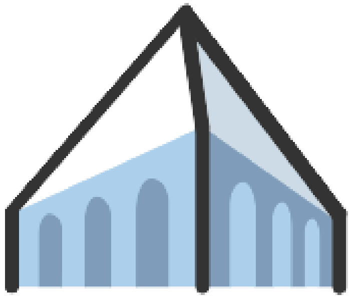
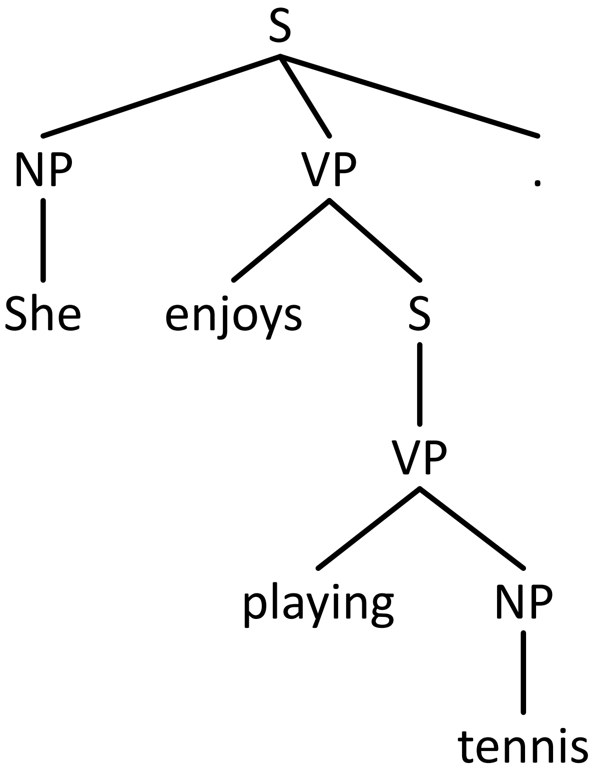
A Minimal Span-Based Neural Constituency Parser

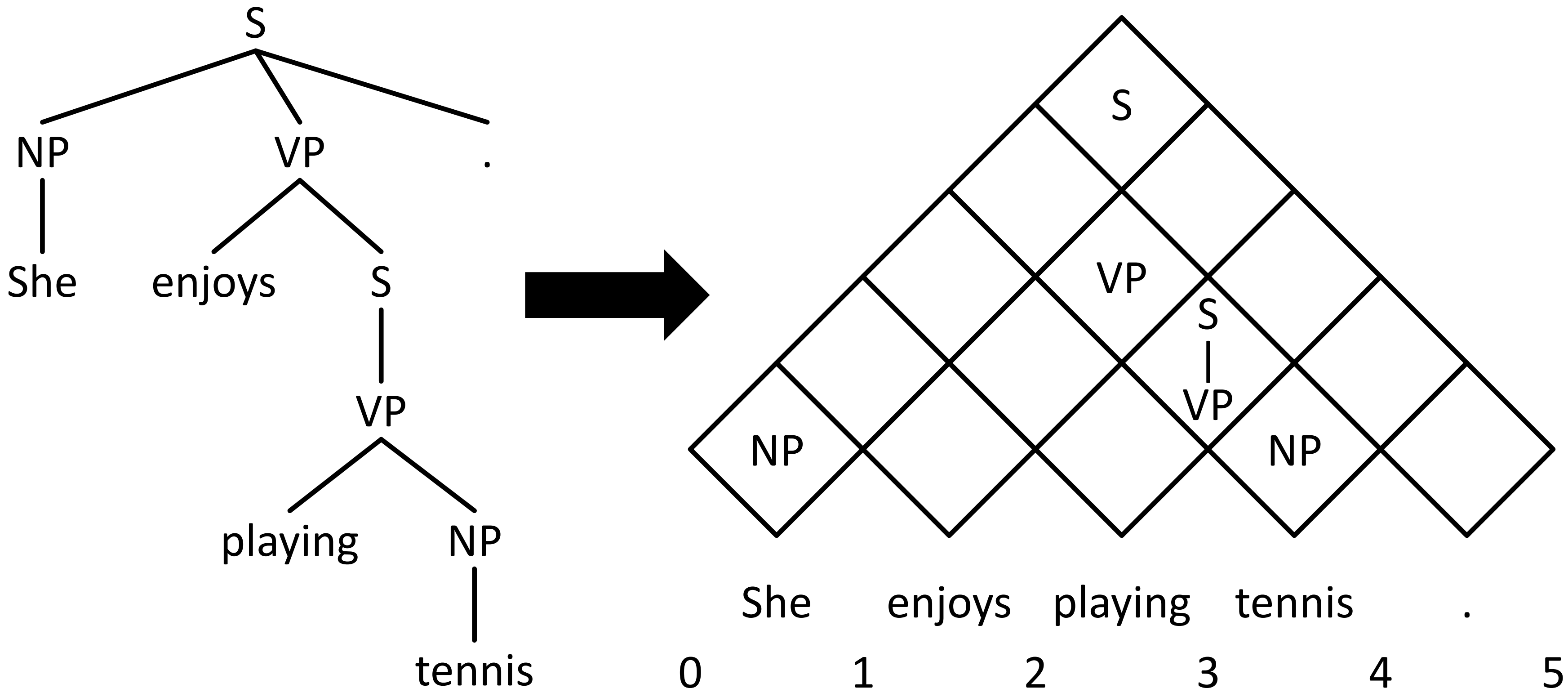


Mitchell Stern, Jacob Andreas, Dan Klein
UC Berkeley

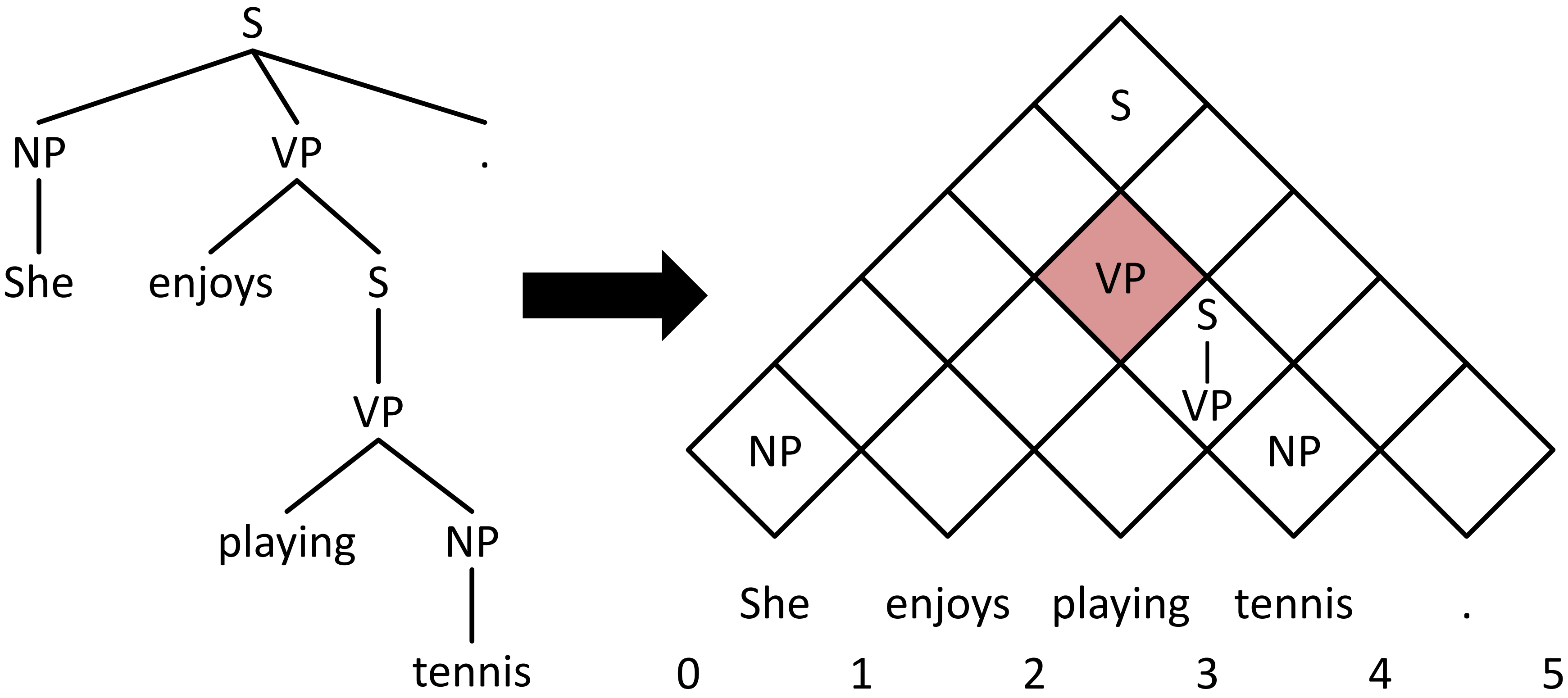
Parsing as Span Classification



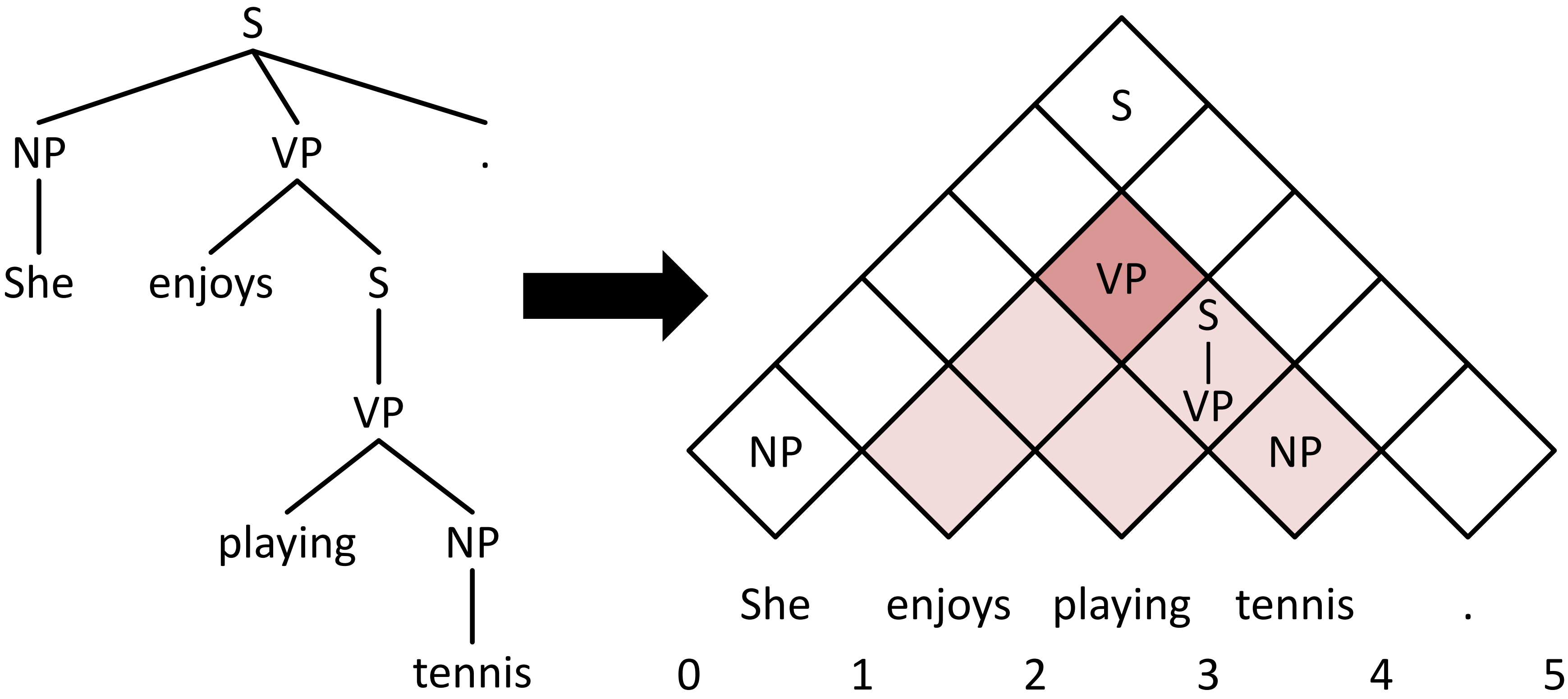
Parsing as Span Classification



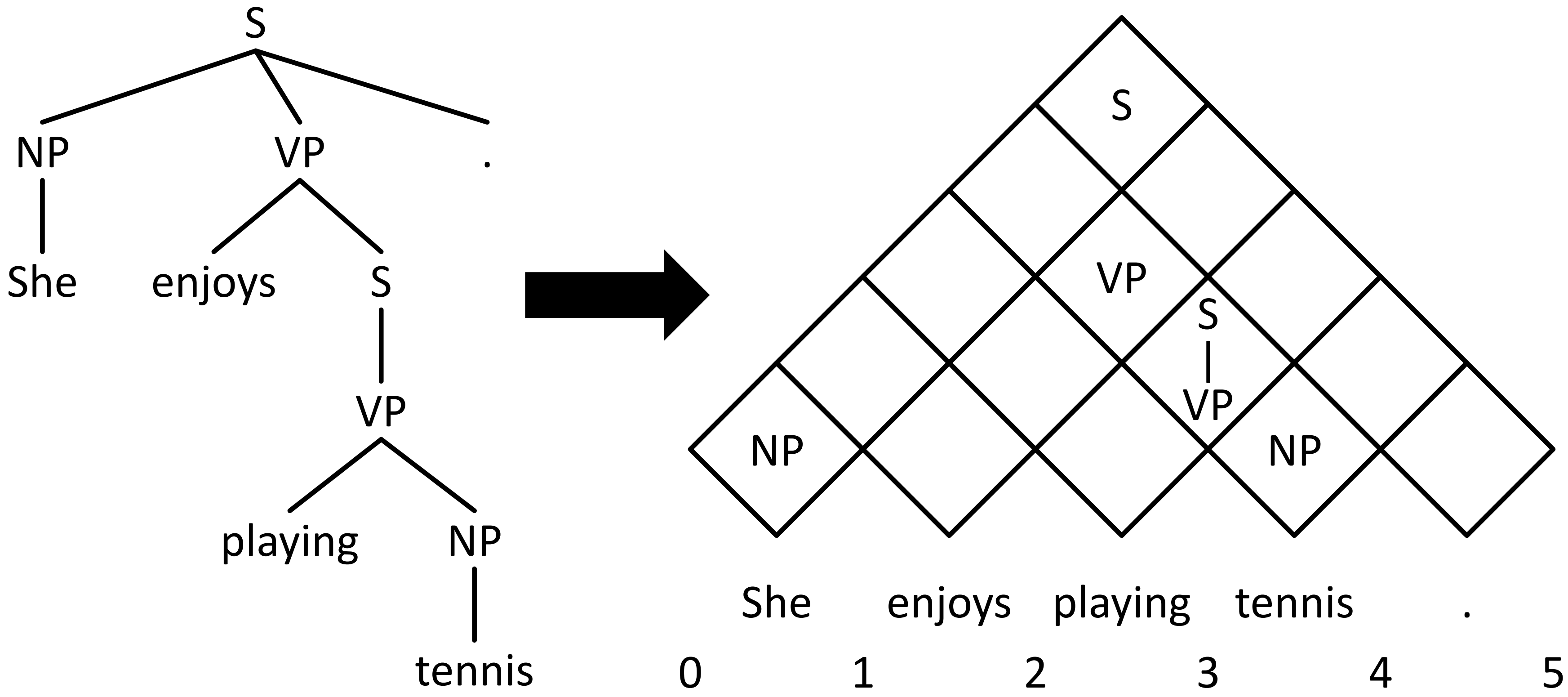
Parsing as Span Classification



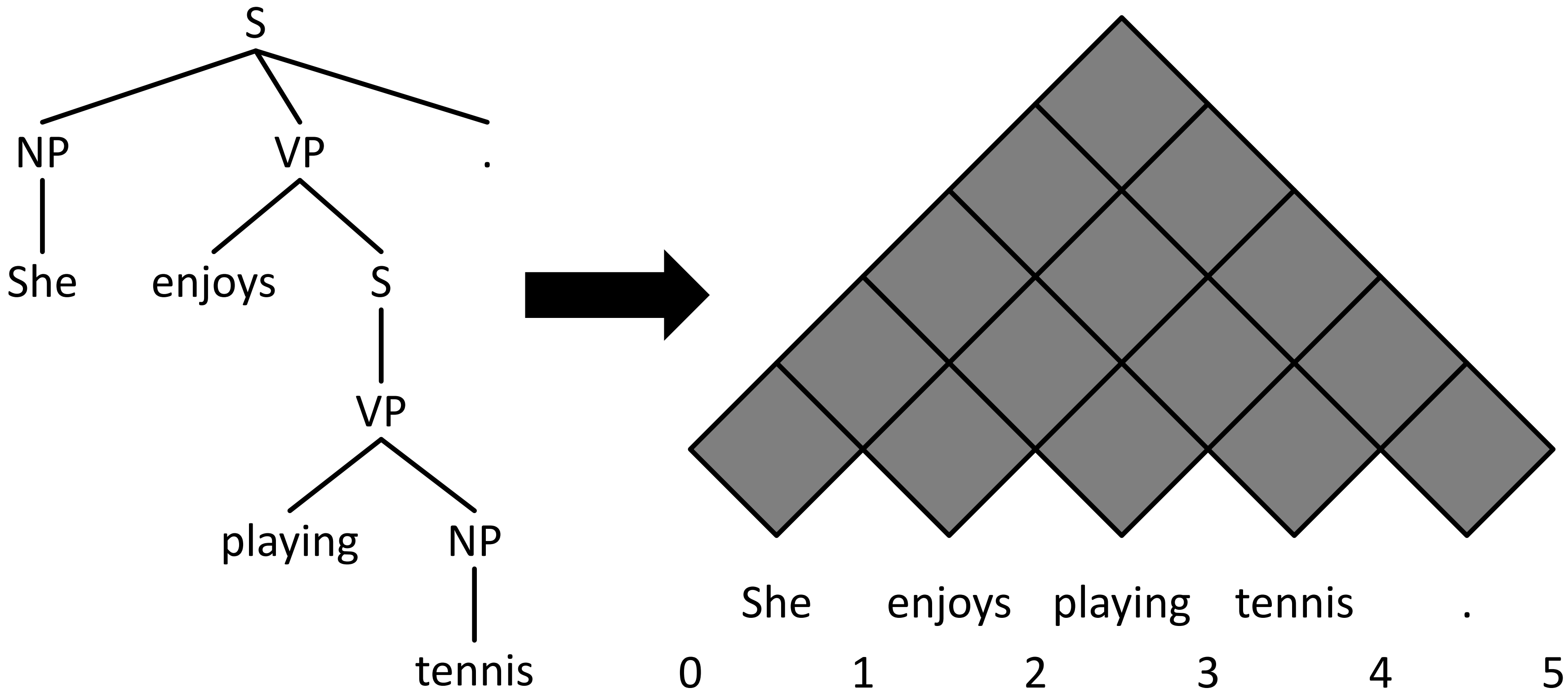
Parsing as Span Classification



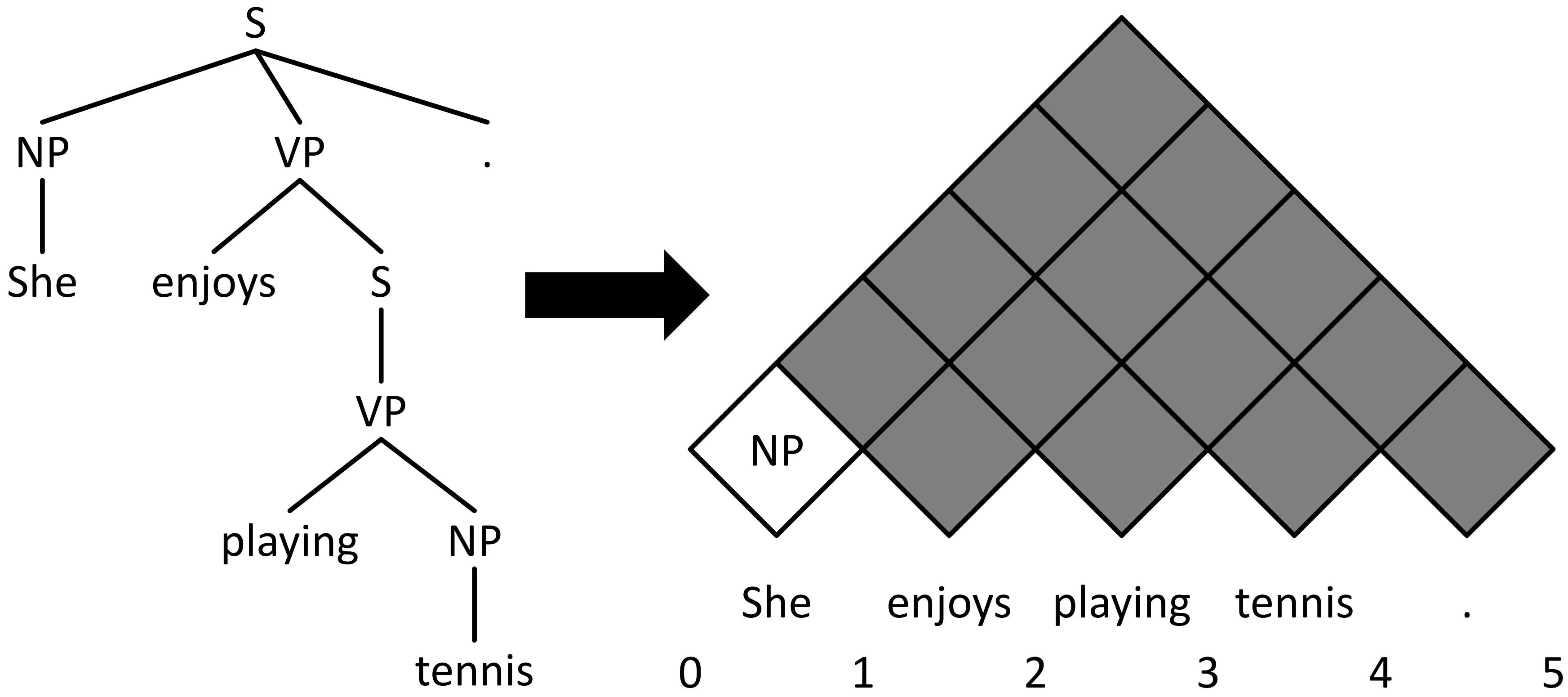
Parsing as Span Classification



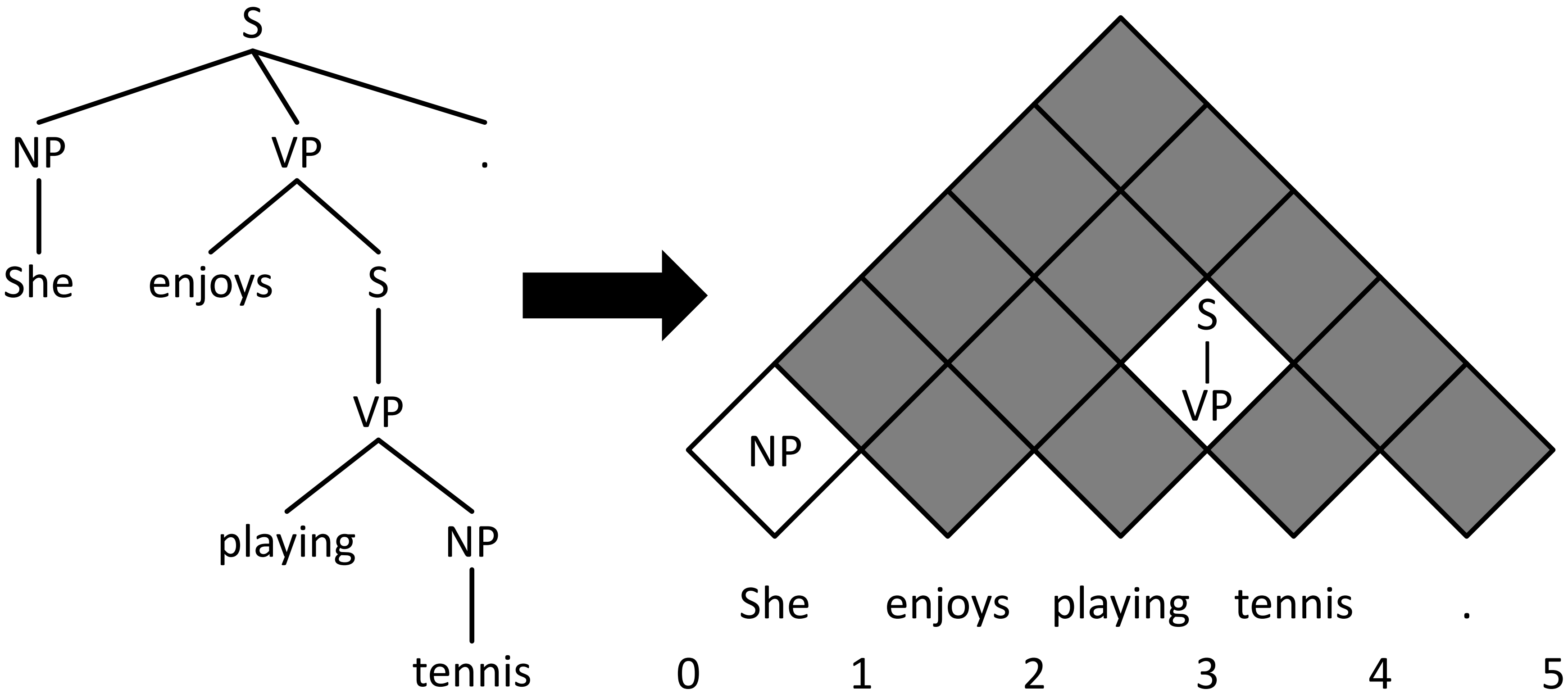
Parsing as Span Classification



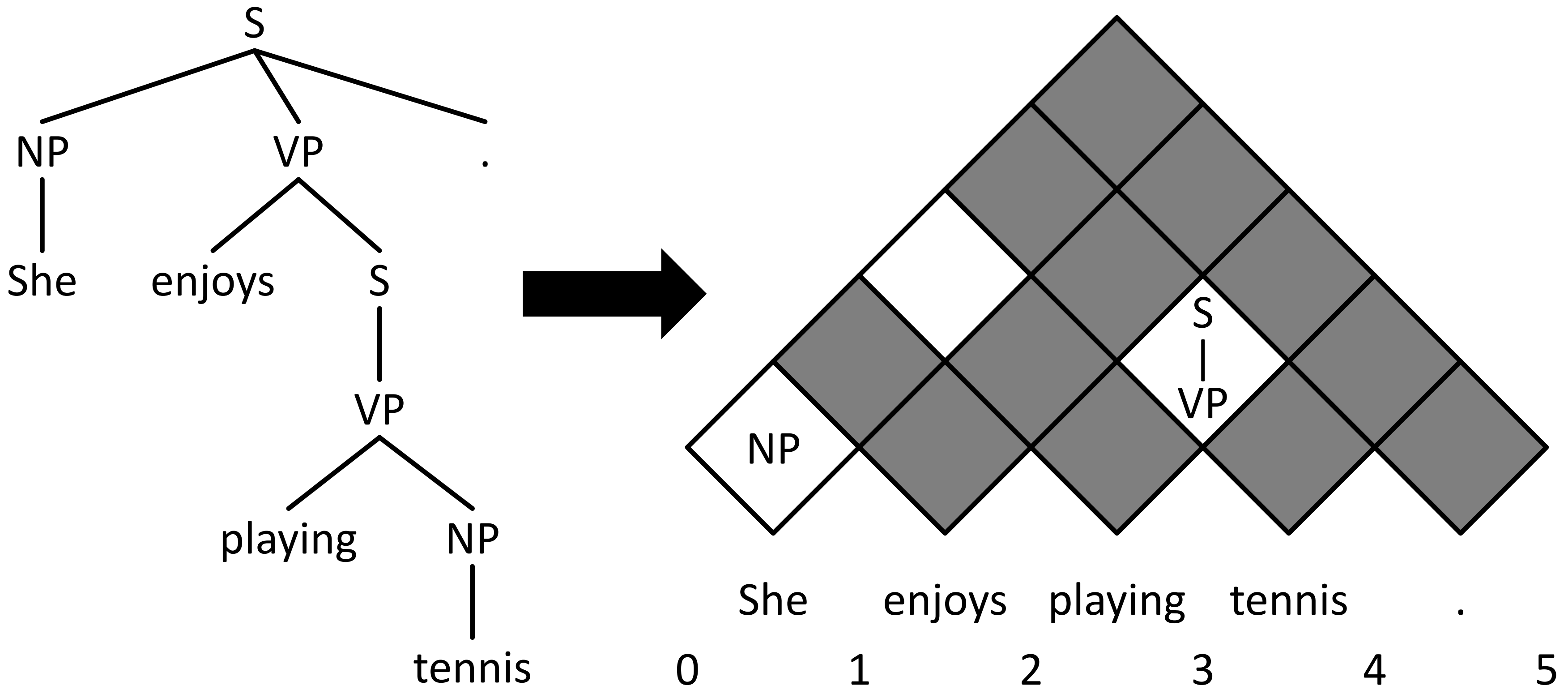
Parsing as Span Classification



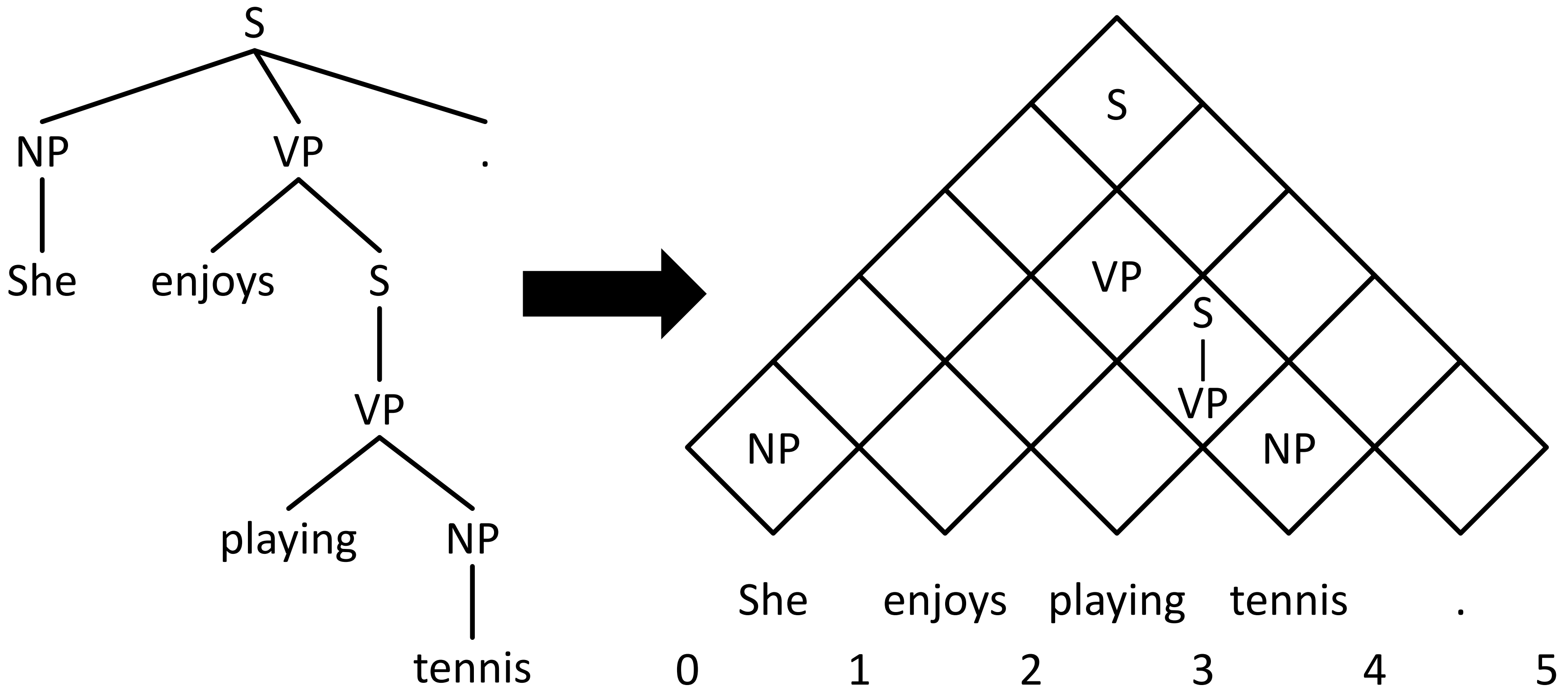
Parsing as Span Classification



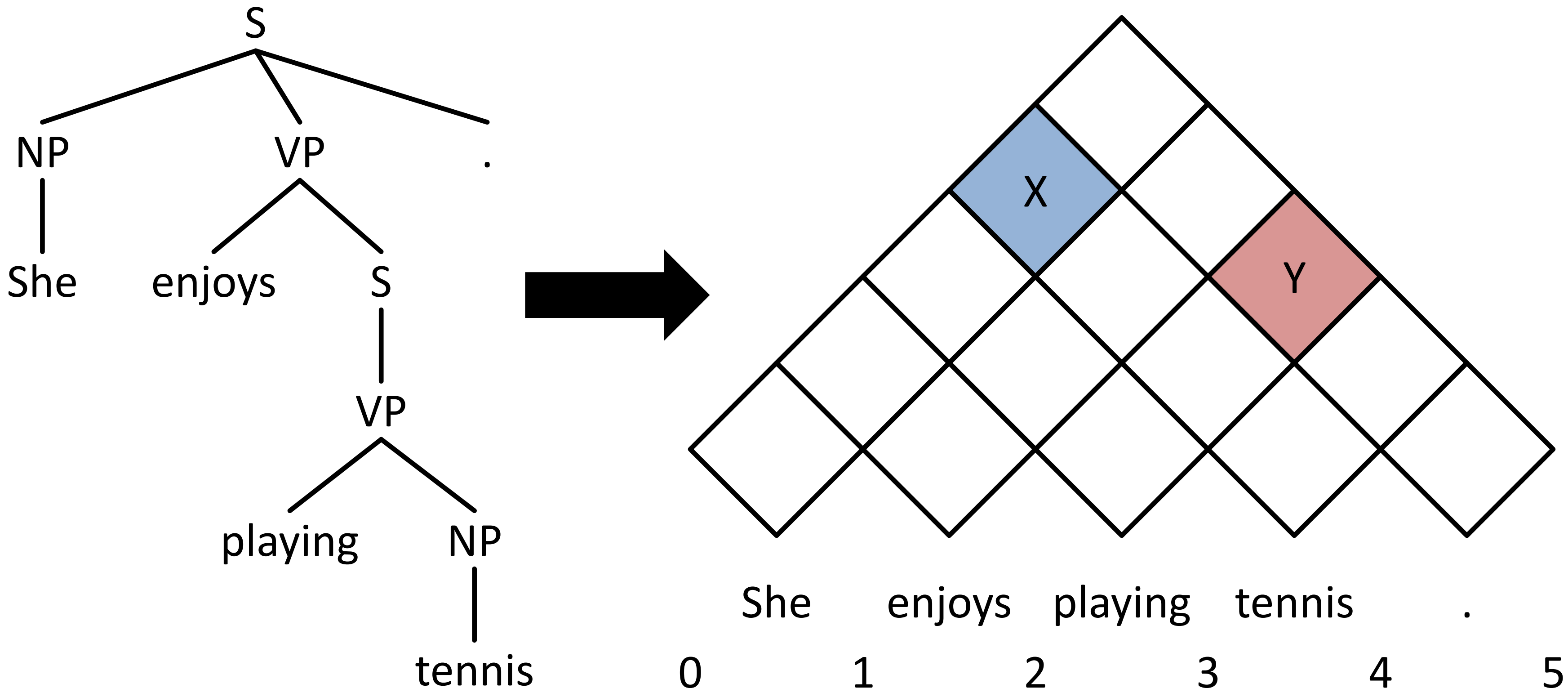
Parsing as Span Classification



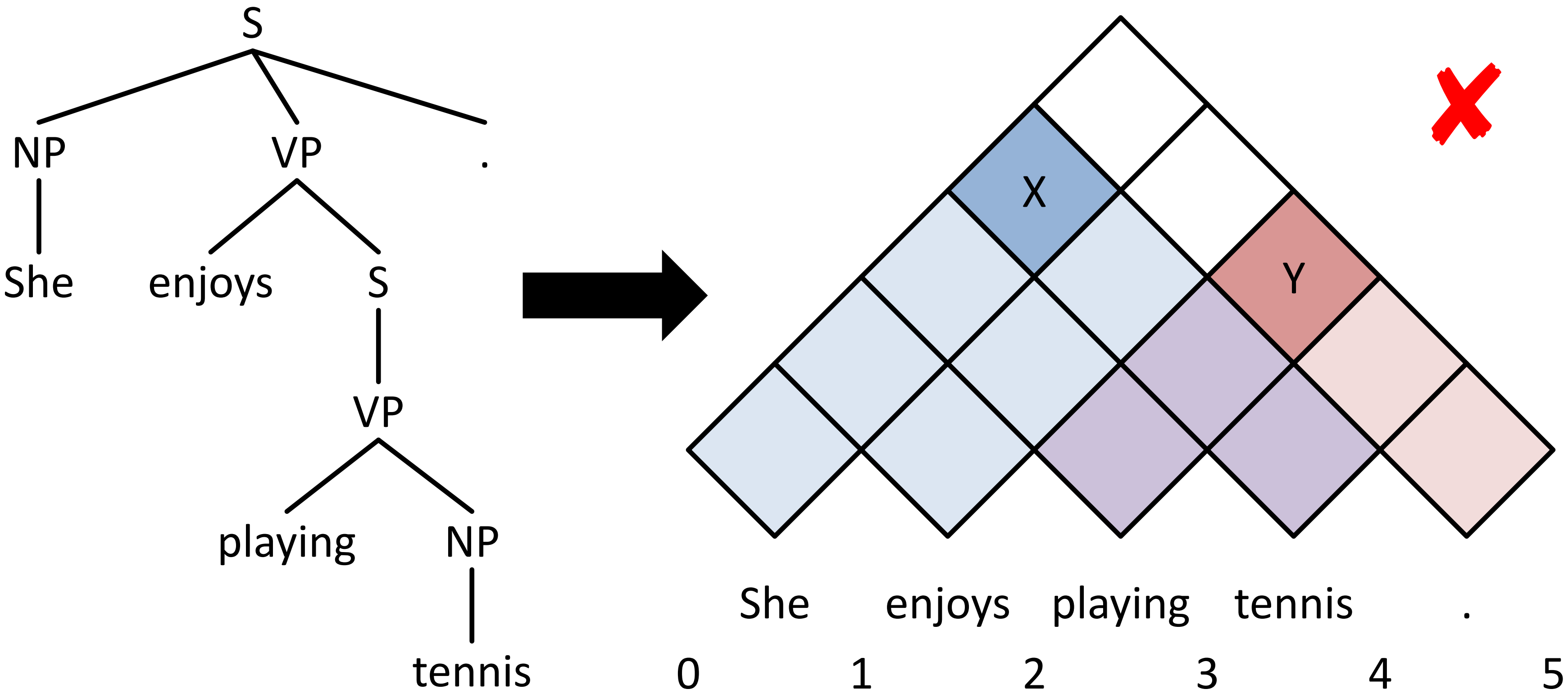
Parsing as Span Classification



Parsing as Span Classification



Parsing as Span Classification



Minimality in Parsing

Grammar:

Minimality in Parsing

Grammar:

S(decided) $\longrightarrow$ NP(workers) VP(decided) [Collins (1999)]

Minimality in Parsing

Grammar:

S(decided) $\rightarrow$ NP(workers) VP(decided) [Collins (1999)]

S $\rightarrow$ NP^S VP^S [Klein and Manning (2003)]

Minimality in Parsing

Grammar:

S(decided)	$\rightarrow$	NP(workers)	VP(decided)	[Collins (1999)]
S	$\rightarrow$	NP^S	VP^S	[Klein and Manning (2003)]
S	$\rightarrow$	NP	VP	[Hall et al. (2014)]

Minimality in Parsing

Grammar:

S(decided)	$\rightarrow$	NP(workers)	VP(decided)	[Collins (1999)]
S	$\rightarrow$	NP^S	VP^S	[Klein and Manning (2003)]
S	$\rightarrow$	NP	VP	[Hall et al. (2014)]
S				[Vinyals et al. (2015)]

Minimality in Parsing

Scoring:

Minimality in Parsing

Scoring:

score(S $\longrightarrow$ NP^S VP^S)

[Klein and Manning (2003)]

Minimality in Parsing

Scoring:

score(S $\longrightarrow$ NP^S VP^S) [Klein and Manning (2003)]

score(i, k, j, S $\longrightarrow$ NP VP) [Hall et al. (2014)]

Minimality in Parsing

Scoring:

score(S $\longrightarrow$ NP^S VP^S) [Klein and Manning (2003)]

score(i, k, j, S $\longrightarrow$ NP VP) [Hall et al. (2014)]

score(i, j, S) and $\text{score}_{\text{action}}$($i$, k, j) [Cross and Huang (2016)]

Minimality in Parsing

Scoring:

score(S $\longrightarrow$ NP^S VP^S) [Klein and Manning (2003)]

score(i, k, j, S $\longrightarrow$ NP VP) [Hall et al. (2014)]

score(i, j, S) and $\text{score}_{\text{action}}(i, k, j)$ [Cross and Huang (2016)]

score(i, j, S) [This work]

Minimality in Parsing

Decoding:

Minimality in Parsing

Decoding:

Chart-based

Globally optimal, $O(n^3)$ time complexity

Minimality in Parsing

Decoding:

Chart-based

Globally optimal, $O(n^3)$ time complexity

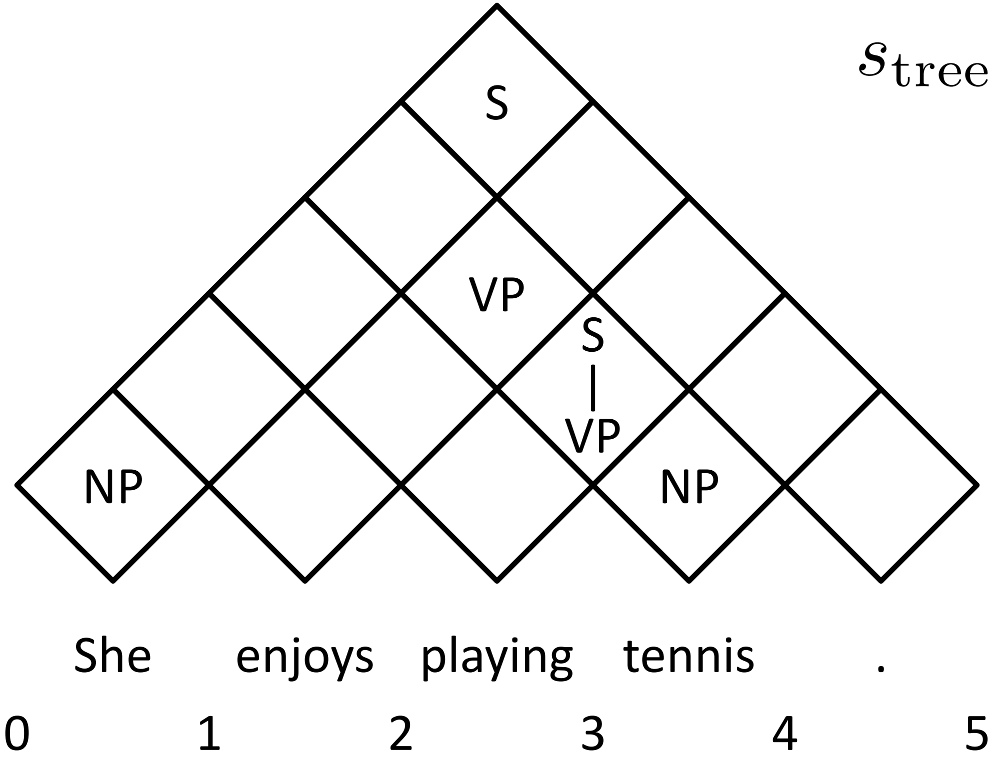
Transition-based

Greedy, $O(n)$ or $O(n^2)$ time complexity

Tree Scoring Function

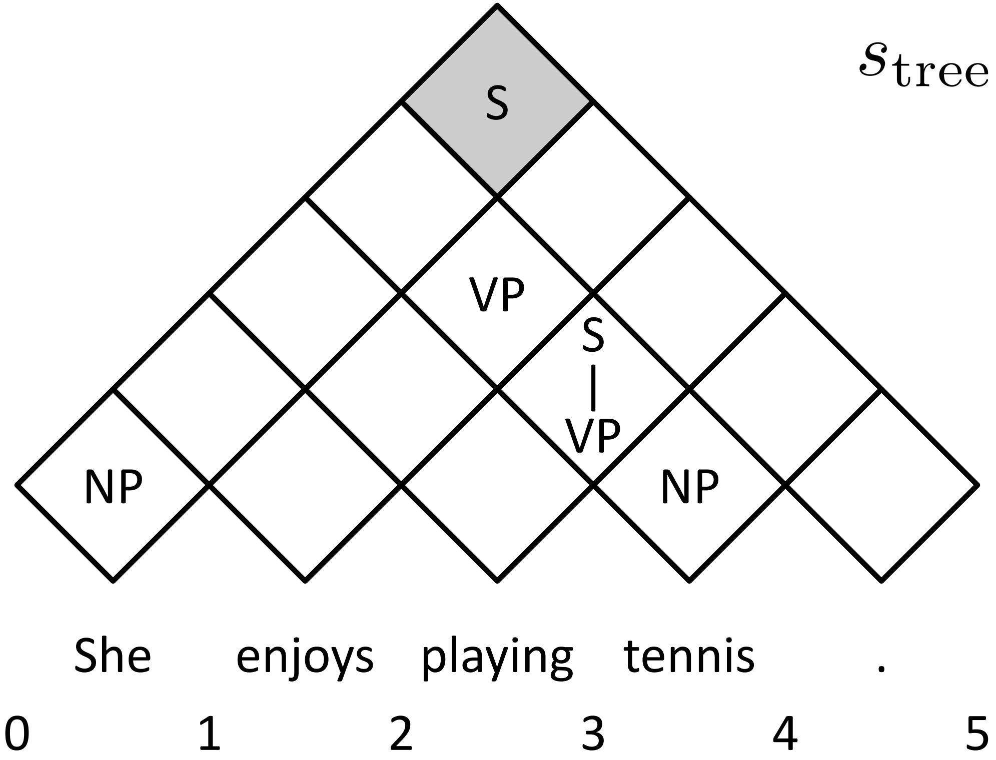
$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

Tree Scoring Function



$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

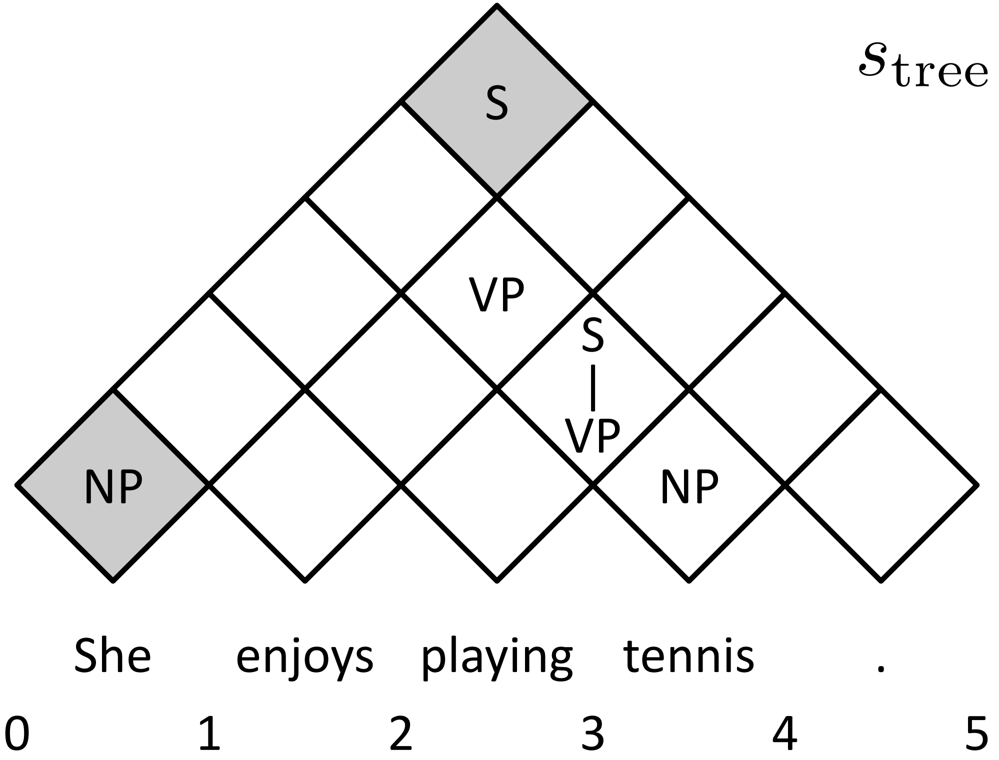
Tree Scoring Function



$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

$$= s(0,5,\mathrm{S})$$

Tree Scoring Function

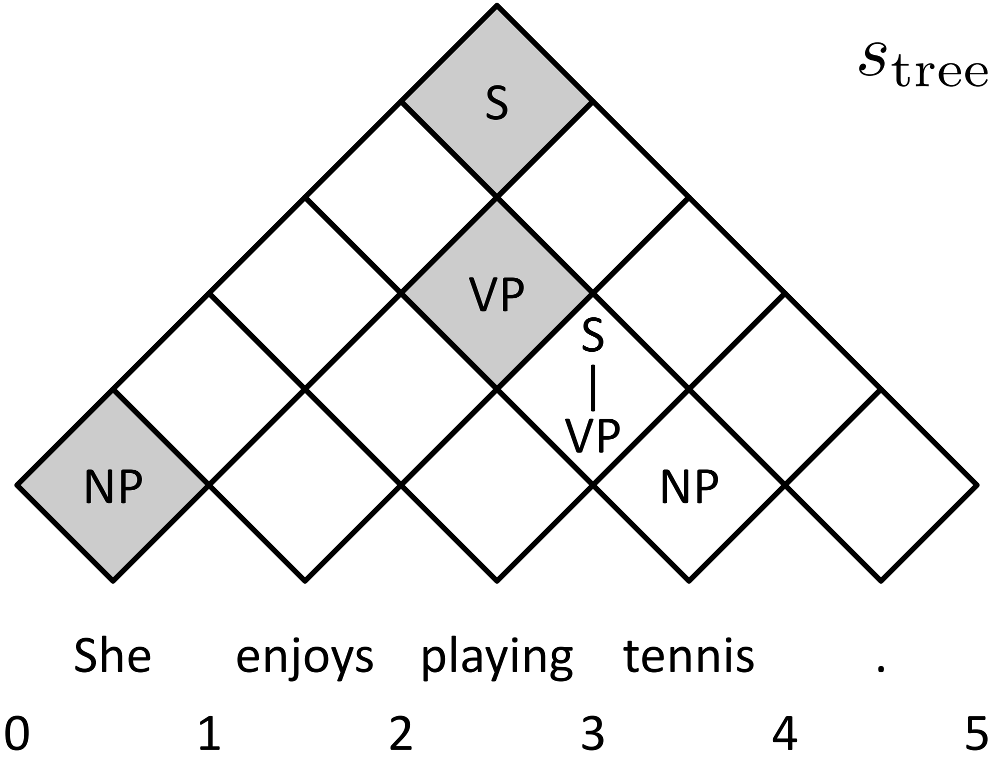


$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j)) \in T} s(i,j,\ell)$$

$$= s(0,5,\mathrm{S})$$

$$+ s(0,1,\mathrm{NP})$$

Tree Scoring Function



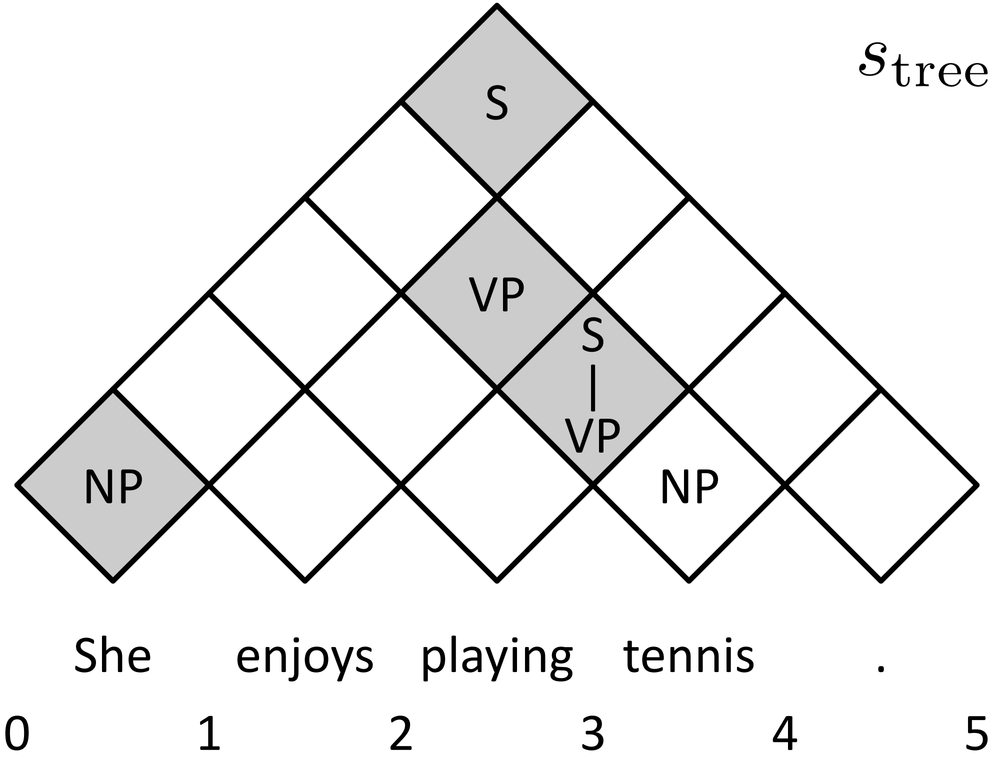
$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

$$= s(0,5,\text{S})$$

$$+ s(0,1,\text{NP})$$

$$+ s(1,4,\text{VP})$$

Tree Scoring Function



$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

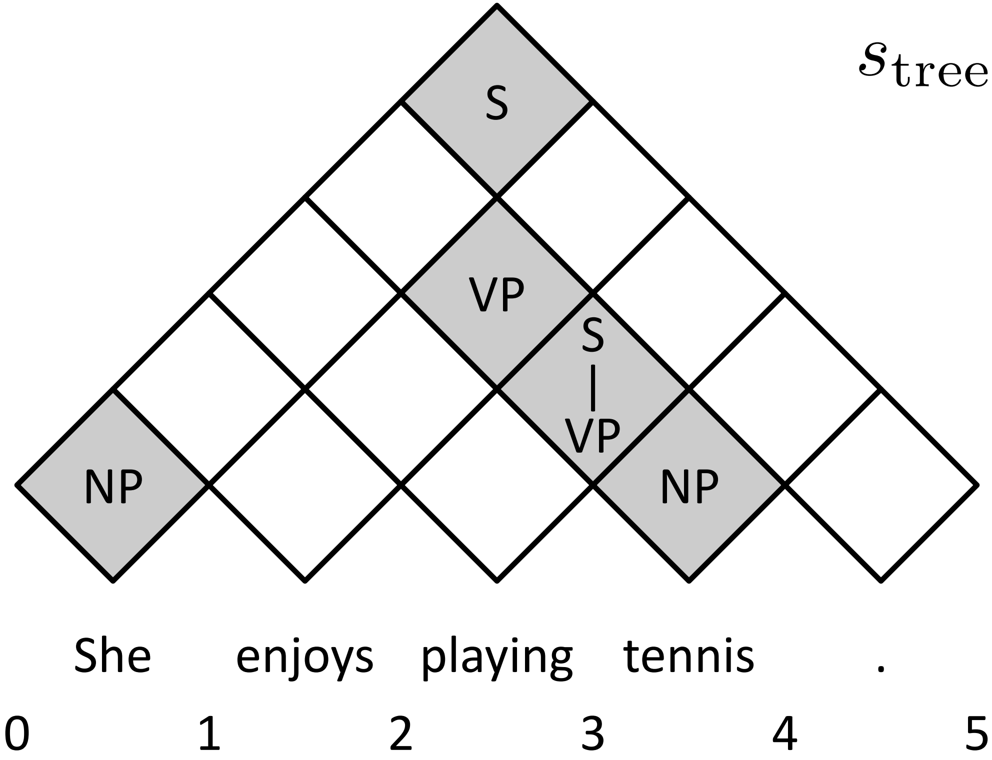
$$= s(0,5,\mathrm{S})$$

$$+\ s(0,1,\mathrm{NP})$$

$$+\ s(1,4,\mathrm{VP})$$

$$+\ s(2,4,\mathrm{S\text{–}VP})$$

Tree Scoring Function



$$s_{\text{tree}}(T) = \sum_{(\ell,(i,j))\in T} s(i,j,\ell)$$

$$= s(0,5,\mathrm{S})$$
$$+ \; s(0,1,\mathrm{NP})$$
$$+ \; s(1,4,\mathrm{VP})$$
$$+ \; s(2,4,\mathrm{S–VP})$$
$$+ \; s(3,4,\mathrm{NP})$$

Dynamic Program: Base Case

$$s_{\text{best}}(i, i+1) = \max_{\ell}[s(i, i+1, \ell)]$$

Dynamic Program: Base Case

$$s_{\text{best}}(i, i+1) = \overbrace{\max_{\ell}[s(i, i+1, \ell)]}^{\text{Pick best label}}$$

Dynamic Program: General Case

$$s_{\text{best}}(i,j) = \max_{\ell}[s(i,j,\ell)] + \max_{k}[s_{\text{best}}(i,k) + s_{\text{best}}(k,j)]$$

Dynamic Program: General Case

$$s_{\text{best}}(i,j) = \overbrace{\max_{\ell}[s(i,j,\ell)]}^{\text{Pick best label}}$$

$$+ \max_{k}[s_{\text{best}}(i,k) + s_{\text{best}}(k,j)]$$

Dynamic Program: General Case

$$s_{\text{best}}(i,j) = \overbrace{\max_{\ell}[s(i,j,\ell)]}^{\text{Pick best label}}$$

$$+ \underbrace{\max_{k}[s_{\text{best}}(i,k) + s_{\text{best}}(k,j)]}_{\text{Pick best split point}}$$

Scoring Function Implementation

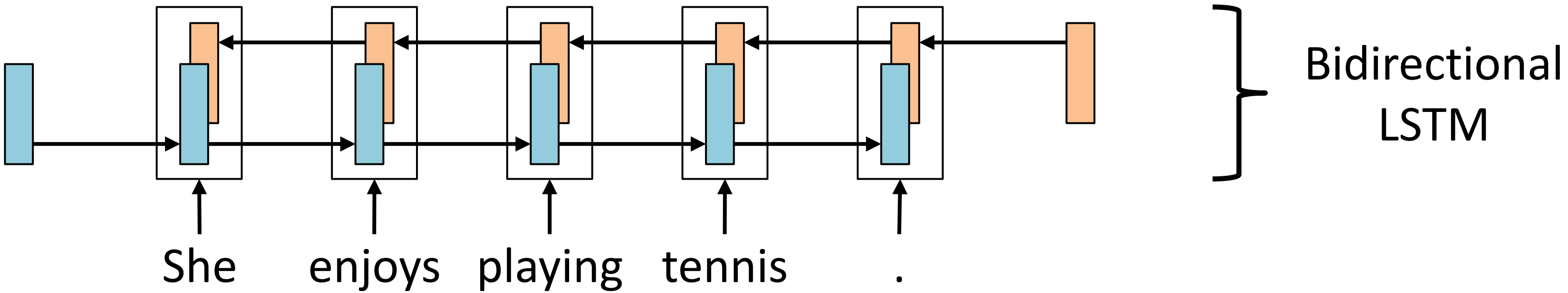
[Inspired by Cross and Huang (2016)]

Scoring Function Implementation

She enjoys playing tennis .

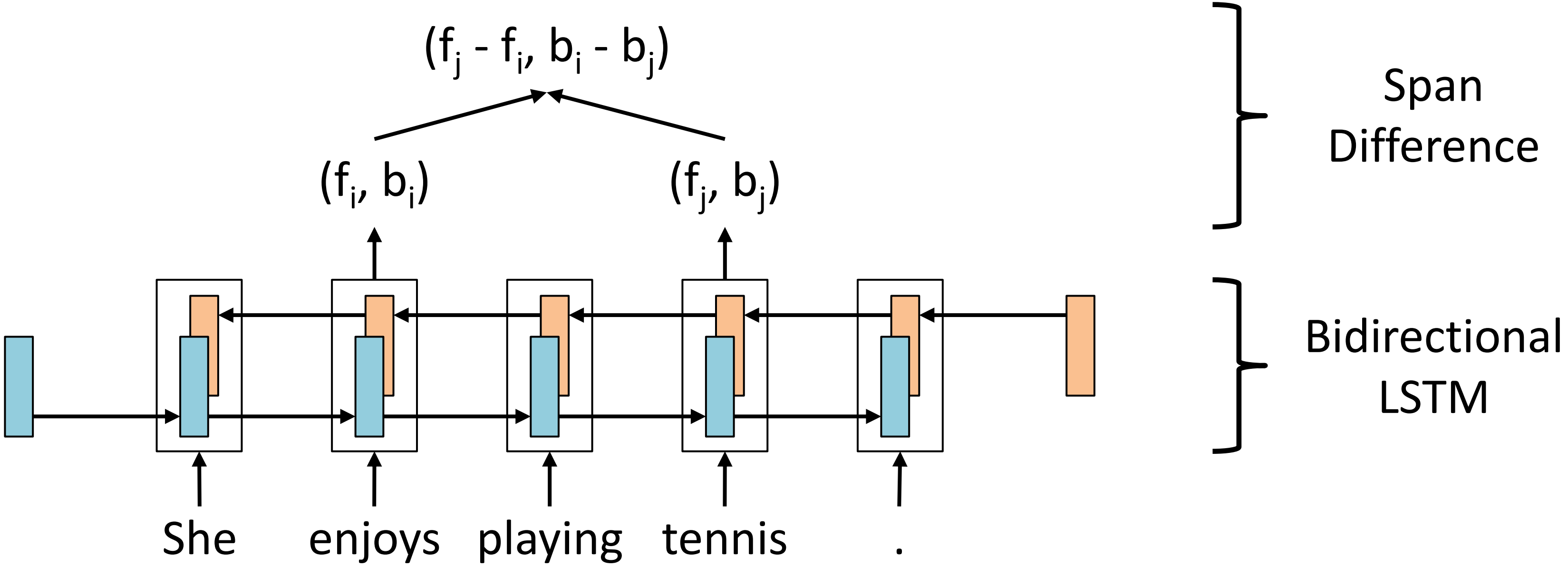
[Inspired by Cross and Huang (2016)]

Scoring Function Implementation



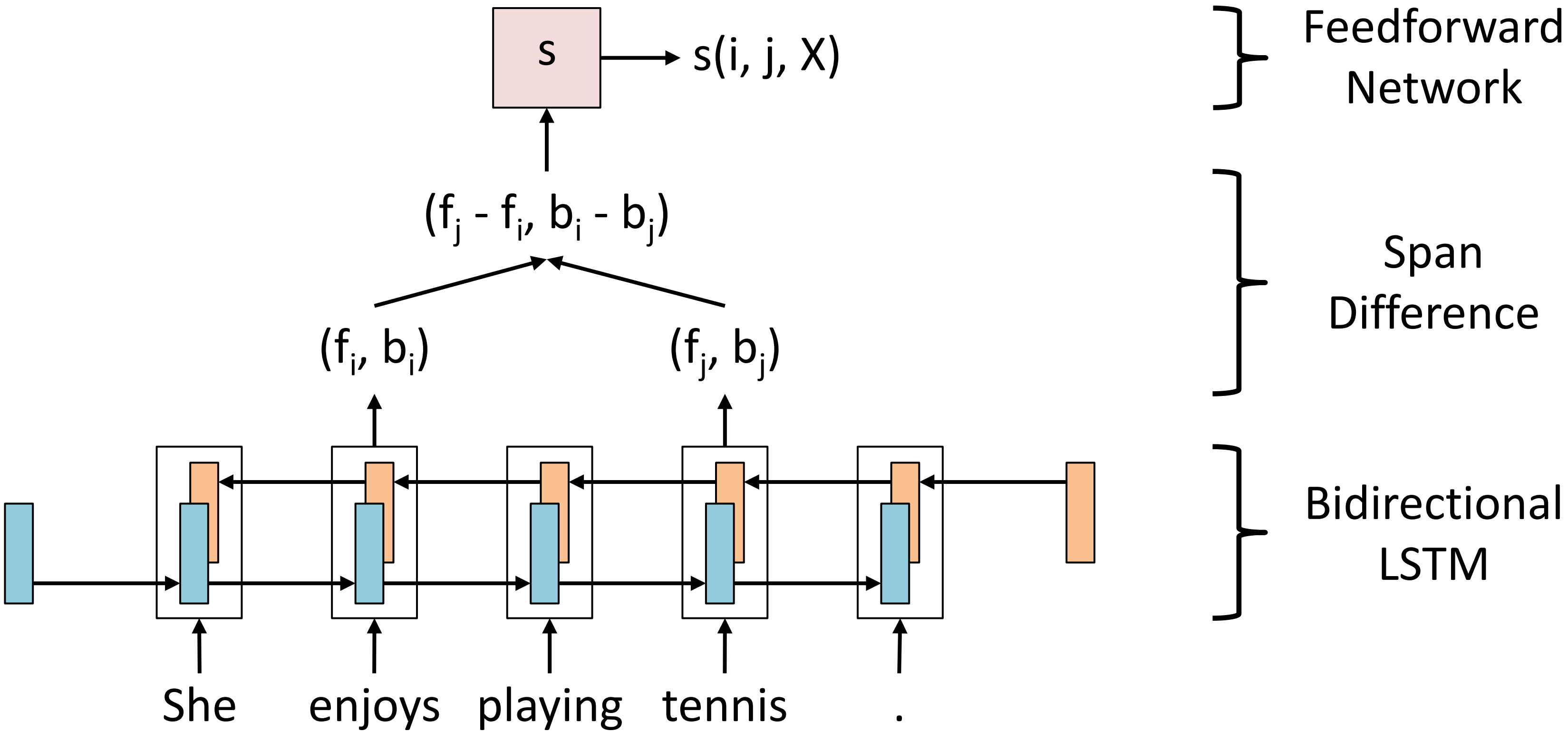
[Inspired by Cross and Huang (2016)]

Scoring Function Implementation



[Inspired by Cross and Huang (2016)]

Scoring Function Implementation



[Inspired by Cross and Huang (2016)]

Training

Training

Want $s_{\mathrm{tree}}(T^*) > s_{\mathrm{tree}}(T)$ for all $T \neq T^*$

Training

Want $s_{\text{tree}}(T^*) > s_{\text{tree}}(T)$ for all $T \neq T^*$

Require larger margin for higher loss:

$$s_{\text{tree}}(T^*) \geq \Delta(T, T^*) + s_{\text{tree}}(T)$$

Training

Want $s_{\text{tree}}(T^*) > s_{\text{tree}}(T)$ for all $T \neq T^*$

Require larger margin for higher loss:

$$s_{\text{tree}}(T^*) \geq \Delta(T, T^*) + s_{\text{tree}}(T)$$

Use hinge penalty function:

$$\max\left(0, \Delta(\hat{T}, T^*) - s_{\text{tree}}(T^*) + s_{\text{tree}}(\hat{T})\right)$$

Training

Use loss-augmented decoding during training:

$$\hat{T} = \max_{T} [\Delta(T, T^*) + s_{\text{tree}}(T)]$$

Training

Use loss-augmented decoding during training:

$$\hat{T} = \max_{T} [\Delta(T, T^*) + s_{\text{tree}}(T)]$$

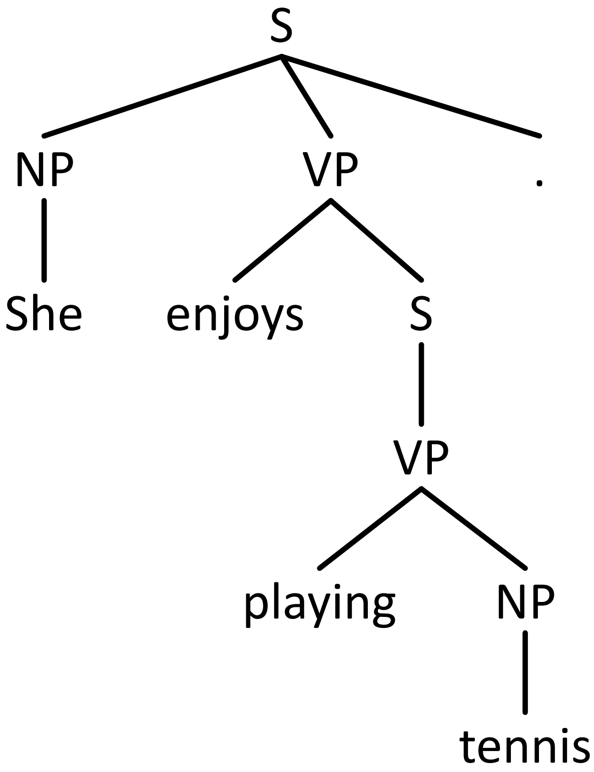
Loss-augmented decoding for Hamming loss:

Replace $s(i, j, \ell)$ with $s(i, j, \ell) + \mathbf{1}(\ell \neq \ell^*_{ij})$

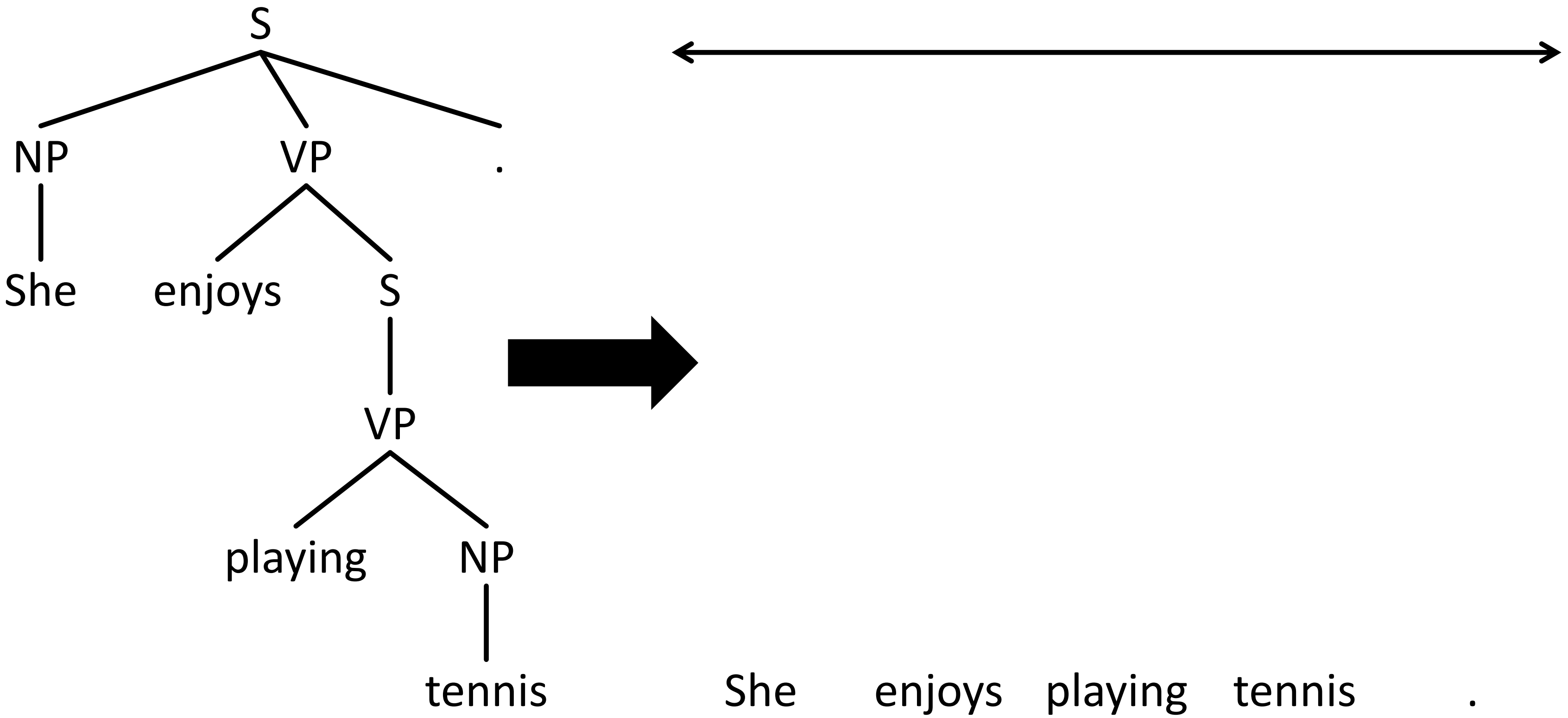
Initial Results

Parser	F1 Score
Hall et al. (2014)	89.2
Vinyals et al. (2015)	88.3
Cross and Huang (2016)	91.3
Dyer et al. (2016)	91.7
Liu and Zhang (2017)	91.7
Our Chart Parser	91.7

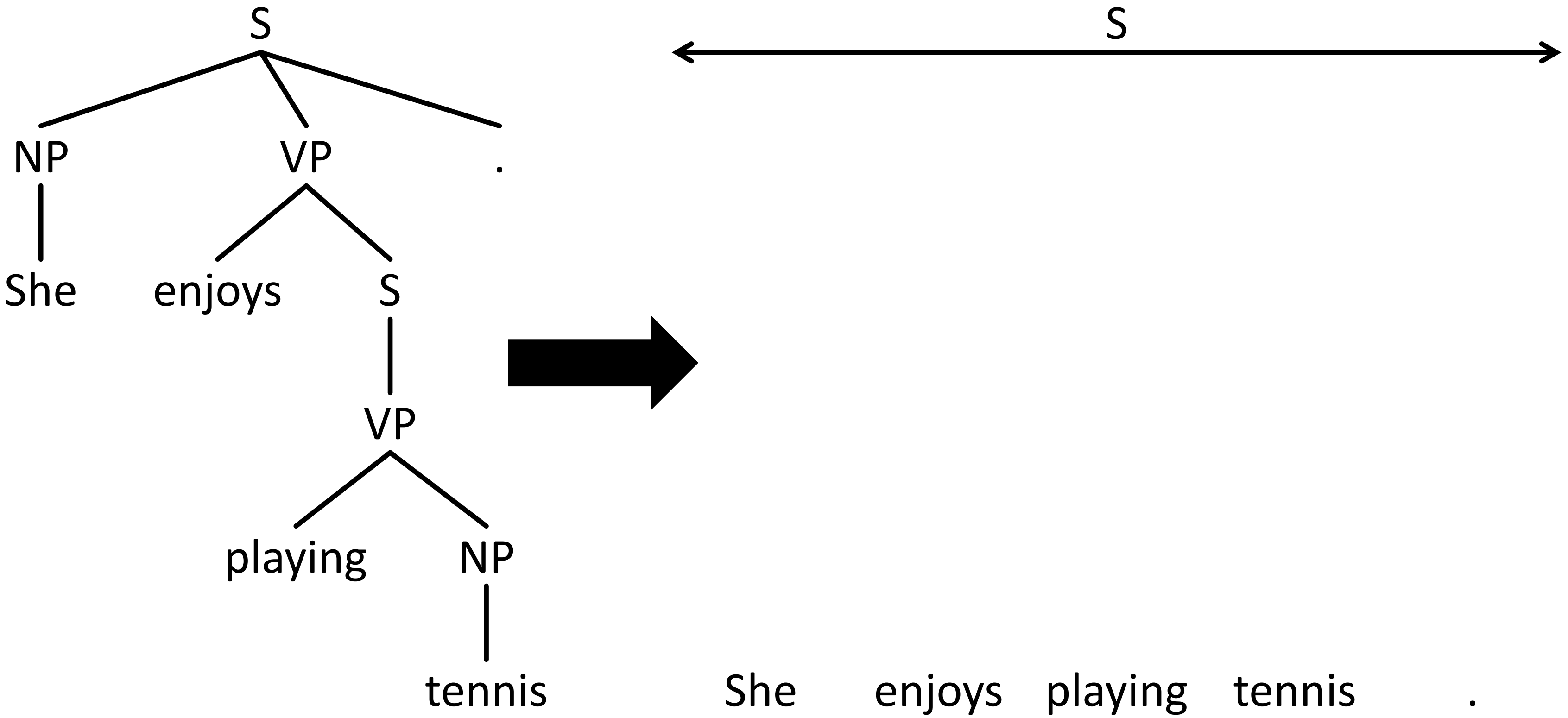
Top-Down Parsing



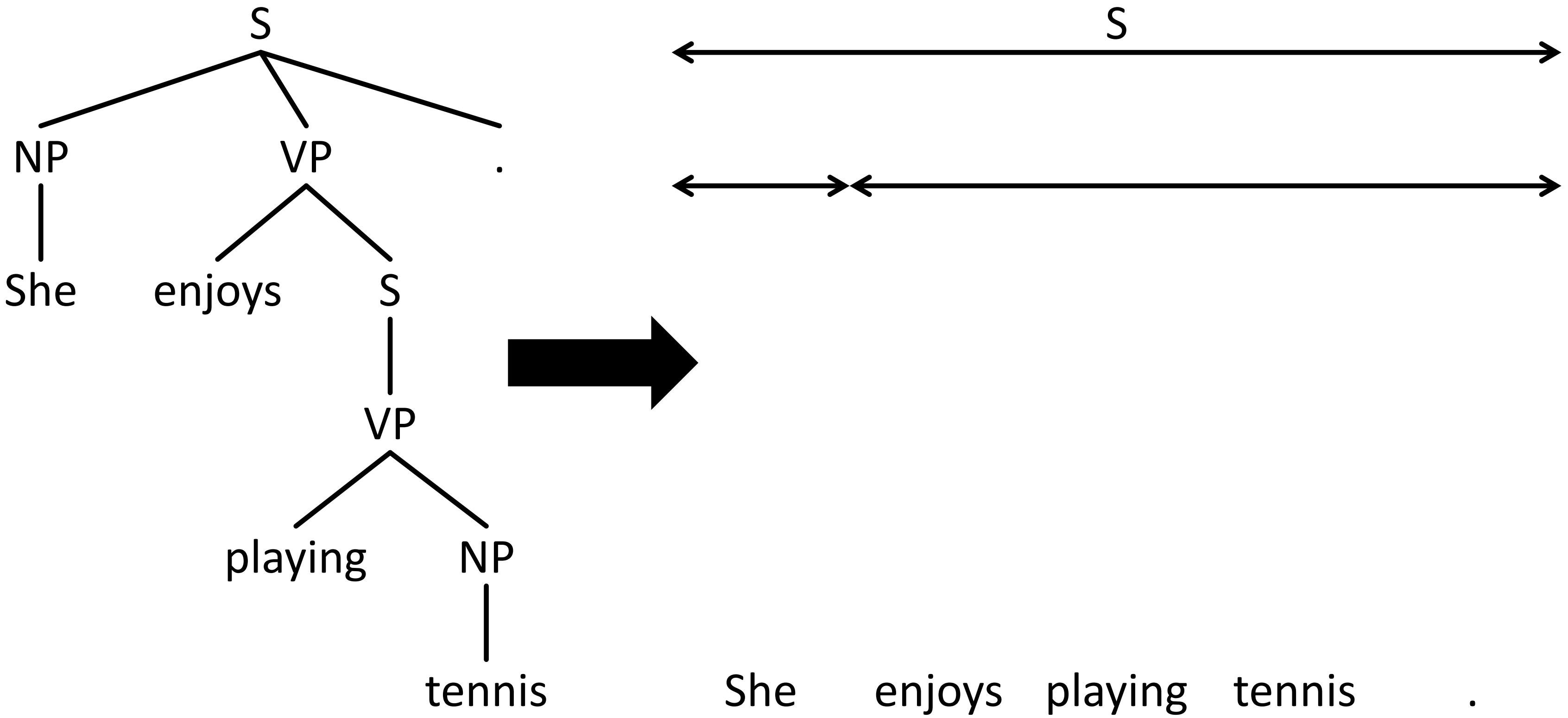
Top-Down Parsing



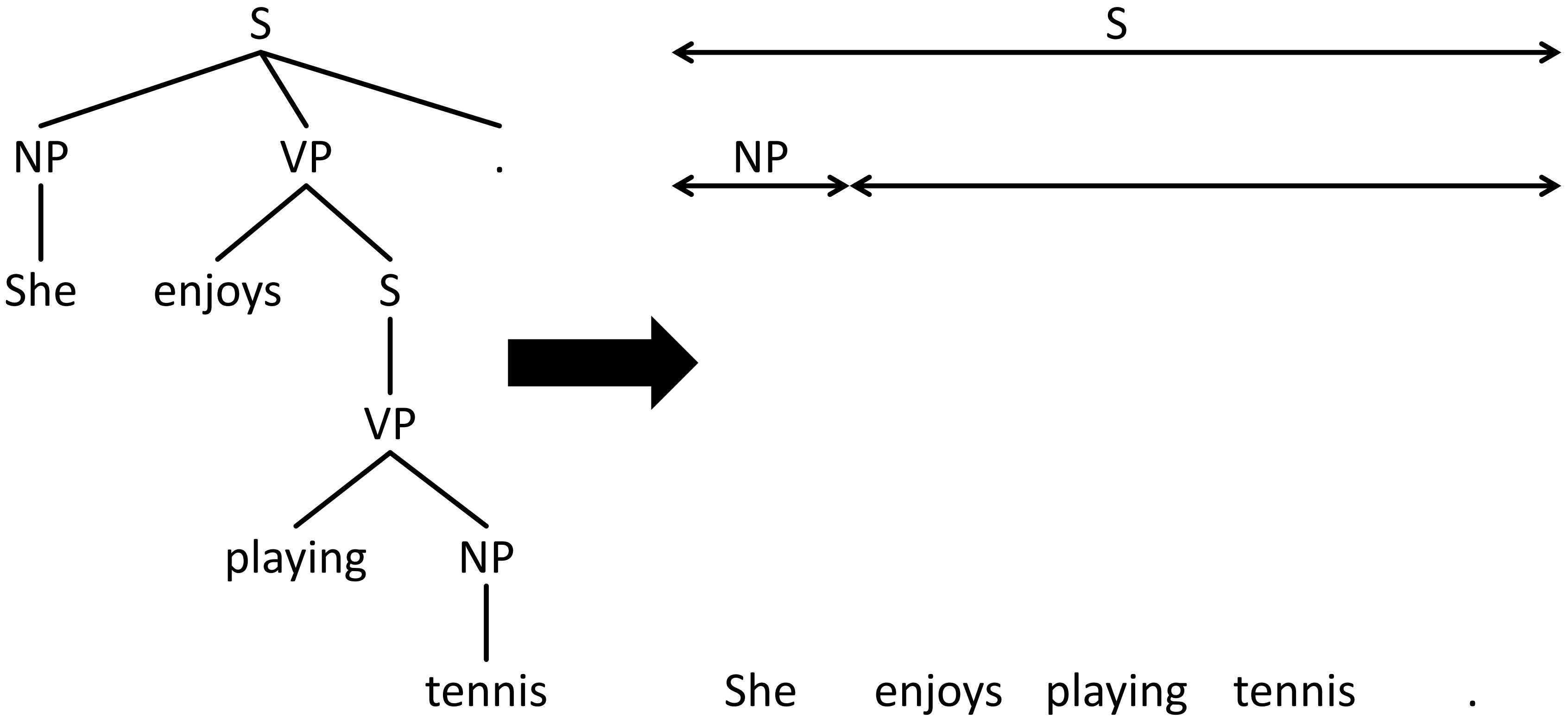
Top-Down Parsing



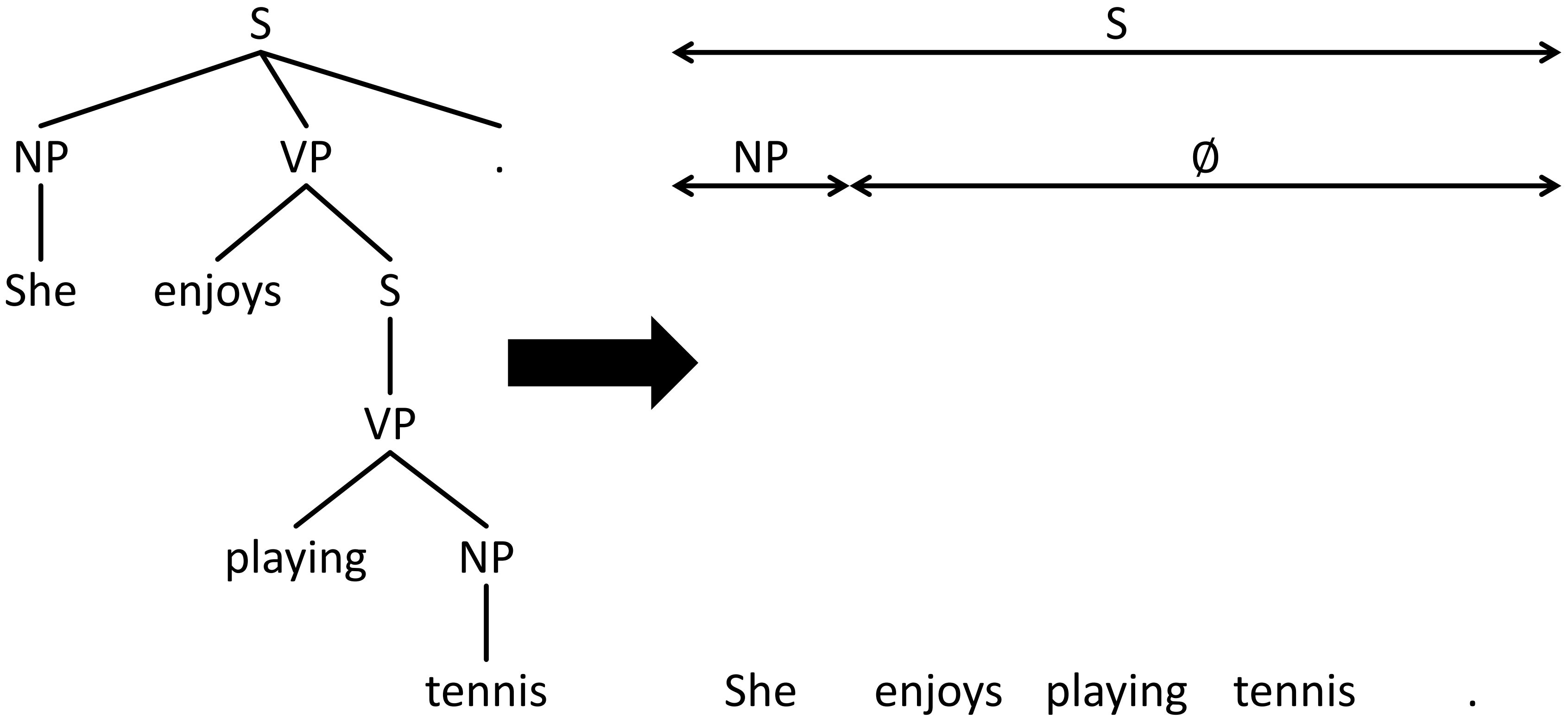
Top-Down Parsing



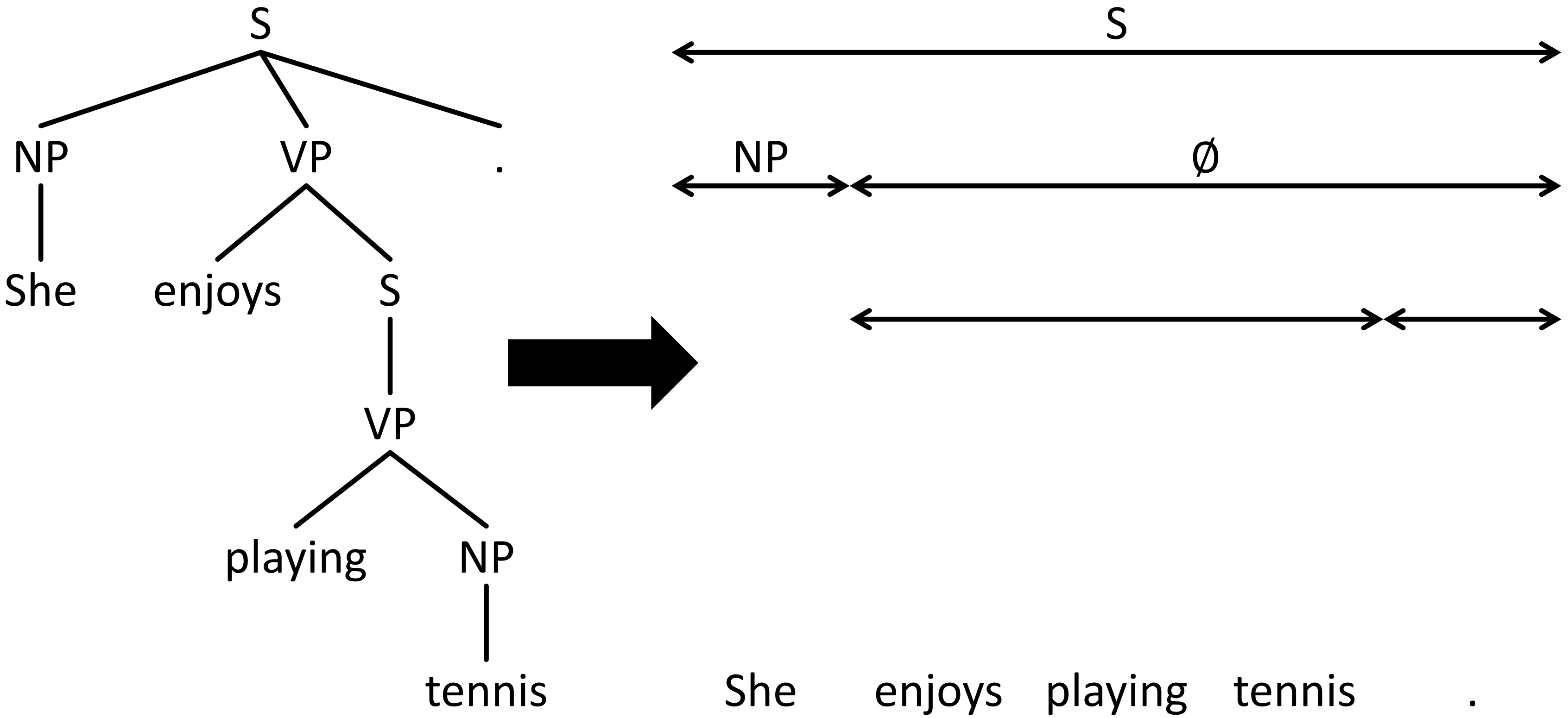
Top-Down Parsing



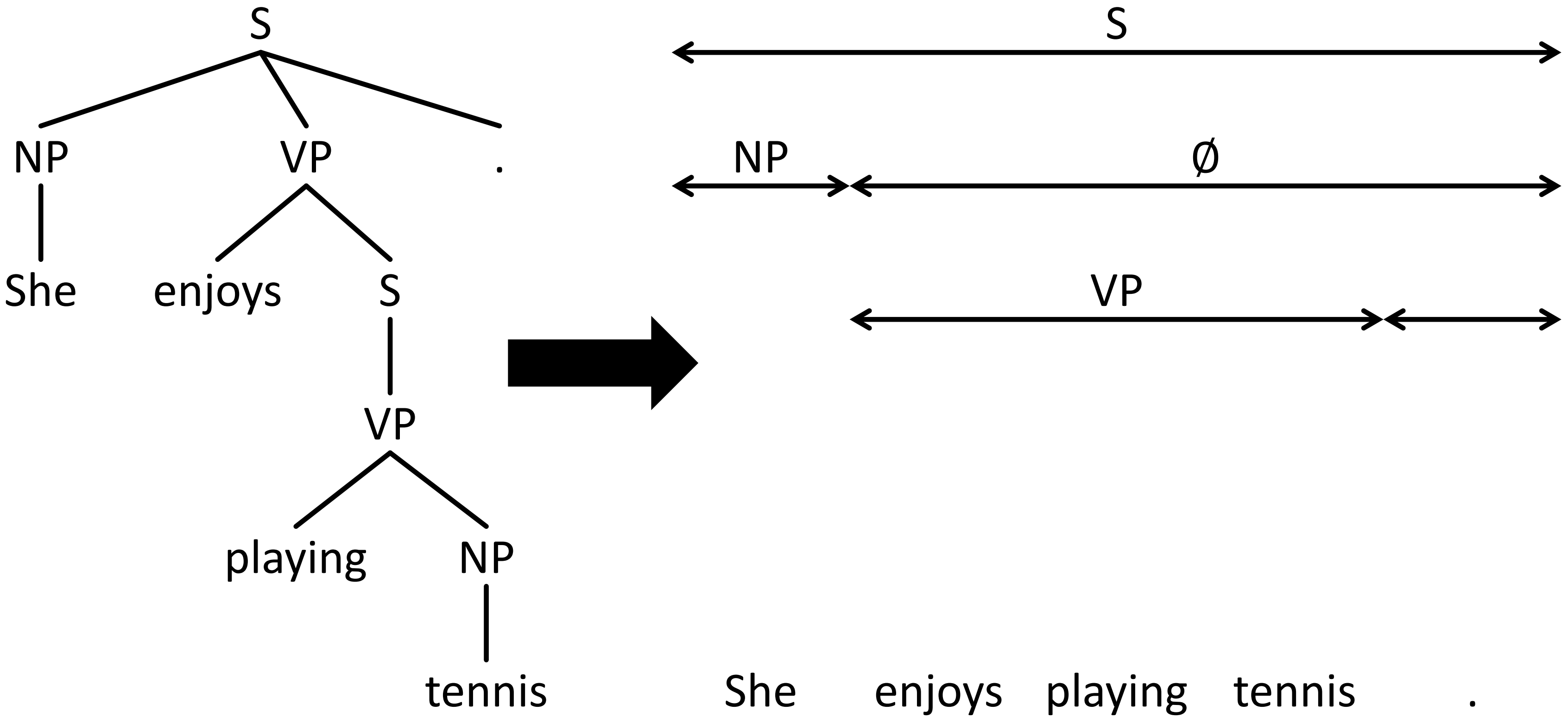
Top-Down Parsing



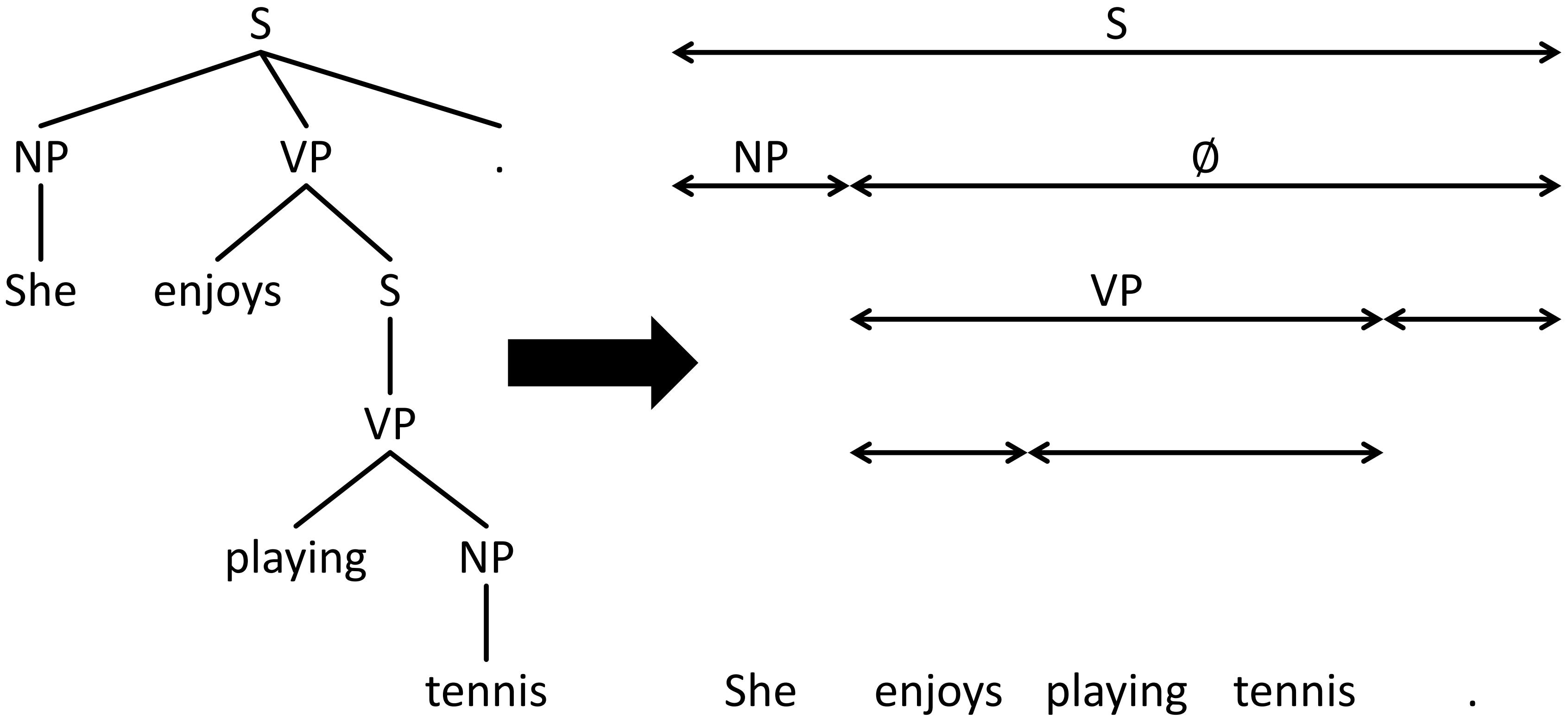
Top-Down Parsing



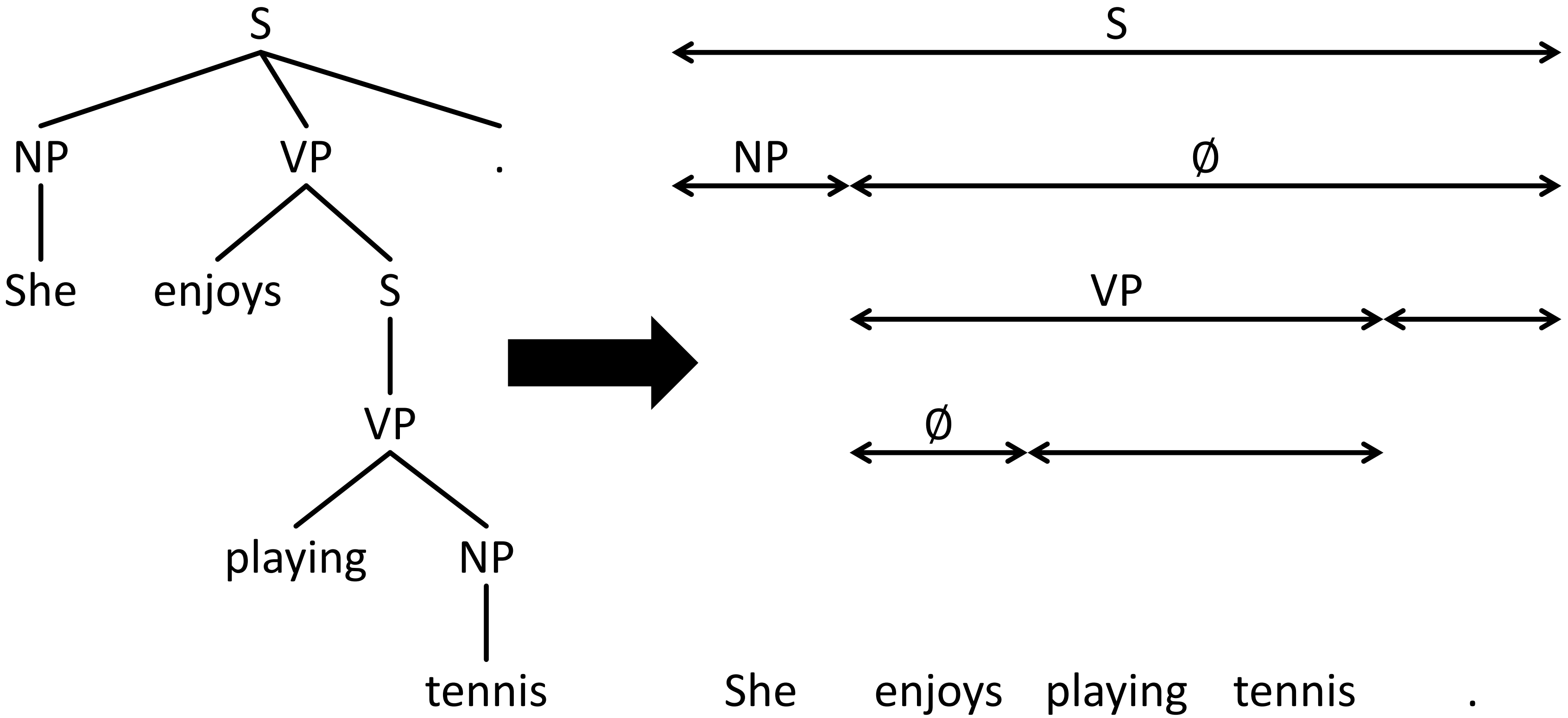
Top-Down Parsing



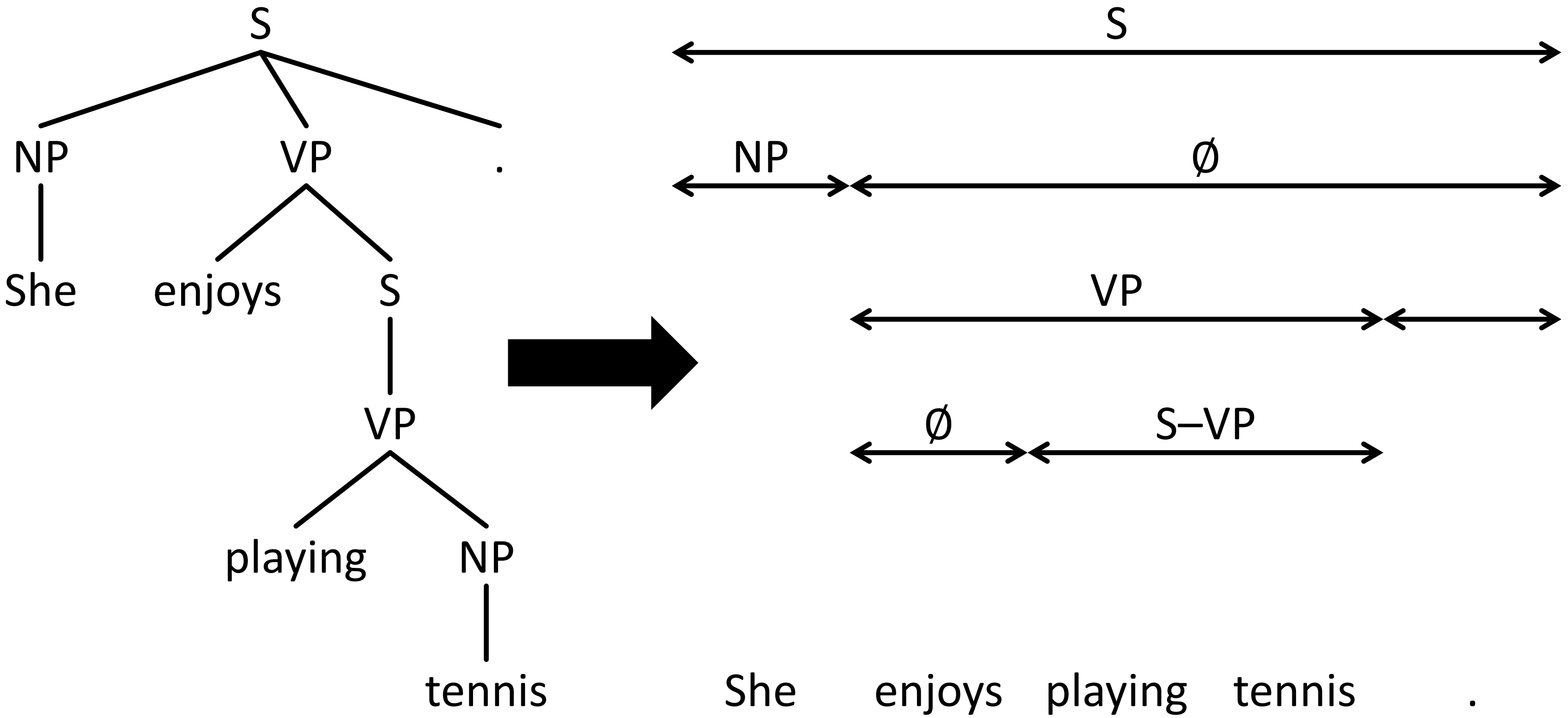
Top-Down Parsing



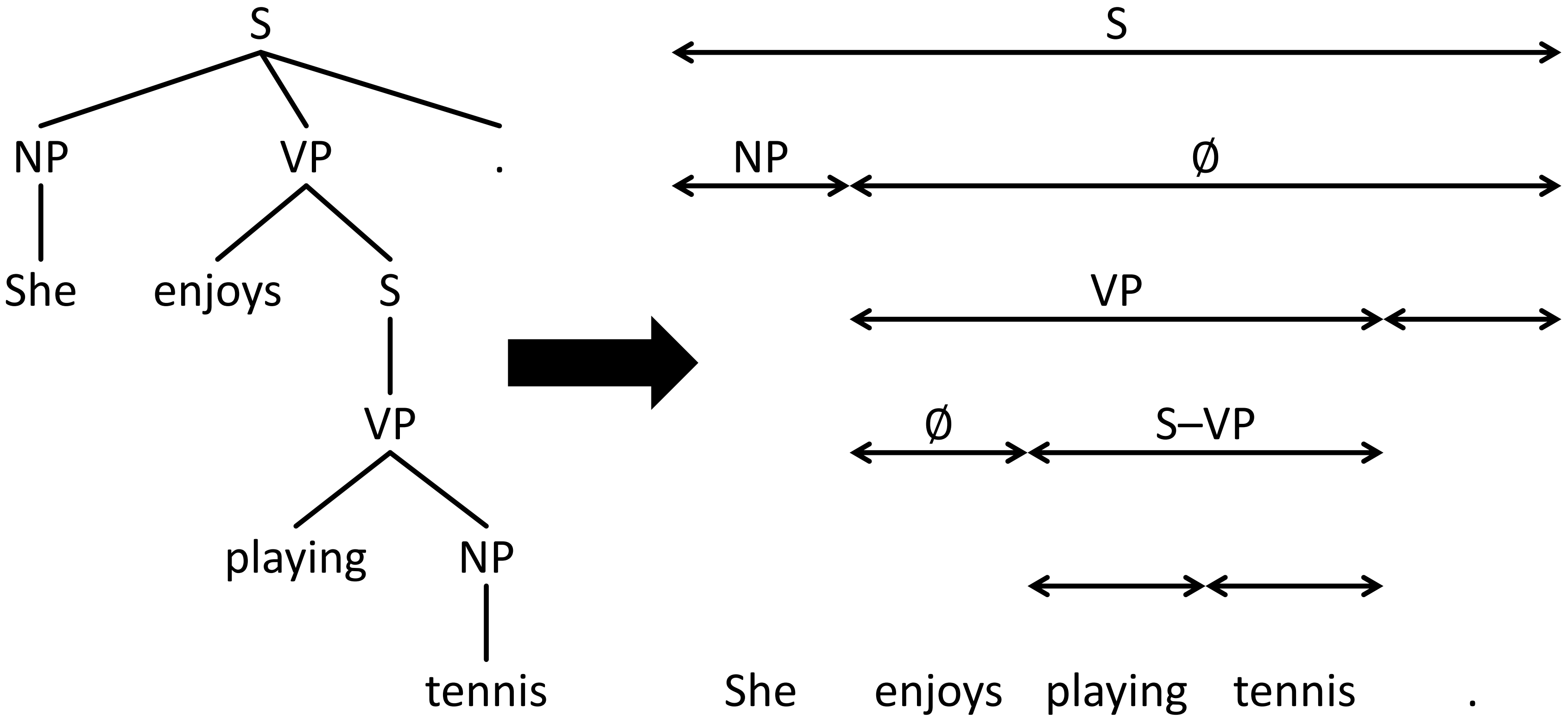
Top-Down Parsing



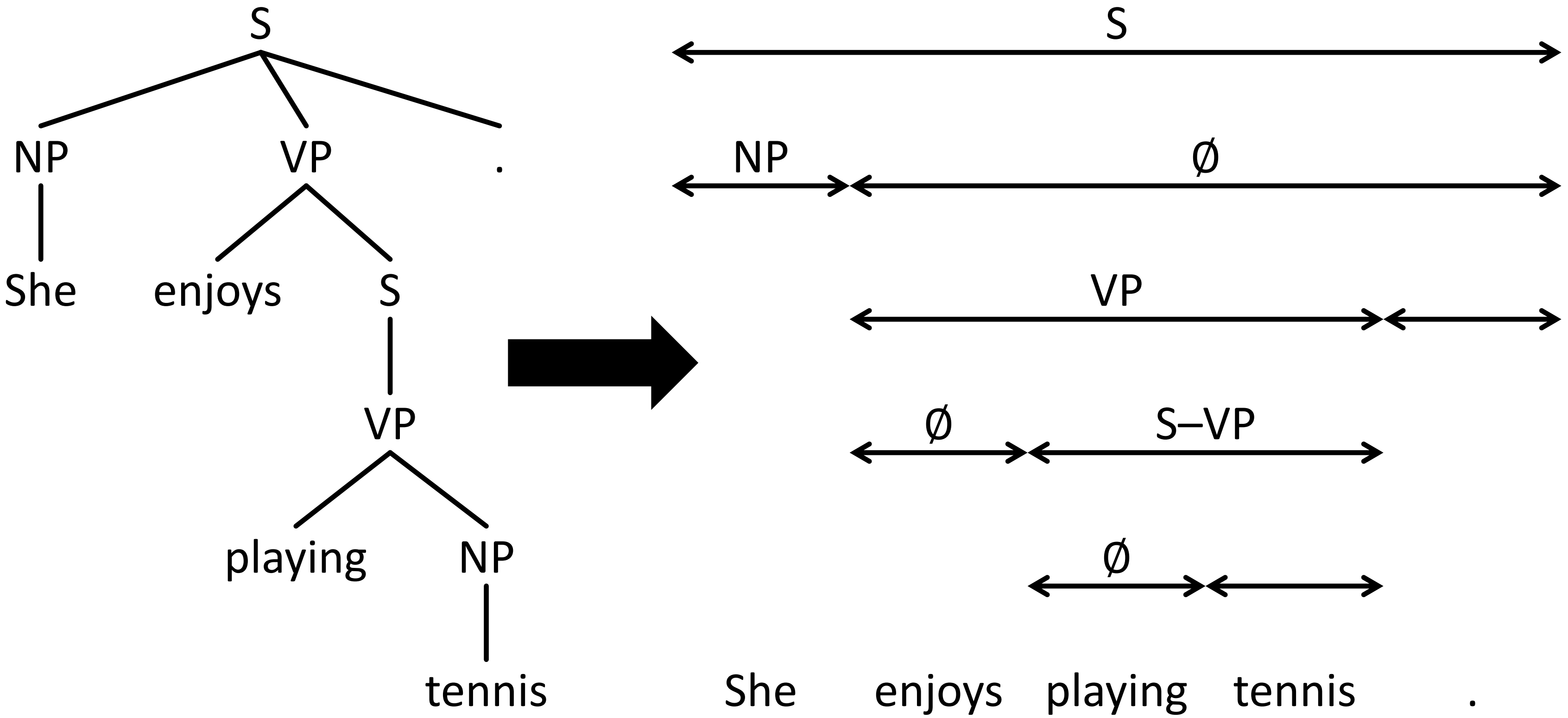
Top-Down Parsing



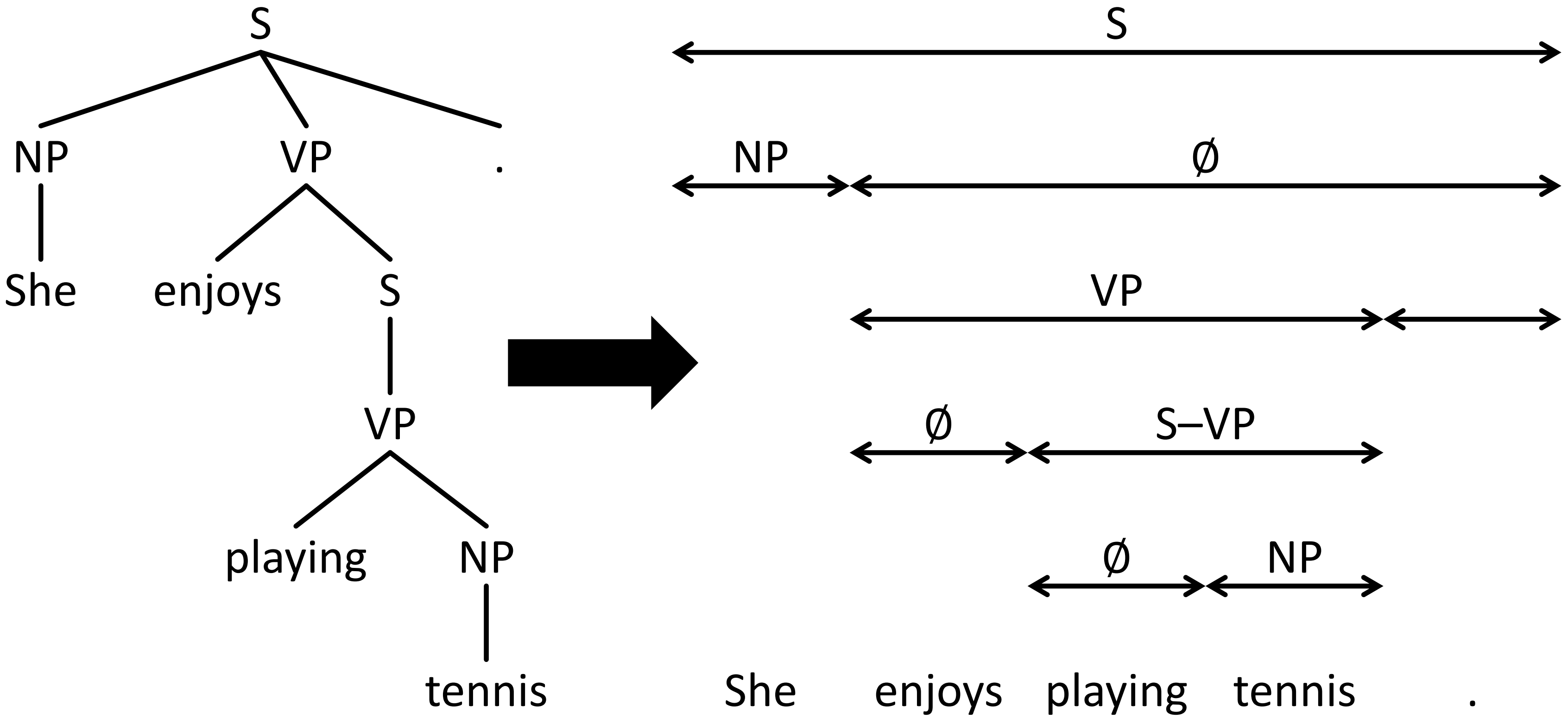
Top-Down Parsing



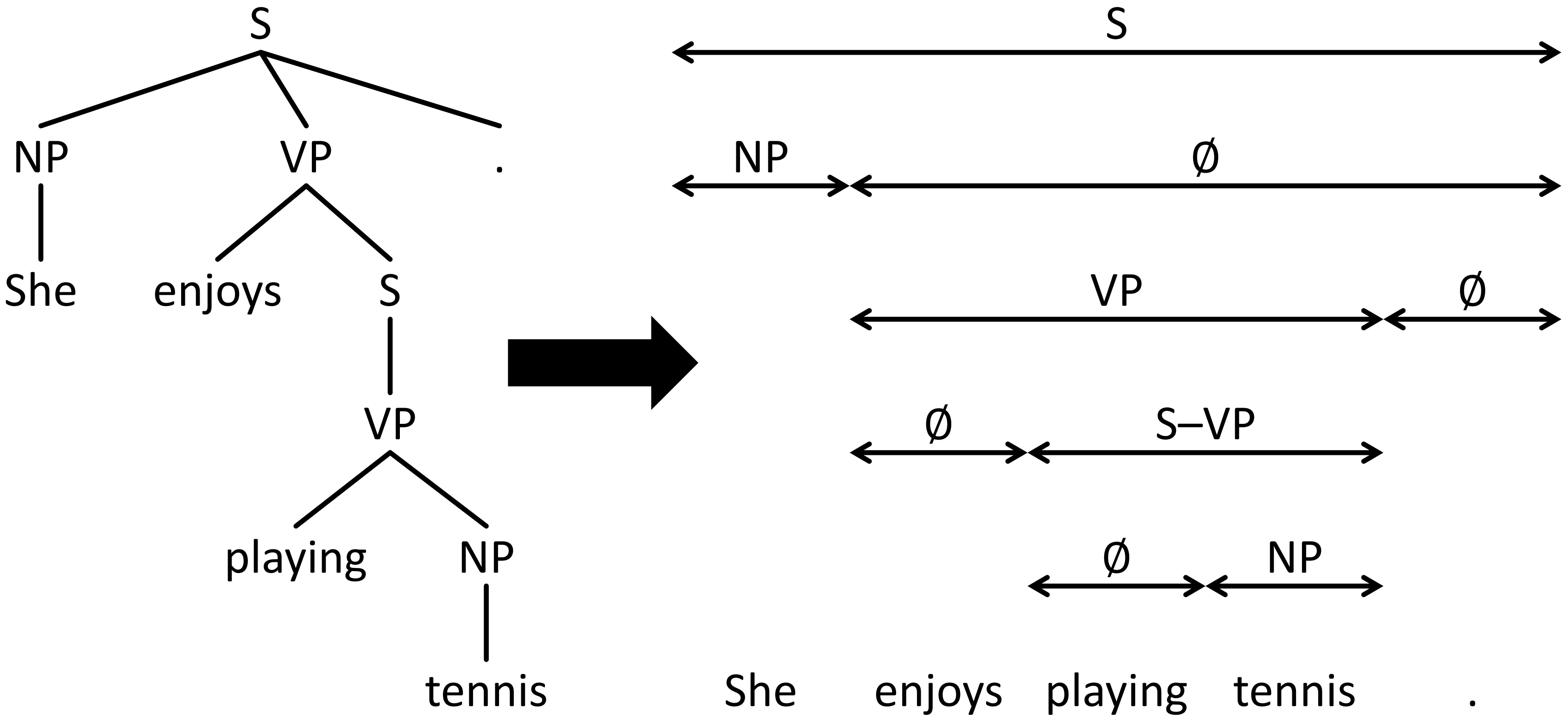
Top-Down Parsing



Top-Down Parsing



Top-Down Parsing



Greedily Label and Split

Greedily Label and Split

$$\hat{\ell} = \underset{\ell}{\operatorname{argmax}}[s_{\text{label}}(i, j, \ell)]$$

Greedily Label and Split

$$\hat{\ell} = \underset{\ell}{\operatorname{argmax}}[s_{\text{label}}(i, j, \ell)]$$

$$\hat{k} = \underset{k}{\operatorname{argmax}}[s_{\text{span}}(i, k) + s_{\text{span}}(k, j)]$$

Top-Down Training

Margin constraint for each decision:

$$\text{score(gold)} \geq 1 + \text{score(other)}$$

Top-Down Training

Margin constraint for each decision:

score(gold) ≥ 1 + score(other)

Train with exploration using a dynamic oracle

[Goldberg and Nivre (2012), Cross and Huang (2016)]

Initial Results

Parser	F1 Score
Hall et al. (2014)	89.2
Vinyals et al. (2015)	88.3
Cross and Huang (2016)	91.3
Dyer et al. (2016)	91.7
Liu and Zhang (2016)	91.7
Our Chart Parser	91.7
Our Top-Down Parser	91.6

Extensions

Extensions

Label scoring for unary chains:

- Split unary chains into top-middle-bottom

Extensions

Label scoring for unary chains:

- Split unary chains into top-middle-bottom

Structured label loss for unary chains:

- Hamming distance on labels (vs. 0-1 loss)

Extensions

Label scoring for unary chains:

- Split unary chains into top-middle-bottom

Structured label loss for unary chains:

- Hamming distance on labels (vs. 0-1 loss)

Split-based (vs. span-based) scoring:

- Left-right, concatenate, deep biaffine

[Cross and Huang (2016)] [Dozat and Manning (2016)]

Final Results

Parser	F1 Score
Hall et al. (2014)	89.2
Vinyals et al. (2015)	88.3
Cross and Huang (2016)	91.3
Dyer et al. (2016)	91.7
Liu and Zhang (2016)	91.7
Our Best Chart Parser	91.8
Our Best Top-Down Parser	91.8

Conclusion

Conclusion

A minimal span-based parser can achieve state-of-the-art results.

Conclusion

A minimal span-based parser can achieve state-of-the-art results.

Little is lost going from global to greedy decoding.

Conclusion

A minimal span-based parser can achieve state-of-the-art results.

Little is lost going from global to greedy decoding.

Various extensions yield only minimal gains beyond the core system.

Thanks!