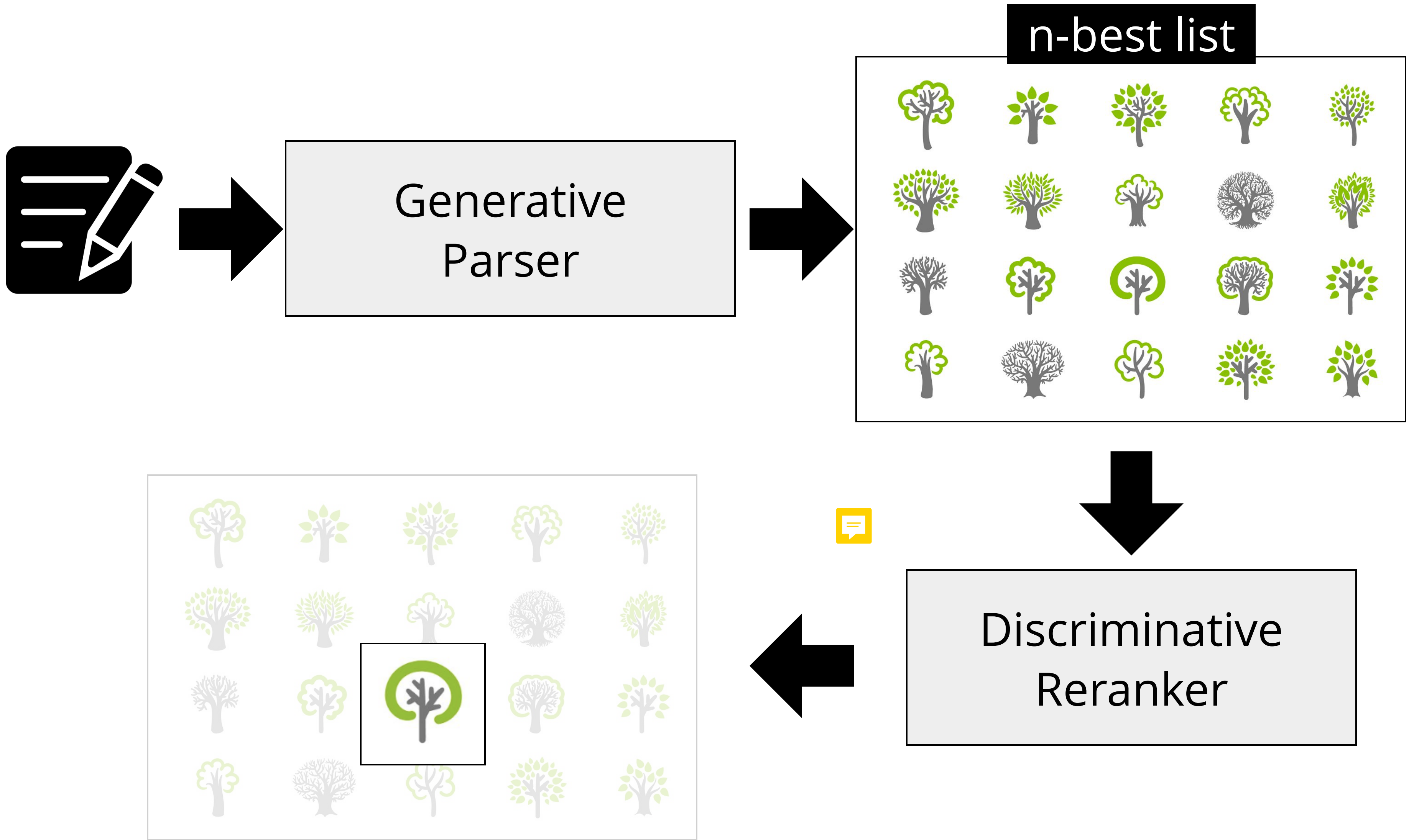
Question

Is it possible to adopt a discriminative approach to parsing?

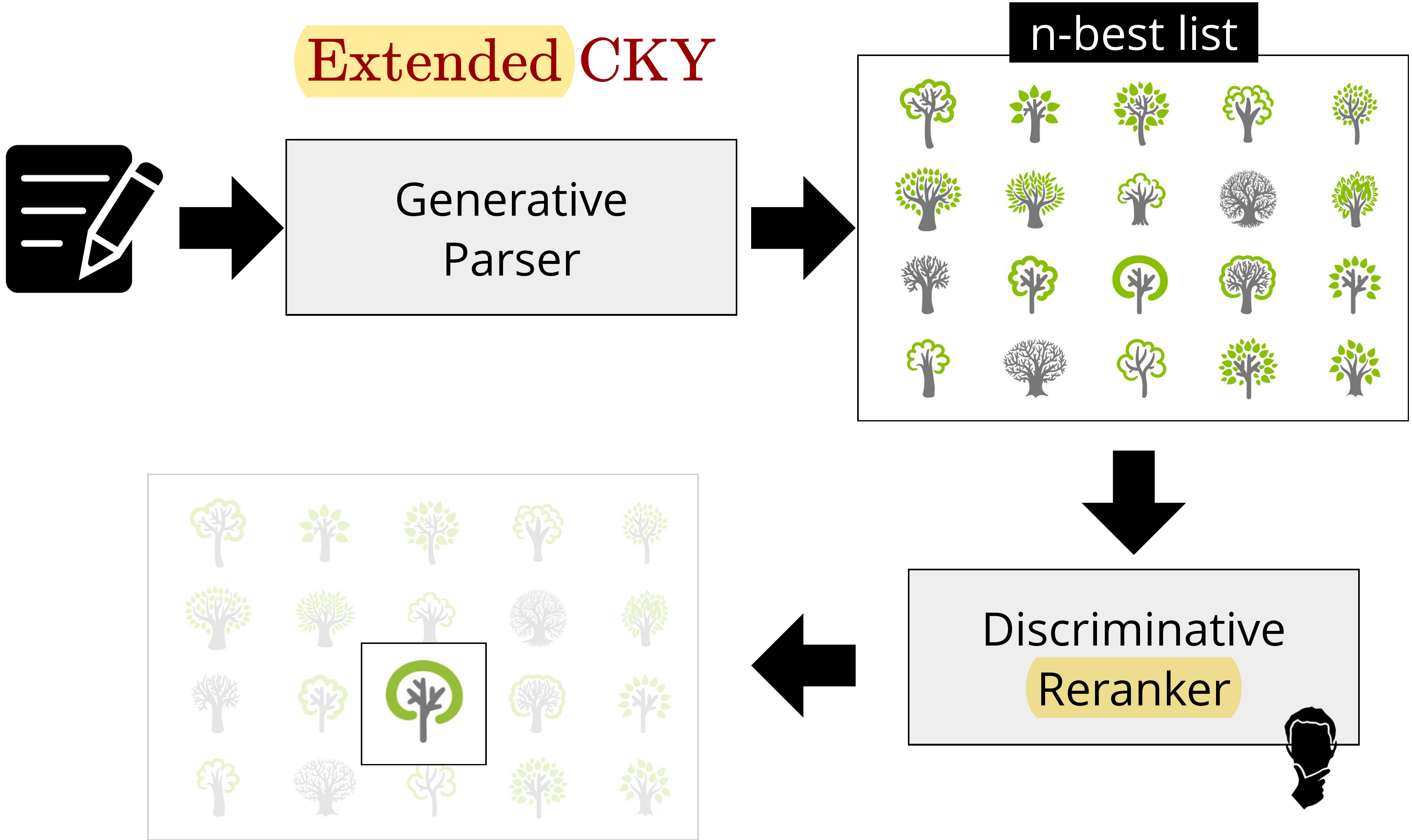
Discriminative Parsing



Discriminative Parsing



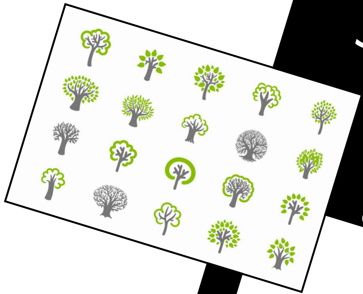
Discriminative Parsing



Discriminative Parsing

reranking n-best list

The best candidate tree



Some candidate trees produced by the generative model

$$\hat{\boldsymbol{y}}_i \leftarrow \arg\max_{\boldsymbol{y} \in \mathbf{GEN}(\boldsymbol{x}_i)} \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}) \cdot \boldsymbol{\theta}$$

$$\text{if } \hat{\boldsymbol{y}}_i \neq \boldsymbol{y}_i \text{ then}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \mathbf{f}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$$

Discriminative Parsing
reranking n-best list

1. Learn a generative model.
2. Use an extended CKY algorithm to produce the n-best trees based on the generative model for each input sentence $\boldsymbol{x}_i$.
This n-best list is $\mathbf{GEN}(\boldsymbol{x}_i)$.
3. Use the structured perceptron algorithm to re-rank the list based on features defined over the parse trees.

You may define some non-local, arbitrary features here!

Discriminative Parsing

reranking n-best list

1. Learn a generative model.
2. Use an extended CKY algorithm to produce the n-best trees based on the generative model for each input sentence $\boldsymbol{x}_i$.
This n-best list is $\mathbf{GEN}(\boldsymbol{x}_i)$.
3. Use the structured perceptron algorithm to re-rank the list based on features defined over the parse trees.

It seems this approach depends on the n-best list. What if the desired/gold parse tree does not appear in the n-best list?

Discriminative Parsing

with all possible trees

Can we search in the complete structured space efficiently?

$$\hat{\boldsymbol{y}}_i \leftarrow \arg\max_{\boldsymbol{y} \in \mathbf{GEN}(\boldsymbol{x}_i)} \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}) \cdot \boldsymbol{\theta}$$

$$\text{if } \hat{\boldsymbol{y}}_i \neq \boldsymbol{y}_i \text{ then}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \mathbf{f}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$$

Discriminative Parsing
with all possible trees

Yes! You may just design "local features" here, similar to what we did for sequence modeling. Then you may use CKY algorithm to find the most probable parse.

$$\hat{\boldsymbol{y}}_i \leftarrow \arg\max_{\boldsymbol{y} \in \mathbf{GEN}(\boldsymbol{x}_i)} \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}) \cdot \boldsymbol{\theta}$$

$$\text{if } \hat{\boldsymbol{y}}_i \neq \boldsymbol{y}_i \text{ then}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \mathbf{f}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$$

Discriminative Parsing
with all possible trees

Since you can do so using structured perceptron, can we also use the CRF based approach?

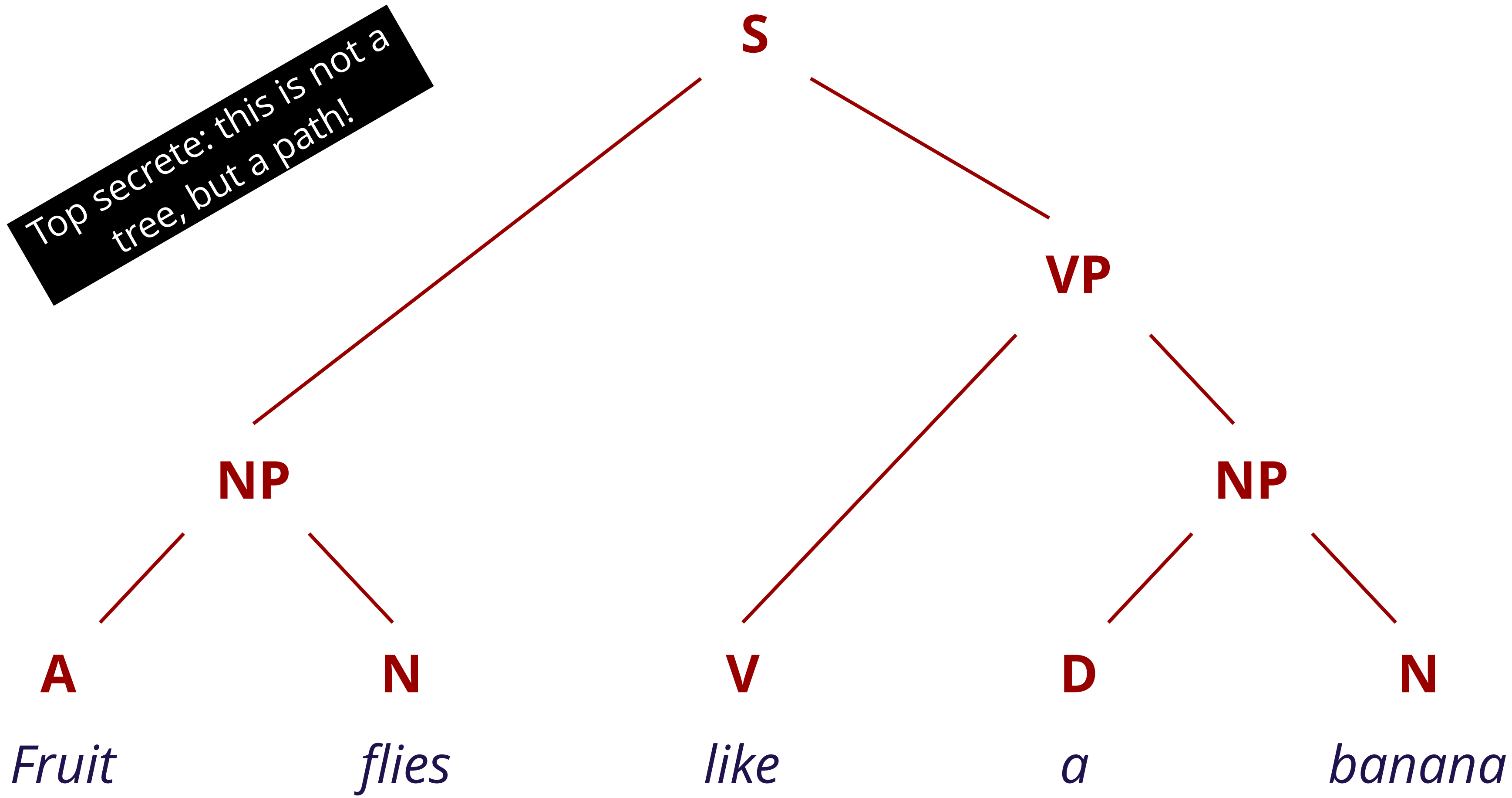
$$\hat{\boldsymbol{y}}_i \leftarrow \arg\max_{\boldsymbol{y} \in \mathbf{GEN}(\boldsymbol{x}_i)} \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}) \cdot \boldsymbol{\theta}$$

$$\text{if } \hat{\boldsymbol{y}}_i \neq \boldsymbol{y}_i \text{ then}$$

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}_i) - \mathbf{f}(\boldsymbol{x}_i, \hat{\boldsymbol{y}}_i)$$

Optional

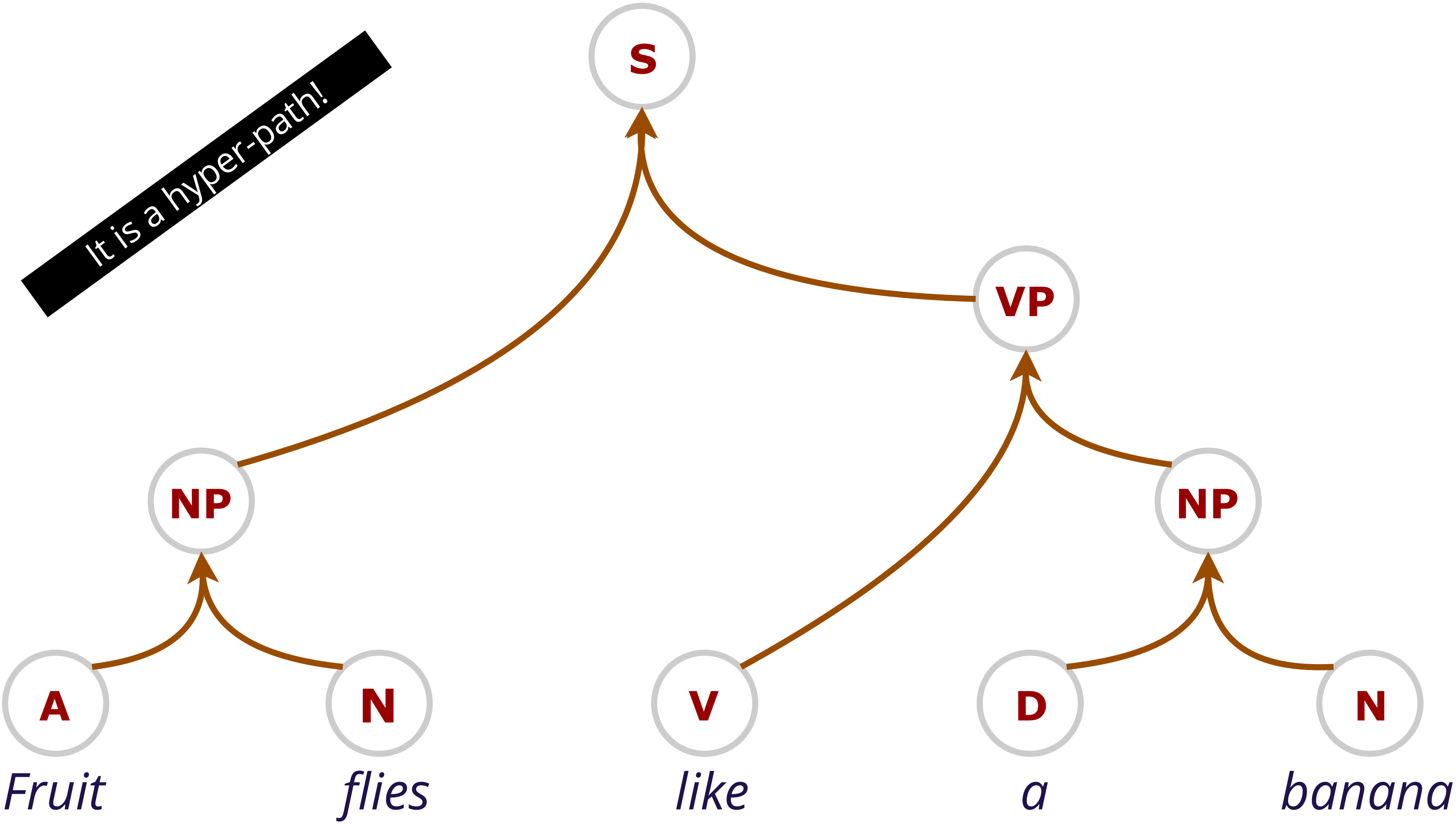
Parsing CRF



Optional

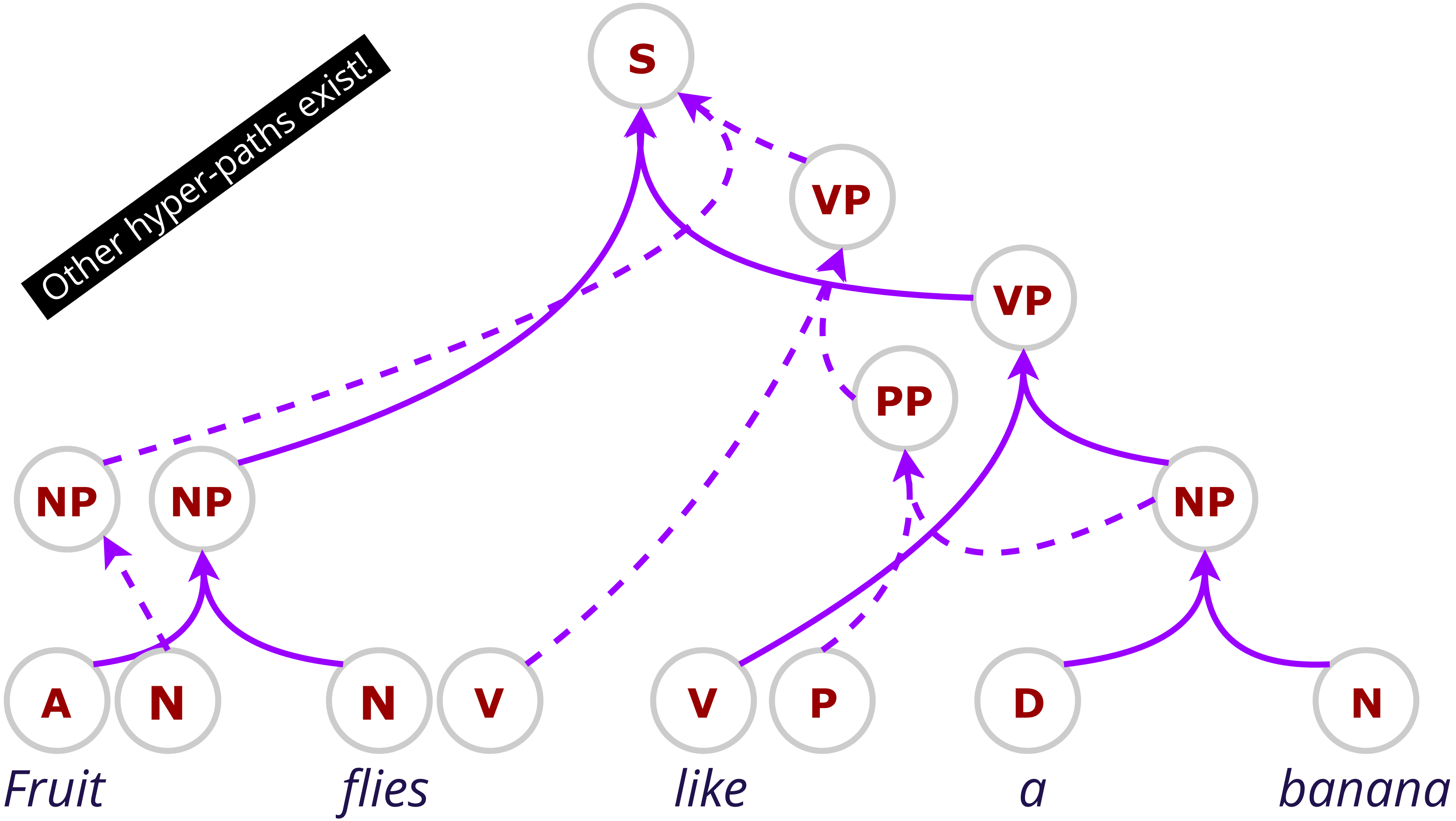
Parsing CRF

It is a hyper-path!



Optional

Parsing CRF


Other hyper-paths exist!
S
VP
VP
PP
NP
NP
NP
A
N
N
V
V
P
D
N
Fruit
flies
like
a
banana

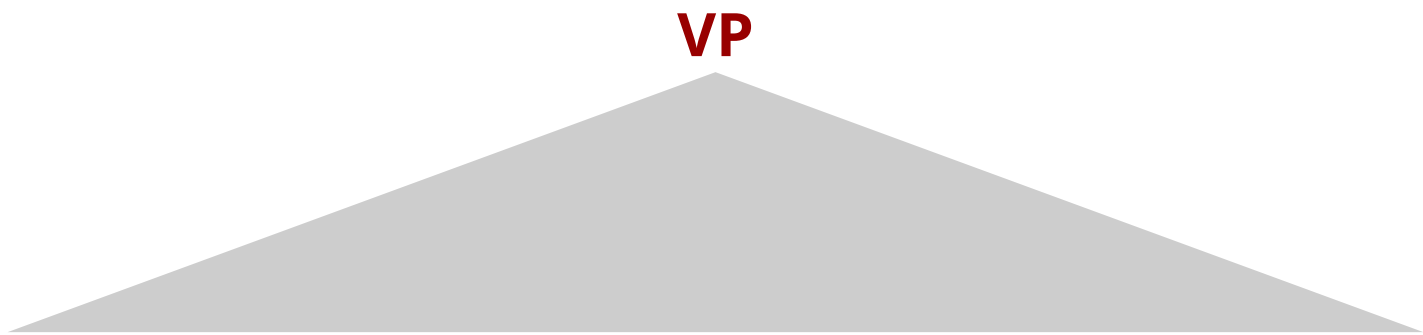
Optional

Inside Algorithm

$$\begin{array}{llll} \mathbf{VP} & \rightarrow & \mathbf{V} \quad \mathbf{NP} & (0.29) \\ \mathbf{VP} & \rightarrow & \mathbf{V} \quad \mathbf{VP} & (0.12) \\ \mathbf{VP} & \rightarrow & \mathbf{V} \quad \mathbf{PP} & (0.08) \end{array}$$

$$\begin{aligned} bestScore[1,5,\mathbf{VP}] = \max\{ & \max_{1\leq k\leq 4}(0.29 \times bestScore[1,k,\mathbf{V}] \times bestScore[k,5,\mathbf{NP}]), \\ & \max_{1\leq k\leq 4}(0.12 \times bestScore[1,k,\mathbf{V}] \times bestScore[k,5,\mathbf{VP}]), \\ & \max_{1\leq k\leq 4}(0.08 \times bestScore[1,k,\mathbf{V}] \times bestScore[k,5,\mathbf{PP}])\} \end{aligned}$$

$$\begin{aligned} totalScore[1,5,\mathbf{VP}] = & \sum_{1\leq k\leq 4}(0.29 \times totalScore[1,k,\mathbf{V}] \times totalScore[k,5,\mathbf{NP}]) \\ & + \sum_{1\leq k\leq 4}(0.12 \times totalScore[1,k,\mathbf{V}] \times totalScore[k,5,\mathbf{VP}]) \\ & + \sum_{1\leq k\leq 4}(0.08 \times totalScore[1,k,\mathbf{V}] \times totalScore[k,5,\mathbf{PP}]) \end{aligned}$$



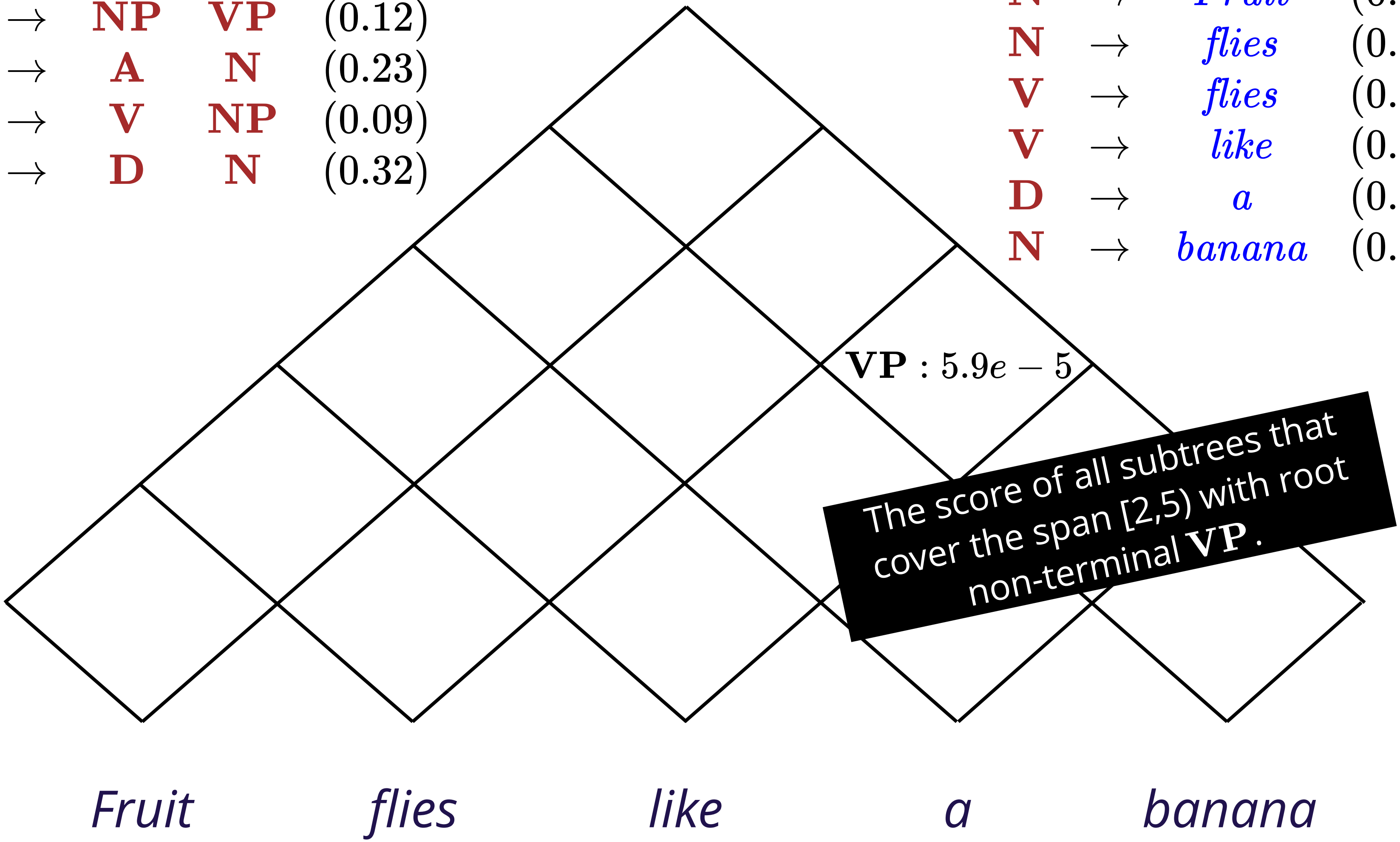
Fruit *flies* *like* *a* *banana*

Optional

Inside Algorithm

$$
\begin{array}{llll l}
\mathbf{S} & \rightarrow & \mathbf{NP} & \mathbf{VP} & (0.12) \\
\mathbf{NP} & \rightarrow & \mathbf{A} & \mathbf{N} & (0.23) \\
\mathbf{VP} & \rightarrow & \mathbf{V} & \mathbf{NP} & (0.09) \\
\mathbf{NP} & \rightarrow & \mathbf{D} & \mathbf{N} & (0.32)
\end{array}
$$

$$
\begin{array}{lll l}
\mathbf{A} & \rightarrow & \mathit{Fruit} & (0.221) \\
\mathbf{N} & \rightarrow & \mathit{Fruit} & (0.711) \\
\mathbf{N} & \rightarrow & \mathit{flies} & (0.112) \\
\mathbf{V} & \rightarrow & \mathit{flies} & (0.291) \\
\mathbf{V} & \rightarrow & \mathit{like} & (0.108) \\
\mathbf{D} & \rightarrow & \mathit{a} & (0.209) \\
\mathbf{N} & \rightarrow & \mathit{banana} & (0.089)
\end{array}
$$



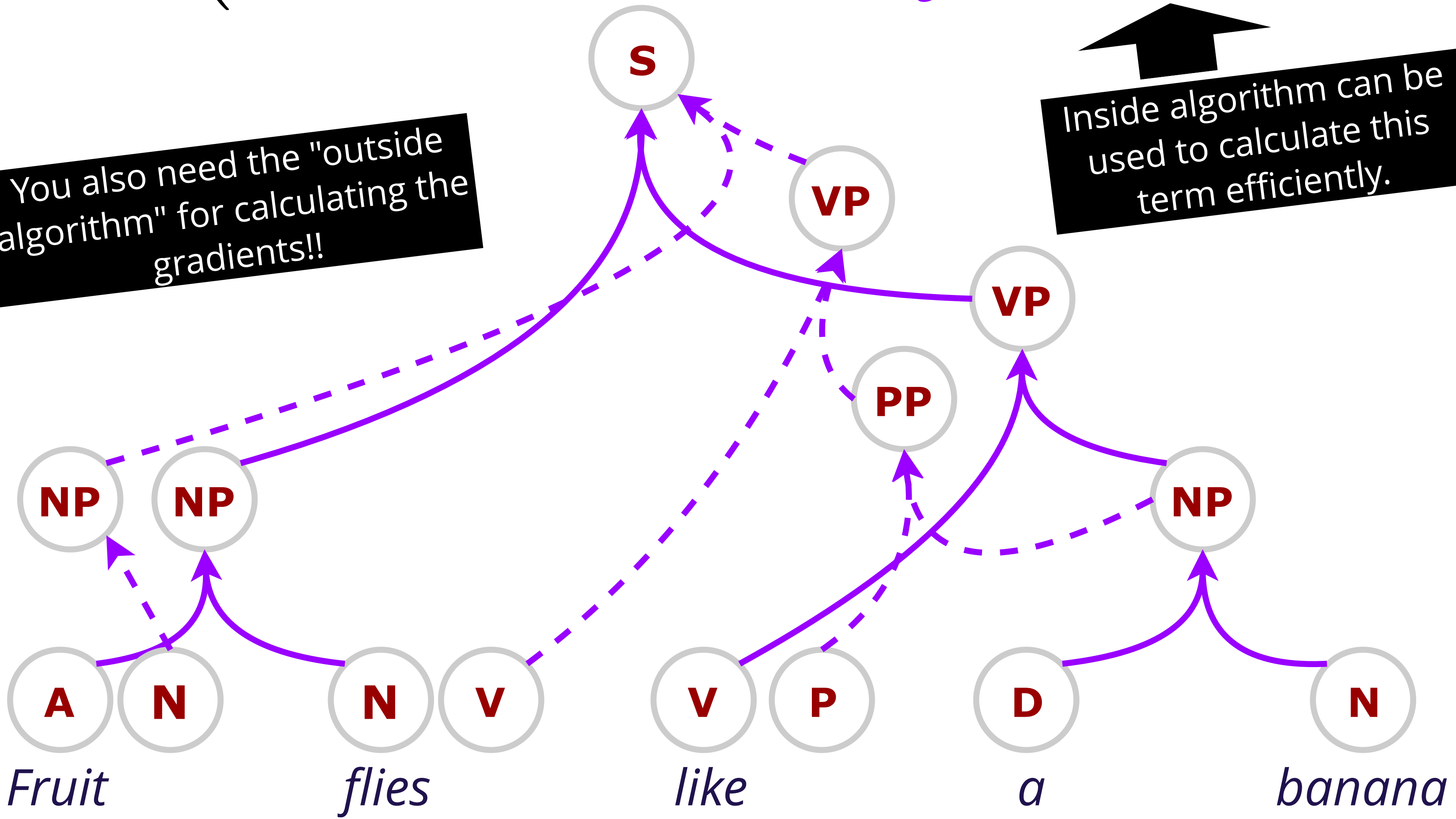
Optional

Inside Algorithm

$$\min_{\boldsymbol{\theta}} \sum_i \left(-\mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}_i) \cdot \boldsymbol{\theta} + \log \sum_{\boldsymbol{y}} \exp\left(\mathbf{f}(\boldsymbol{x}_i, \boldsymbol{y}) \cdot \boldsymbol{\theta}\right)\right)$$

Inside algorithm can be used to calculate this term efficiently.

You also need the "outside algorithm" for calculating the gradients!!

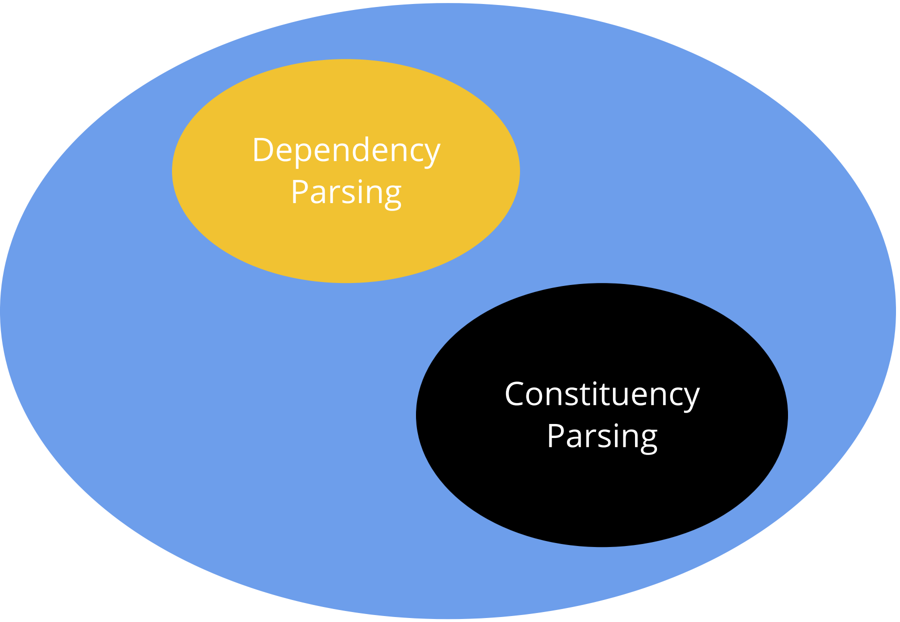


Discriminative Parsing

Summary

Approaches	Pros	Cons
Discriminative Reranking	Easy to implement, able to exploit arbitrary, global features	There is no guarantee the desired output is within n-best list
Global Discriminative Models (Structured Perceptron, Parsing CRF ...)	Able to consider all possible structures, without relying on n-best list from the generative model	Must design local features so as to perform dynamic programming

Syntactic Parsing



Syntactic Parsing

Techniques

Probabilistic (lexicalized) CFG

~~Adapter Grammars~~

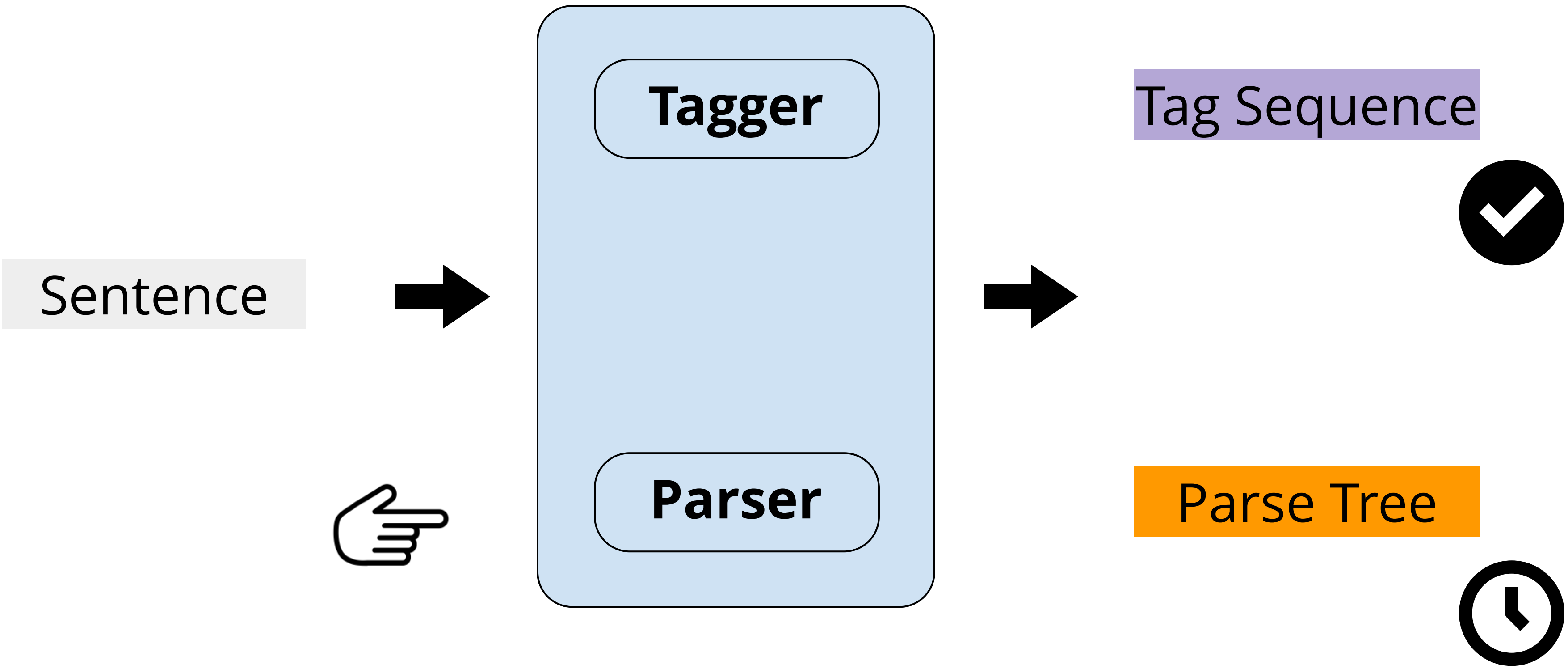
Discriminative Rerankers

~~Parsing CRF~~

~~Max-margin~~/Perceptron Parser

Shift-Reduce/Transition-based Parser

Structured Prediction



Structured Space $\mathcal{S}_i$ Structured Prediction Model M Structured Space $\mathcal{S}_o$

$\mathsf{M} : \mathcal{S}_i \rightarrow \mathcal{S}_o$