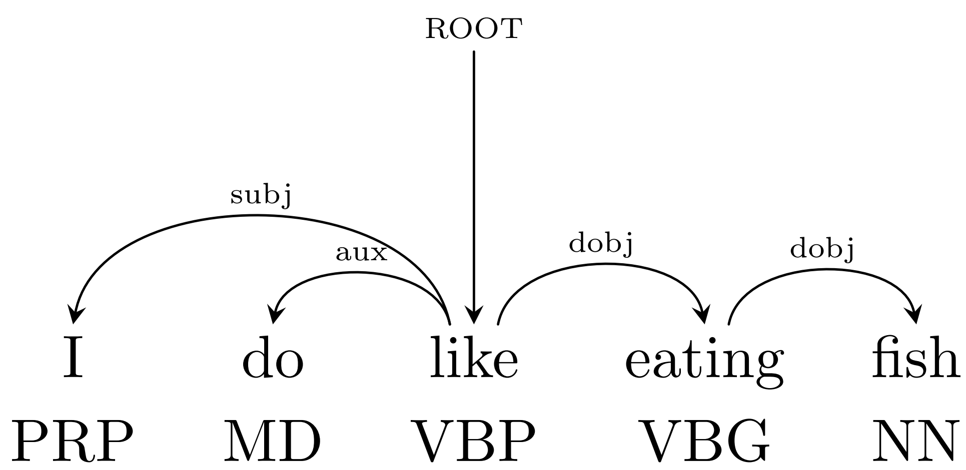
Span-Based Constituency Parsing with Provably Optimal Dynamic Oracles

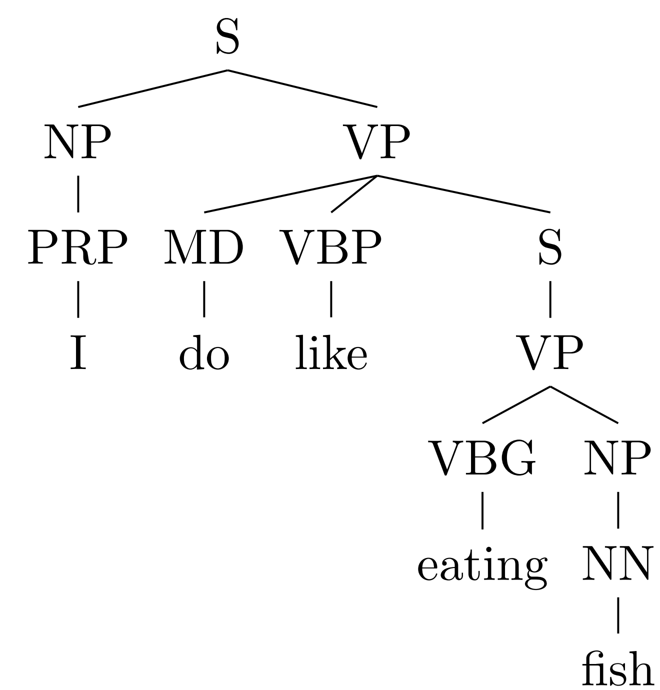
James Cross and Liang Huang
Oregon State University

EMNLP, Austin, TX
November 2, 2016

Oregon State UNIVERSITY

Dependency vs. Constituency



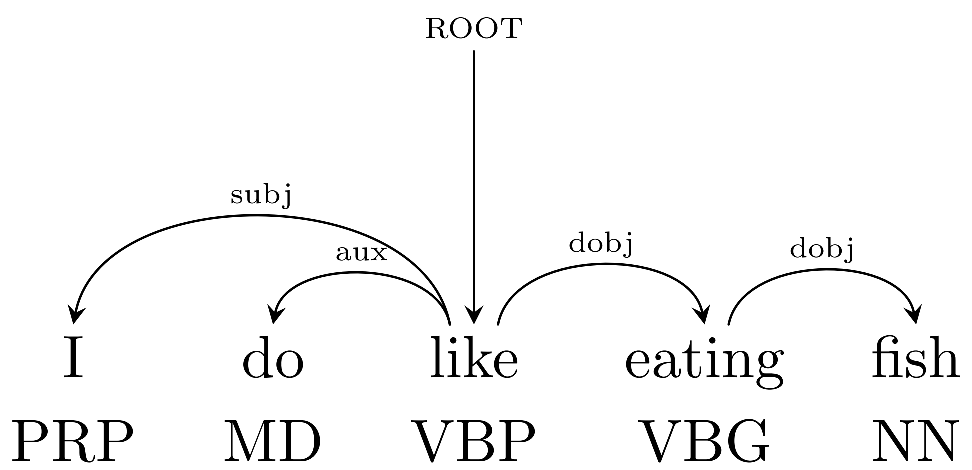


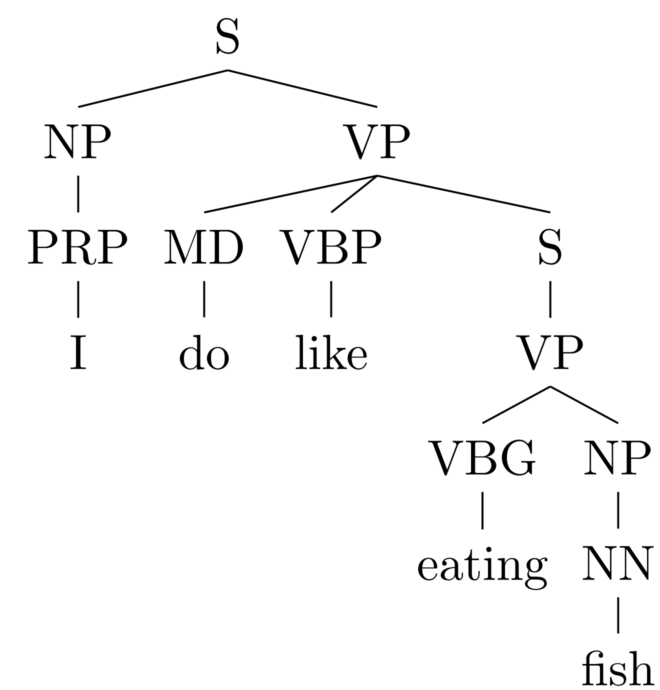
	search	UAS
Zhang & Nivre 2011	beam	92.9
Chen & Manning 2014	**greedy**	91.8
Zhou et al. (2015)	beam	93.3
Weiss et al. (2015)	beam	94.0
our work (ACL 2016)	**greedy**	93.4
Andor et al. (2016)	beam	**94.4**

	search	F_1
Carreras et al. (2008)	cubic	91.1
Shindo et al. (2012)	cubic	91.1
Thang et al. (2015) (A*)	~cubic	91.1
Watanabe et al. (2015)	beam	90.7
Vinyals et al. (2015) (WSJ)	beam	90.5

Red = Neural

Dependency vs. Constituency





	search	UAS
Zhang & Nivre 2011	beam	92.9
Chen & Manning 2014	**greedy**	91.8
Zhou et al. (2015)	beam	93.3
Weiss et al. (2015)	beam	94.0
our work (ACL 2016)	**greedy**	93.4
Andor et al. (2016)	beam	**94.4**

	search	F_1
Carreras et al. (2008)	cubic	91.1
Shindo et al. (2012)	cubic	91.1
Thang et al. (2015) (A*)	~cubic	91.1
Watanabe et al. (2015)	beam	90.7
Vinyals et al. (2015) (WSJ)	beam	90.5
This Work	**greedy**	**91.3**

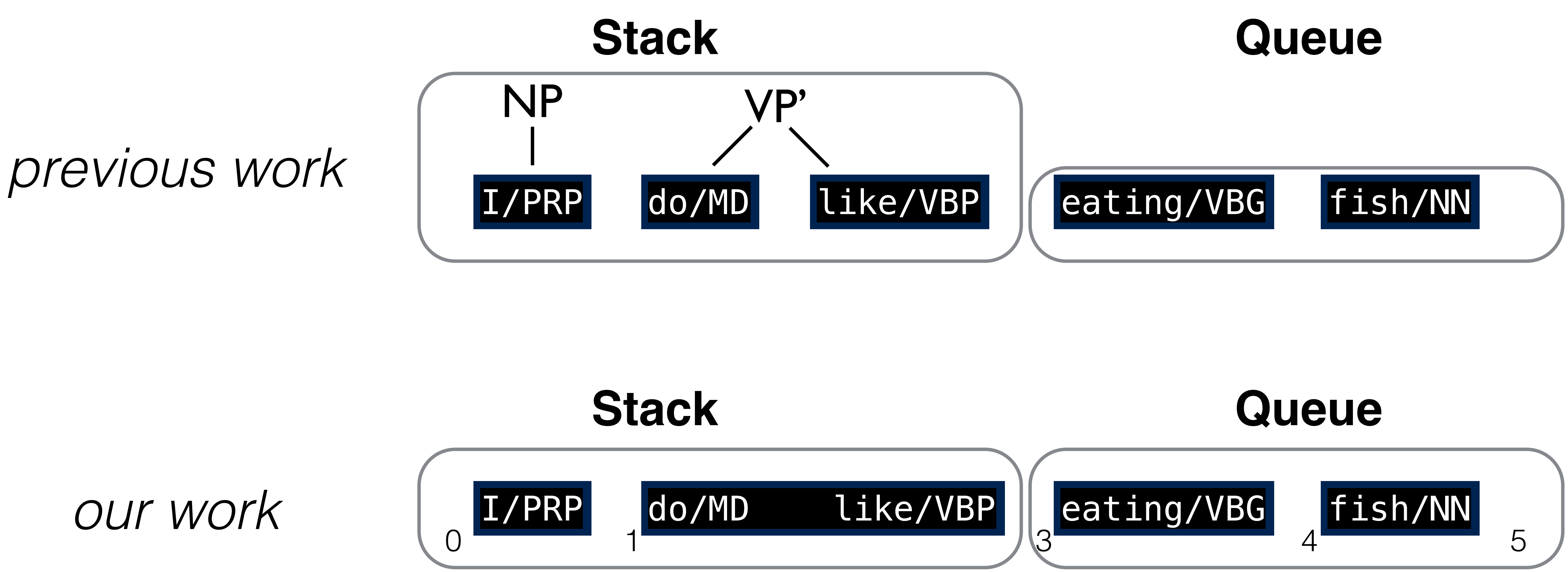
Red = Neural

Outline

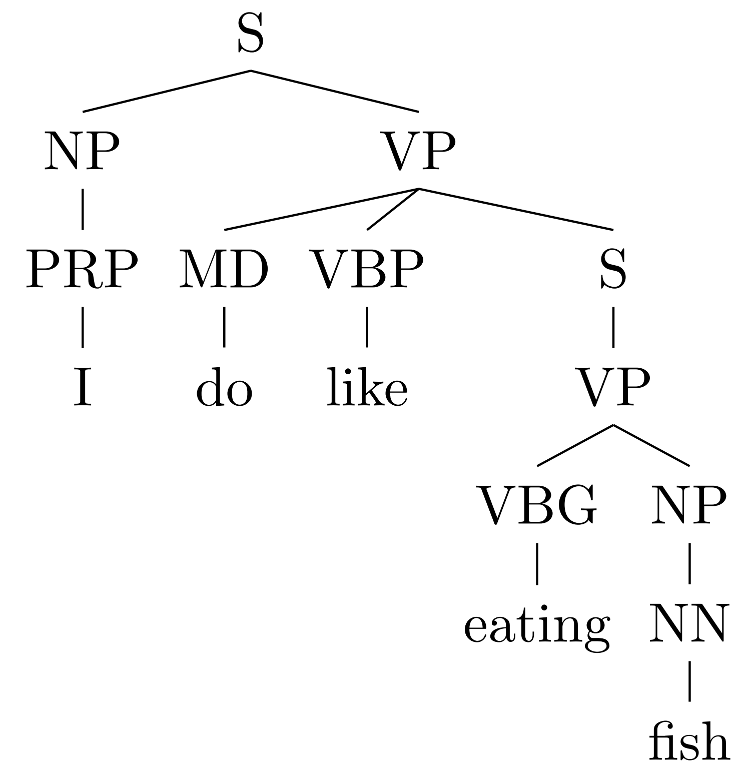
- Span-Based Constituency Parsing
- Bi-Directional LSTM Span Features
- Provably Optimal Dynamic Oracle
- Experiments

Span-Based Parsing

- Previous work uses tree structures on stack
- We simplify to operate directly on sentence spans
- Simple-to-implement linear-time parsing

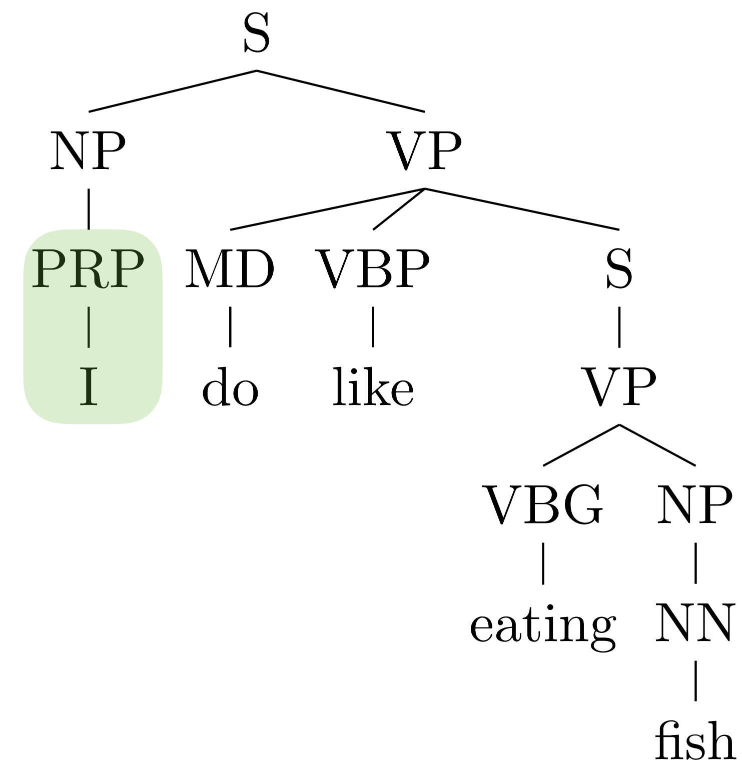


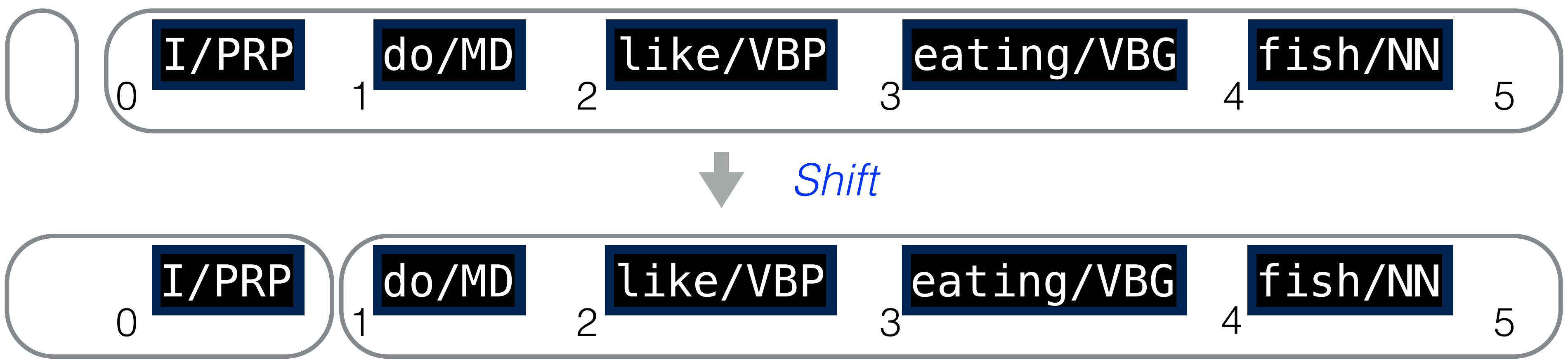
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



current brackets t = {}

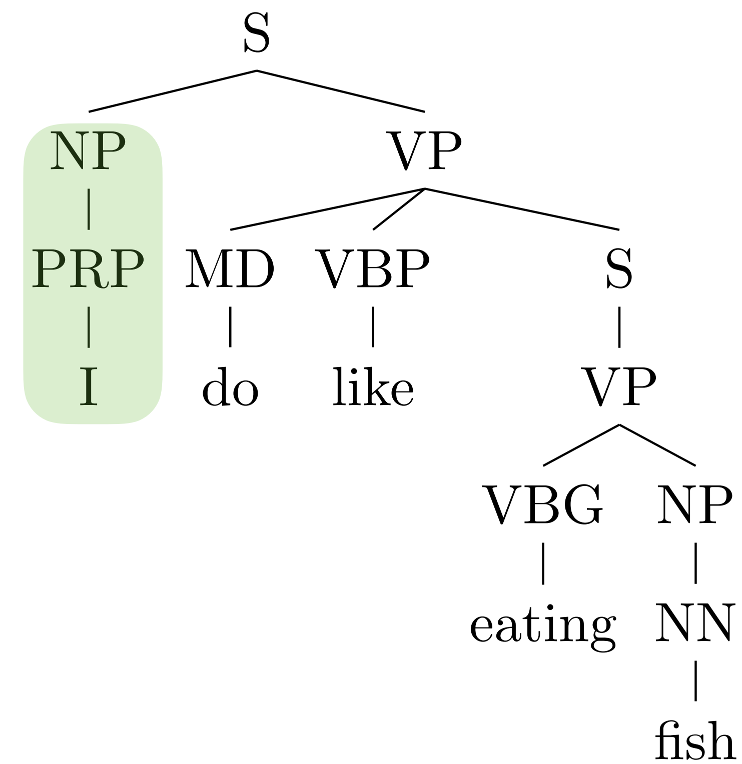
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label

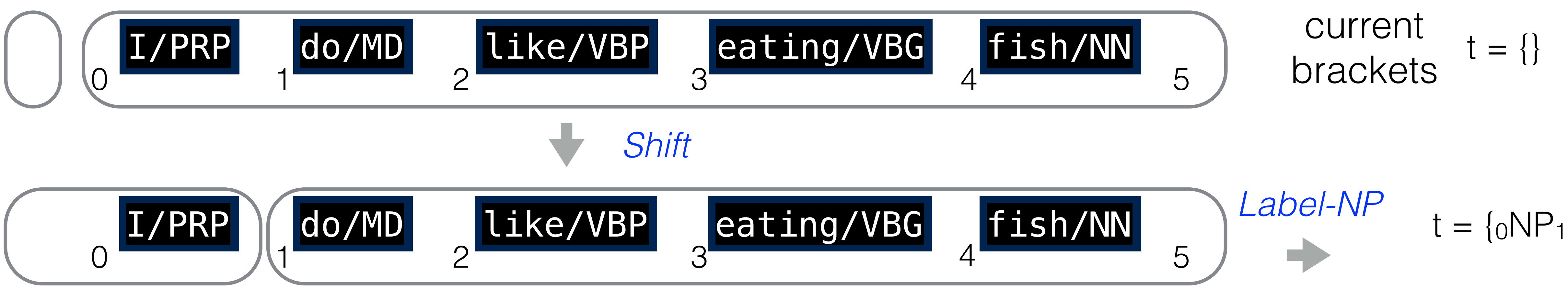




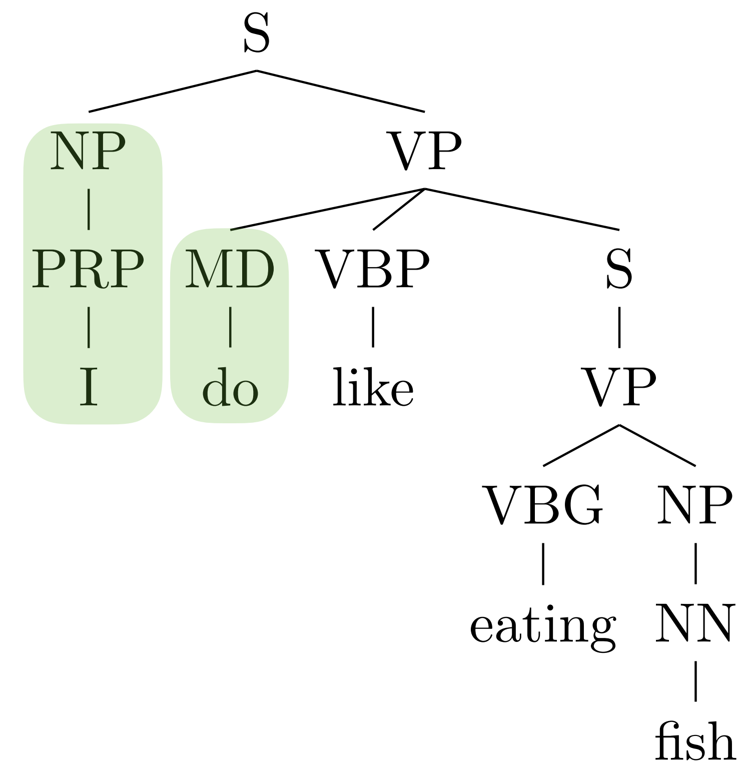
current brackets t = {}

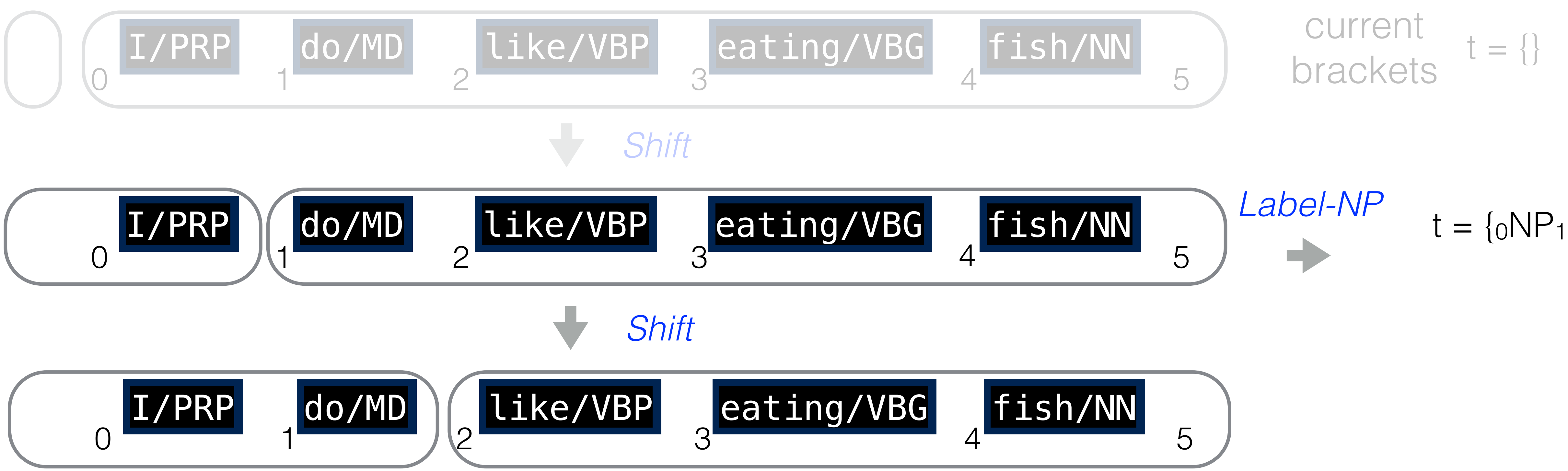
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



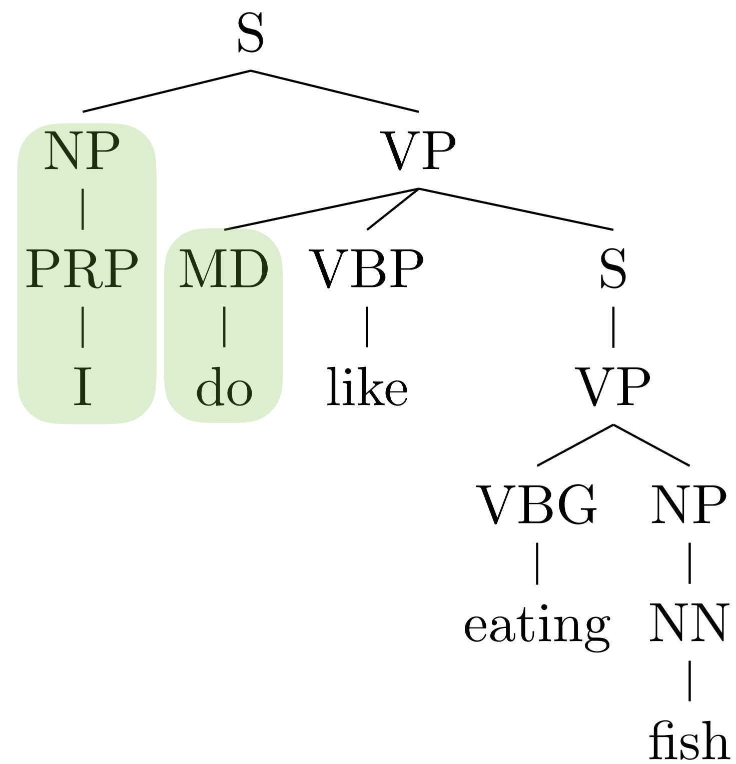


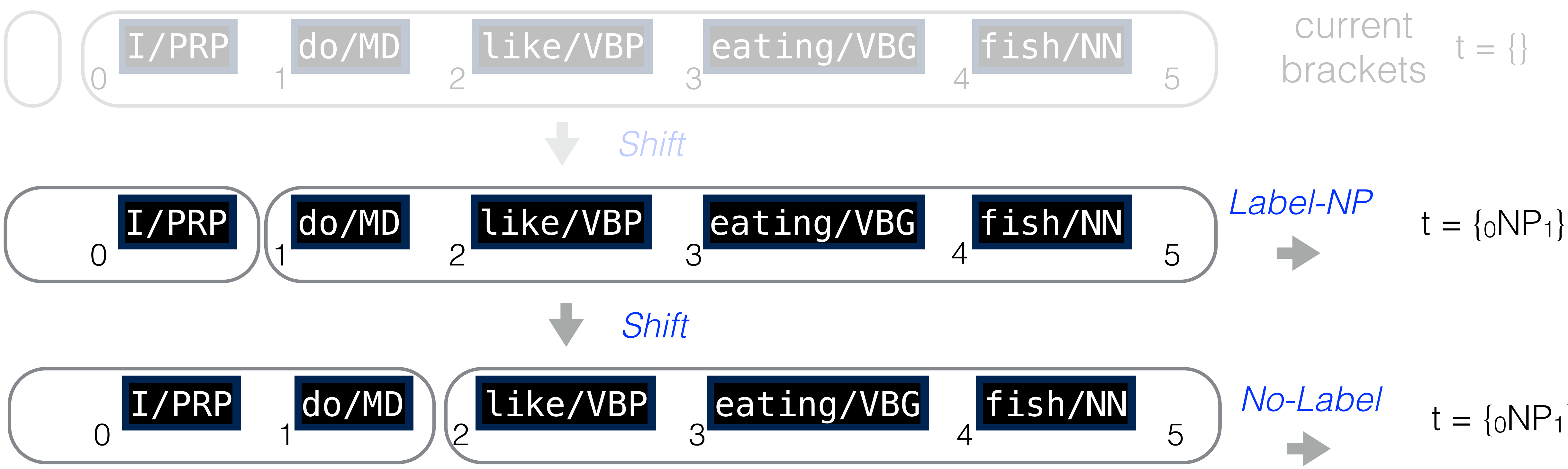
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



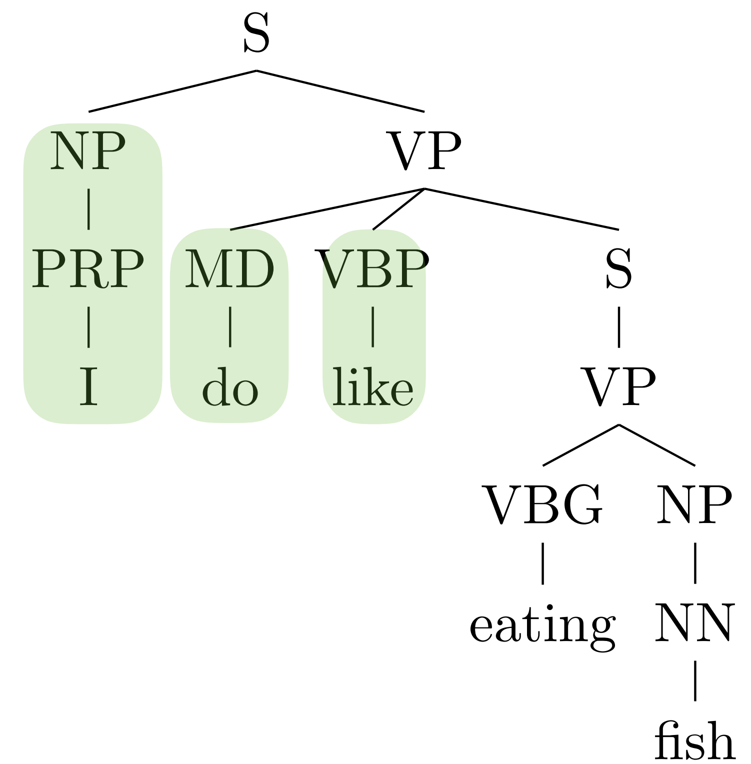


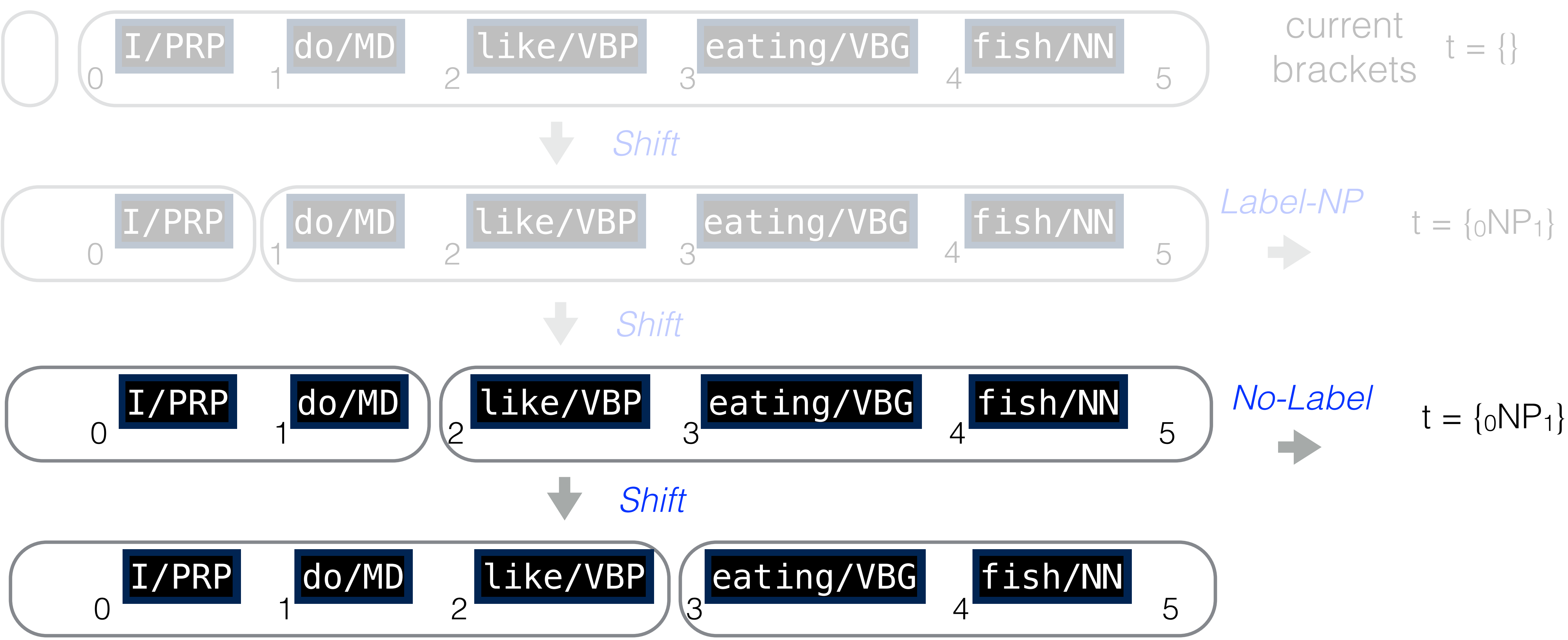
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



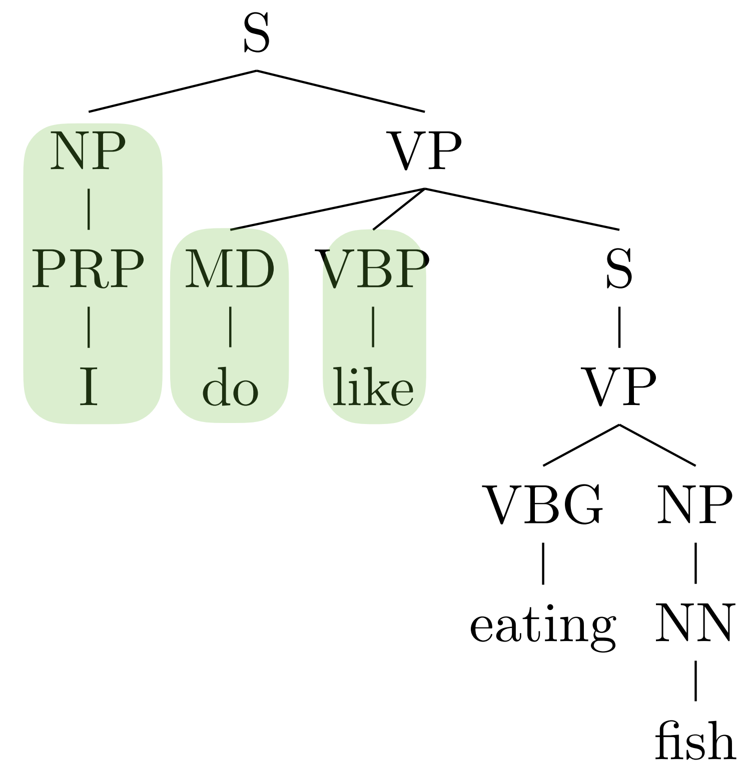


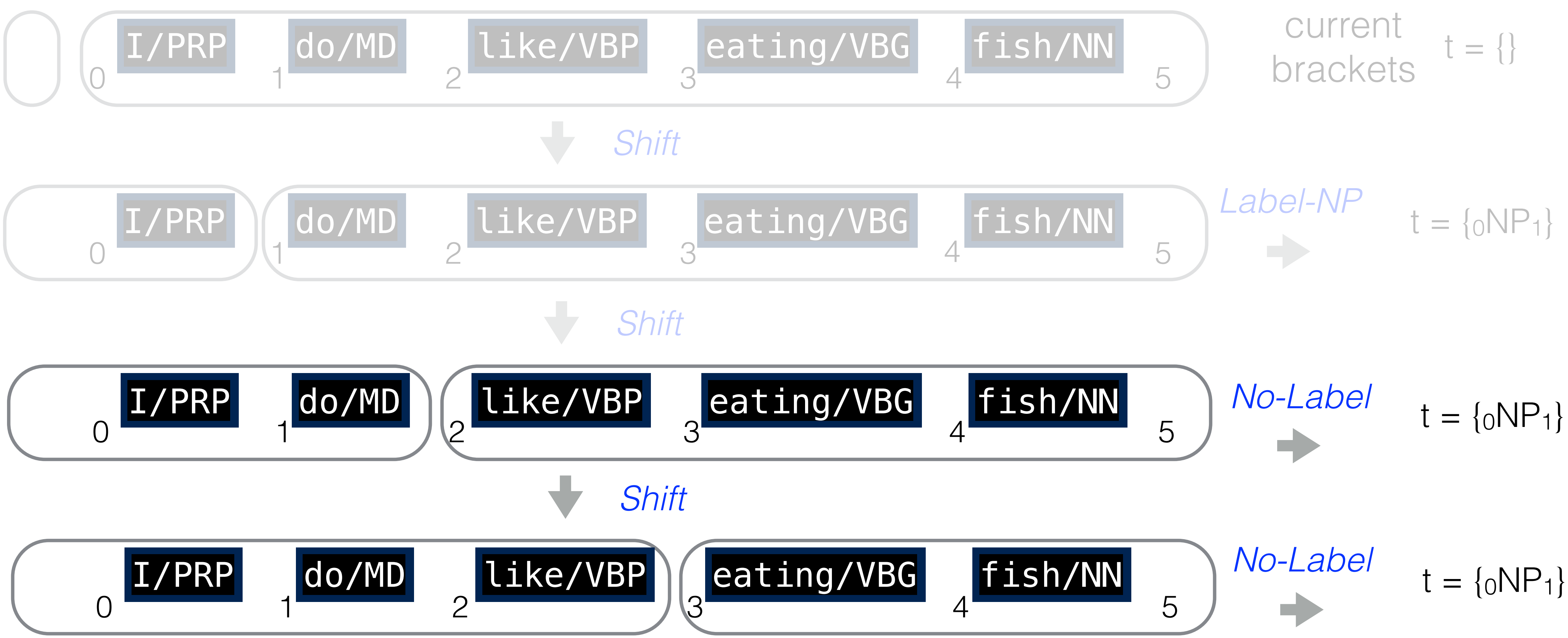
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



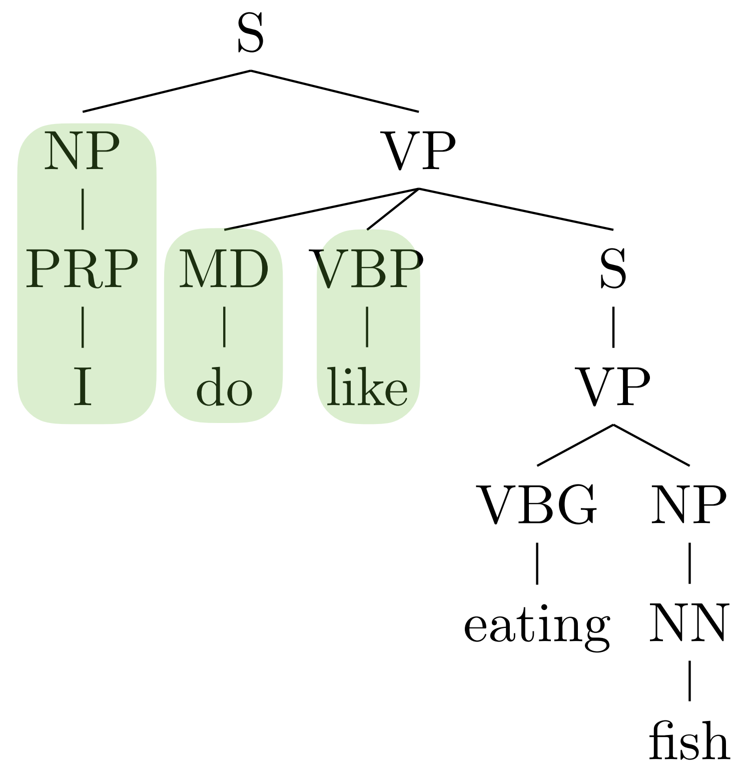


Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label





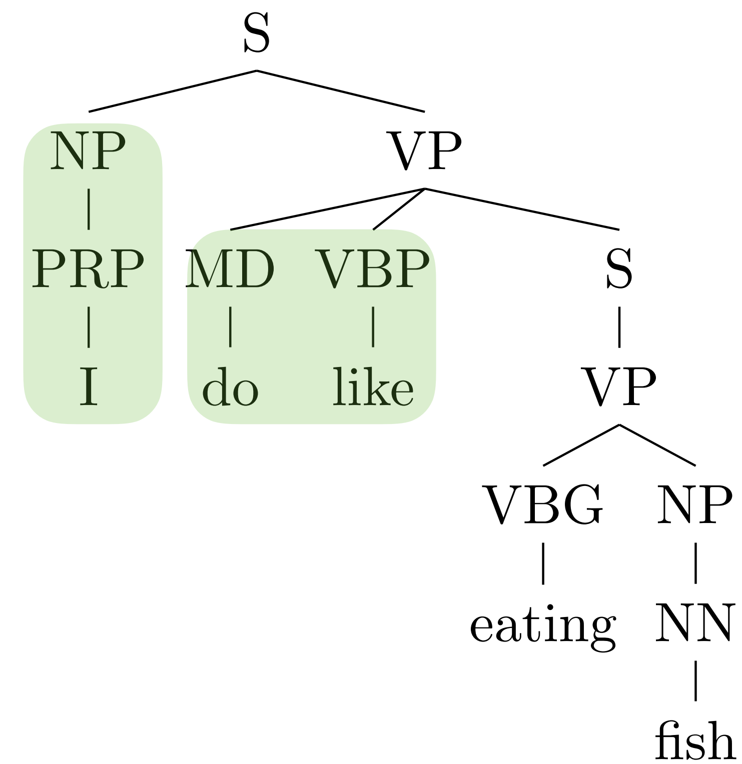
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label

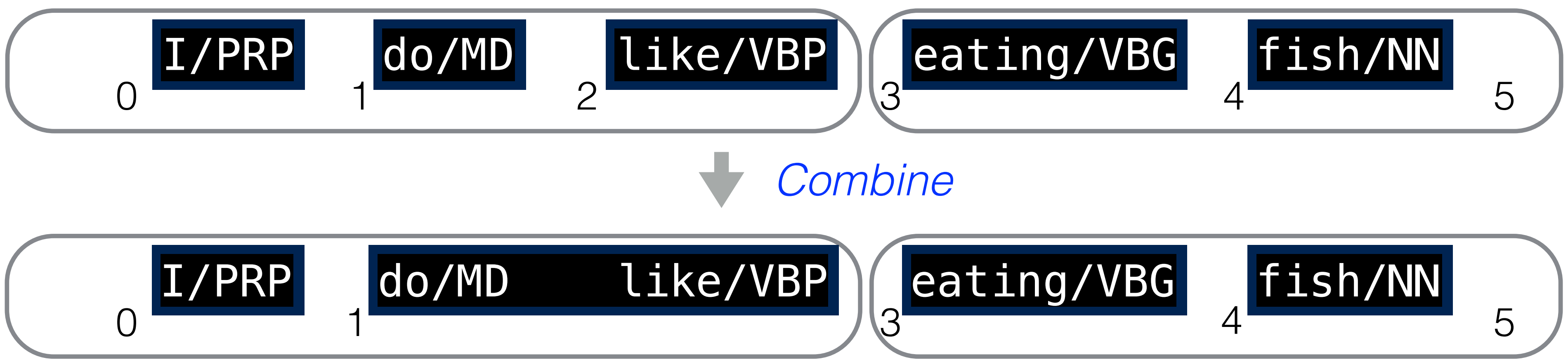


$_0$ `I/PRP` $_1$ `do/MD` $_2$ `like/VBP` $_3$ `eating/VBG` $_4$ `fish/NN` $_5$

$t = \{{}_0\mathrm{NP}_1\}$

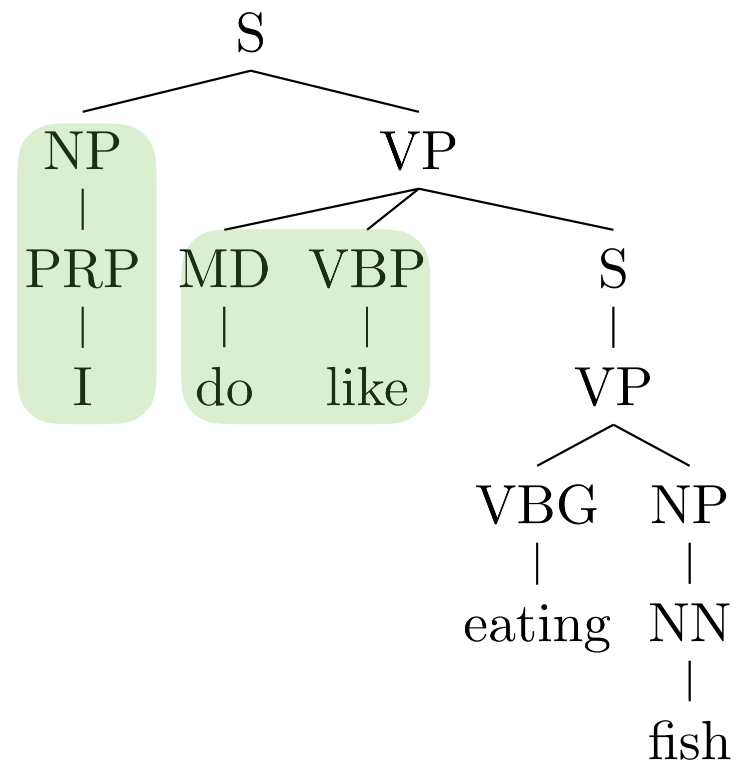
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label

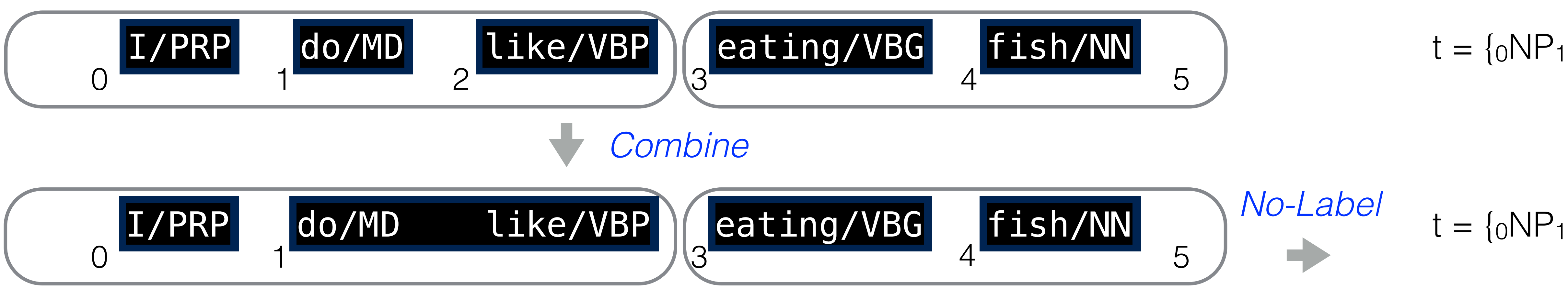




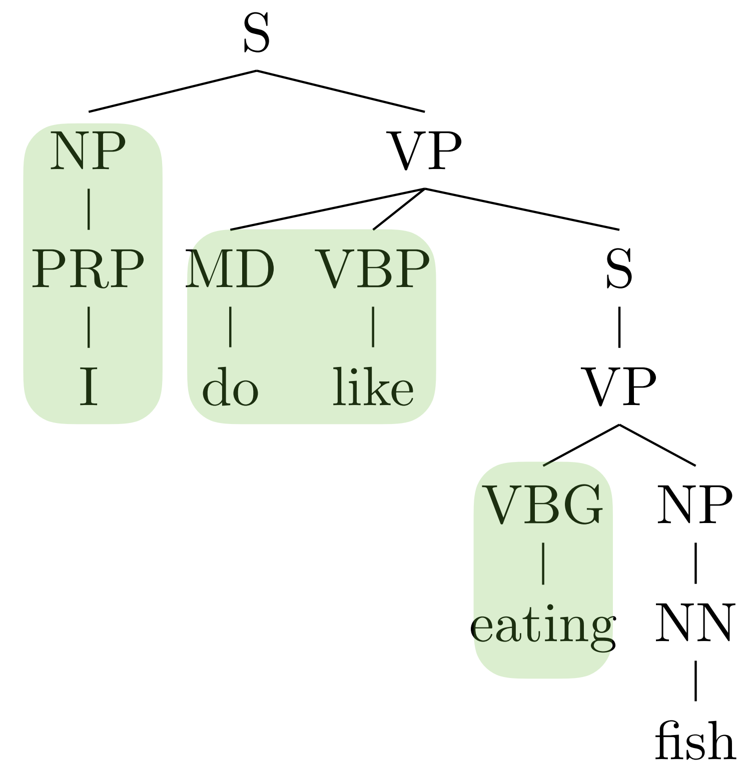
$t = \{_0NP_1\}$

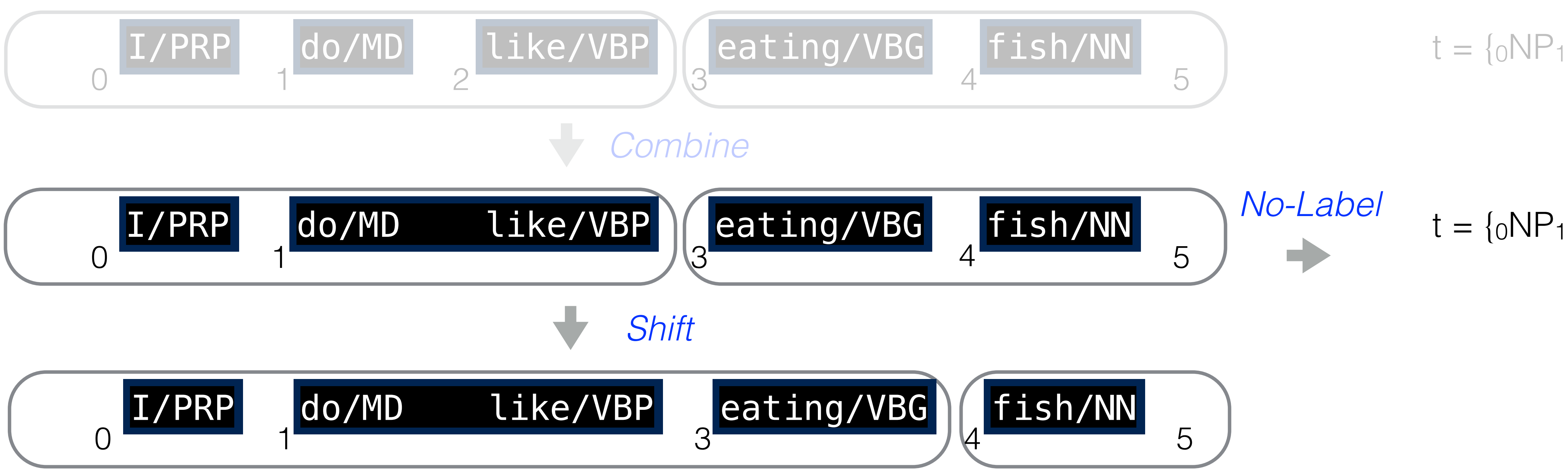
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



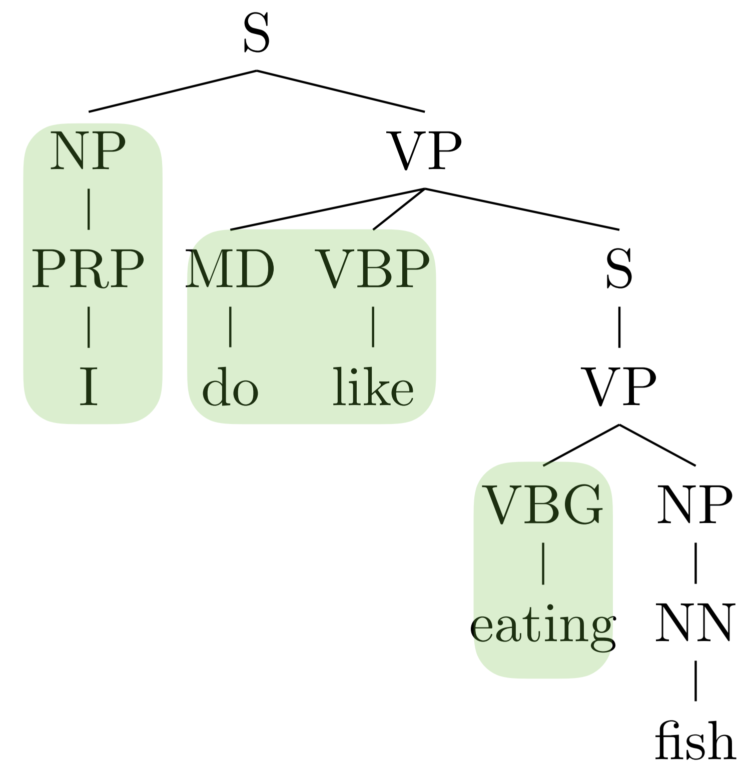


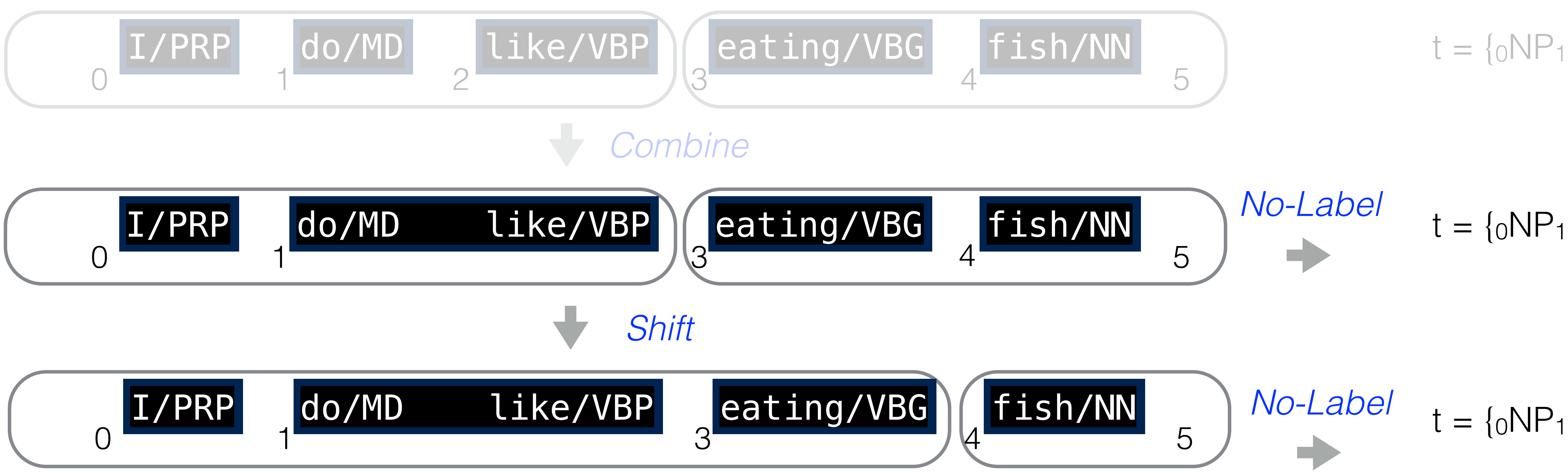
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



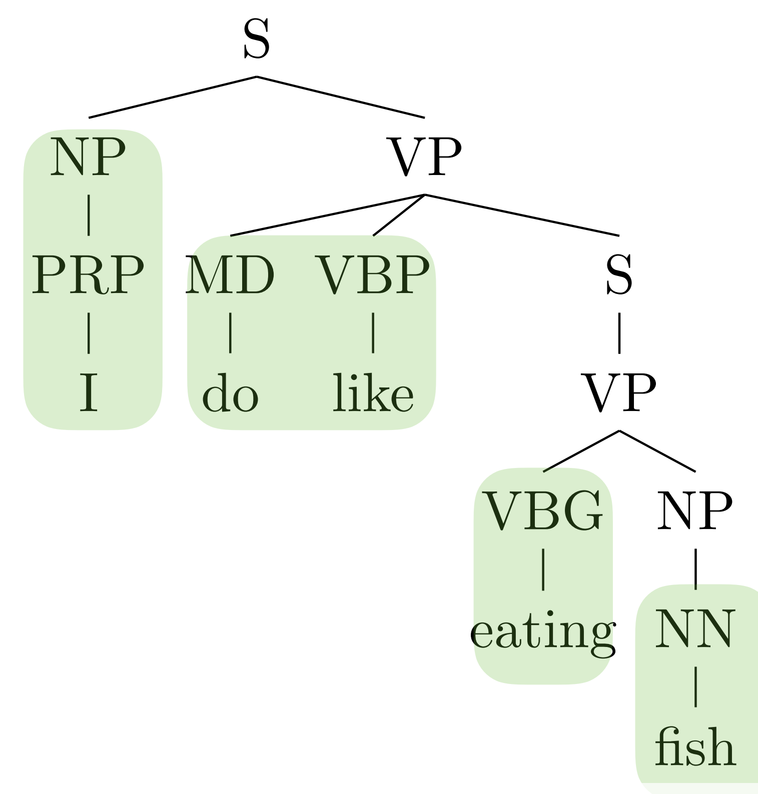


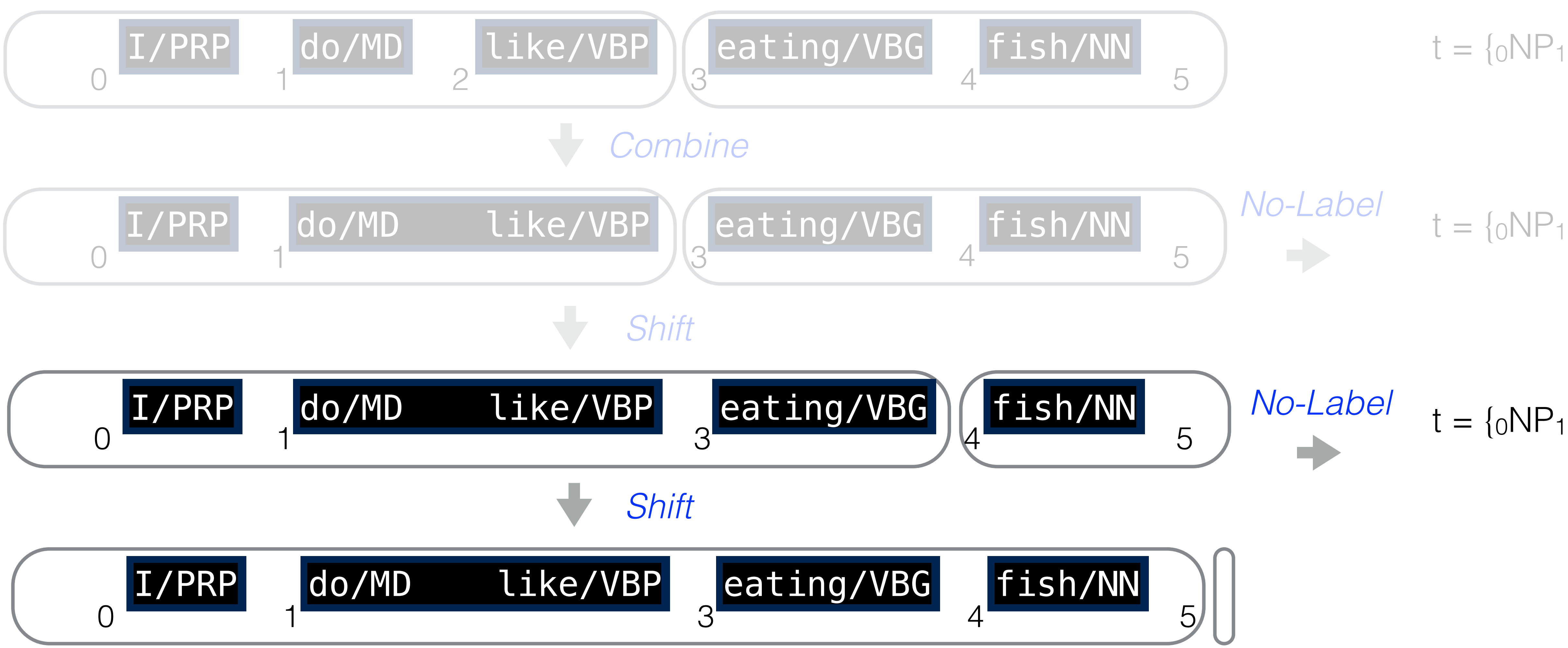
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



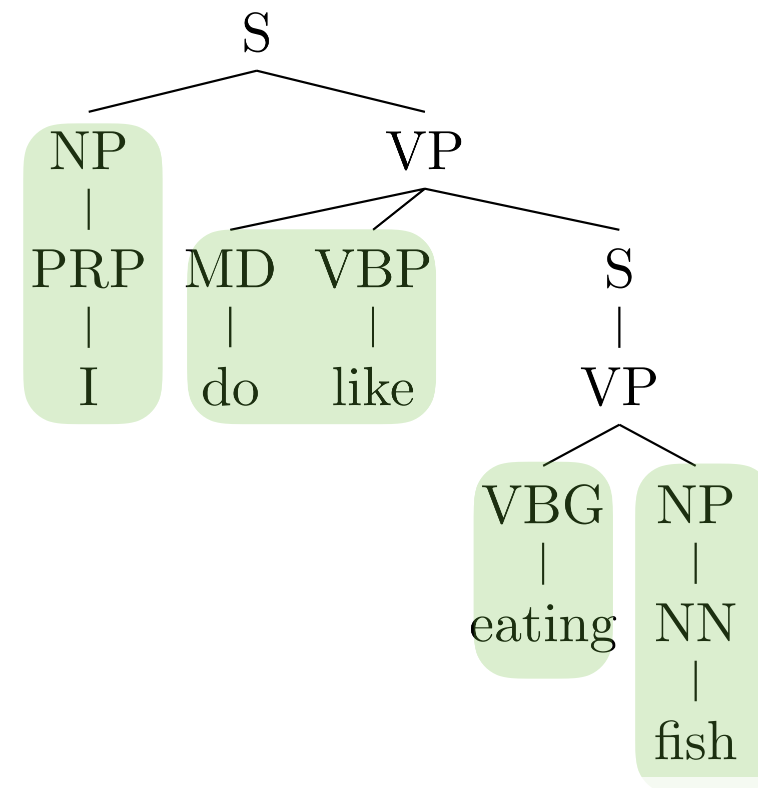


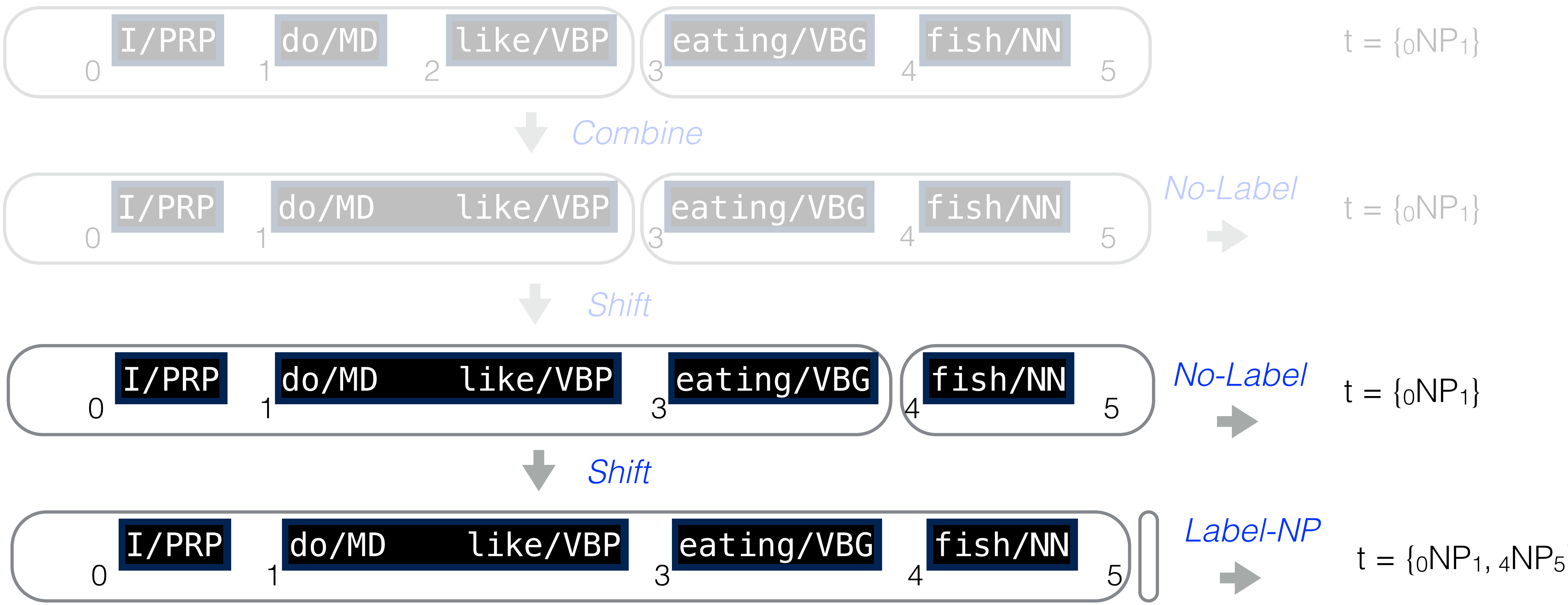
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



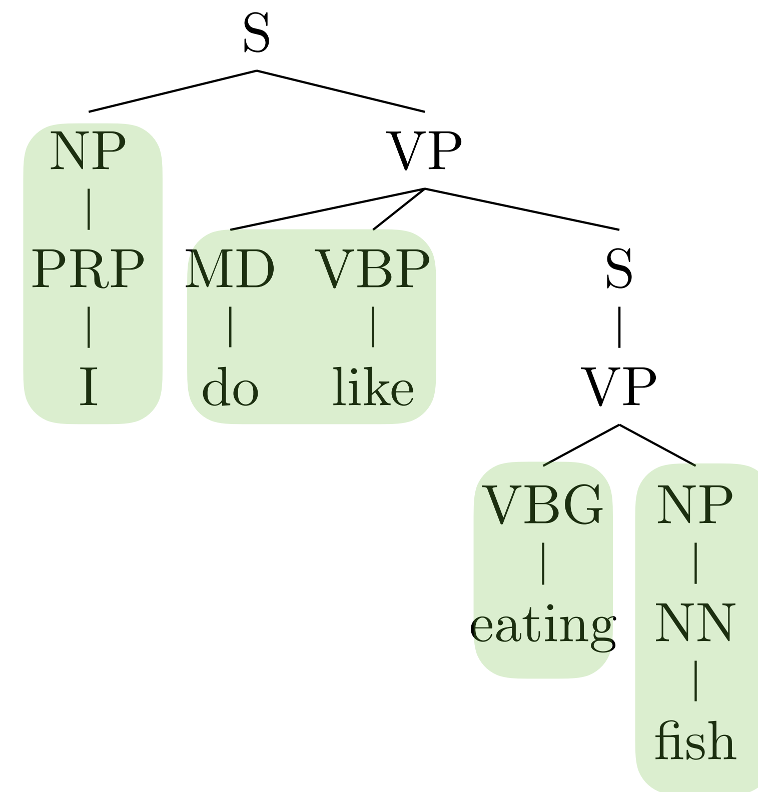


Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



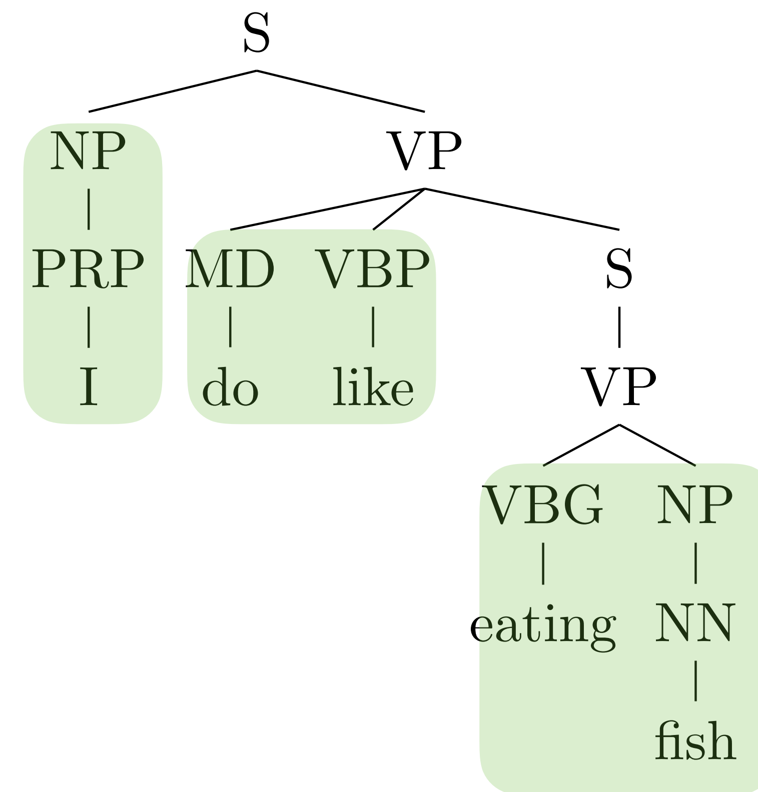


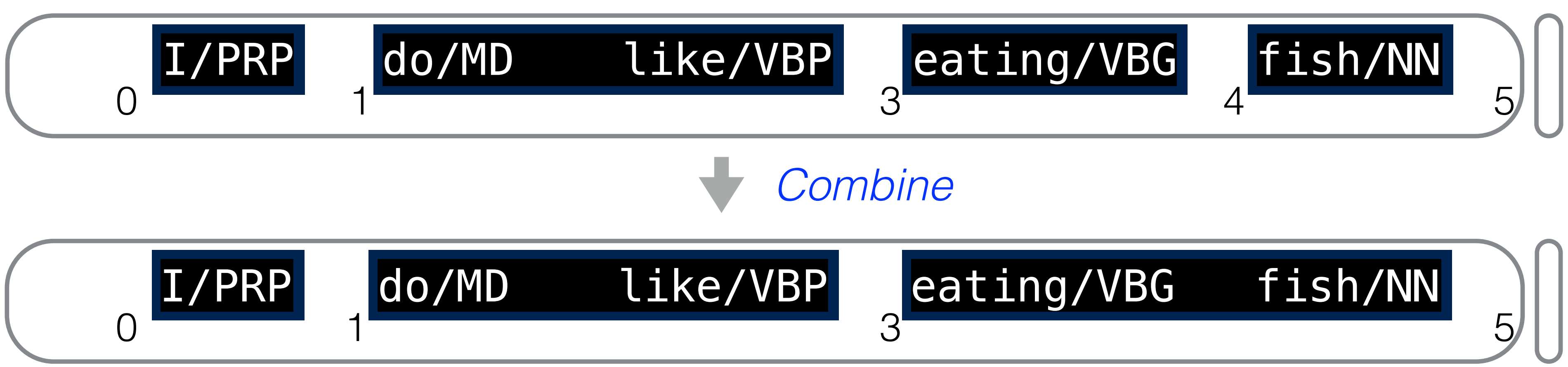
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



t = $\{{}_0\mathrm{NP}_1, {}_4\mathrm{NP}_5\}$

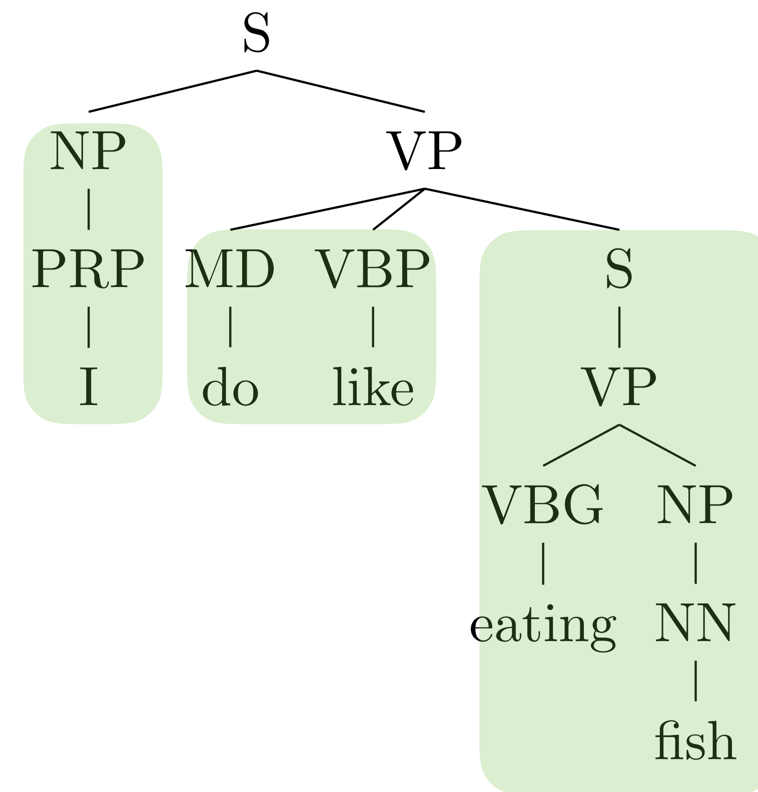
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label

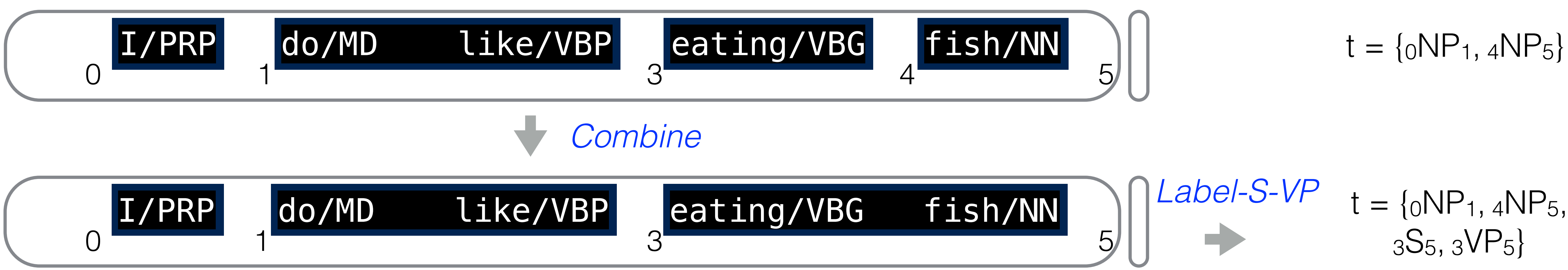




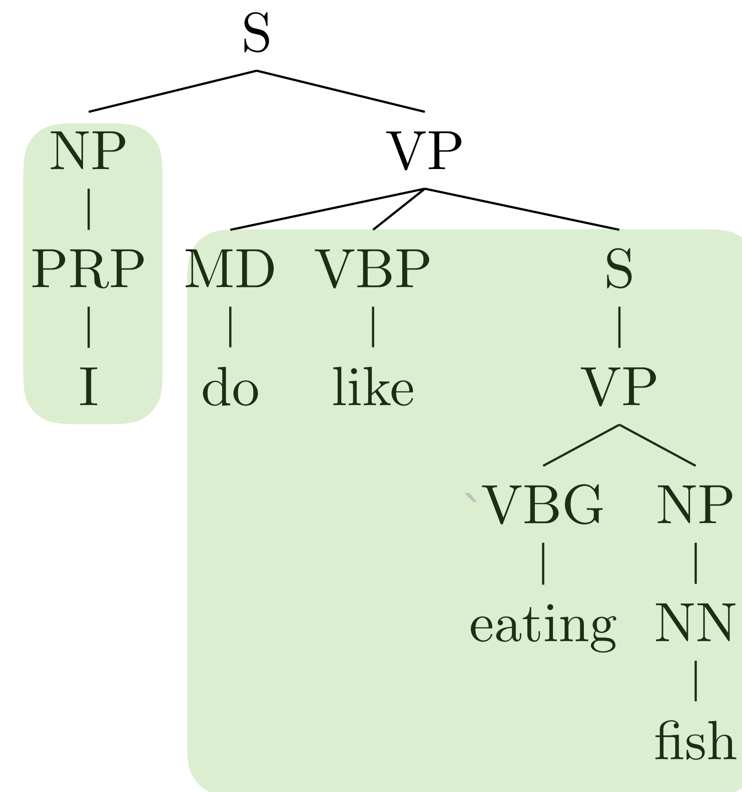
$t = \{{}_0NP_1, {}_4NP_5\}$

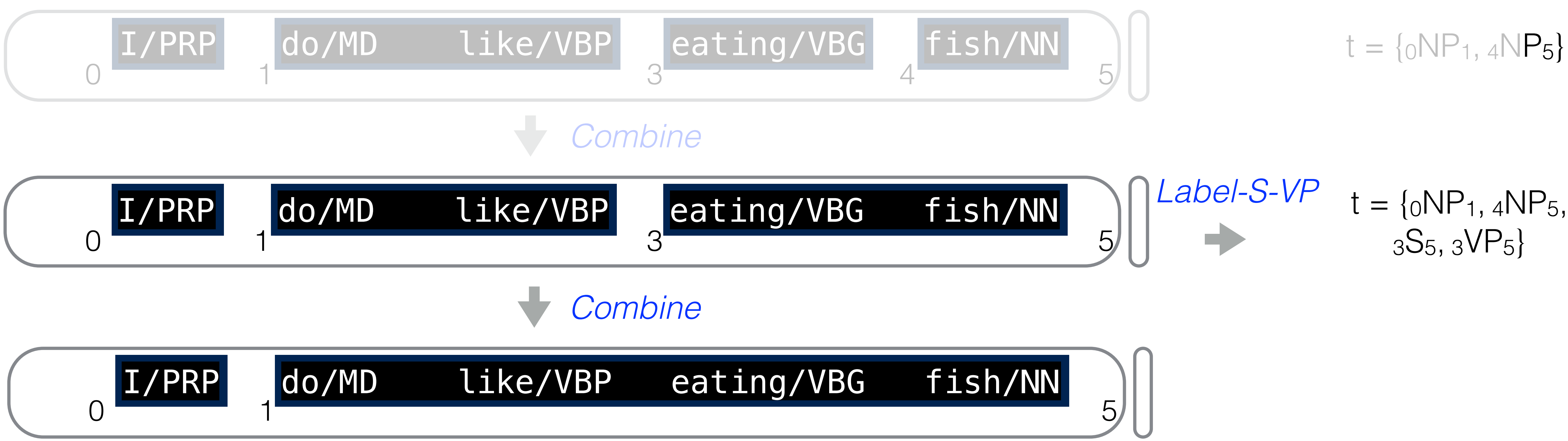
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



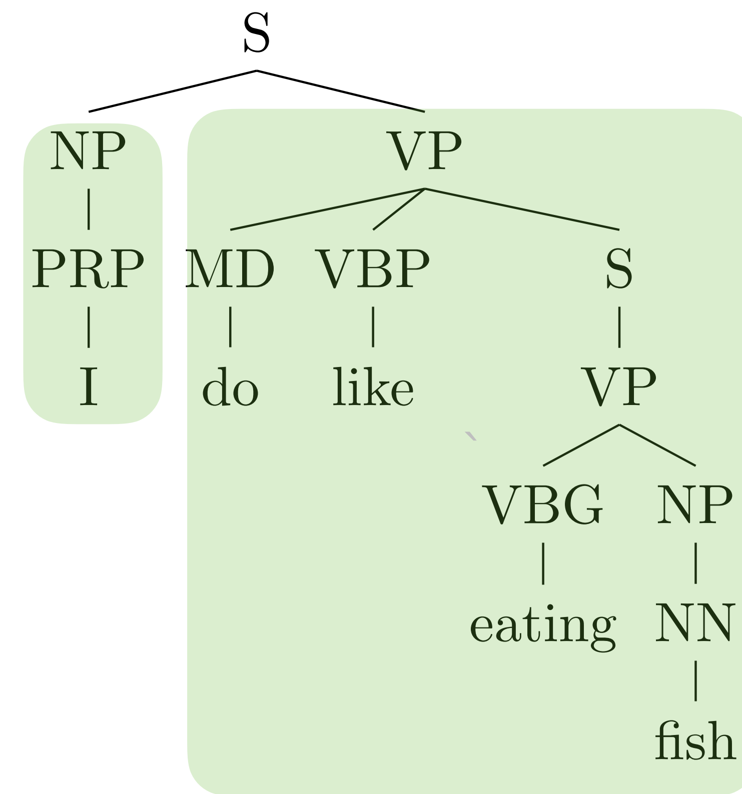


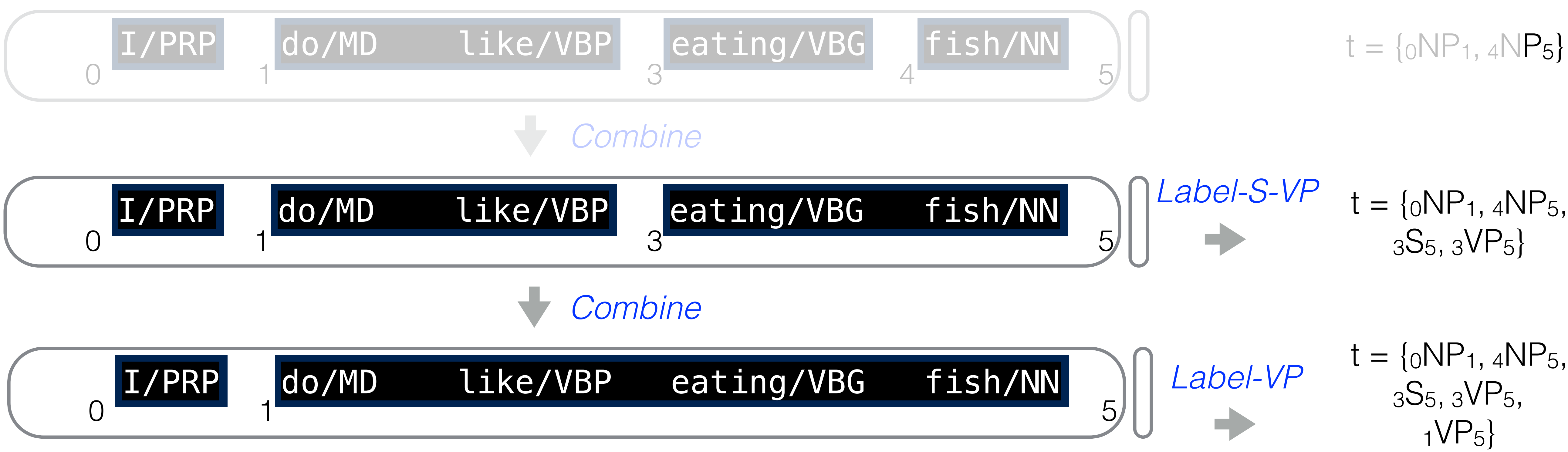
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



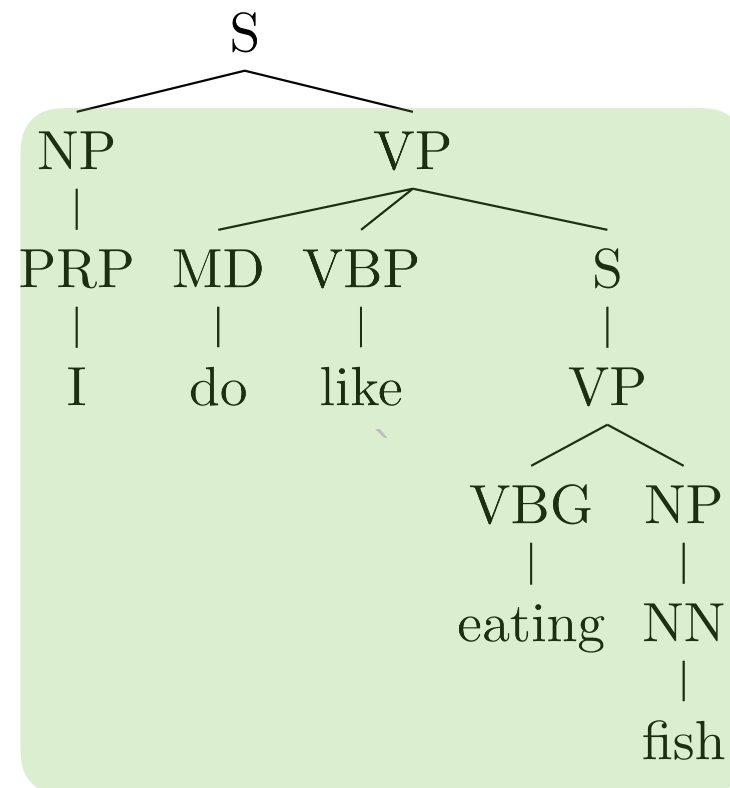


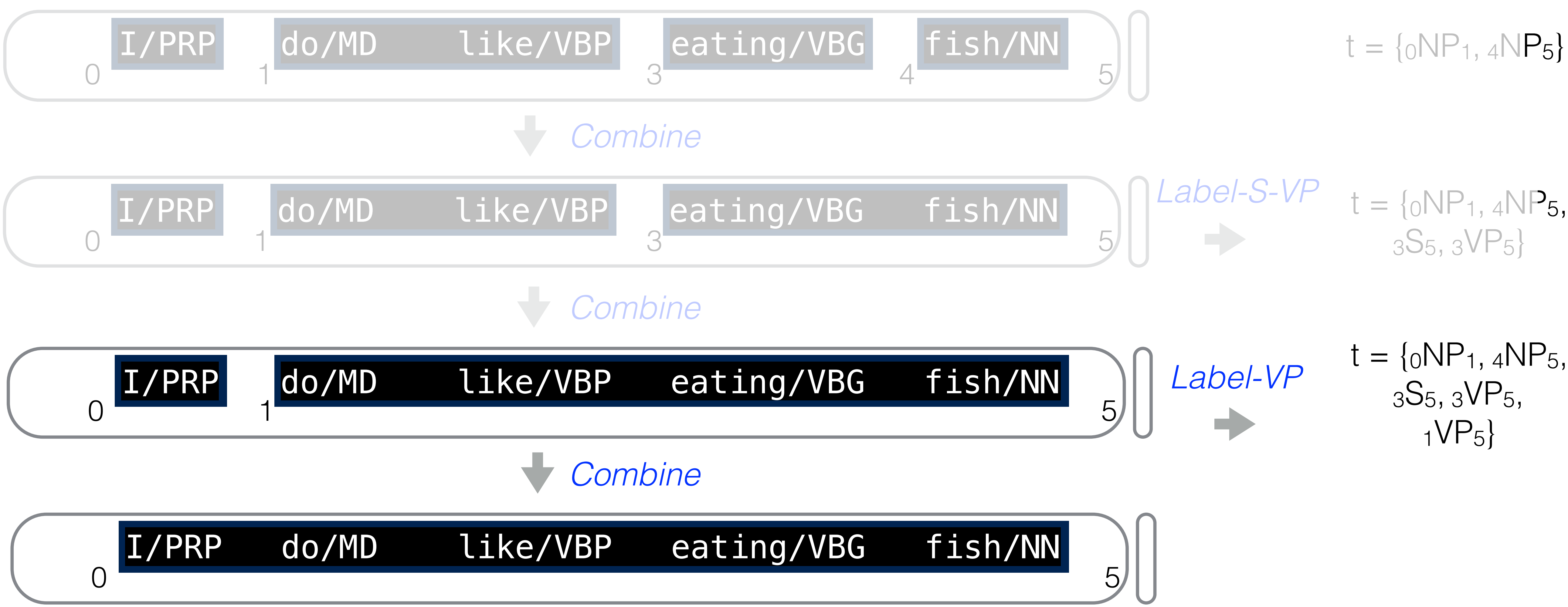
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label



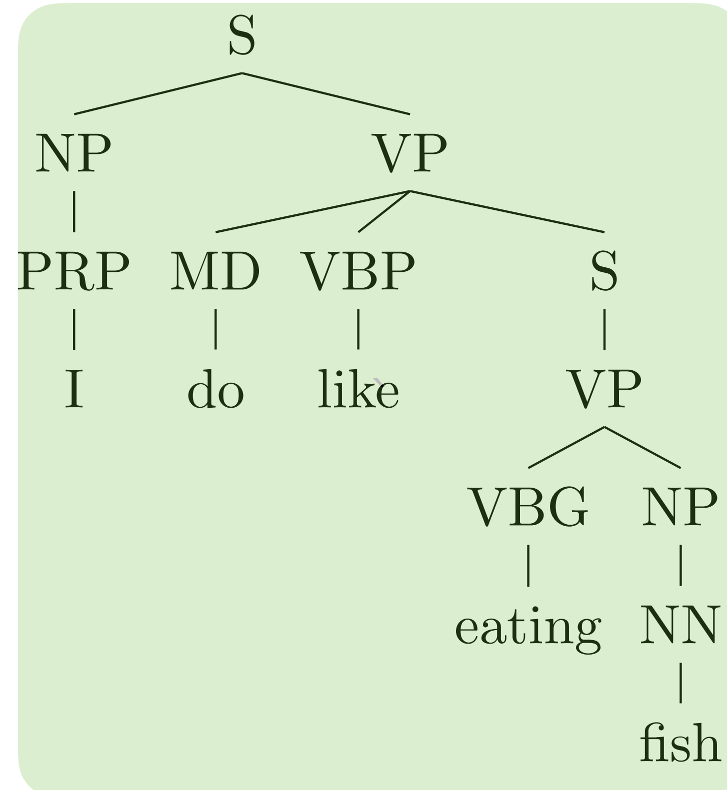


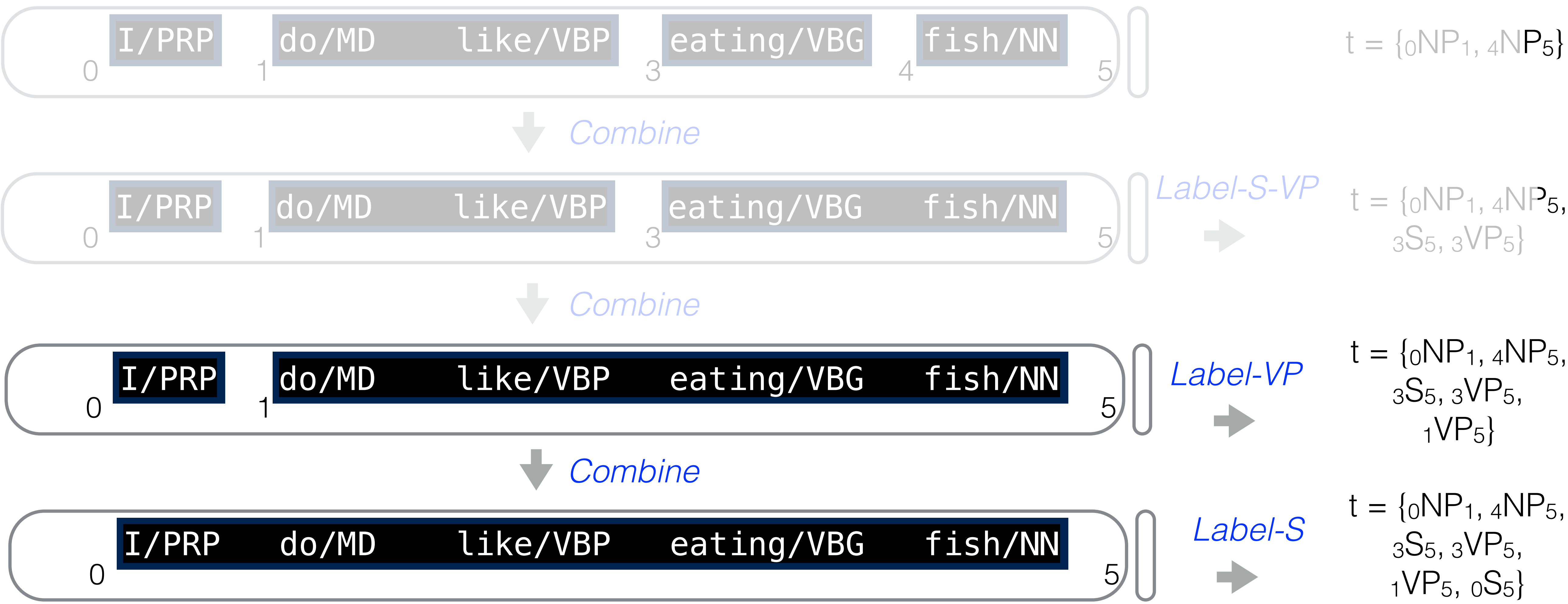
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label





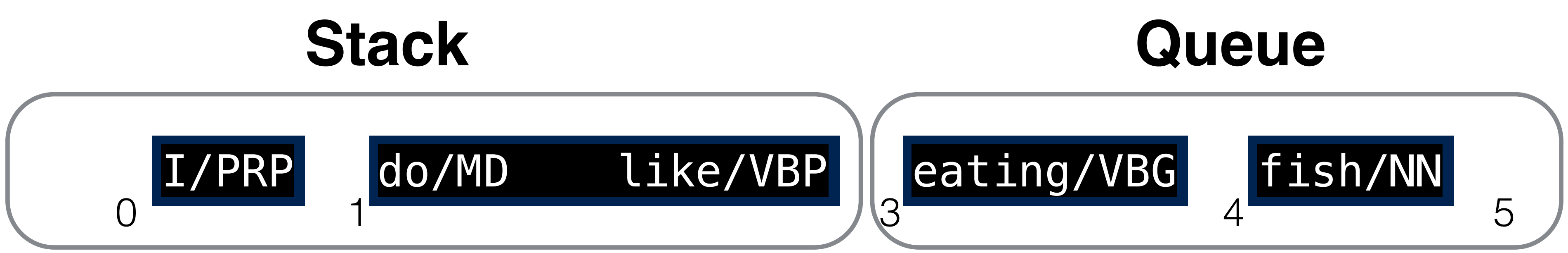
Structural (even step)	Shift
	Combine
Label (odd step)	Label-*X*
	No-Label





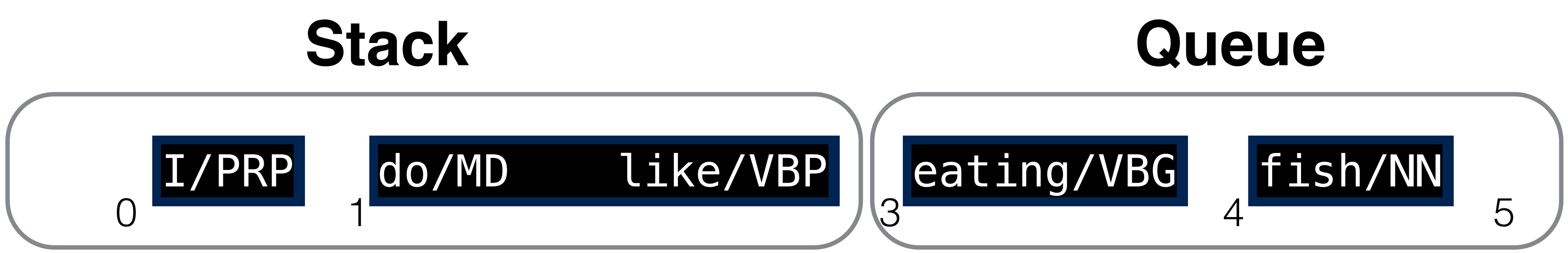
Advantages of Span-Based System

- Linear-time and fixed number of steps (well-suited for beam search)
- Separates prediction of structure and labels
- Predicts rules of arbitrary arity with no binarization



Advantages of Span-Based System

- Linear-time and fixed number of steps (well-suited for beam search)
- Separates prediction of structure and labels
- Predicts rules of arbitrary arity with no binarization

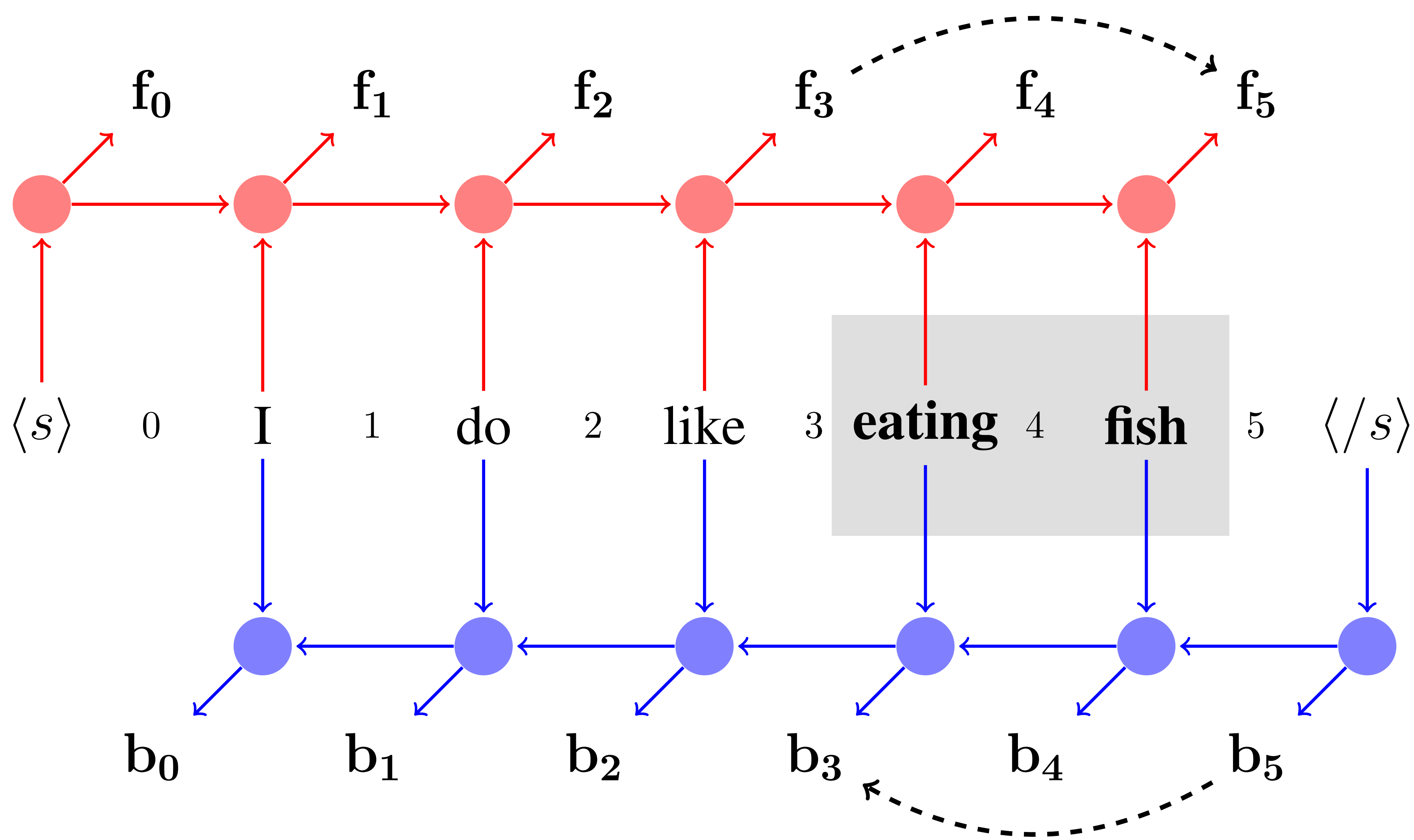


Q: How to decide which action to take?
What features represent spans?

Outline

- Span-Based Constituency Parsing
- **Bi-Directional LSTM Span Features**
- Provably Optimal Dynamic Oracle
- Experiments

Bi-LSTM Span Features

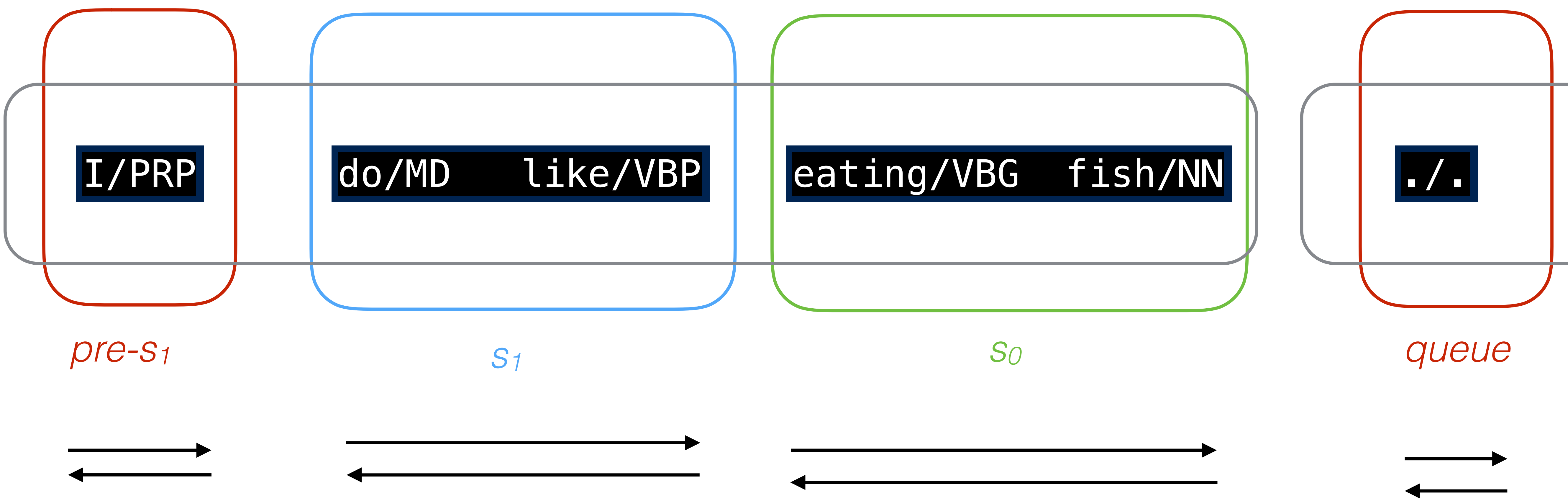


Sentence segment “eating fish” represented by two vectors:

- Forward component: f_5 - f_3 (Wang and Chang, ACL 2016)
- Backward component: b_3 - b_5

Span Features for Structure Action

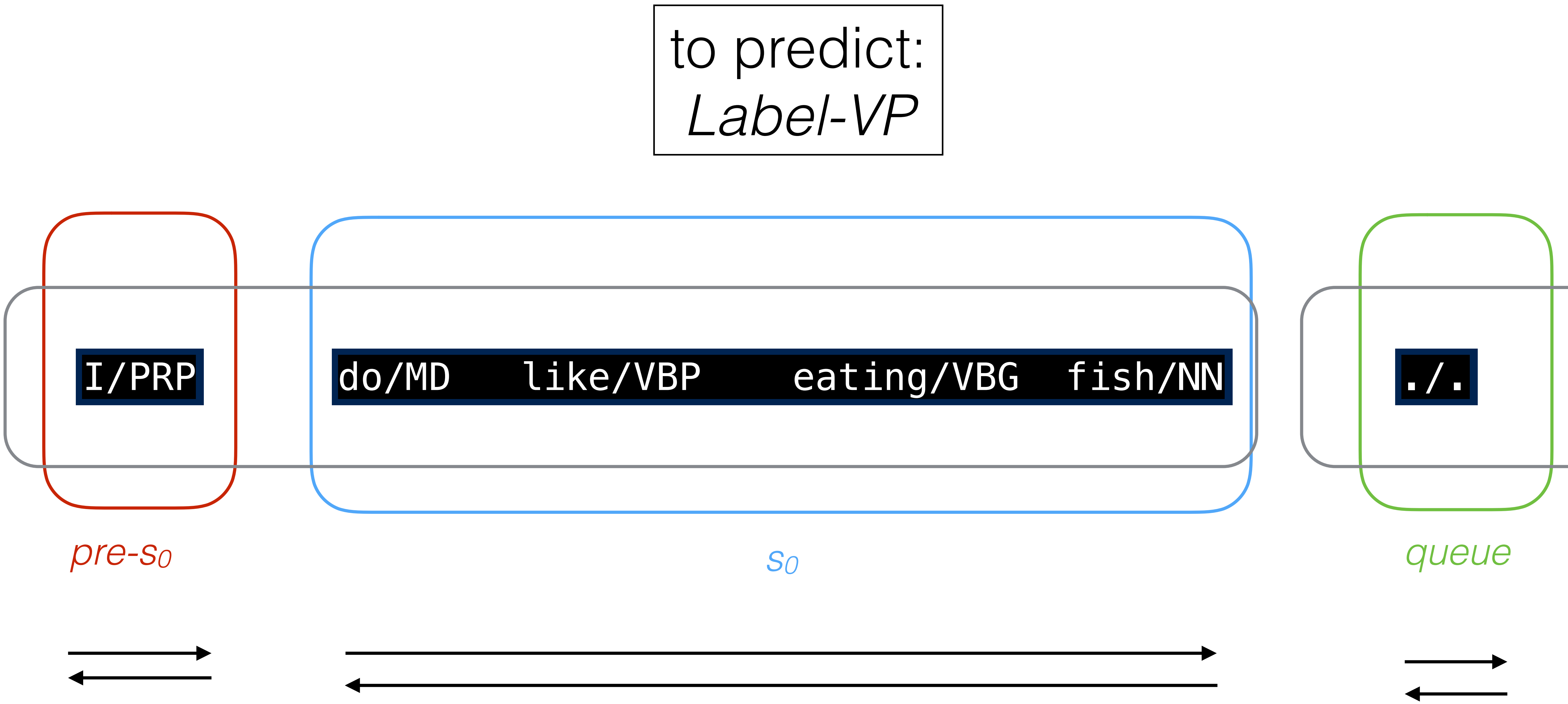
to predict:
Combine



4 bi-LSTM span features

(no tree-structure information used)

Span Features for Label Action

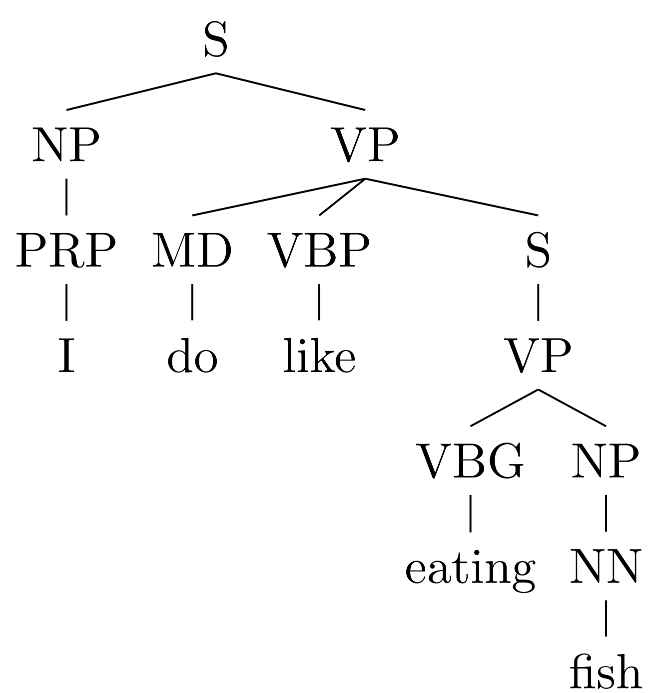


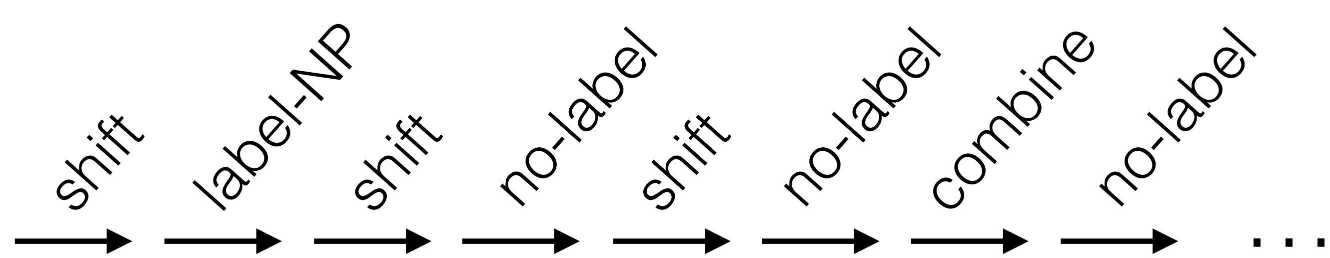
3 bi-LSTM span features

(no tree-structure information used)

Training Scheme: Local

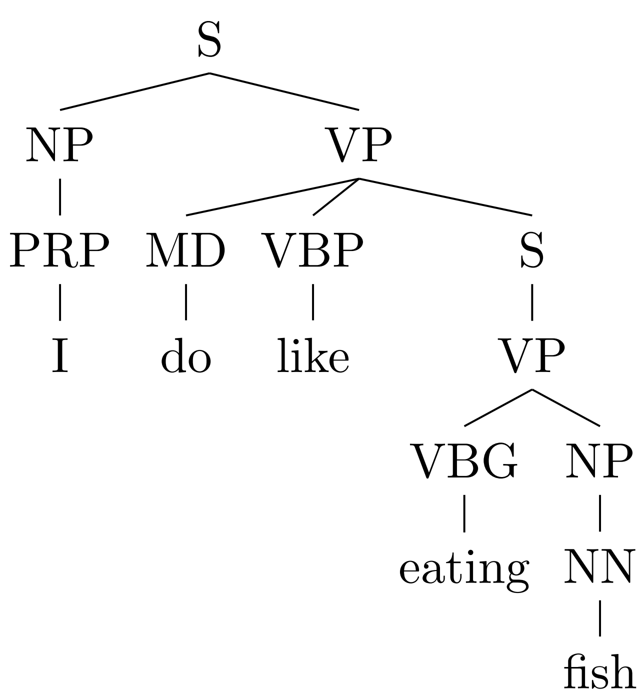
- Every parser state is paired with a correct action
- Separate multilayer perceptron for each action type
- Baseline training scheme (static oracle) uses canonical order with short-stack preference

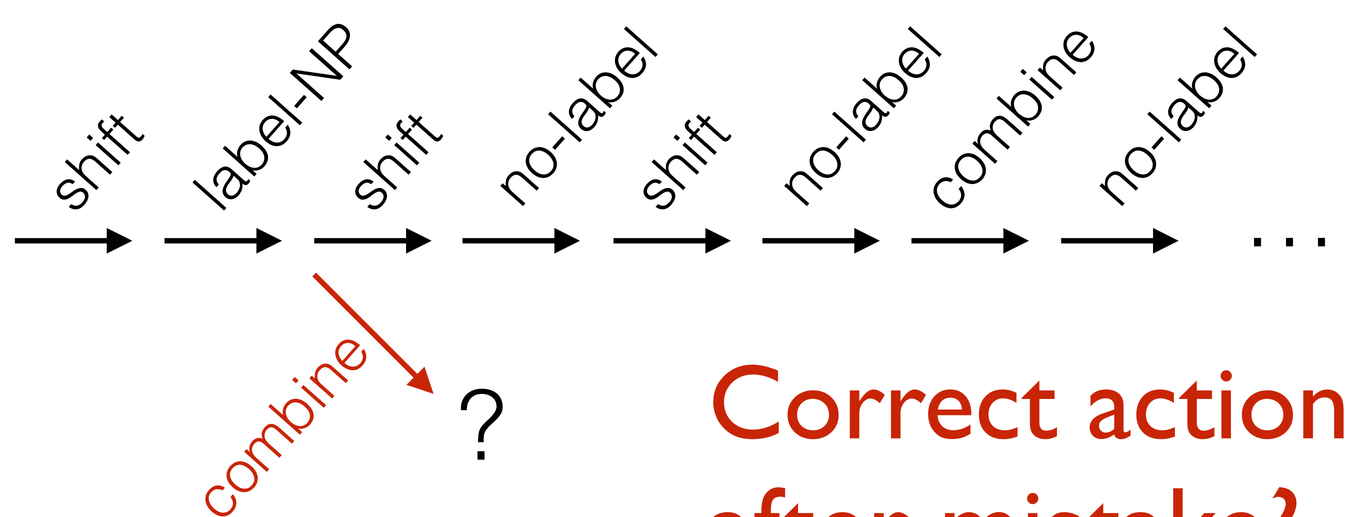




Training Scheme: Local

- Every parser state is paired with a correct action
- Separate multilayer perceptron for each action type
- Baseline training scheme (static oracle) uses canonical order with short-stack preference





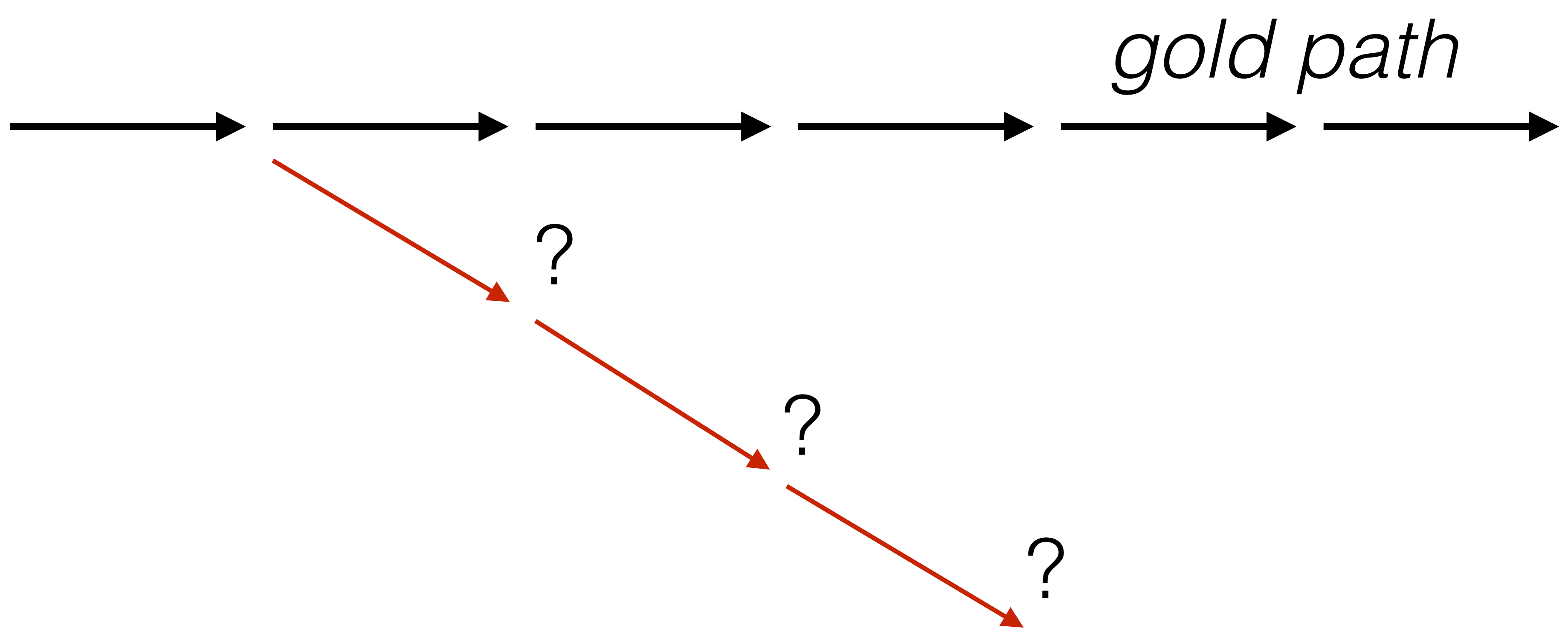
Correct action after mistake?

Outline

- Span-Based Constituency Parsing
- Bi-Directional LSTM Span Features
- **Provably Optimal Dynamic Oracle**
- Experiments

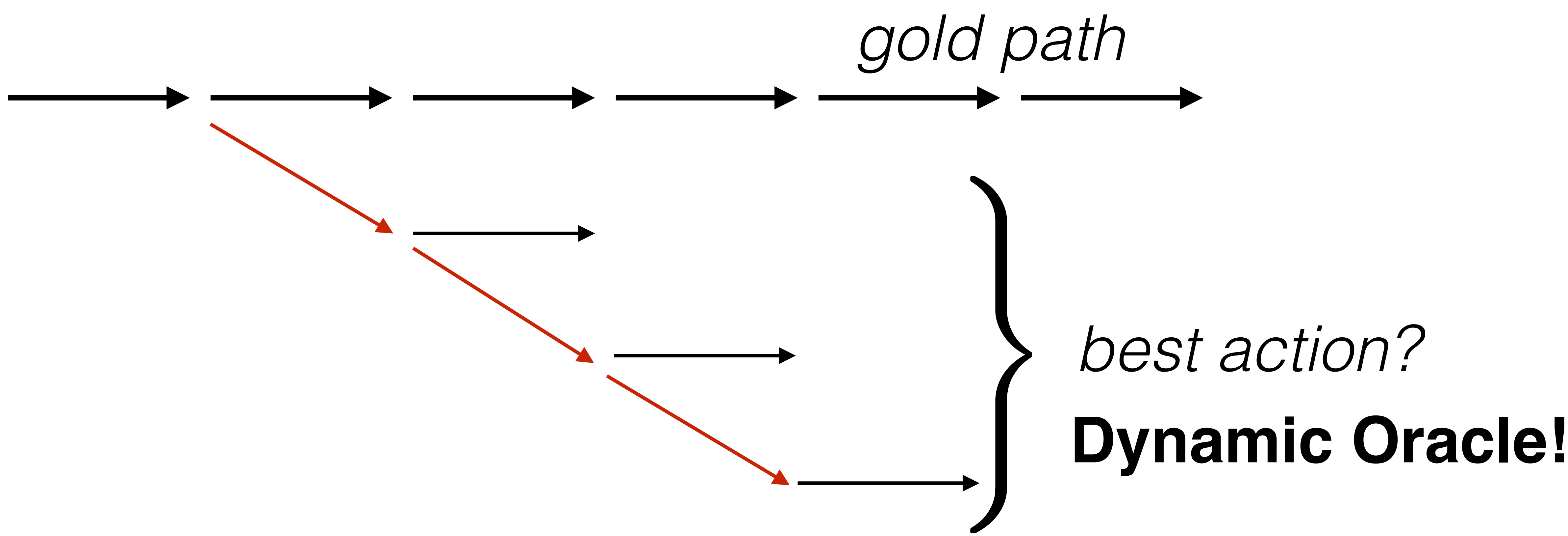
Dynamic Oracle: Motivation

- Static oracle training assumes all correct actions
- What to do after decoding mistakes?

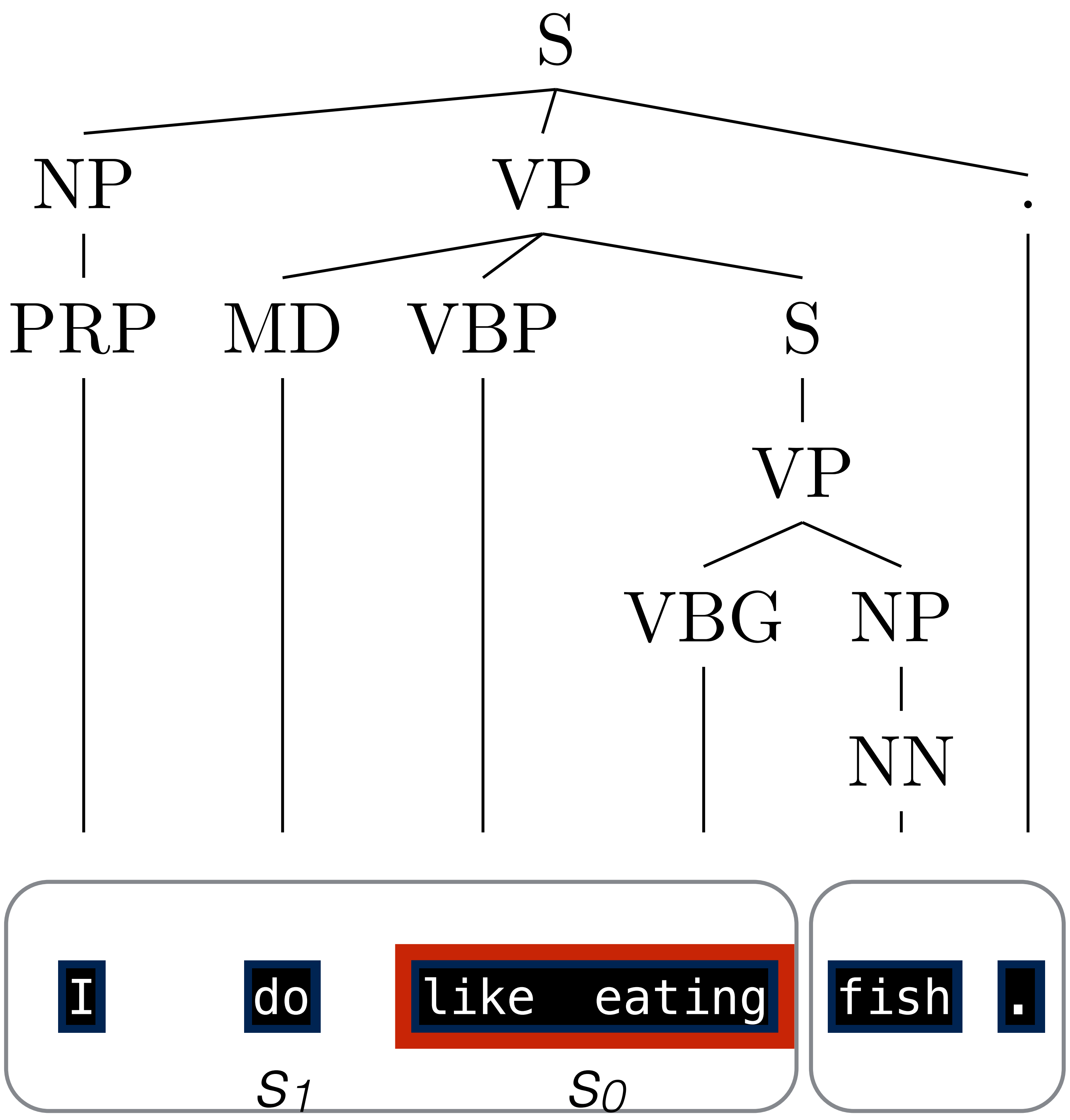


Dynamic Oracle: Motivation

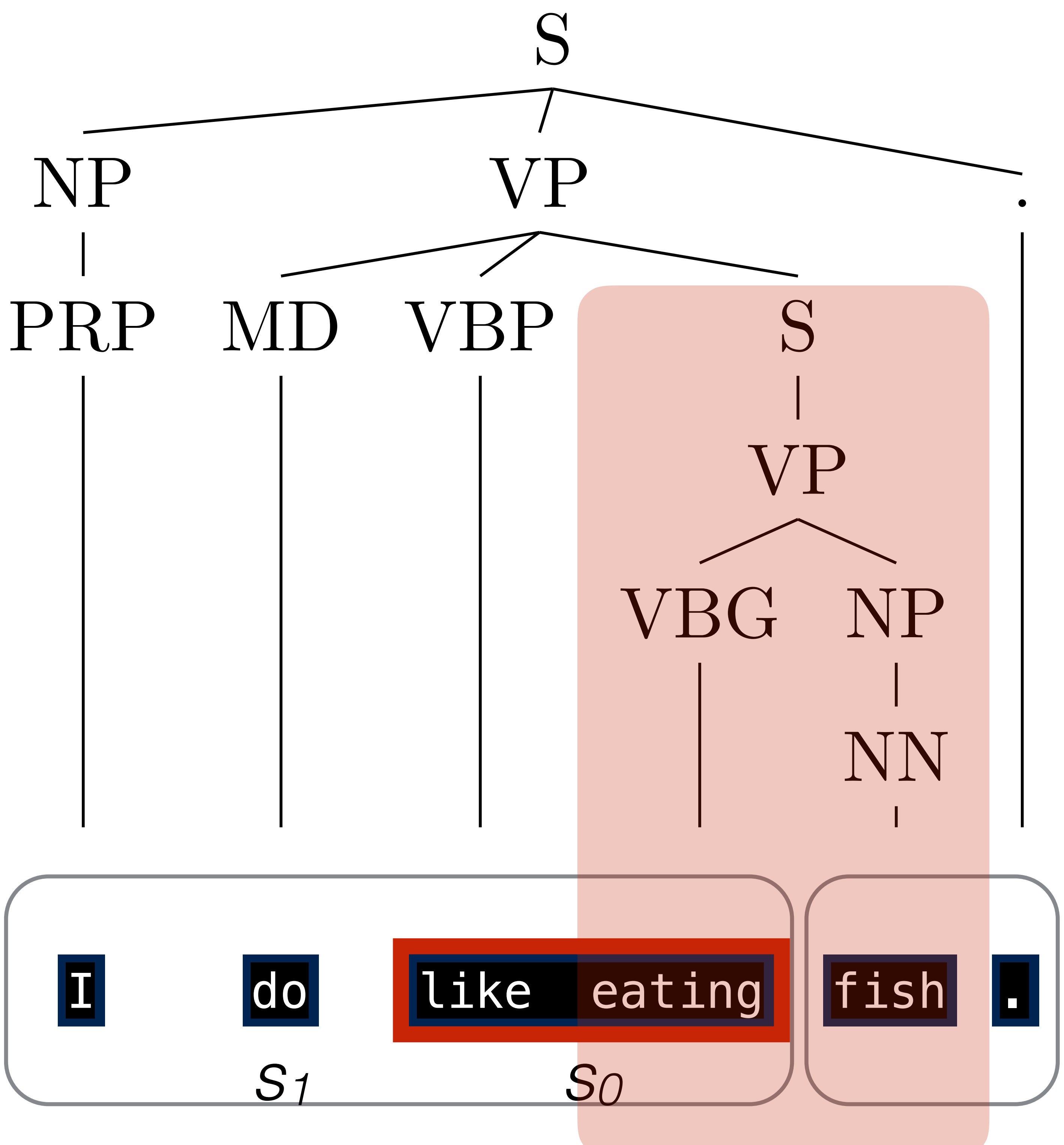
- Static oracle training assumes all correct actions
- What to do after decoding mistakes?
- Need a way to decide best action in arbitrary state:
  Dynamic Oracle (everywhere-defined optimal policy)



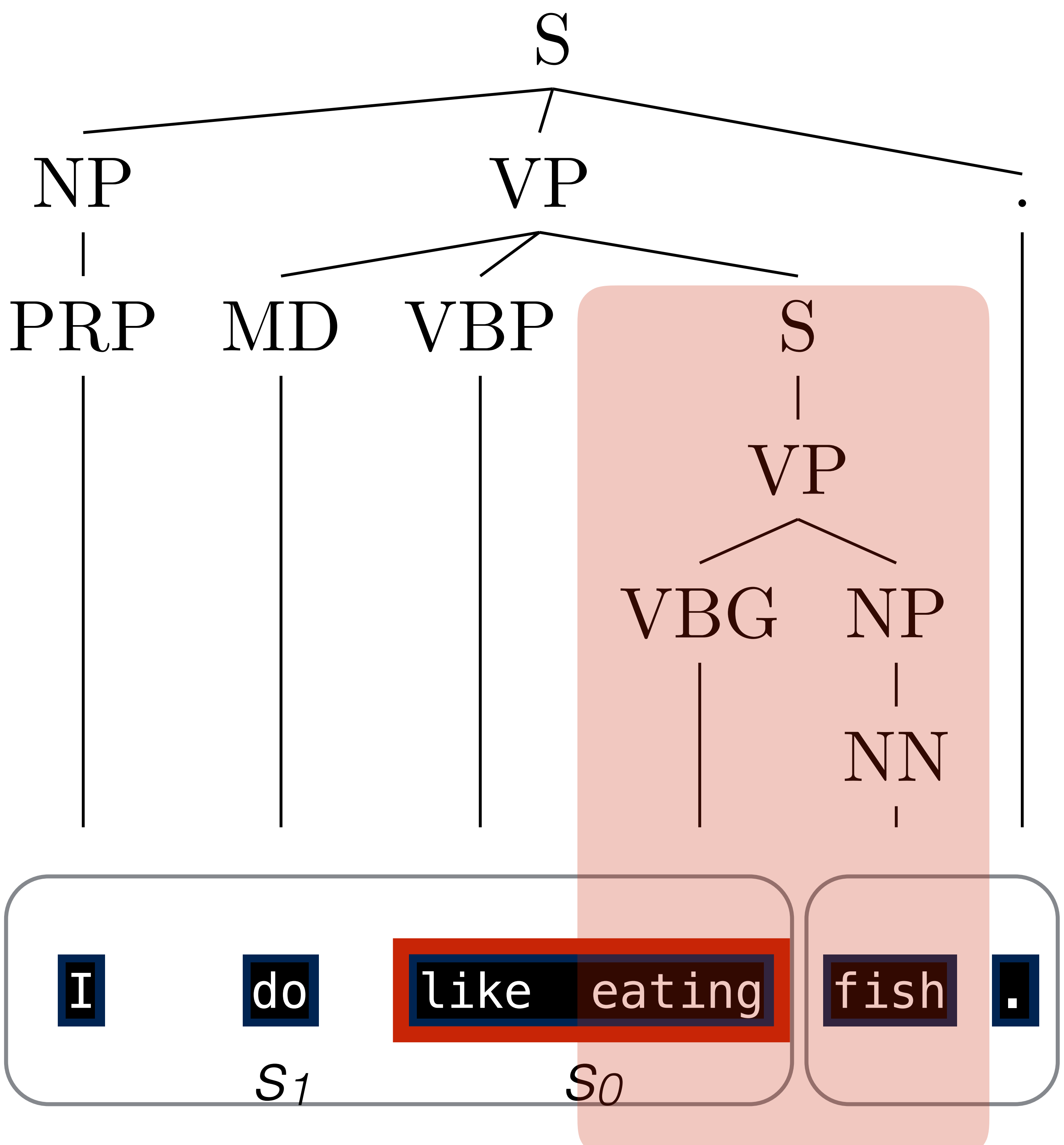
Dynamic Oracle: Example



Dynamic Oracle: Example

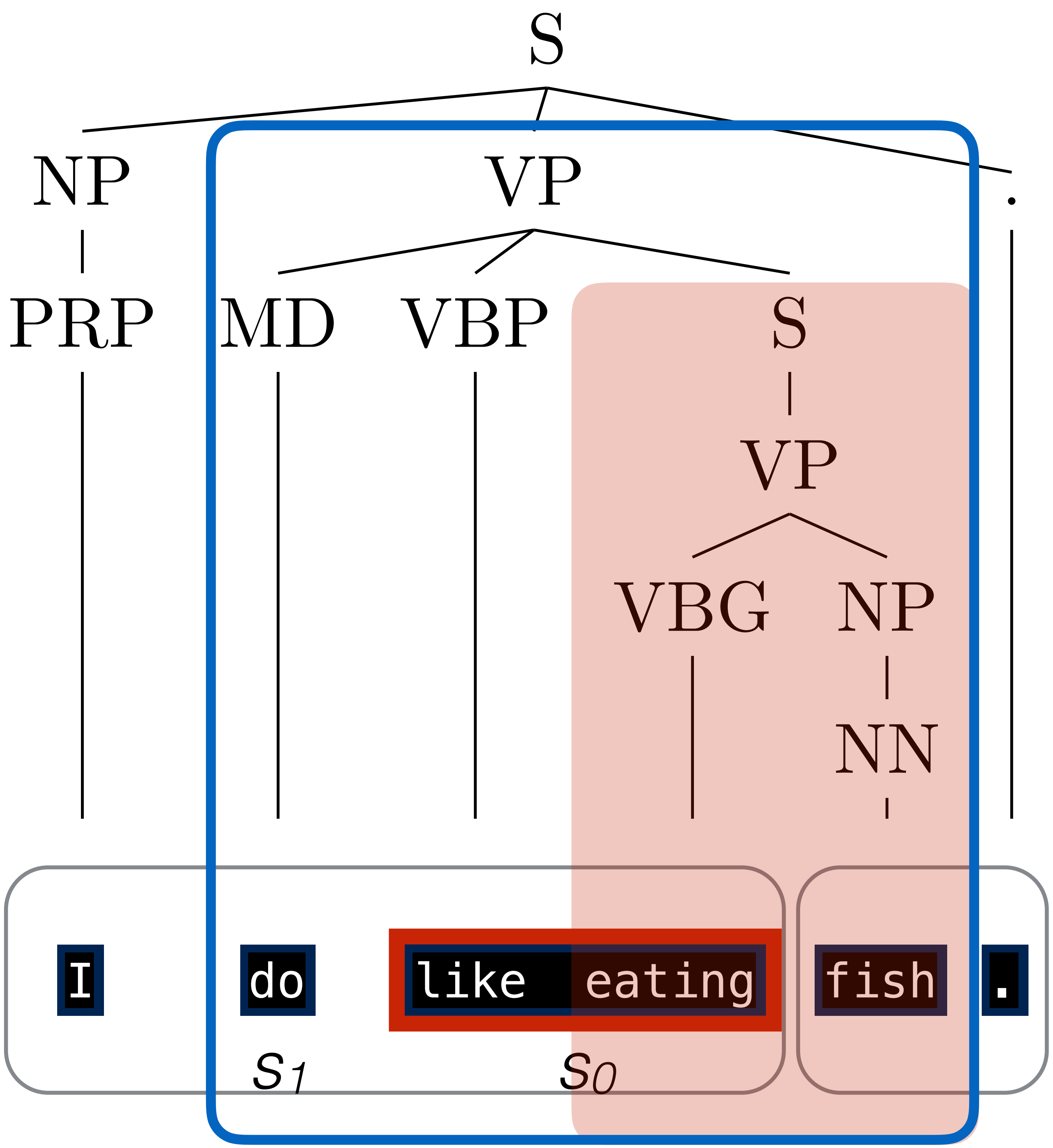


Dynamic Oracle: Example



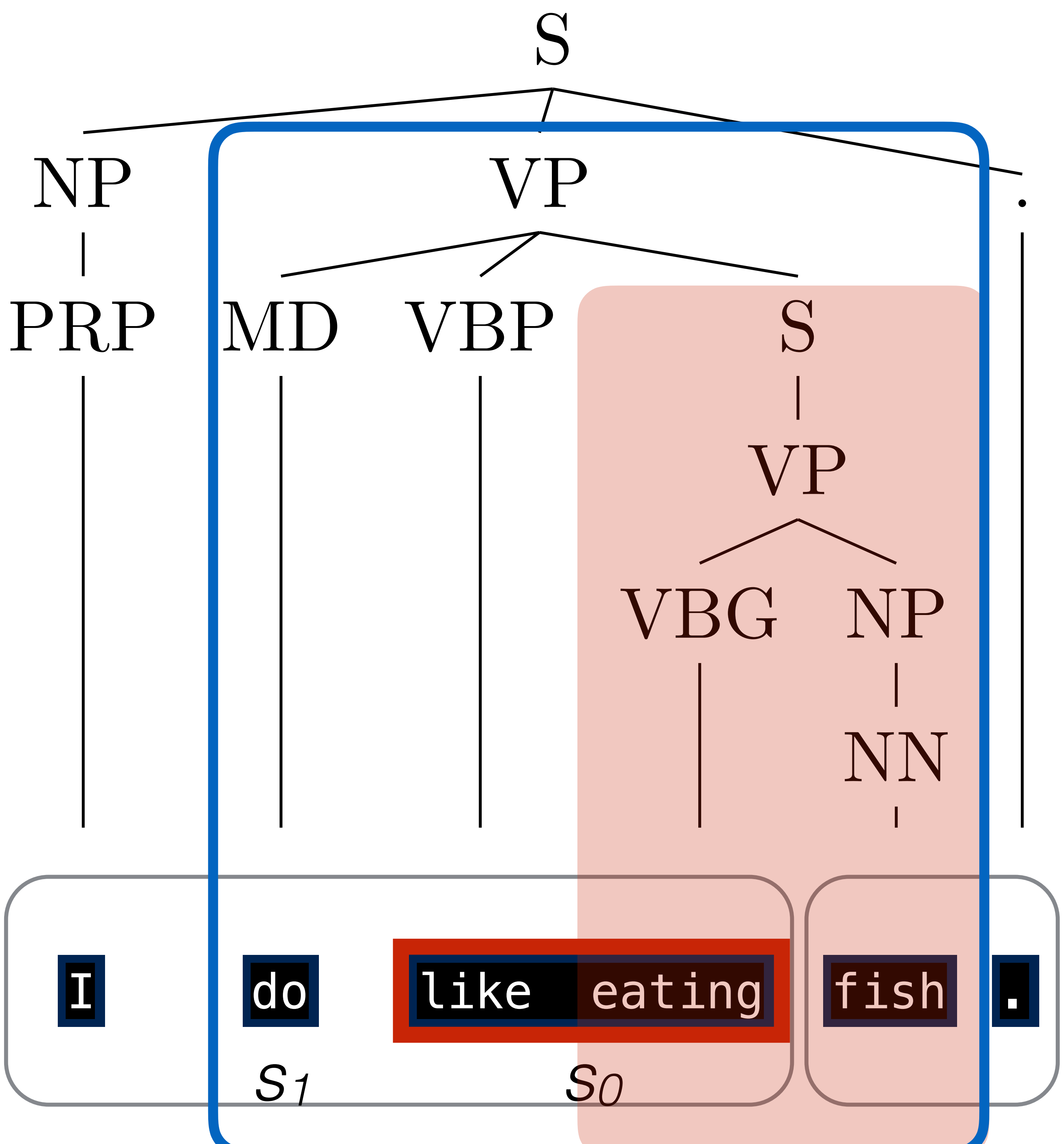
smallest reachable gold bracket incl. s_0

Dynamic Oracle: Example

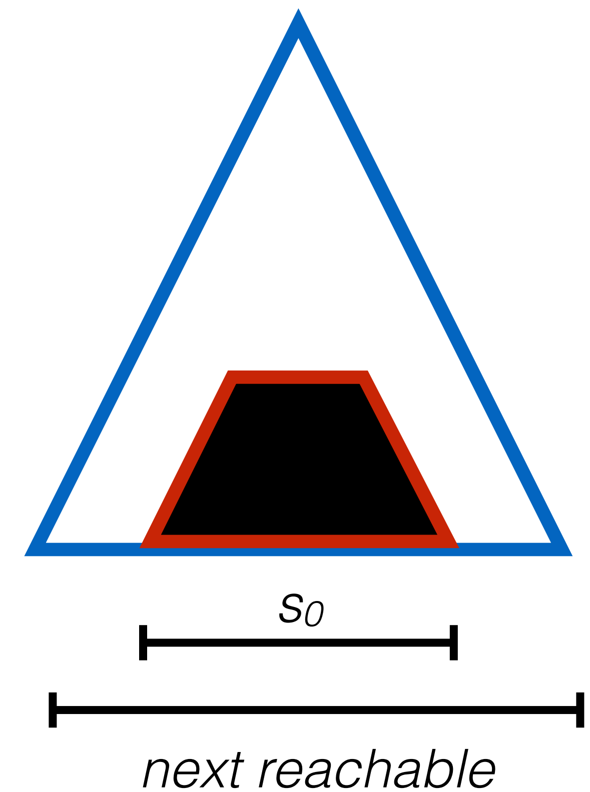


smallest reachable gold bracket incl. s_0

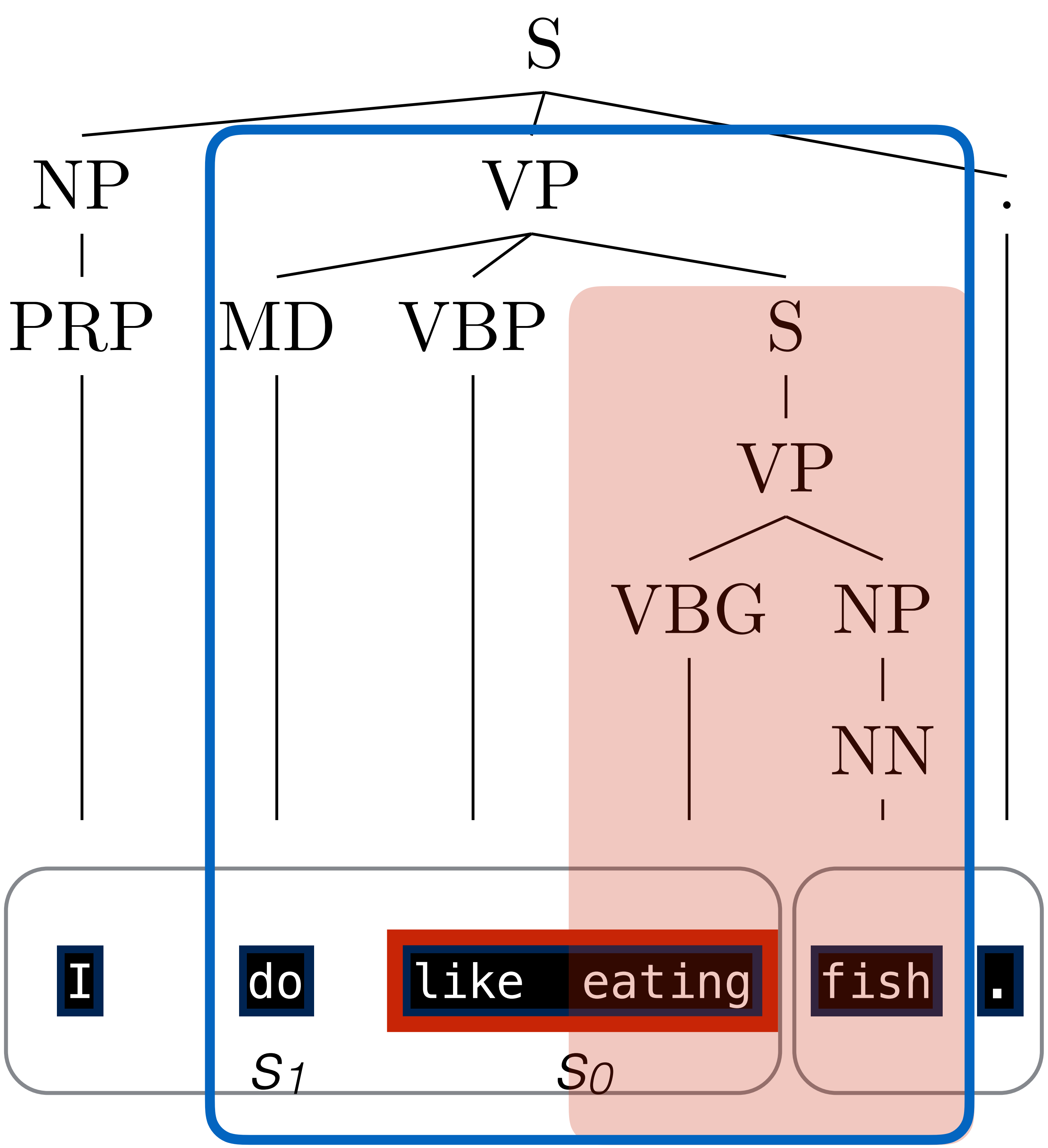
Dynamic Oracle: Example



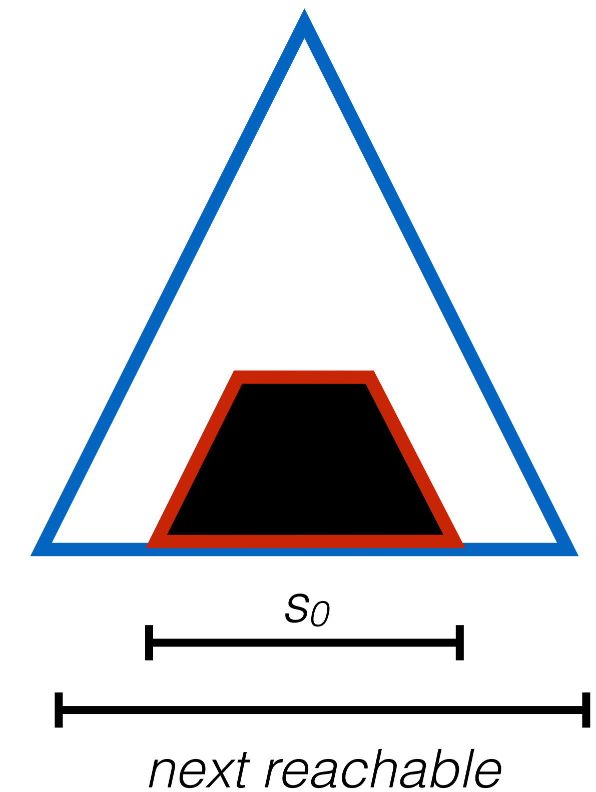
smallest reachable gold bracket incl. s_0



Dynamic Oracle: Example

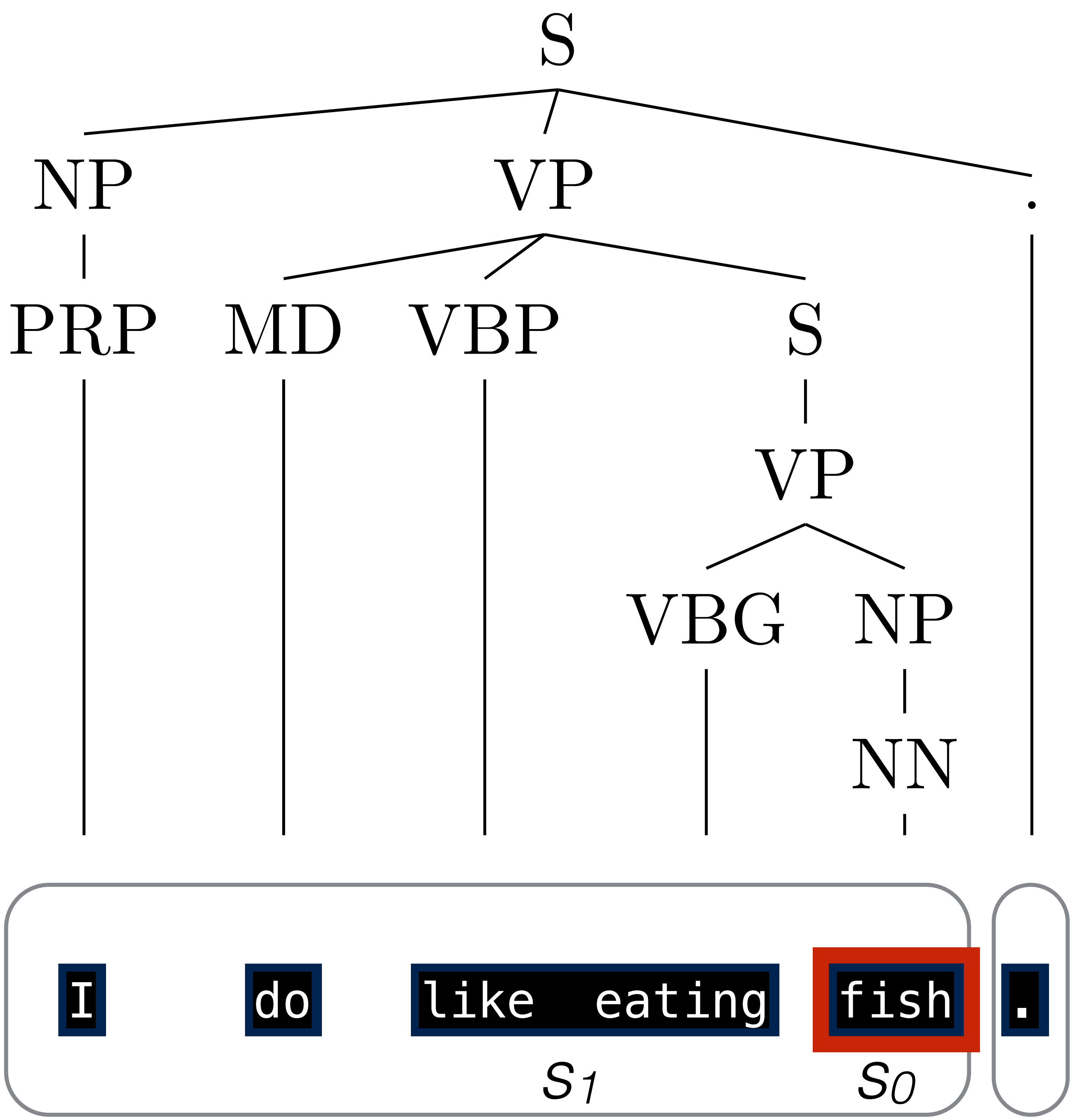


smallest reachable gold bracket incl. s_0

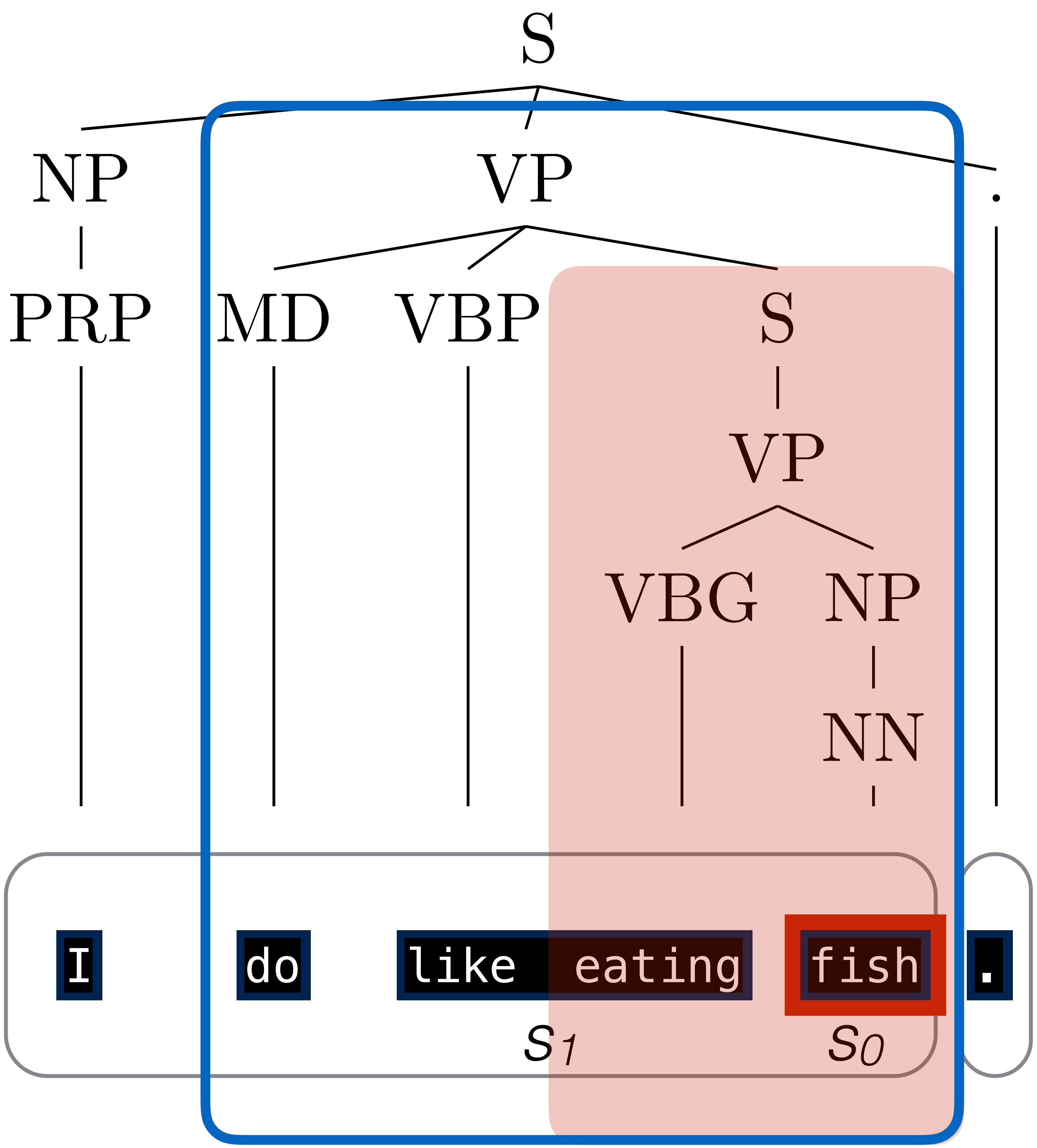


Dynamic Oracle: Shift or Combine

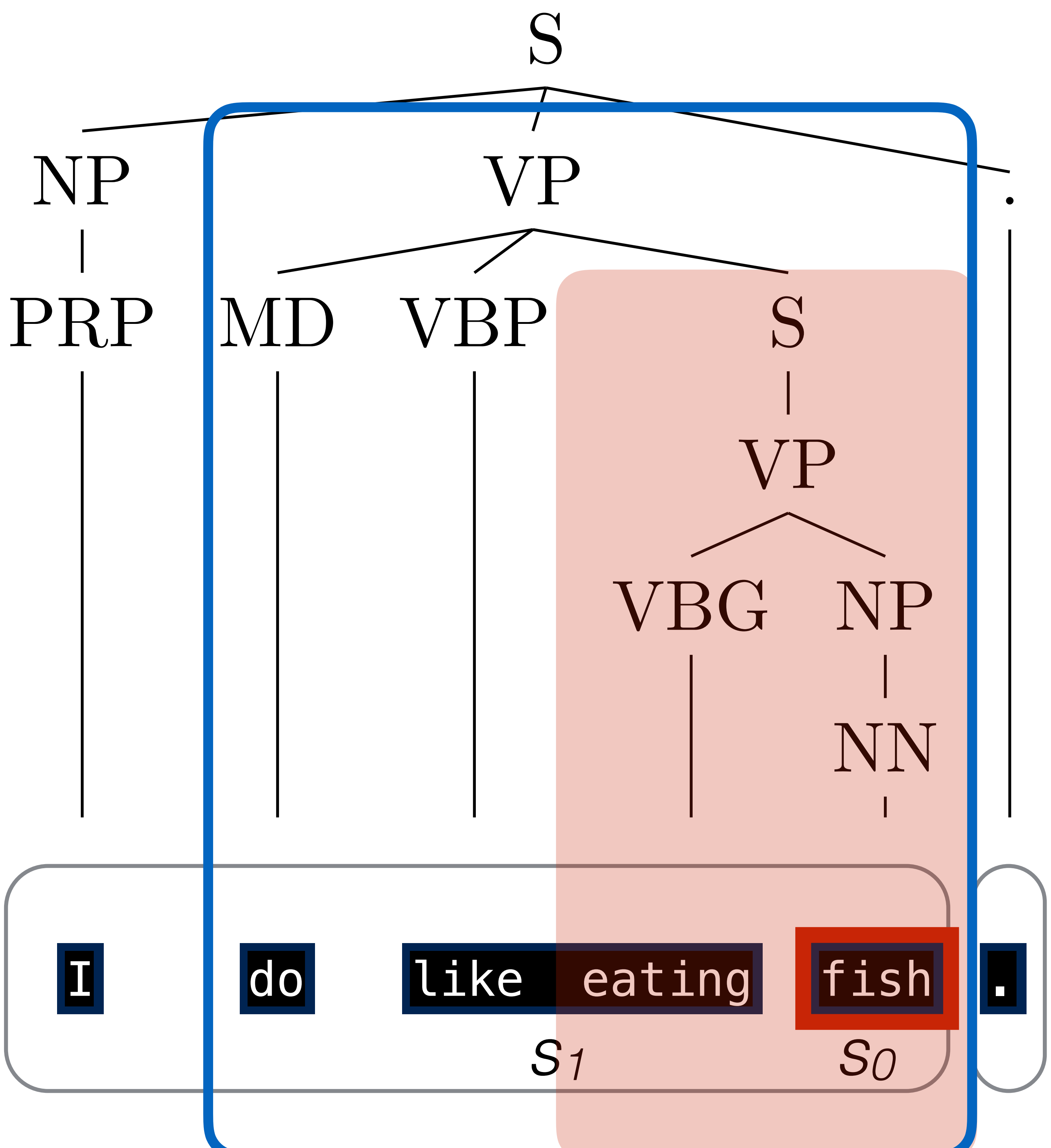
Dynamic Oracle: Example



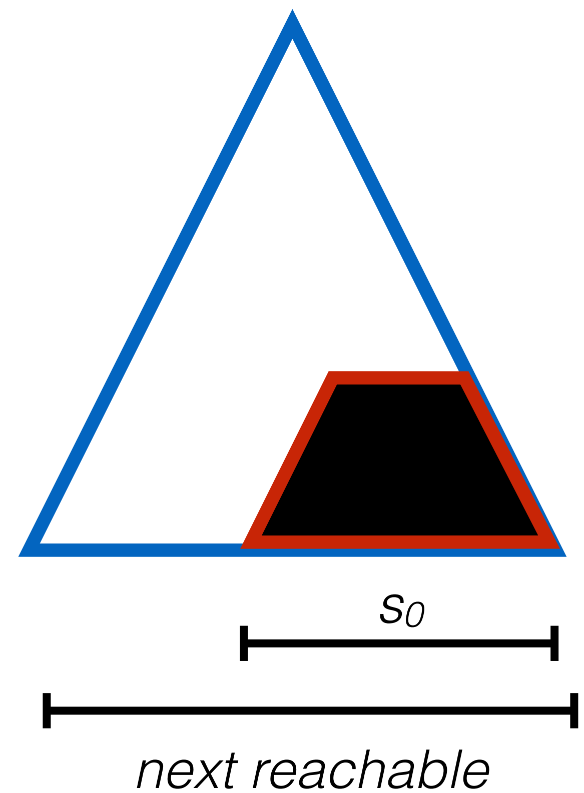
Dynamic Oracle: Example



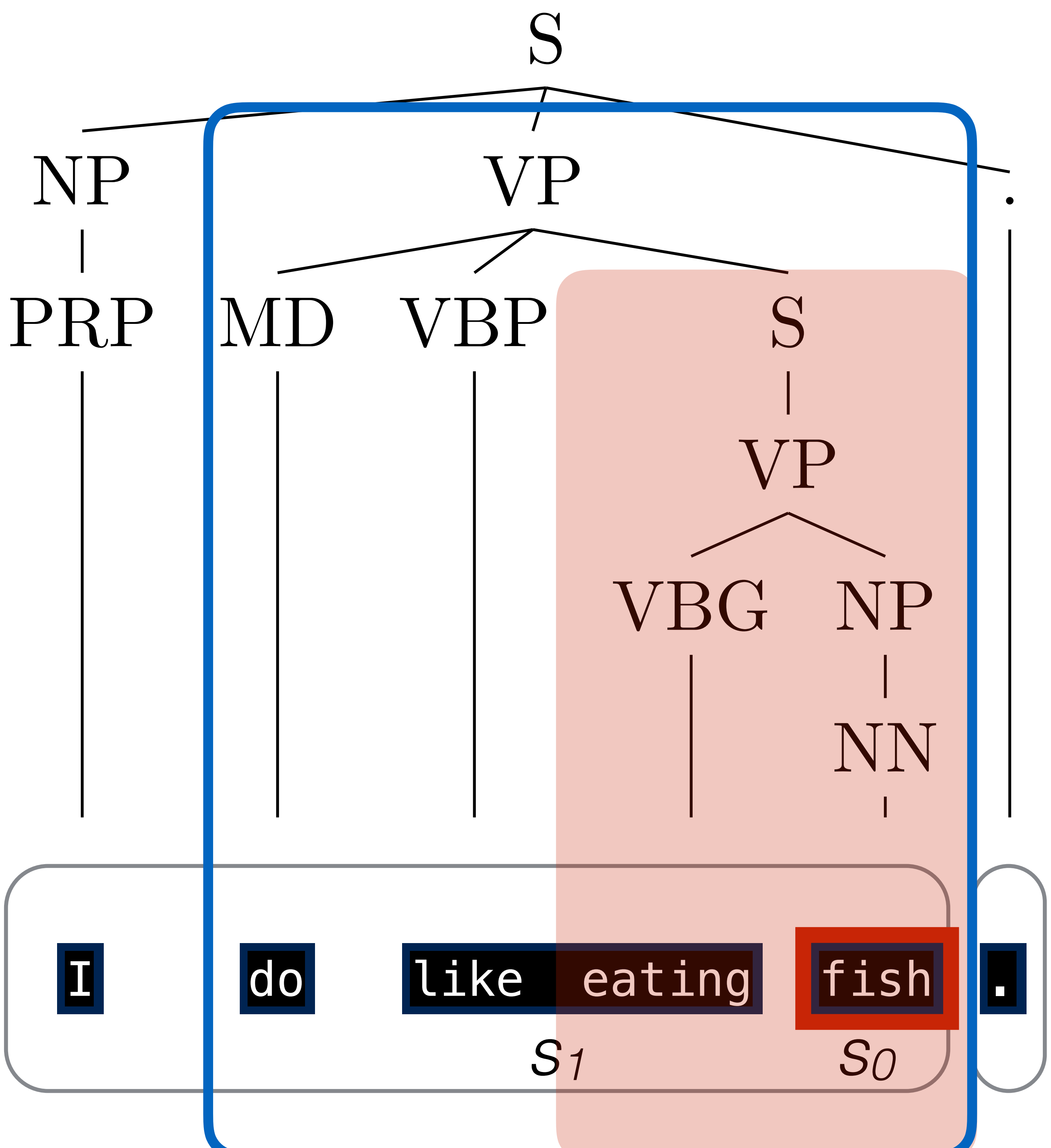
Dynamic Oracle: Example



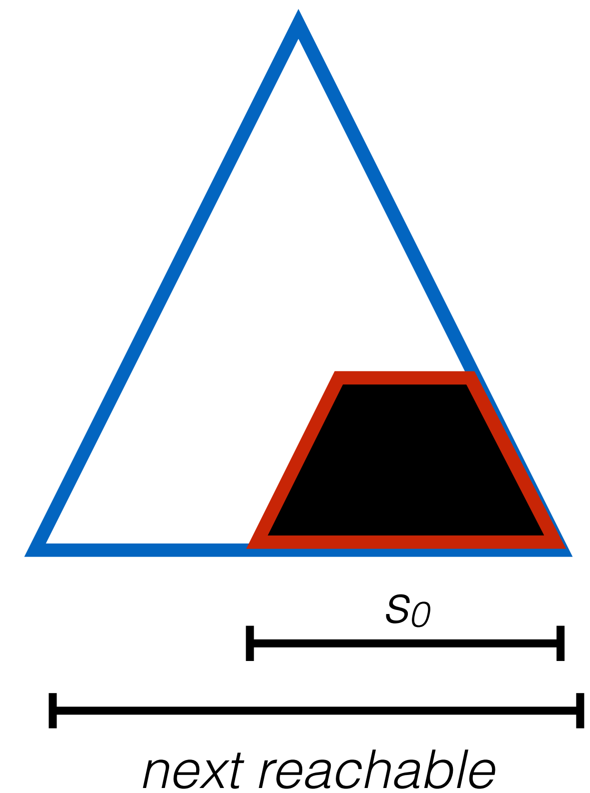
smallest reachable gold bracket incl. s_0



Dynamic Oracle: Example

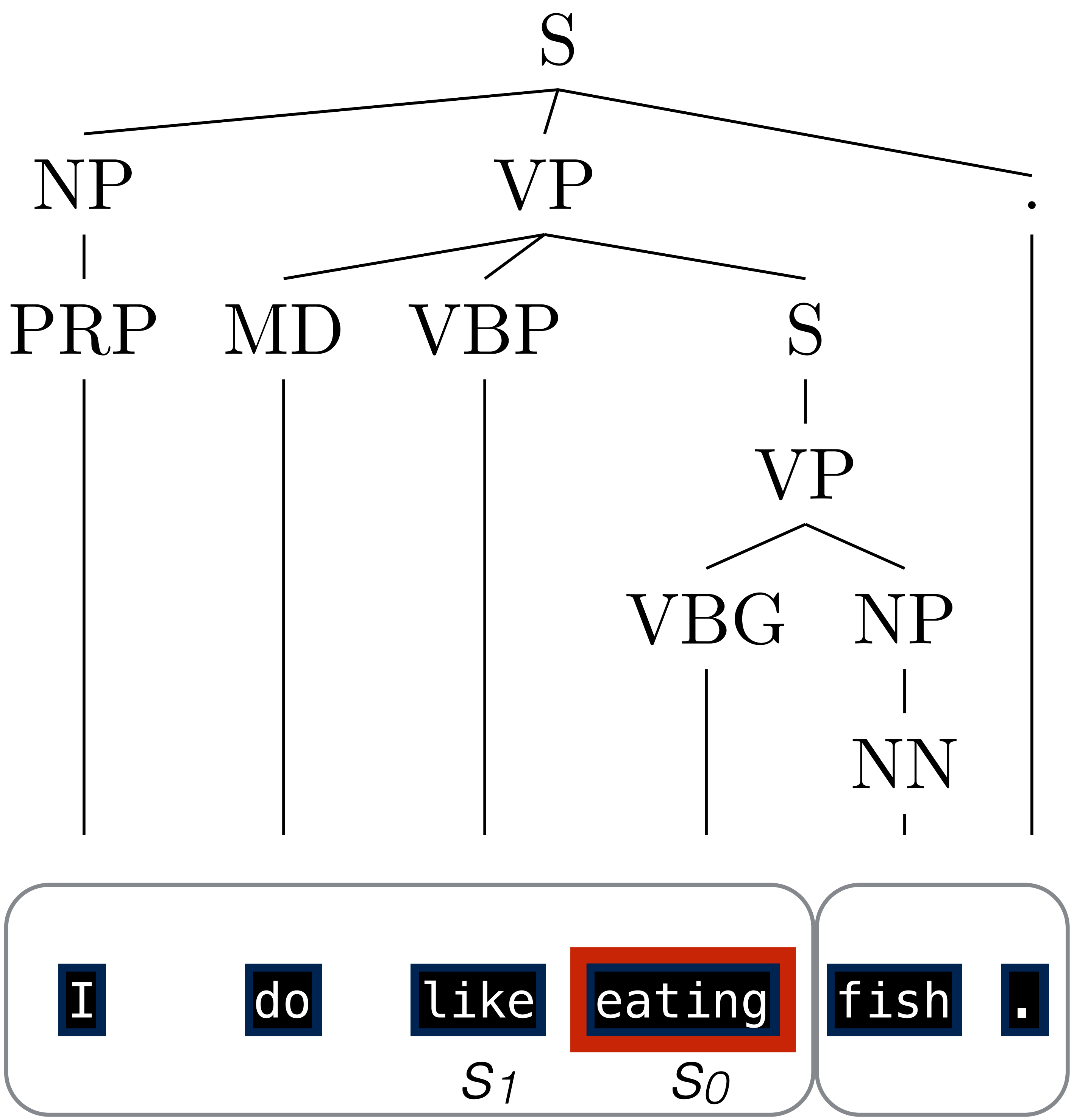


smallest reachable gold bracket incl. s_0

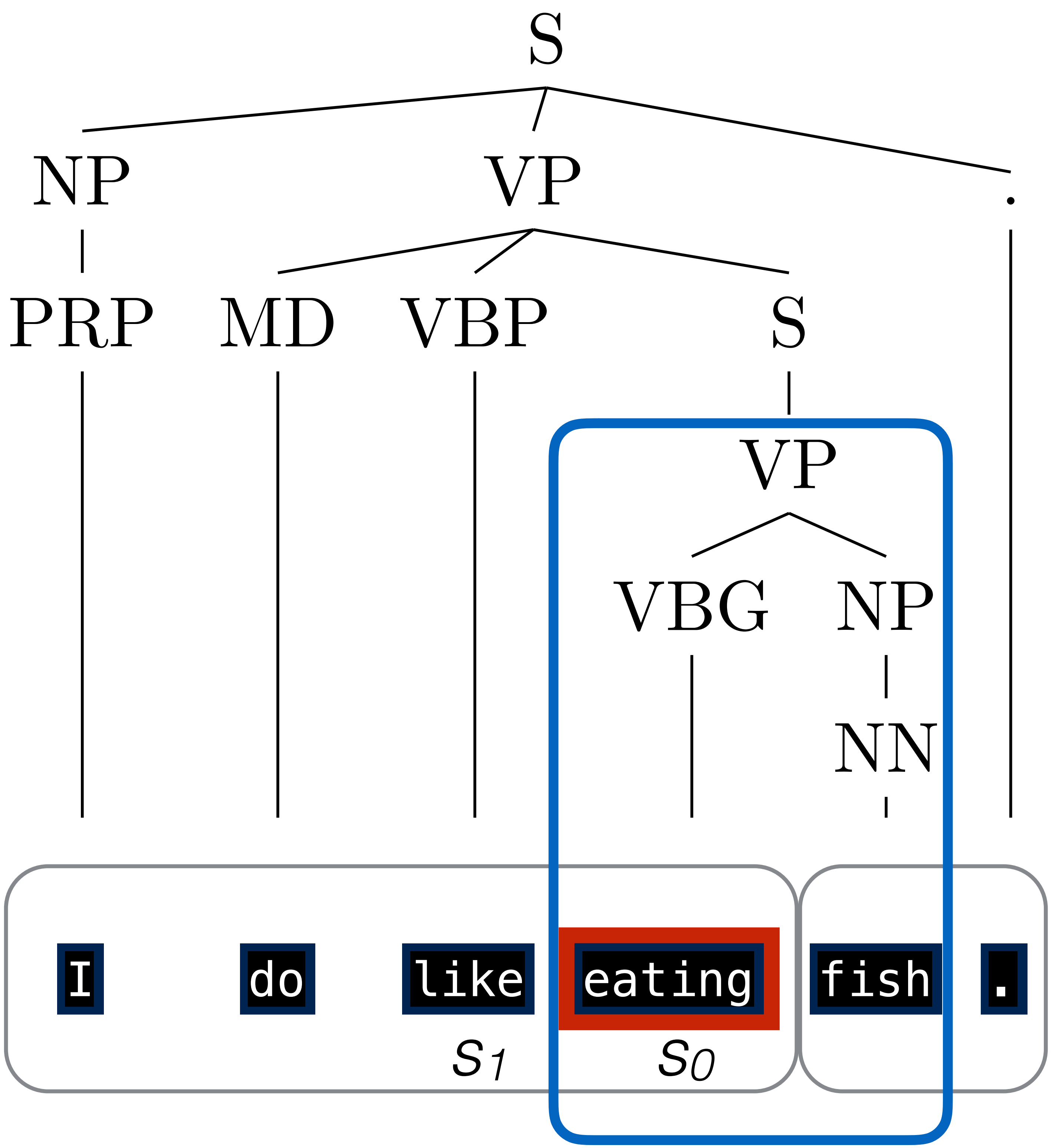


Dynamic Oracle: Combine

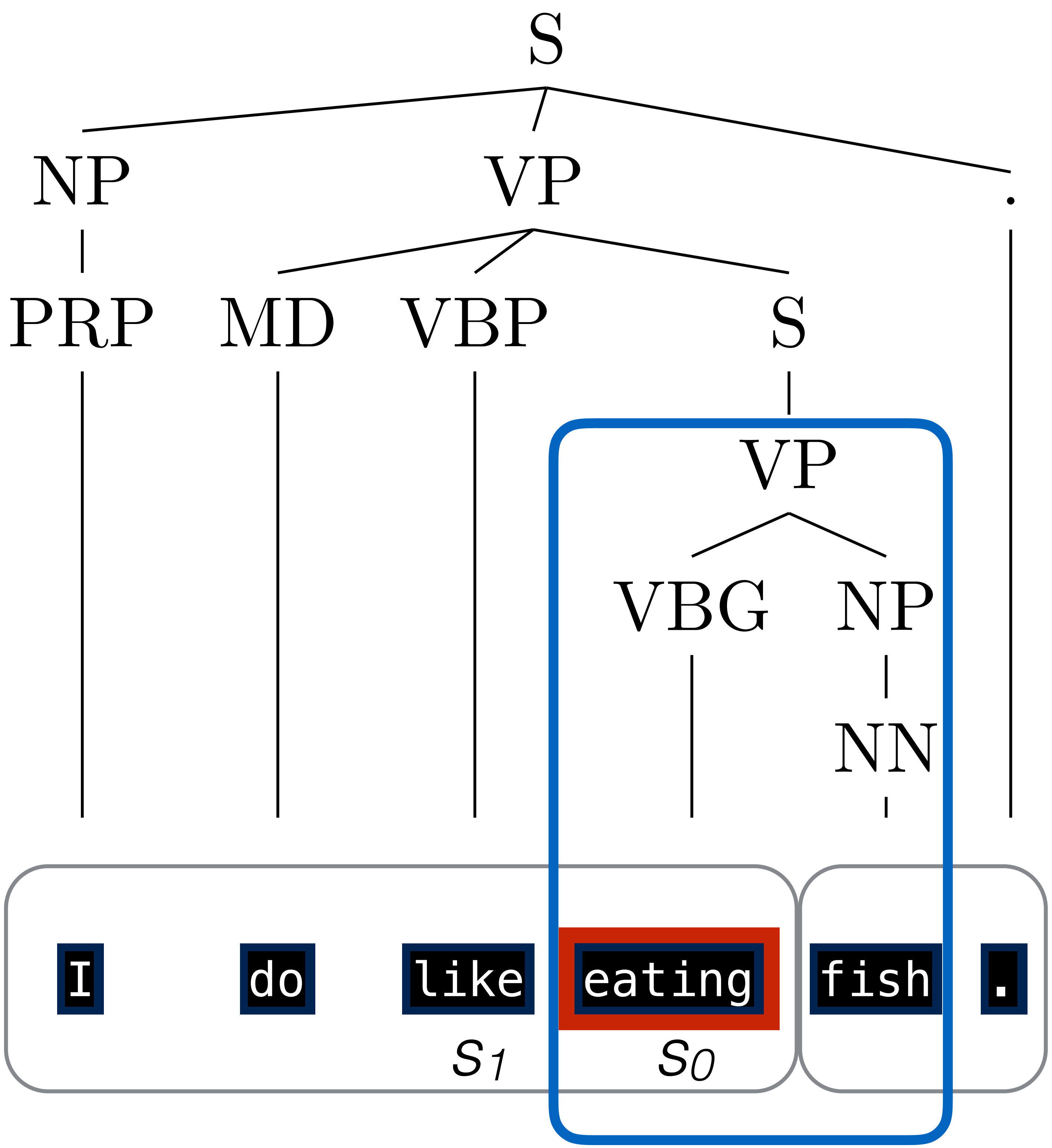
Dynamic Oracle: Example



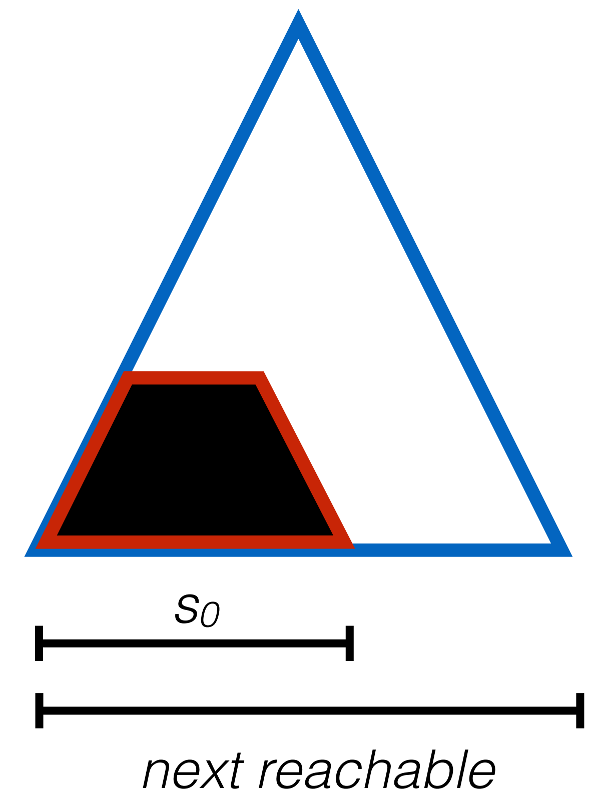
Dynamic Oracle: Example



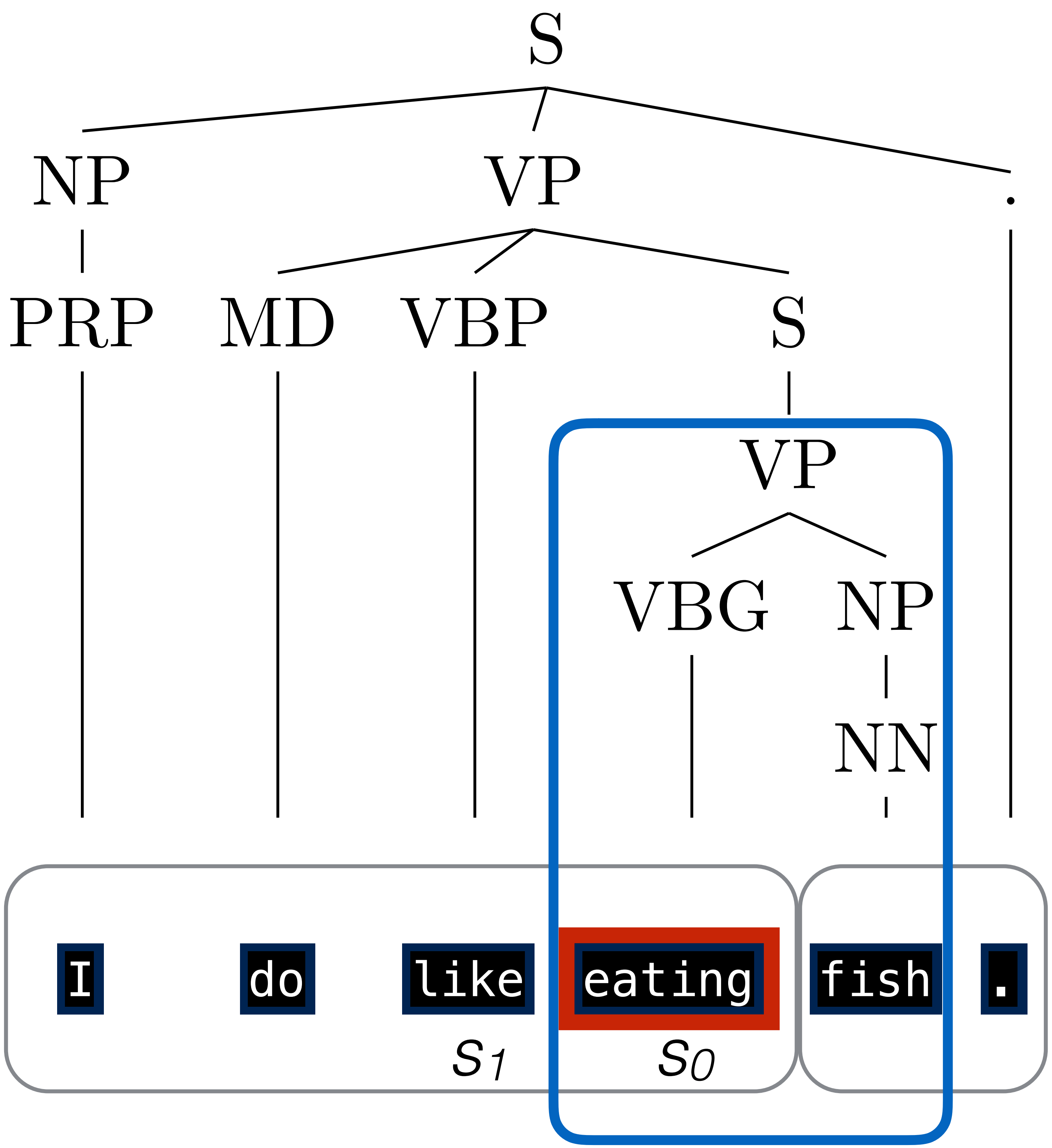
Dynamic Oracle: Example



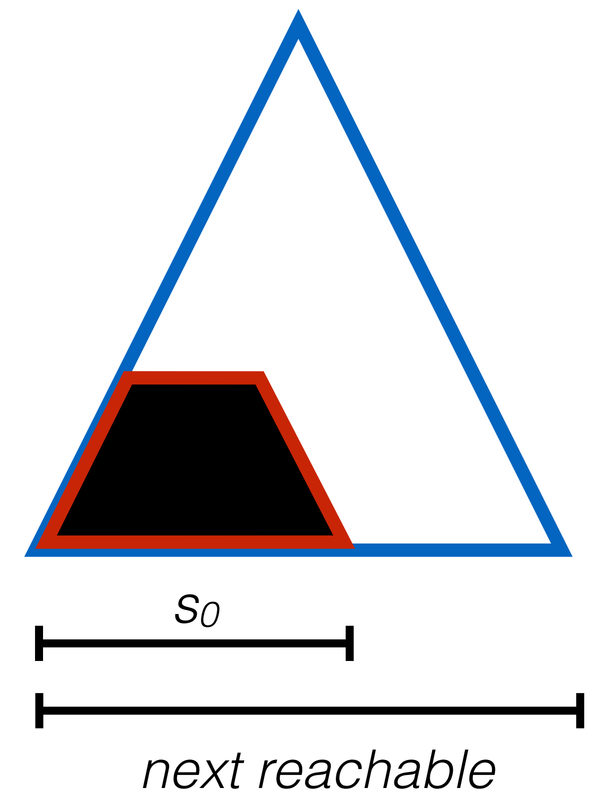
smallest reachable gold bracket incl. s_0



Dynamic Oracle: Example

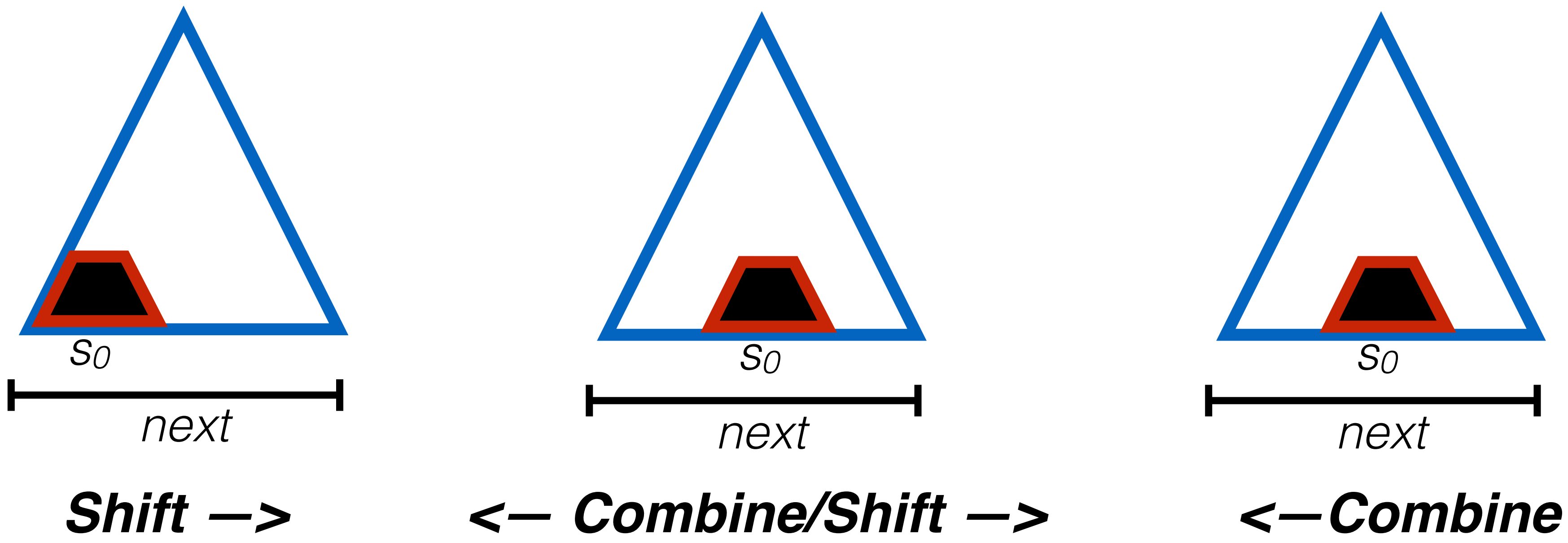


smallest reachable gold bracket incl. s_0



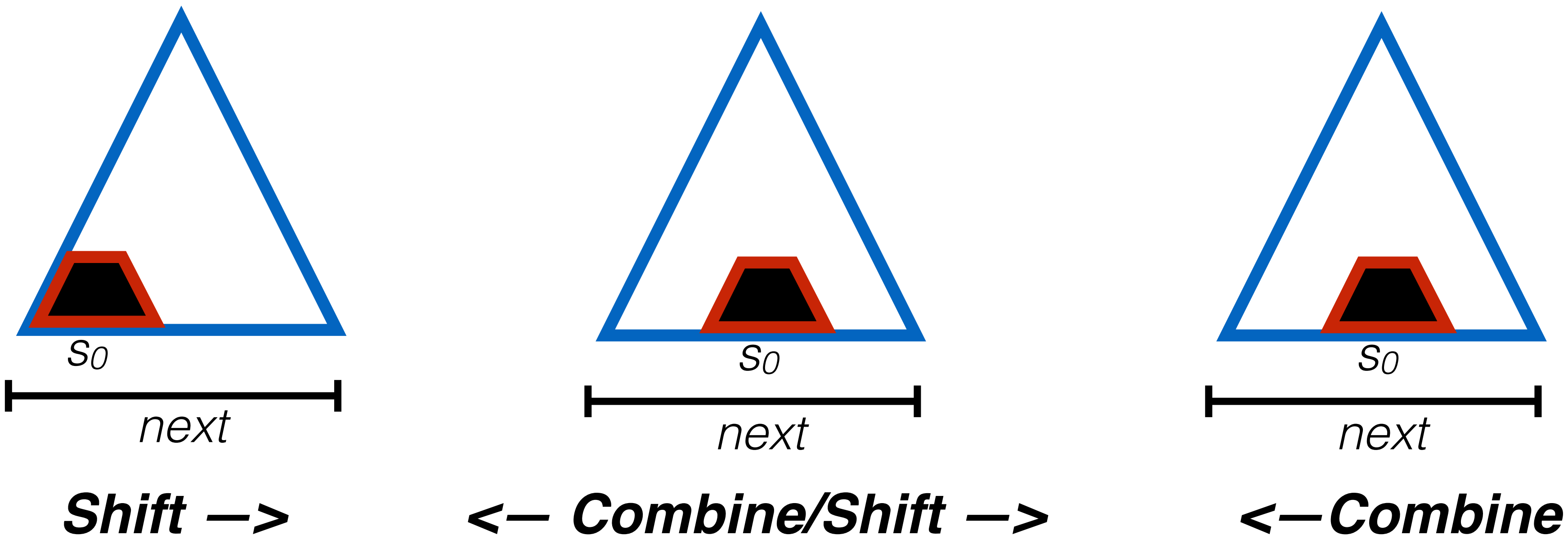
Dynamic Oracle: Shift

Dynamic Oracle: Full Definition

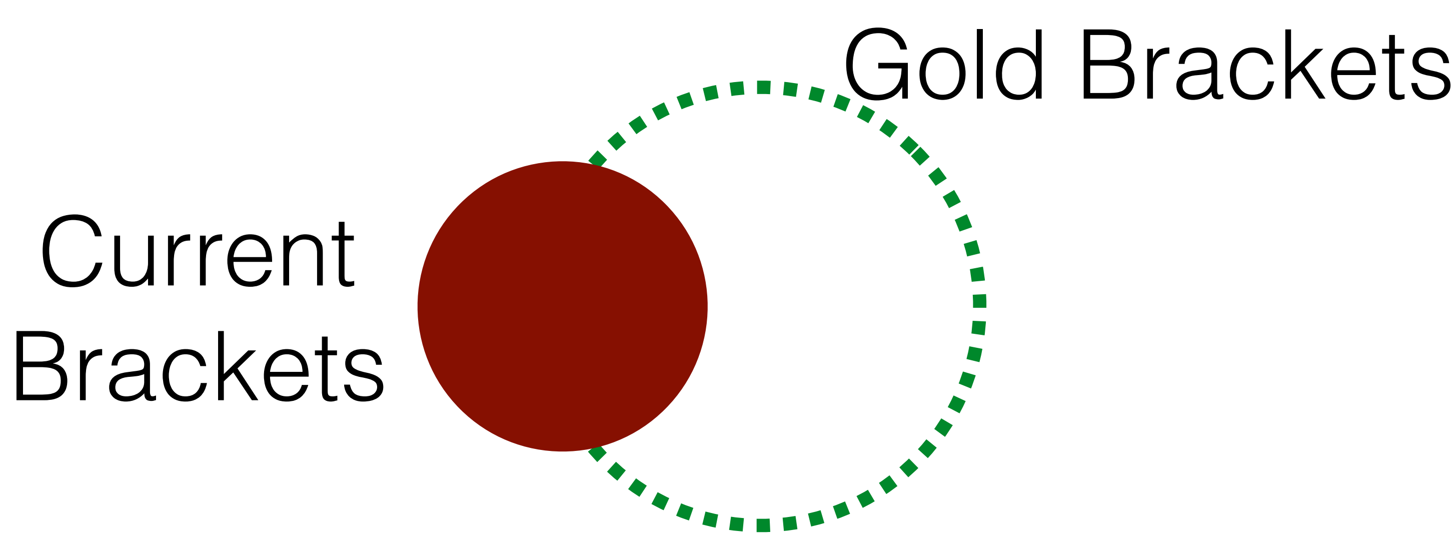


- Structure actions depend on next reachable bracket in gold tree
- All non-bracket label states —> *No-Label*
- All gold-bracket label states —> Correct label(s)

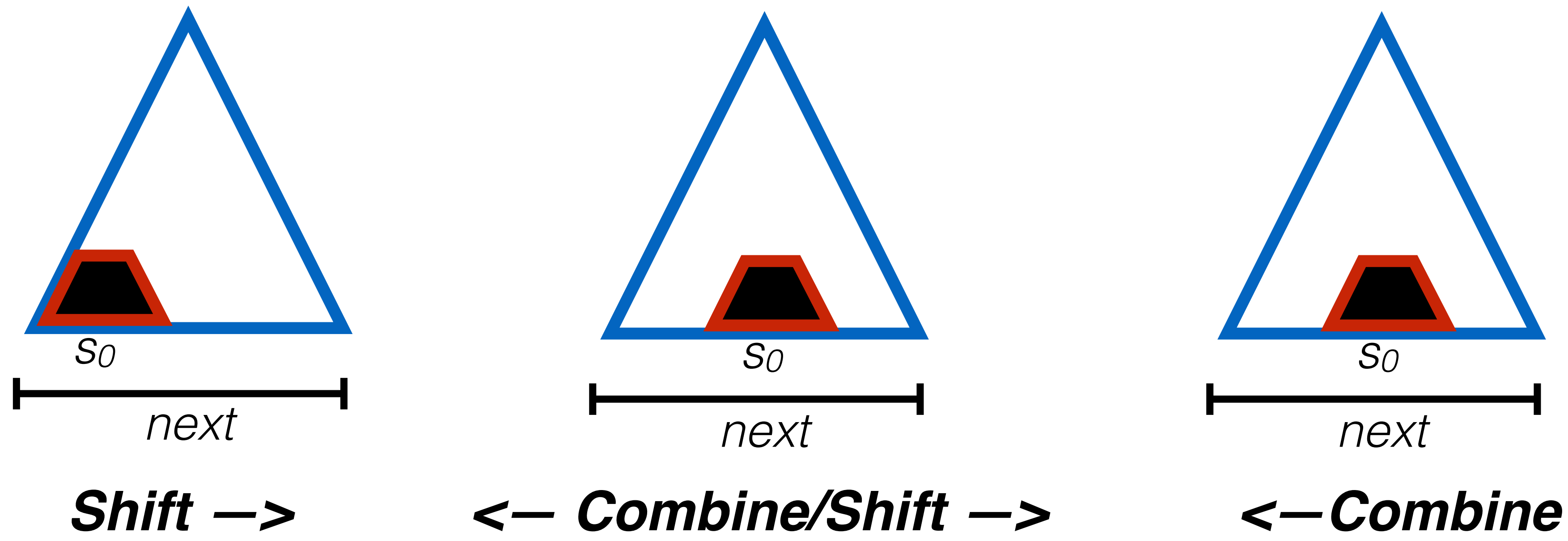
Dynamic Oracle: Full Definition



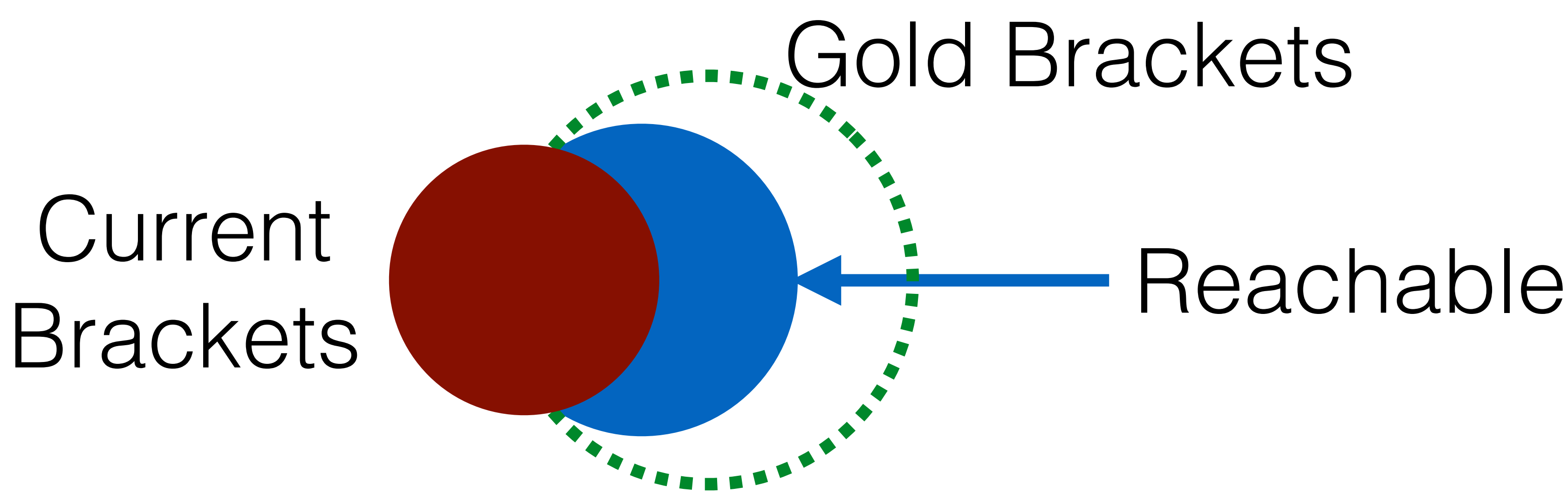
- Structure actions depend on next reachable bracket in gold tree
- All non-bracket label states —> *No-Label*
- All gold-bracket label states —> Correct label(s)



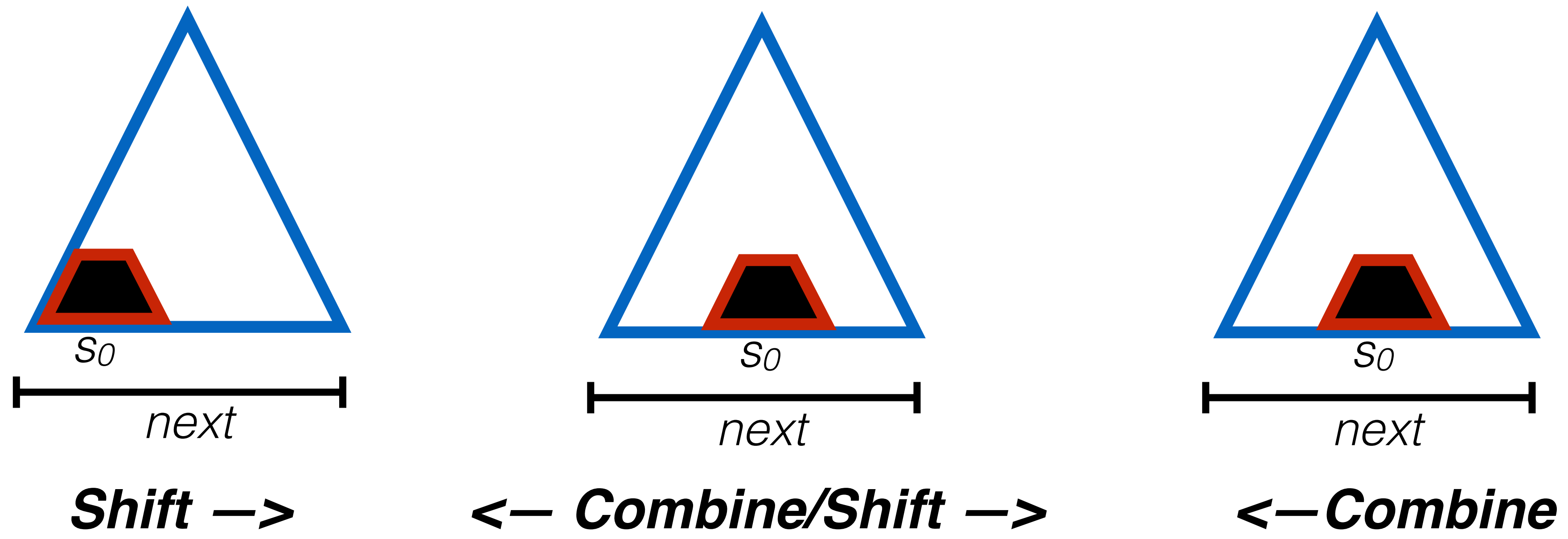
Dynamic Oracle: Full Definition



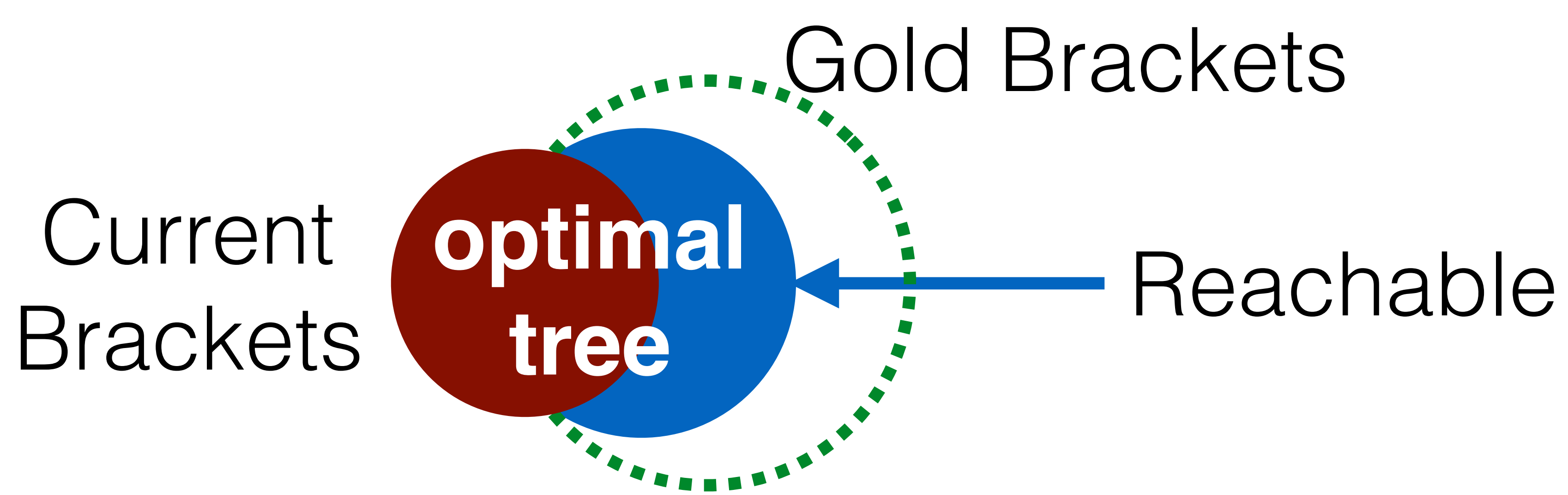
- Structure actions depend on next reachable bracket in gold tree
- All non-bracket label states —> *No-Label*
- All gold-bracket label states —> Correct label(s)



Dynamic Oracle: Full Definition

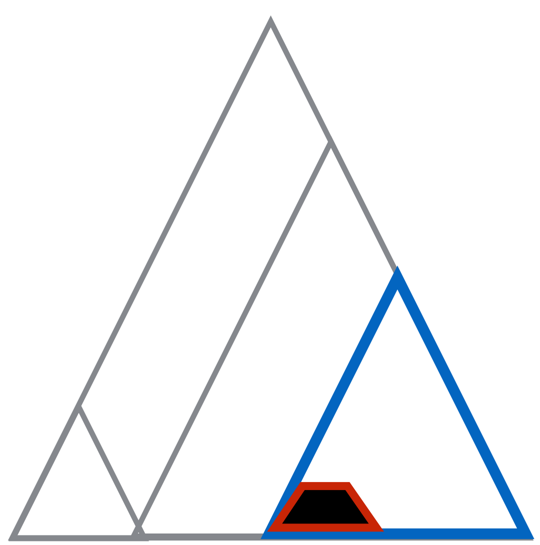


- Structure actions depend on next <u>reachable</u> bracket in gold tree
- All non-bracket label states —> *No-Label*
- All gold-bracket label states —> Correct label(s)



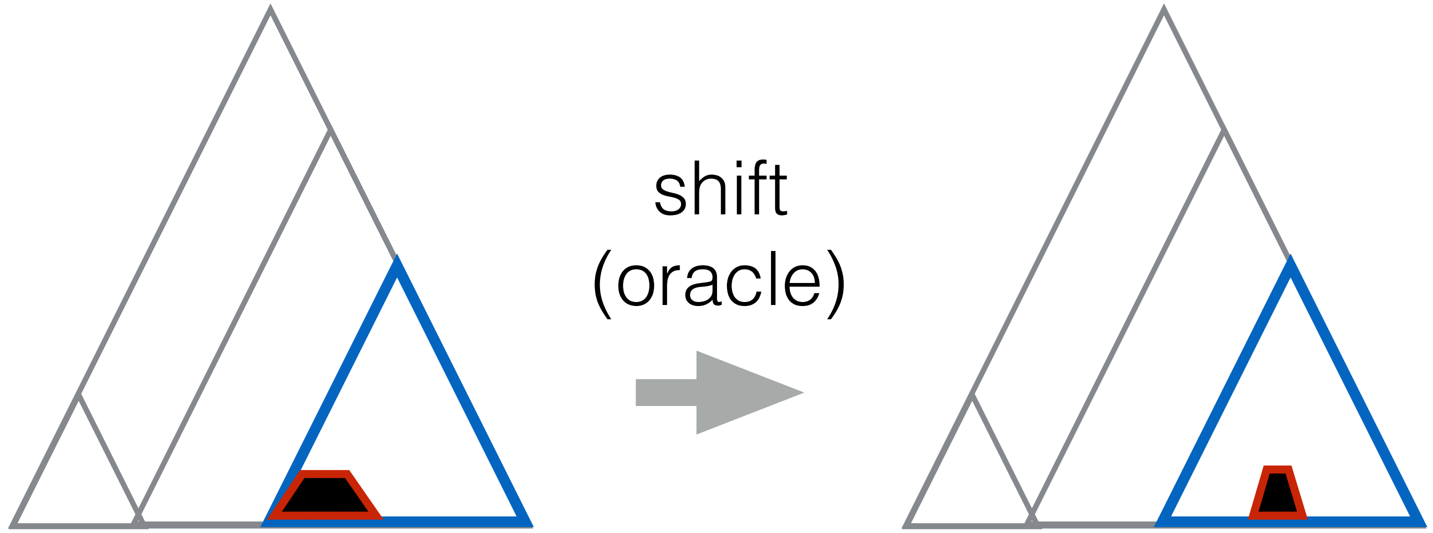
Dynamic Oracle: Optimality/Complexity

- First provably optimal oracle for constituency parsing (optimal in both precision and recall)
- After each action next reachable may (or may not) be updated by tracing parent link in gold tree
- Also $O(n)$ steps, thus amortized $O(1)$ time
- Dependency parsing oracle (arc-std): worst case $O(n^3)$ per step



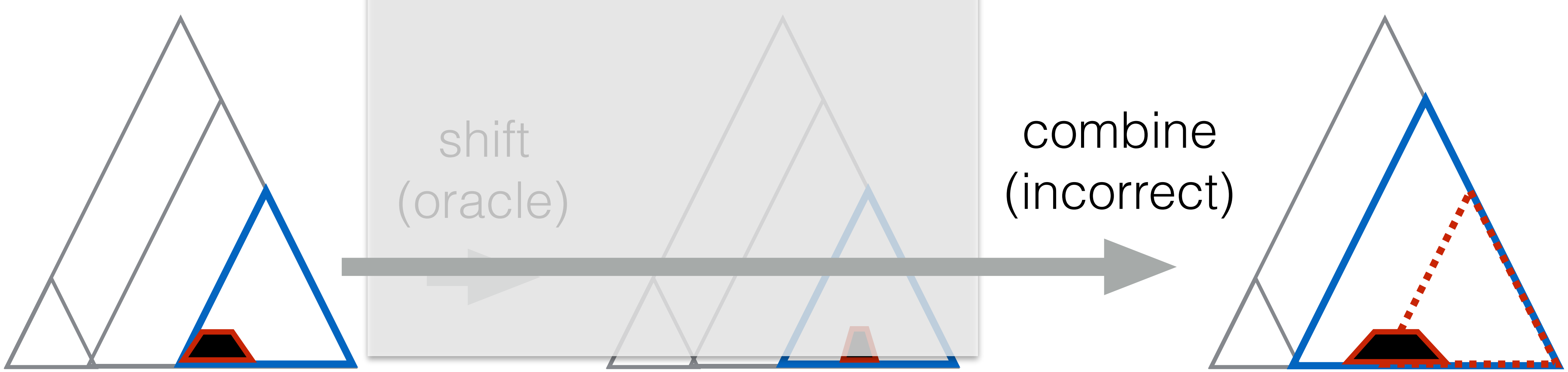
Dynamic Oracle: Optimality/Complexity

- First provably optimal oracle for constituency parsing (optimal in both precision and recall)
- After each action next reachable may (or may not) be updated by tracing parent link in gold tree
- Also $O(n)$ steps, thus amortized $O(1)$ time
- Dependency parsing oracle (arc-std): worst case $O(n^3)$ per step

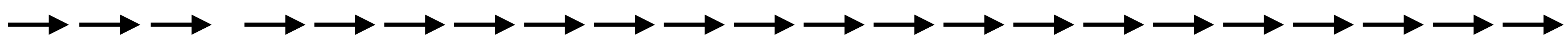


Dynamic Oracle: Optimality/Complexity

- First provably optimal oracle for constituency parsing (optimal in both precision and recall)
- After each action next reachable may (or may not) be updated by tracing parent link in gold tree
- Also $O(n)$ steps, thus amortized $O(1)$ time
- Dependency parsing oracle (arc-std): worst case $O(n^3)$ per step



Training with Dynamic Oracle

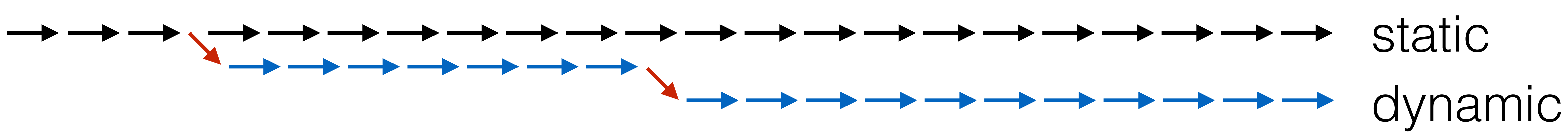


static

(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38

Training with Dynamic Oracle

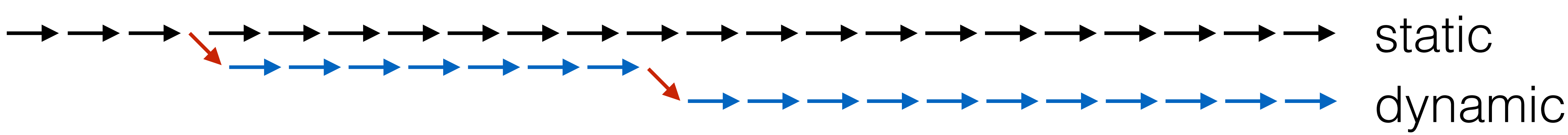
- Basic dynamic oracle: follow current model



(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38

Training with Dynamic Oracle

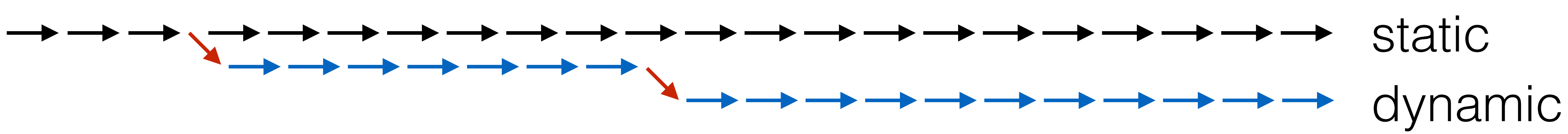
- Basic dynamic oracle: follow current model



(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38
Dynamic Oracle	91.14	91.61	91.38

Training with Dynamic Oracle

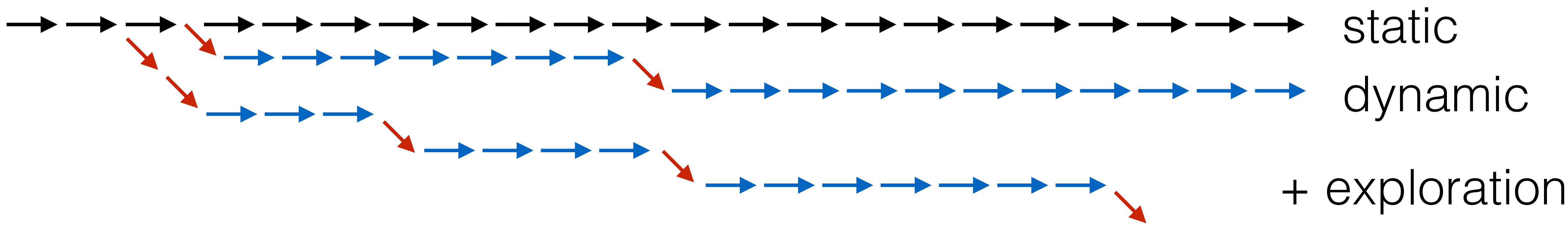
- Basic dynamic oracle: follow current model
- Problem: overfits training data, making fewer mistakes than test



(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38
Dynamic Oracle	91.14	91.61	91.38

Training with Dynamic Oracle

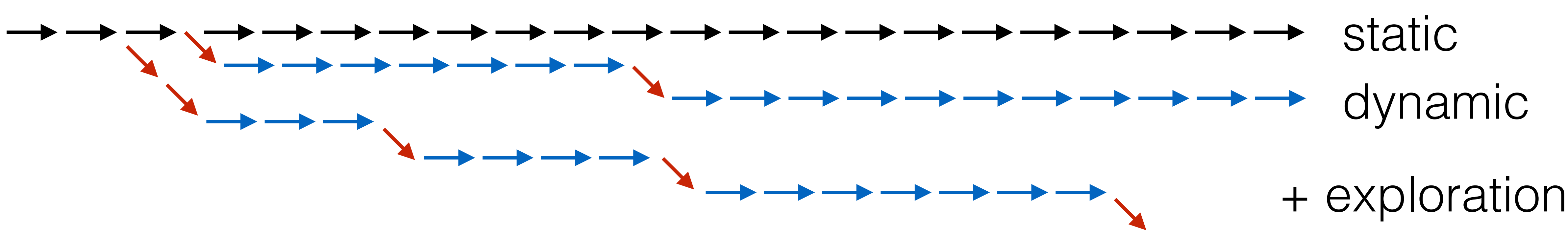
- Basic dynamic oracle: follow current model
- Problem: overfits training data, making fewer mistakes than test
- Exploration: sample from softmax distribution (Ballesteros et al., 2016) to encourage more mistakes



(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38
Dynamic Oracle	91.14	91.61	91.38

Training with Dynamic Oracle

- Basic dynamic oracle: follow current model
- Problem: overfits training data, making fewer mistakes than test
- Exploration: sample from softmax distribution (Ballesteros et al., 2016) to encourage more mistakes

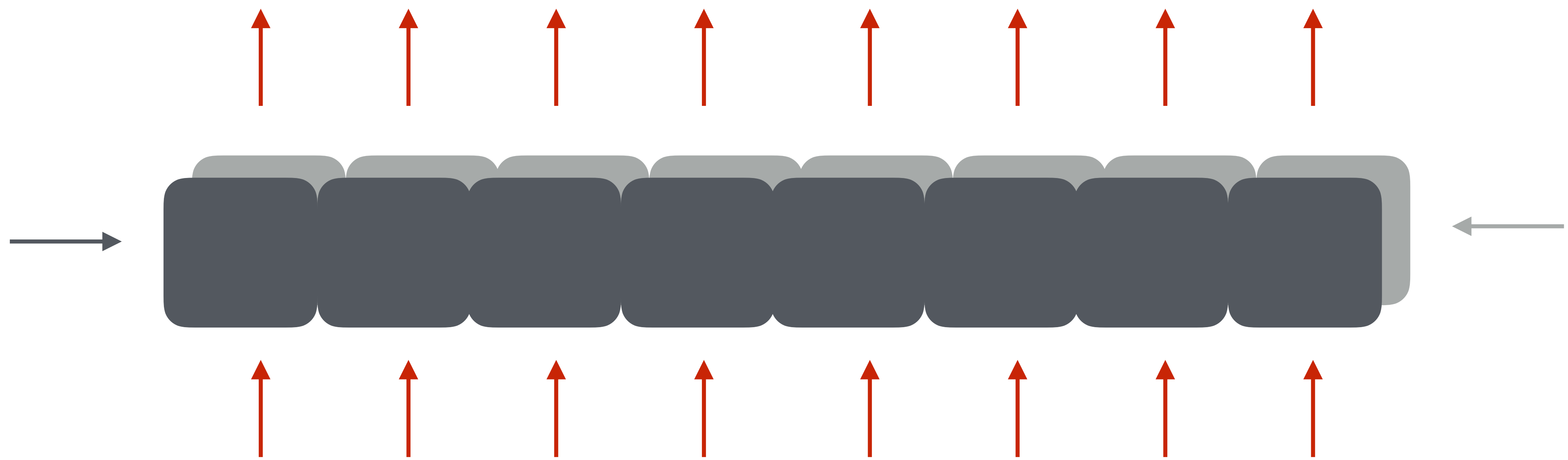


(scores on PTB 22)	Recall	Prec.	F_1
Static Oracle	91.34	91.43	91.38
Dynamic Oracle	91.14	91.61	91.38
Dynamic + Exploration	91.07	92.22	**91.64**

Outline

- Span-Based Constituency Parsing
- Bi-Directional LSTM Span Features
- Provably Optimal Dynamic Oracle
- **Experiments**

Architecture



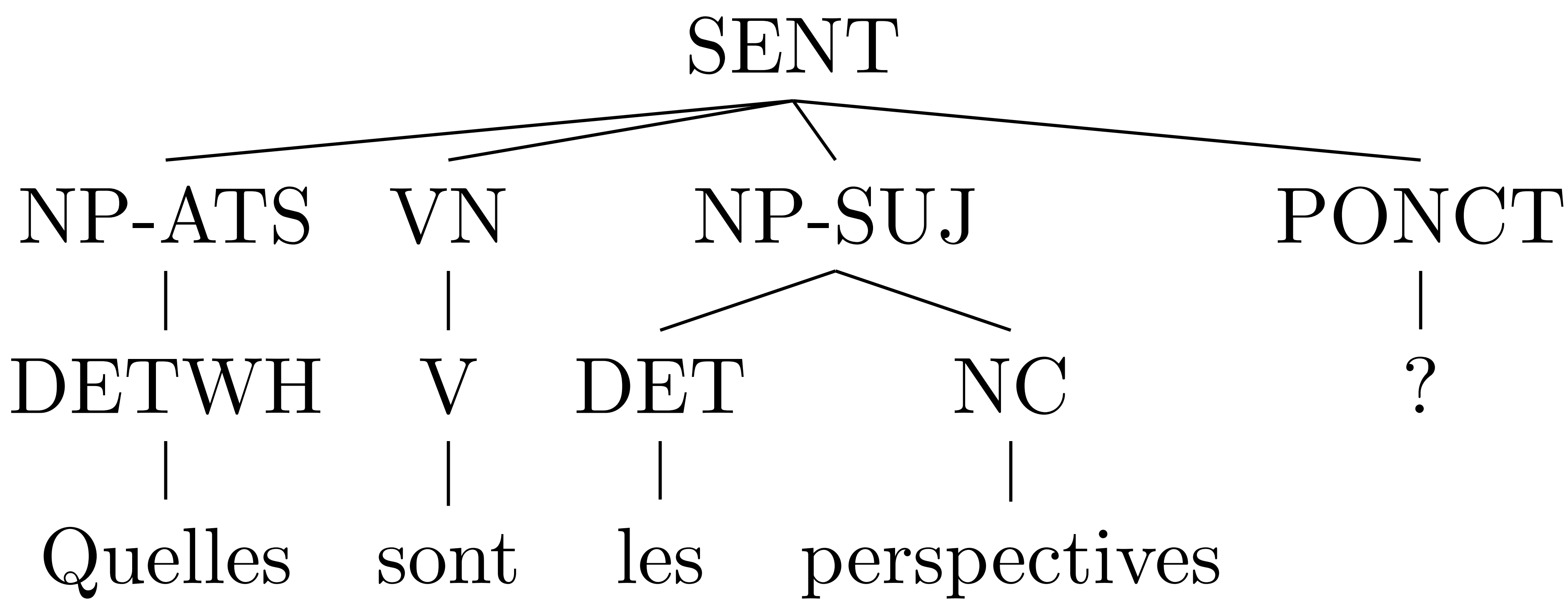
- 50-dim word and 20-dim tag embeddings
- No pre-training
- Each LSTM layer 200 units each direction
- 200 ReLU units for each of structure and label predictors

Results on Penn Treebank

Parser	Search	Recall	Prec.	F_1
Carreras et al. (2008)	cubic	90.7	91.4	91.1
Shindo et al. (2012)	cubic			91.1
Thang et al. (2015)	~cubic			91.1
Watanabe et al. (2015)	beam			90.7
Static Oracle	**greedy**	90.7	91.4	91.0
Dynamic + Exploration	**greedy**	90.5	92.1	**91.3**

- State of the art despite: simple system with greedy actions and small embeddings trained from scratch

Parsing Morphologically Rich Languages



lemma = perspective
coarse_POS = N
gender = feminine
number = plural
subcategory = common

Results on French Treebank

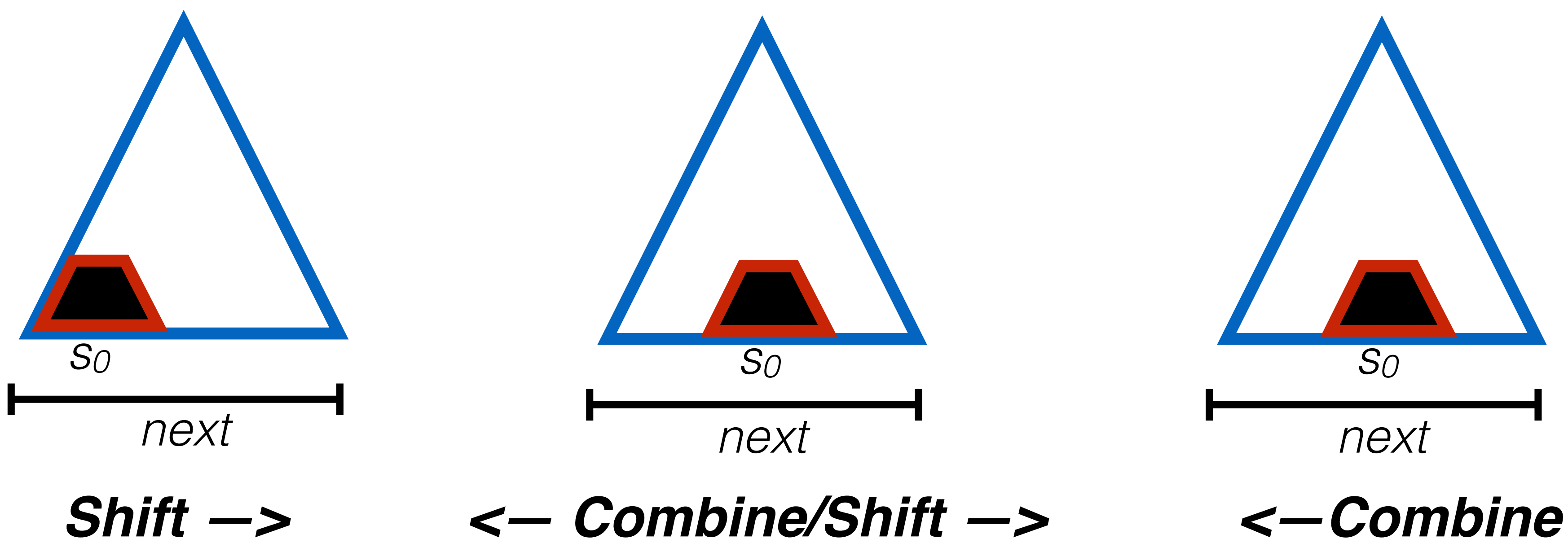
- Morphological feature embeddings (10 dim. each)
- Additional input to recurrent network
- For French, we used SPMRL 2014 predicted features

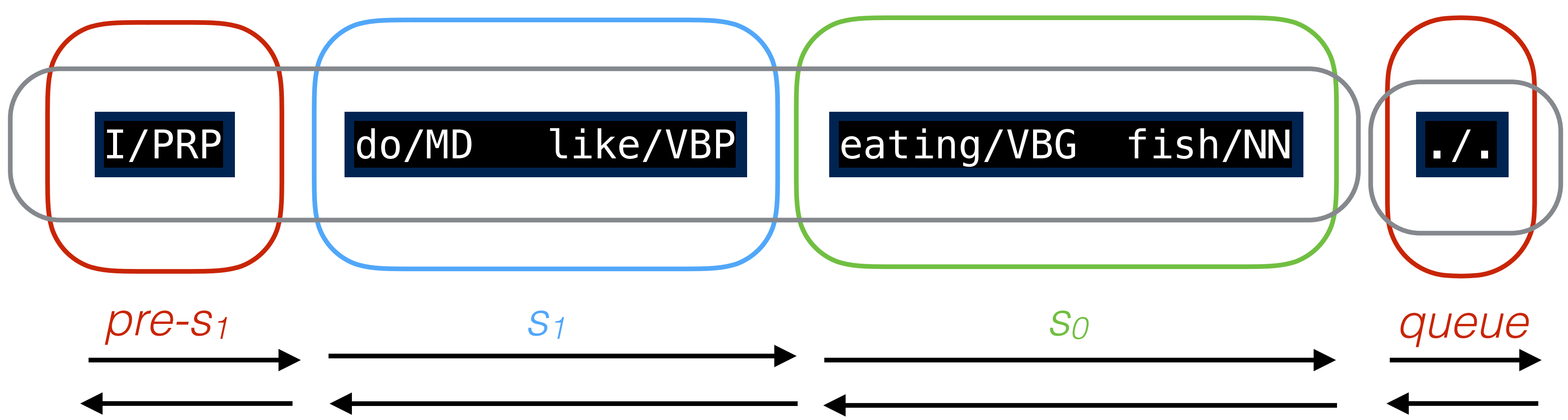
Parser	Recall	Prec.	F_1
Björkelund et al. (2014)			82.53
Static Oracle	83.50	82.87	83.18
Dynamic + Exploration	81.90	84.77	**83.31**

Summary

- Simple, easy-to-implement span-based parsing system
- No tree/label information in features (good candidate for dynamic programming)
- Linear time parsing with greedy decoding
- No pre-trained embeddings, small architecture, and minimal hyper-parameter tuning (trained on CPU)
- First optimal dynamic oracle for constituency parsing

Thank You!


s_0
next
Shift —>
s_0
next
<— Combine/Shift —>
s_0
next
<—Combine


I/PRP
do/MD like/VBP
eating/VBG fish/NN
./.
pre-s_1
s_1
s_0
queue