61A Lecture 37

Wednesday, December 3

Announcements

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm
- No videos for Lecture 38 on Friday 12/5

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm
- No videos for Lecture 38 on Friday 12/5
  - Come to class and take the final survey

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm
- No videos for Lecture 38 on Friday 12/5
  - Come to class and take the final survey
  - There will be a screencast of live lecture (http://goo.gl/hyUTca)

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm
- No videos for Lecture 38 on Friday 12/5
  - Come to class and take the final survey
  - There will be a screencast of live lecture (http://goo.gl/hyUTca)
- Final exam held on Thursday 12/18 3pm–6pm

Announcements

- Homework 10 due Wednesday 12/3 @ 11:59pm
- Quiz 3 released Wednesday, due Thursday 12/4 @ 11:59pm
- No videos for Lecture 38 on Friday 12/5
  - Come to class and take the final survey
  - There will be a screencast of live lecture (http://goo.gl/hyUTca)
- Final exam held on Thursday 12/18 3pm–6pm
  - 30 hours of review sessions next week! Monday – Friday 11am–6pm (mostly in 271 Soda)

Ambiguity

Syntactic Ambiguity in English

```
Programs must be written for people to read
```

Syntactic Ambiguity in English

Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

Syntactic Ambiguity in English

Sentence

Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

Syntactic Ambiguity in English

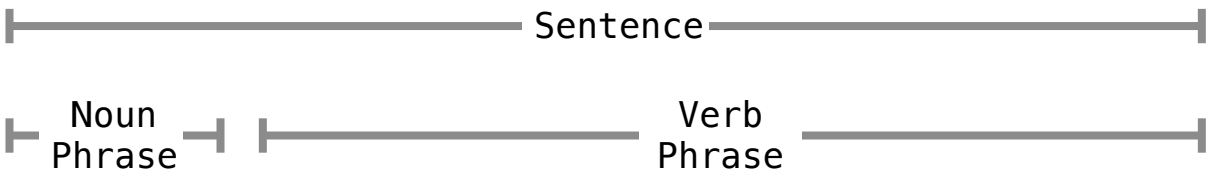
Sentence

Noun Phrase

Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

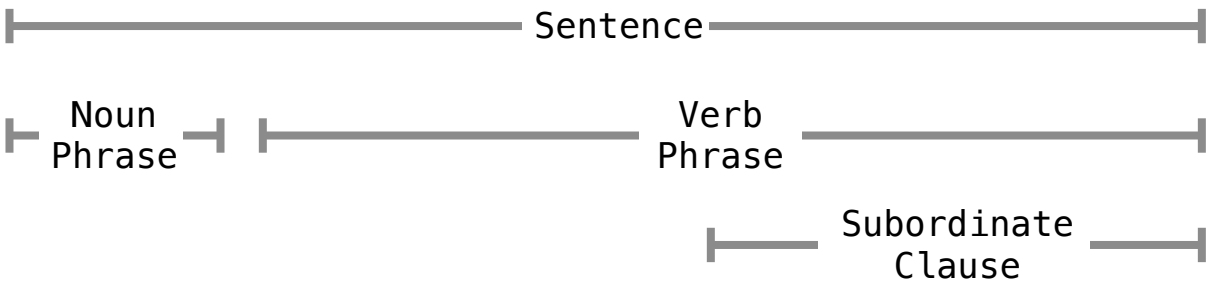
Syntactic Ambiguity in English



Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

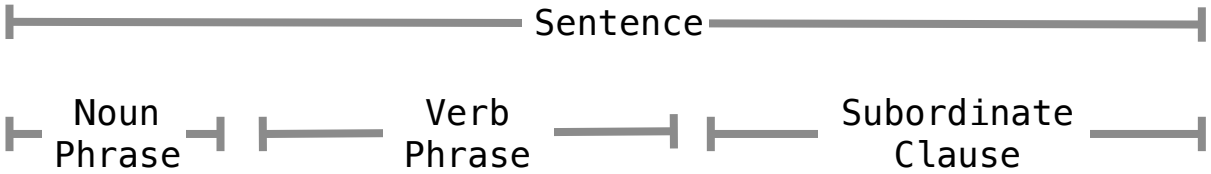
Syntactic Ambiguity in English



Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

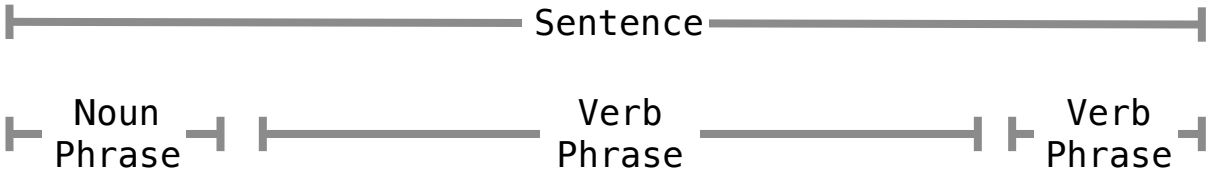
Syntactic Ambiguity in English



Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

Syntactic Ambiguity in English



Programs must be written for people to read[1]

[1]Preface of **Structure and Interpretation of Computer Programs**
by Harold Abelson and Gerald Sussman with Julie Sussman

Syntactic Ambiguity in English

pro•gram (noun)
  a series of coded software instructions

pro•gram (verb)
  provide a computer with coded instructions

Programs must be written for people to read

must (verb)
  be obliged to

must (noun)
  dampness or mold

Definitions from the New Oxford American Dictionary

Syntax Trees

Representing Syntactic Structure

Representing Syntactic Structure

```
buffalo
```

Representing Syntactic Structure

Representing Syntactic Structure

Noun

buffalo

Representing Syntactic Structure

Noun

buffalo buffalo

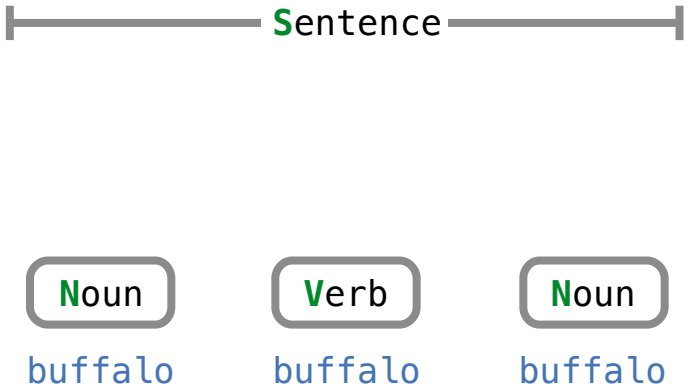
Representing Syntactic Structure

Noun	Verb
buffalo	buffalo

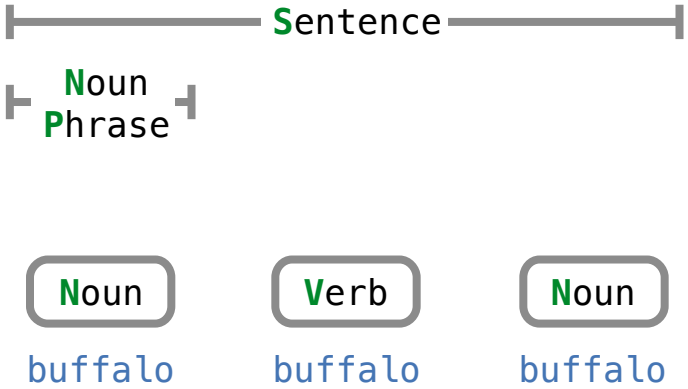
Representing Syntactic Structure

Noun	Verb	Noun
buffalo	buffalo	buffalo

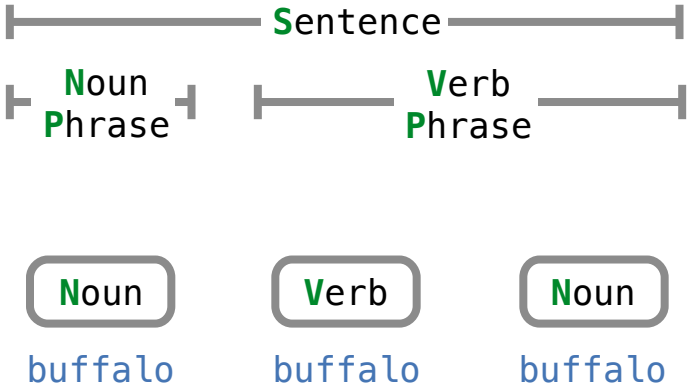
Representing Syntactic Structure



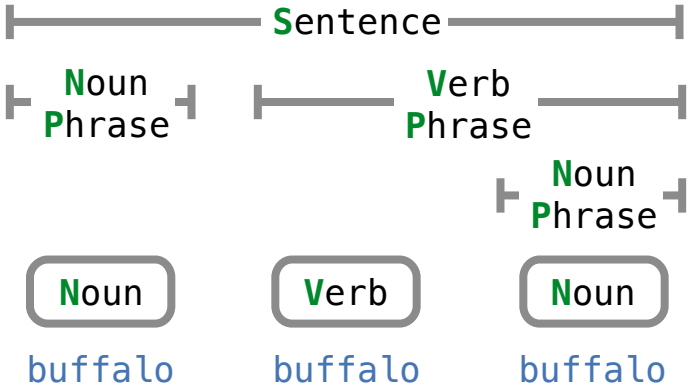
Representing Syntactic Structure



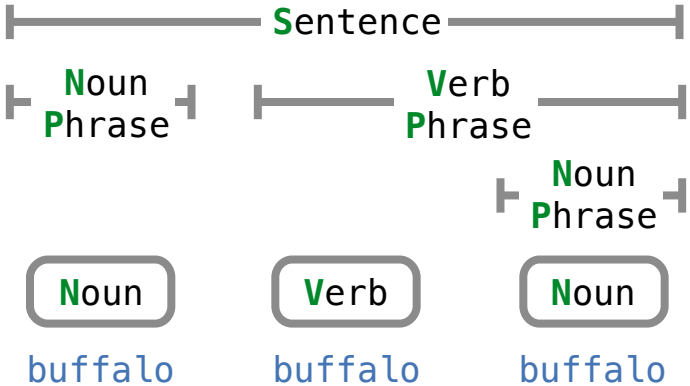
Representing Syntactic Structure



Representing Syntactic Structure



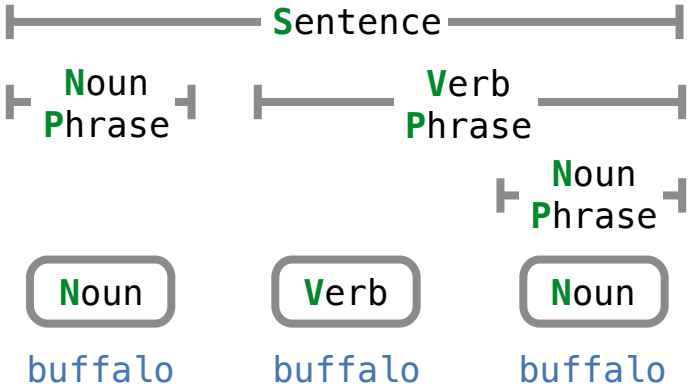
Representing Syntactic Structure



A **Tree** represents a phrase:

Representing Syntactic Structure

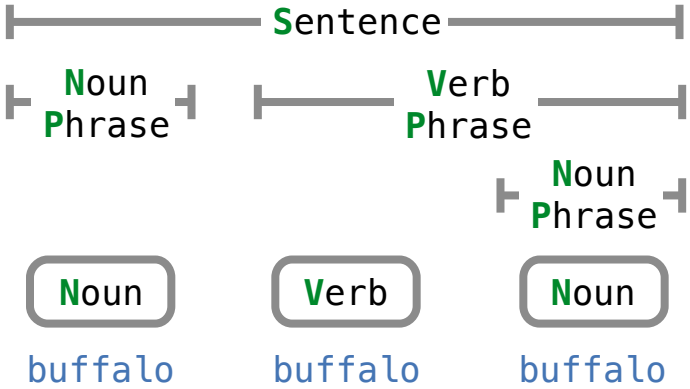
Photo by Vince O'Sullivan licensed under
http://creativecommons.org/licenses/by-nc-nd/2.0/



A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)

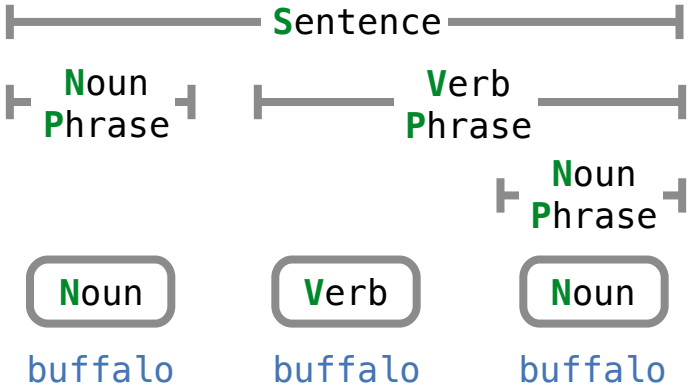
Representing Syntactic Structure



A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

Representing Syntactic Structure

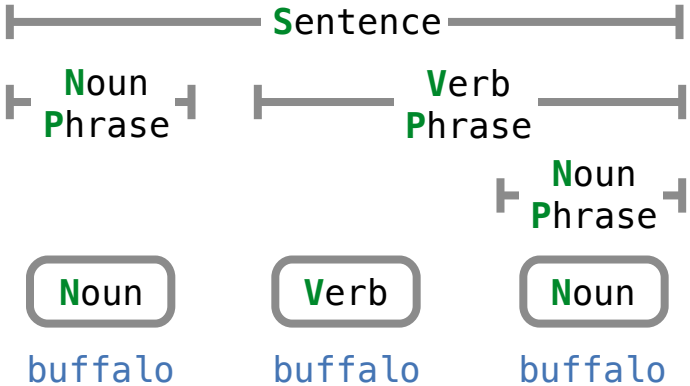


A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

Representing Syntactic Structure



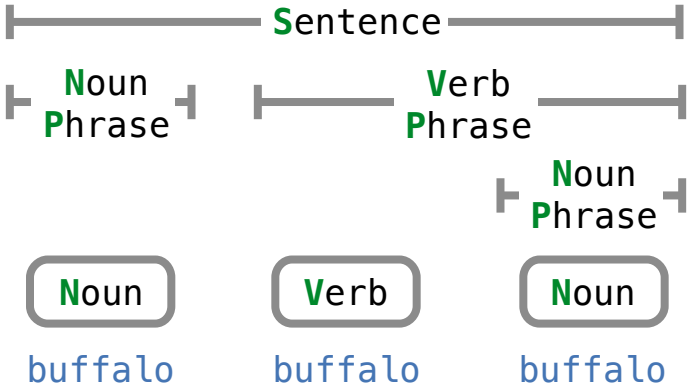
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)

Representing Syntactic Structure



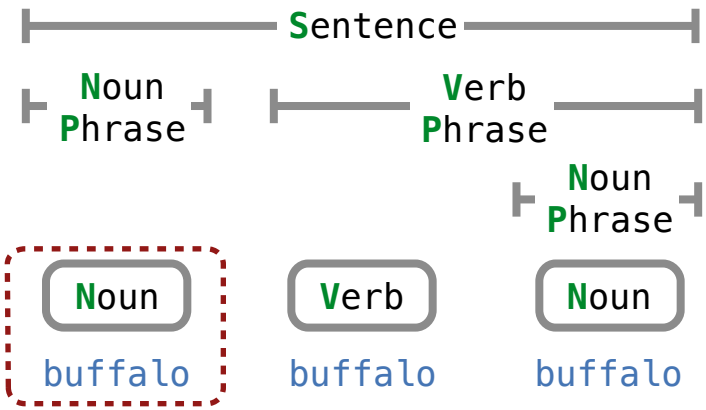
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



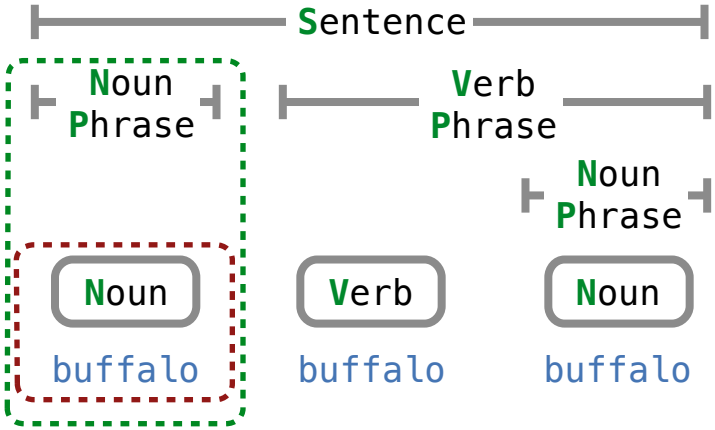
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



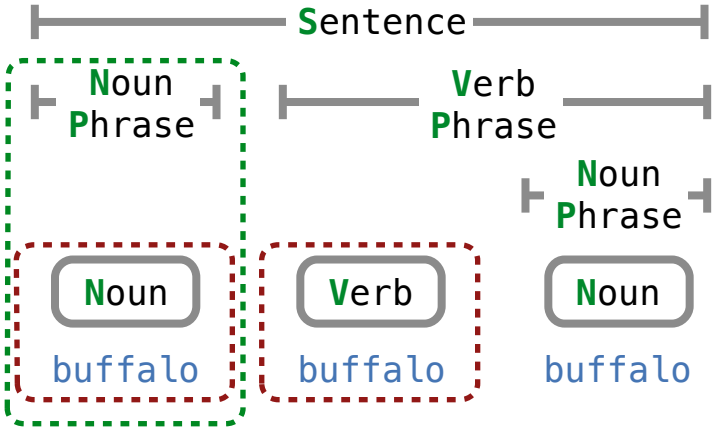
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



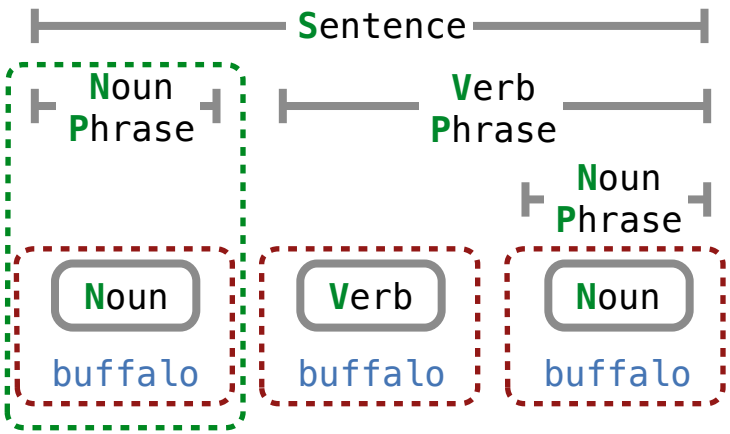
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



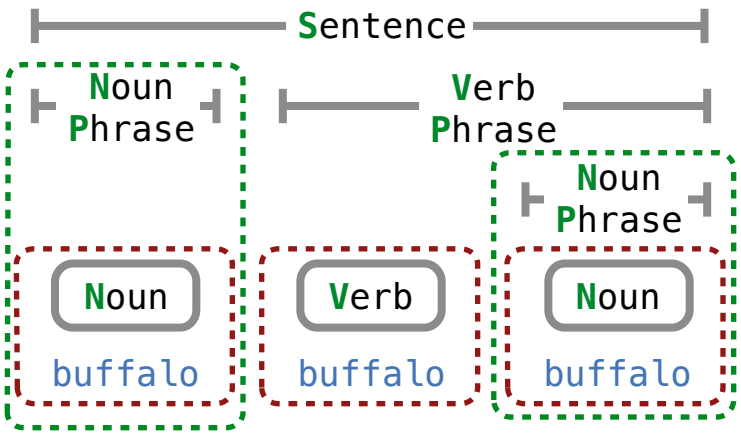
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



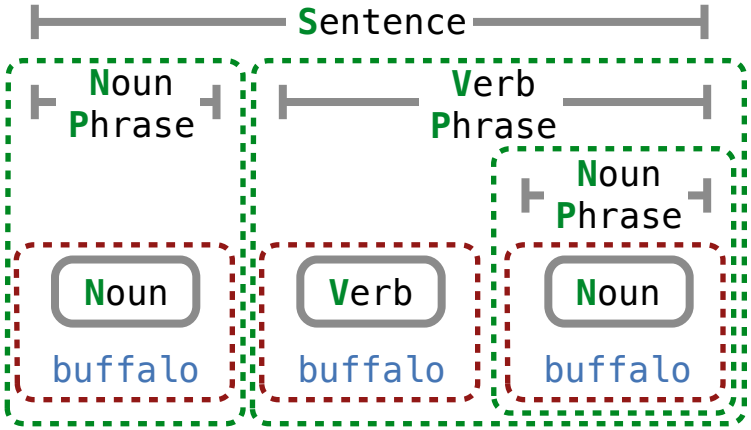
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



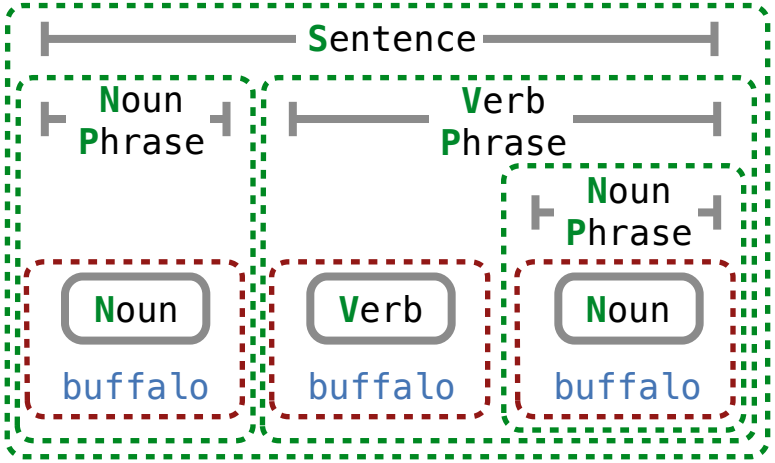
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



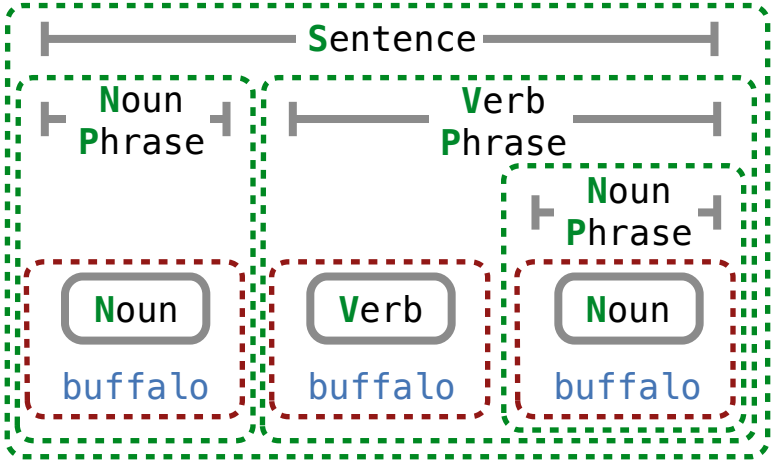
A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

Representing Syntactic Structure



A **Tree** represents a phrase:

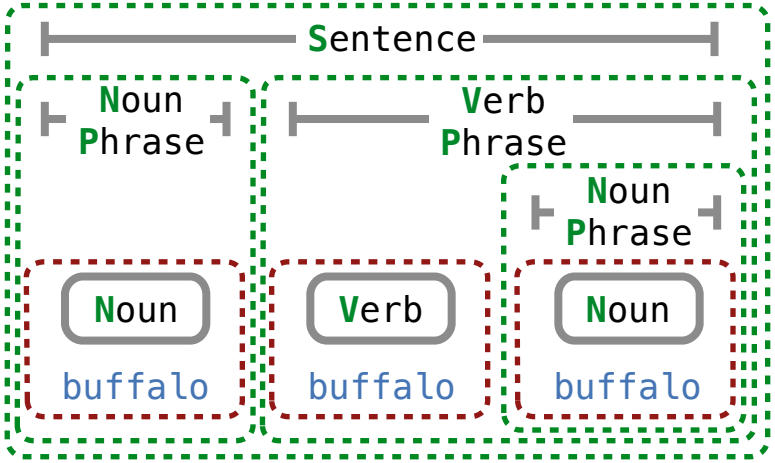
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
```

Representing Syntactic Structure



A **Tree** represents a phrase:

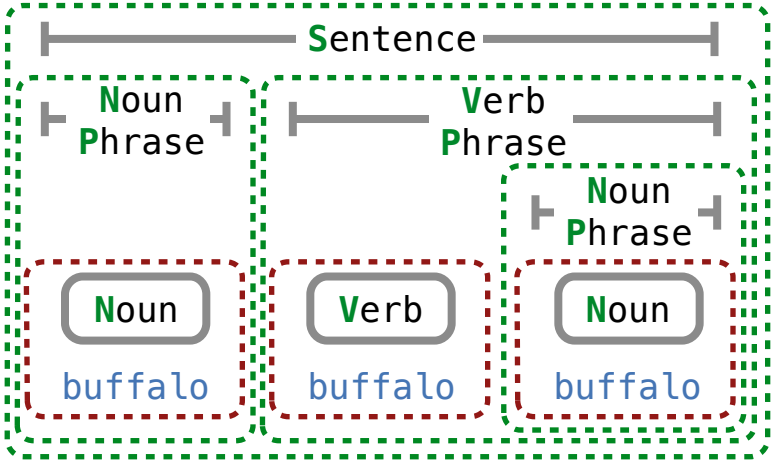
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
```

Representing Syntactic Structure



A **Tree** represents a phrase:

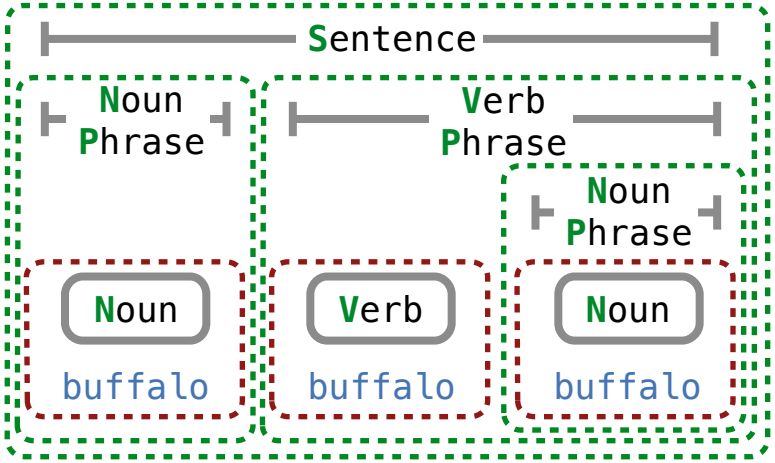
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
S, NP, VP = 'S', 'NP', 'VP'
```

Representing Syntactic Structure



A **Tree** represents a phrase:

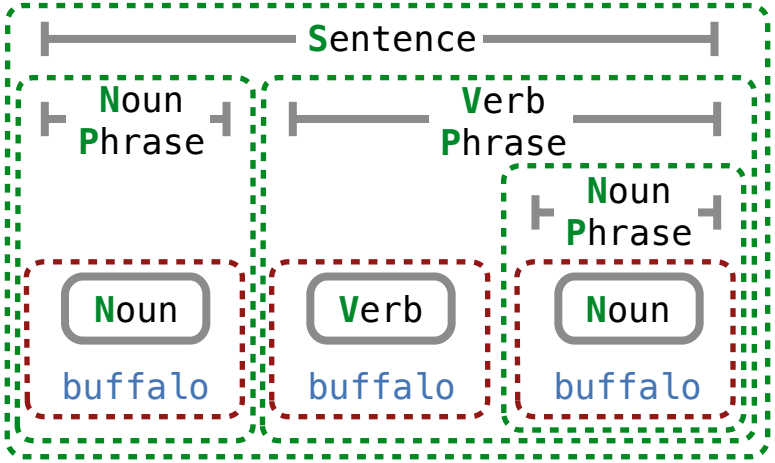
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
S, NP, VP = 'S', 'NP', 'VP'
Tree(S, [Tree(NP, [beasts]),
```

Representing Syntactic Structure



A **Tree** represents a phrase:

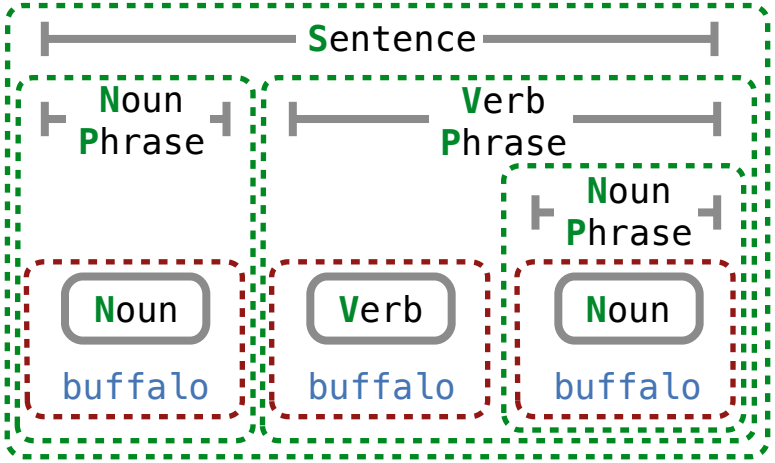
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
S, NP, VP = 'S', 'NP', 'VP'
Tree(S, [Tree(NP, [beasts]),
         Tree(VP, [intimidate,
```

Representing Syntactic Structure



A **Tree** represents a phrase:

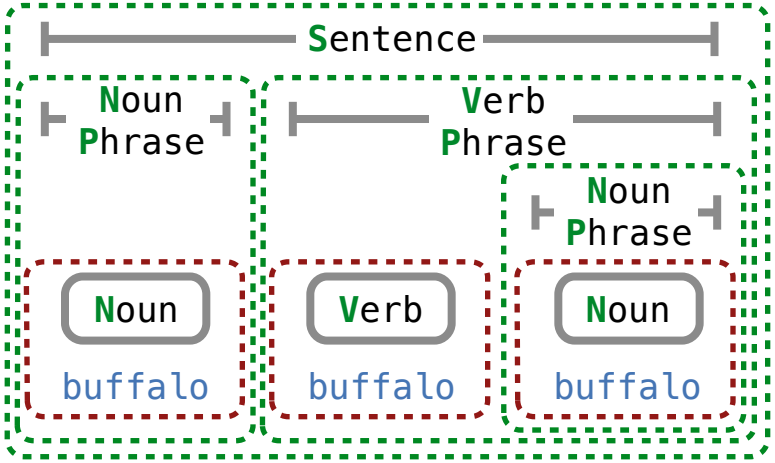
- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
S, NP, VP = 'S', 'NP', 'VP'
Tree(S, [Tree(NP, [beasts]),
         Tree(VP, [intimidate,
                   Tree(NP, [beasts])])])
```

Representing Syntactic Structure



A **Tree** represents a phrase:

- **tag** -- What kind of phrase (e.g., **S**, **NP**, **VP**)
- **branches** -- Sequence of **Tree** or **Leaf** components

A **Leaf** represents a single word:

- **tag** -- What kind of word (e.g., **N**, **V**)
- **word** -- The word

```
beasts = Leaf('N', 'buffalo')
intimidate = Leaf('V', 'buffalo')
S, NP, VP = 'S', 'NP', 'VP'
Tree(S, [Tree(NP, [beasts]),
         Tree(VP, [intimidate,
                   Tree(NP, [beasts])])])
```

(Demo)

Grammars

Context-Free Grammar Rules

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

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

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → **NP** **VP**

A **S**entence ...

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → **NP** **VP**

A **S**entence ...

... can be expanded as ...

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → **NP VP**

A **S**entence ...

... can be expanded as ...

... a **N**oun **P**hrase then a **V**erb **P**hrase.

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → **NP VP**

A **S**entence ...

... can be expanded as ...

... a **N**oun **P**hrase then a **V**erb **P**hrase.

Grammar

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S ⟶ **NP VP**

A **S**entence ...

... can be expanded as ...

... a **N**oun **P**hrase then a **V**erb **P**hrase.

Grammar

S ⟶ **NP VP**

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S ⟶ **NP VP**

A **S**entence ...

... can be expanded as ...

... a **N**oun **P**hrase then a **V**erb **P**hrase.

S

Grammar

S ⟶ **NP VP**

Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S ⟶ **NP VP**

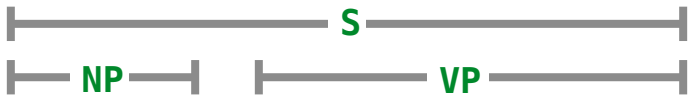
A **S**entence ...

... can be expanded as ...

... a **N**oun **P**hrase then a **V**erb **P**hrase.

Grammar

S ⟶ **NP VP**



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

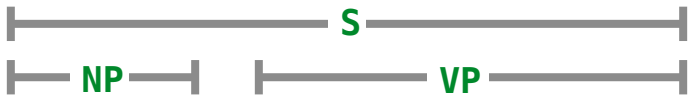
... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP

NP → N



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

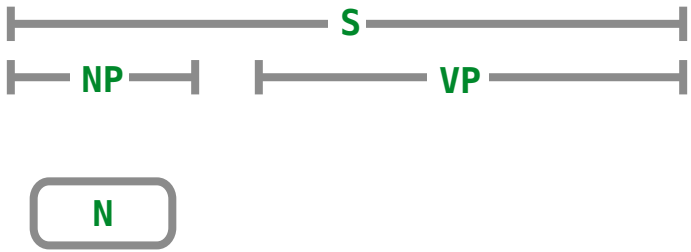
... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP

NP → N



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

... can be expanded as ...

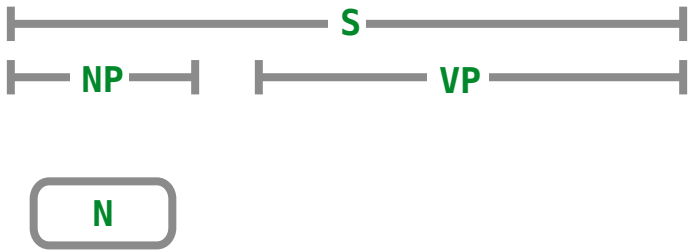
... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP

NP → N

N → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

... can be expanded as ...

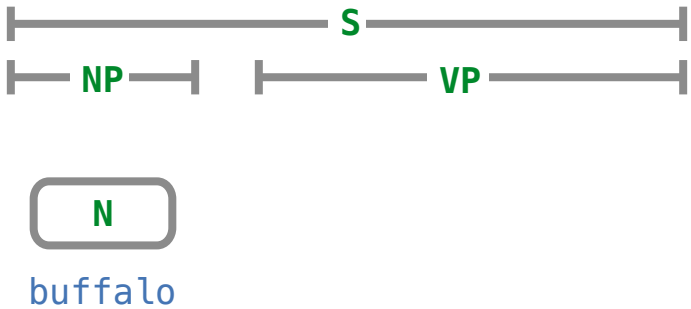
... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP

NP → N

N → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

... can be expanded as ...

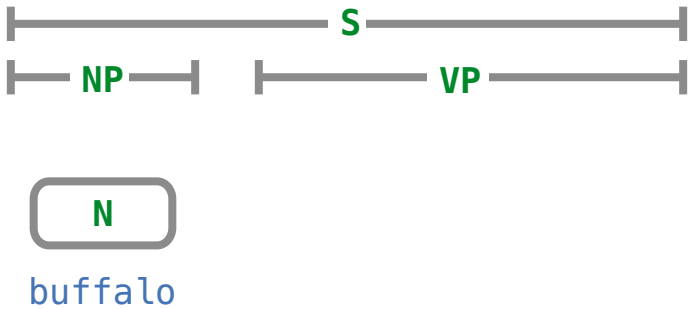
... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP

NP → N

N → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

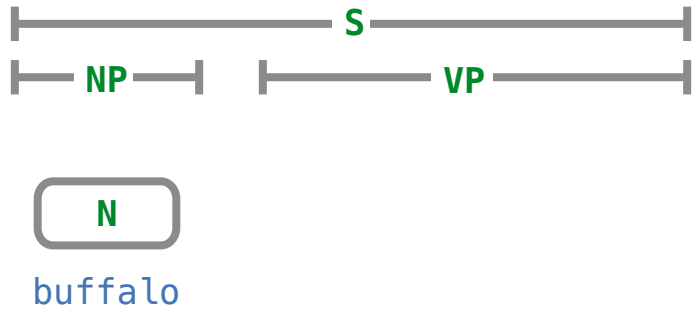
Grammar

S → NP VP

NP → N

N → buffalo

VP → V NP



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

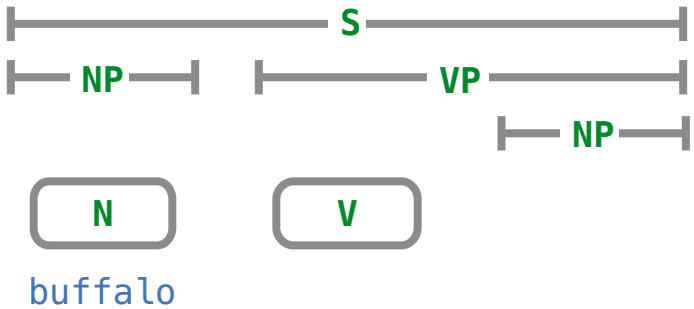
Grammar

S → NP VP

NP → N

N → buffalo

VP → V NP



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

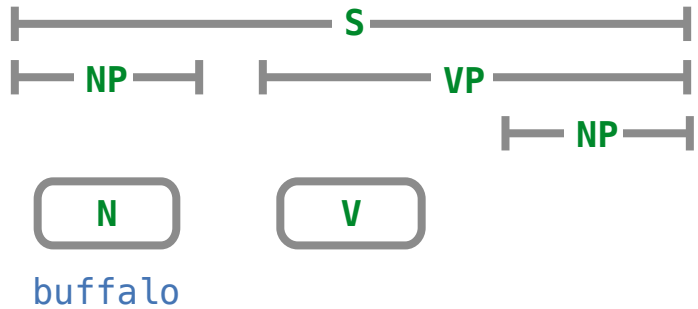
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

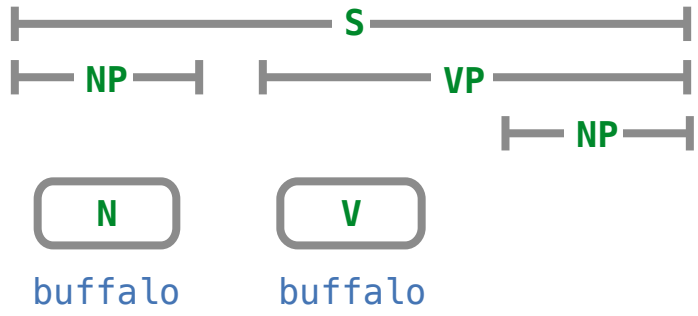
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

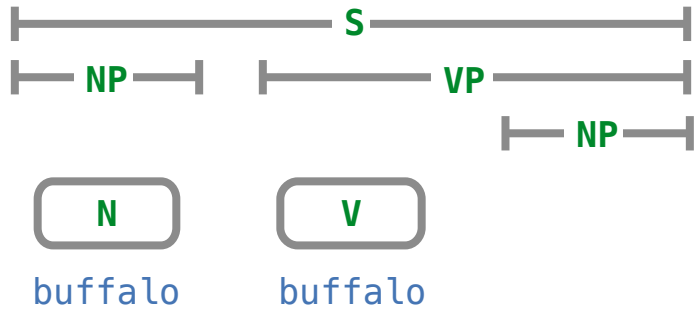
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

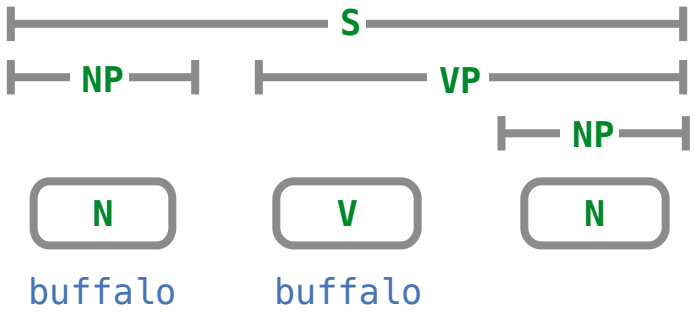
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

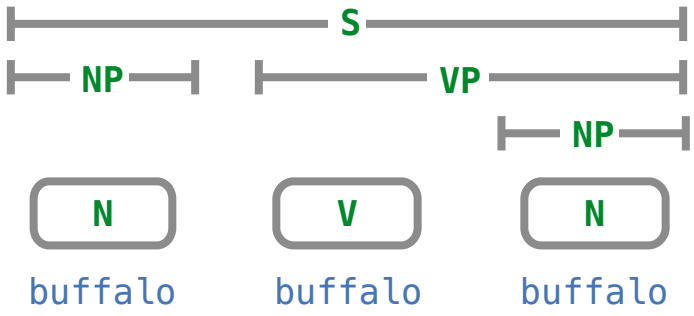
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

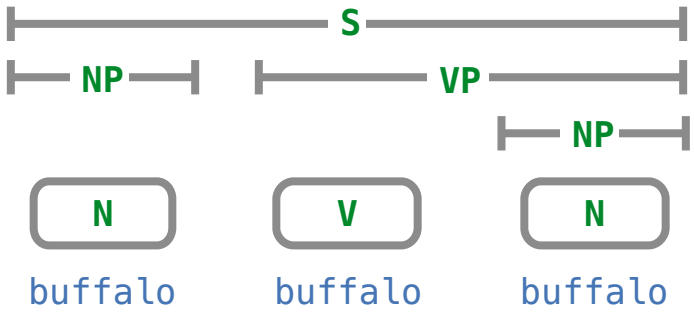
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



Context-Free Grammar Rules

A grammar rule describes how a tag can be expanded as a sequence of tags or words

S → NP VP

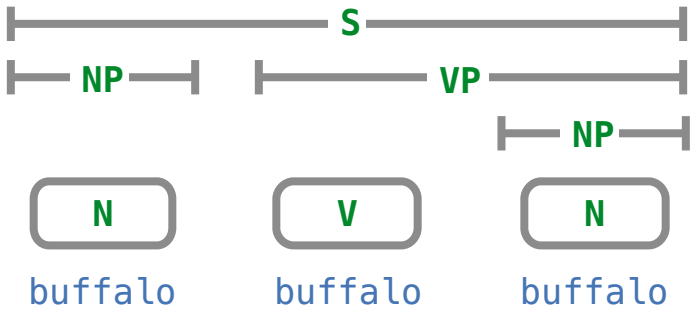
A Sentence ...

... can be expanded as ...

... a Noun Phrase then a Verb Phrase.

Grammar

S → NP VP
NP → N
N → buffalo
VP → V NP
V → buffalo



(Demo)

Parsing

Exhaustive Parsing

```
Expand all tags recursively, but constrain words to match input
```

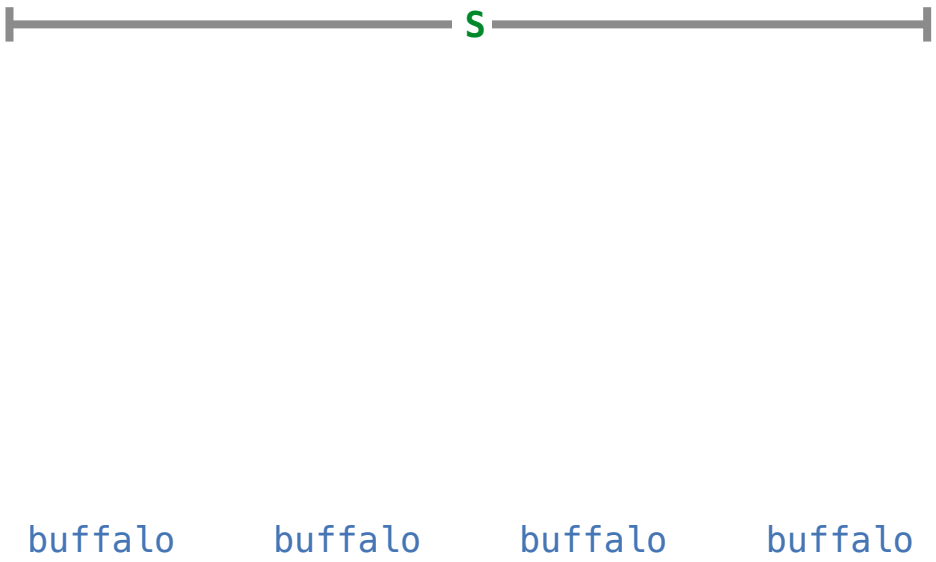
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input

buffalo buffalo buffalo buffalo

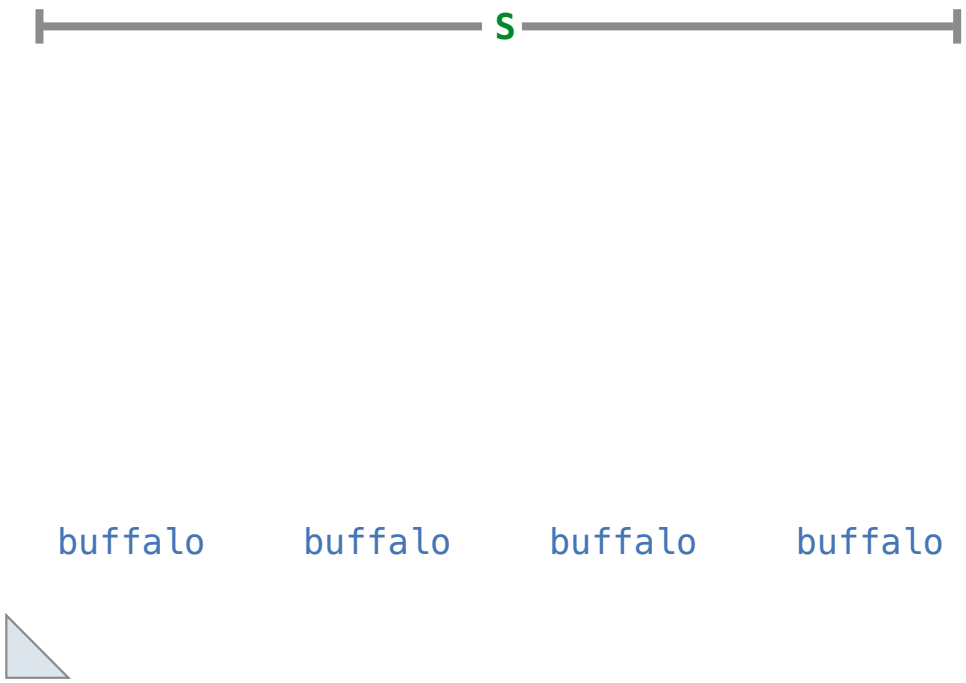
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



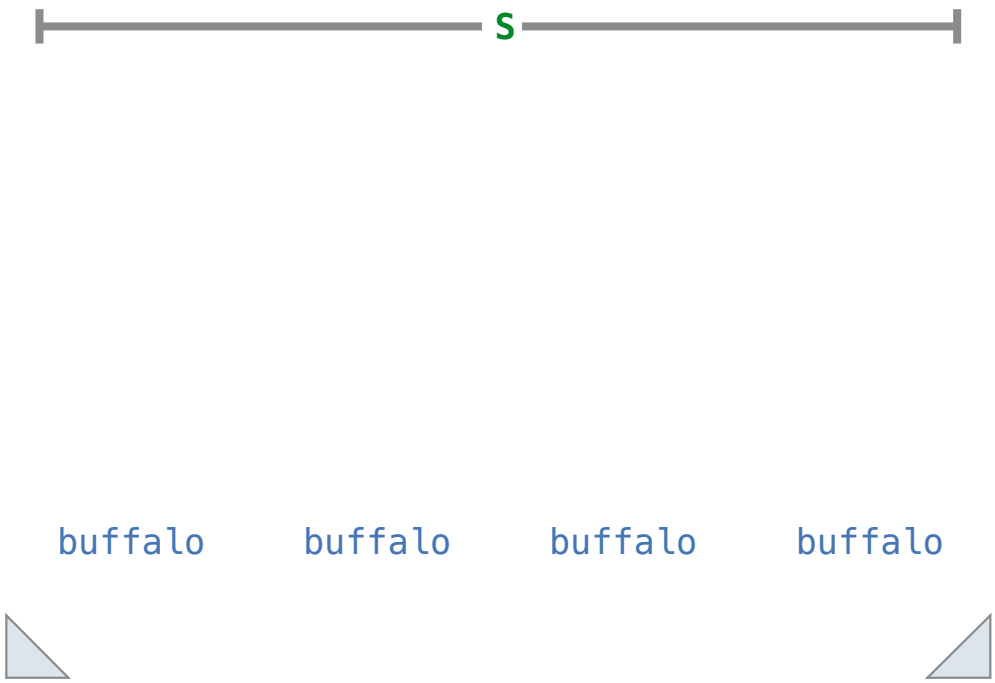
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



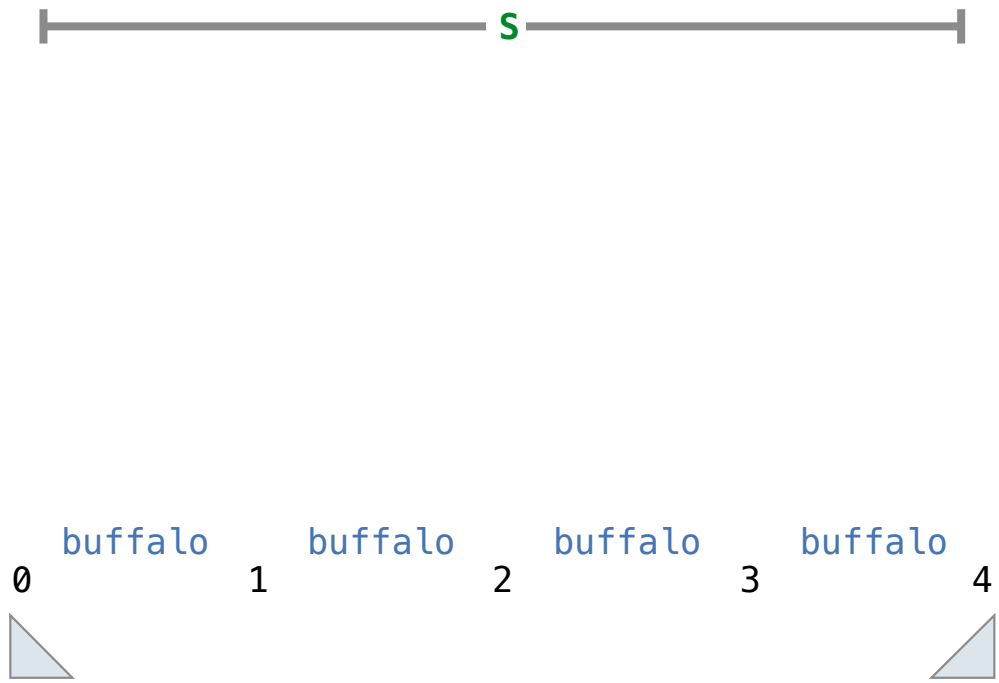
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



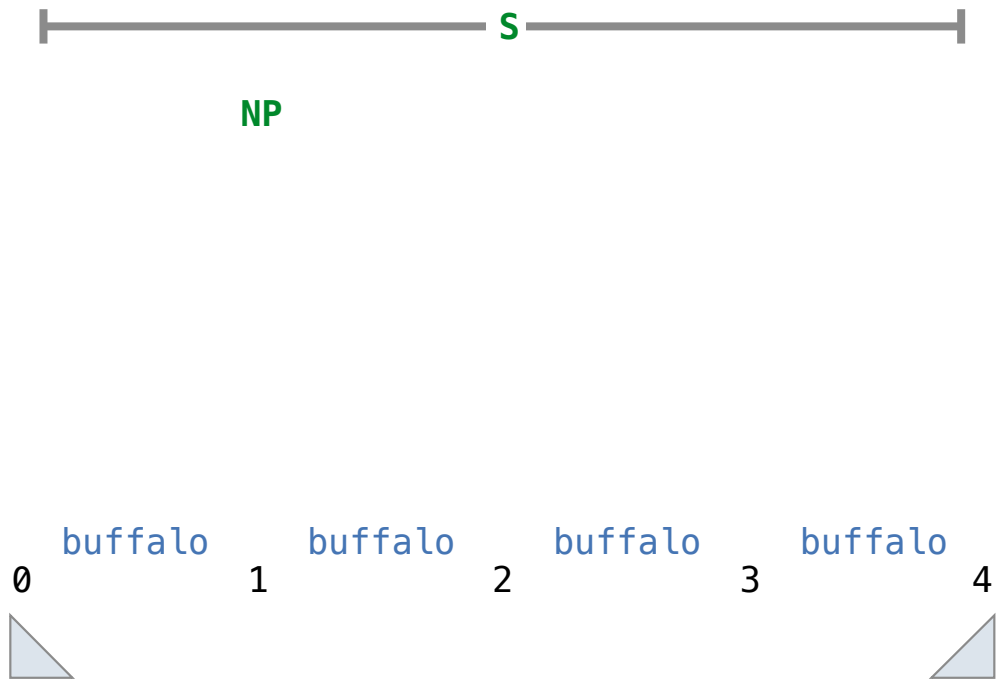
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



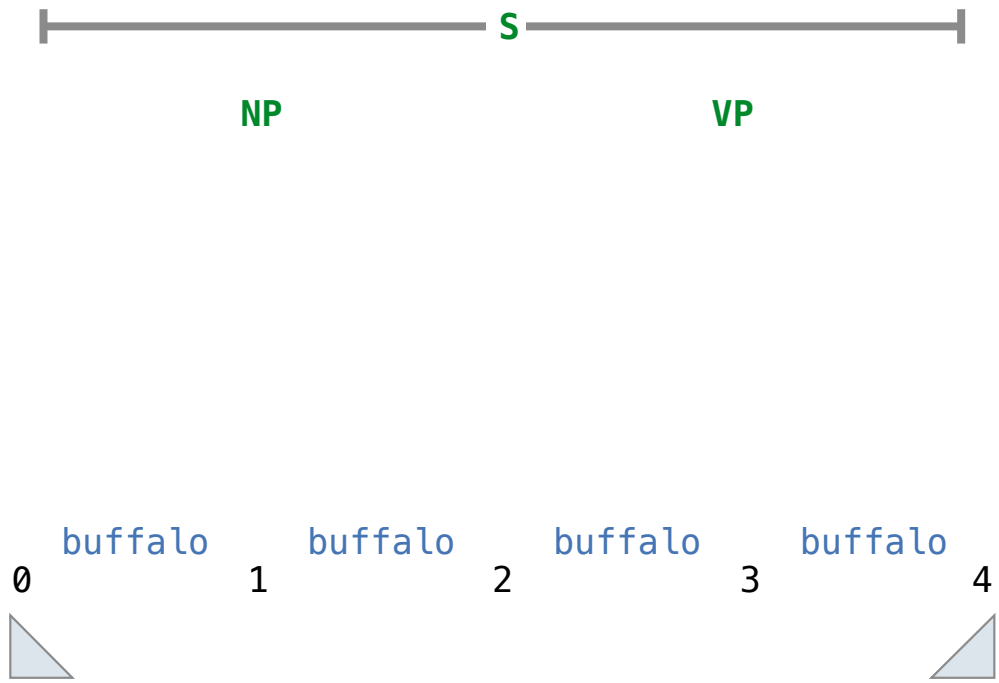
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



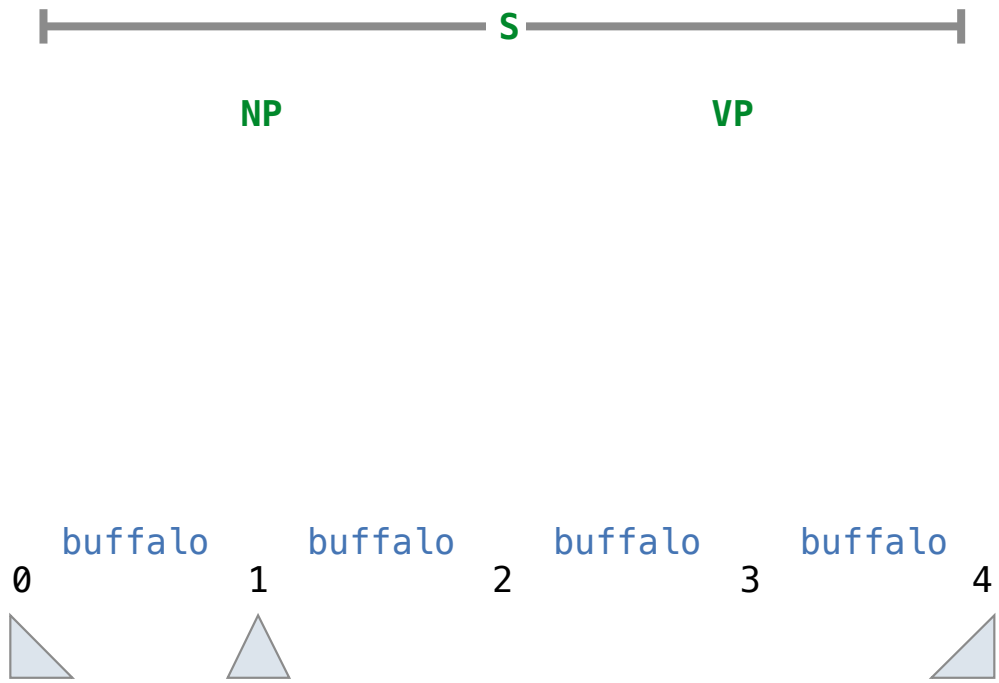
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



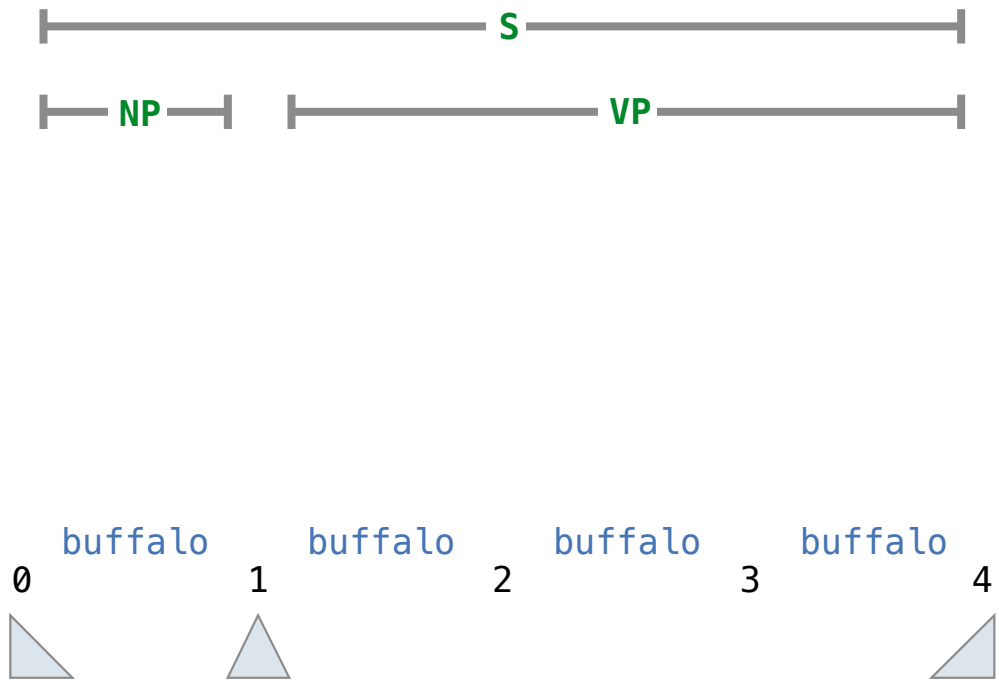
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



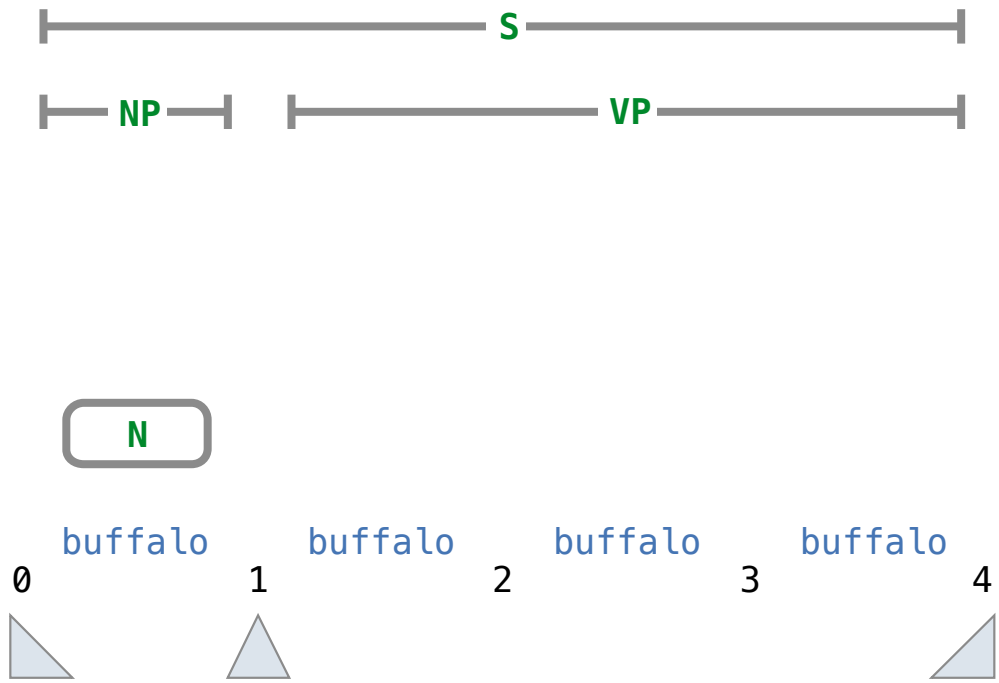
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



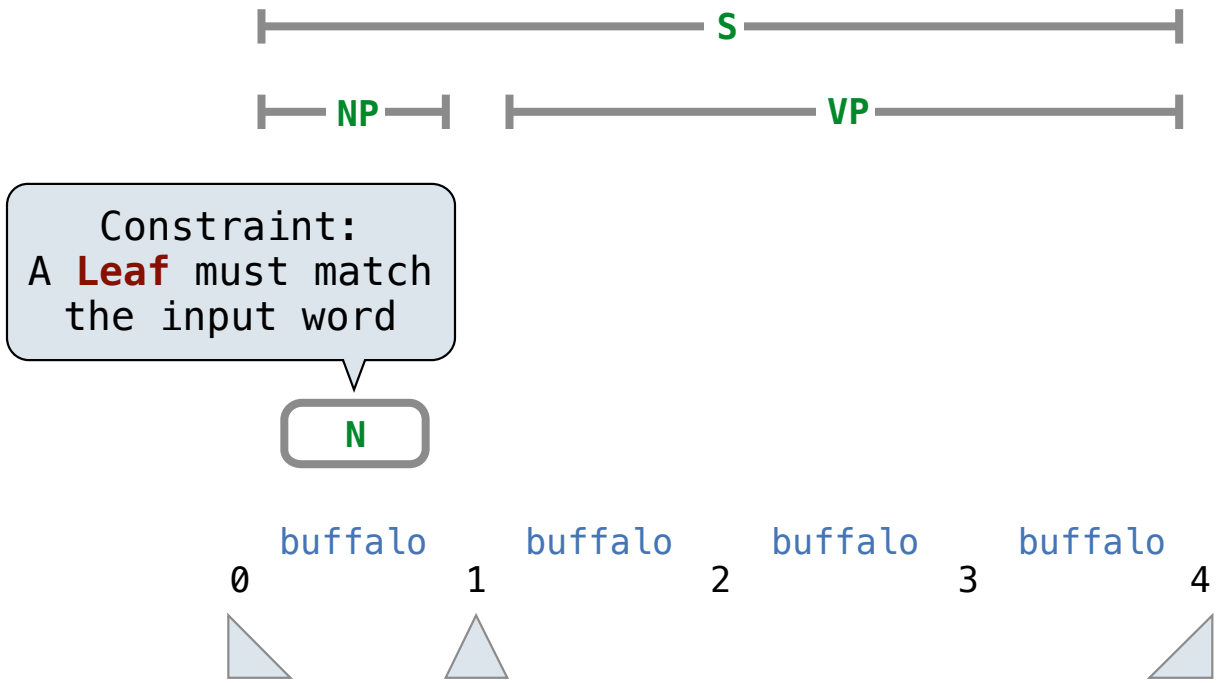
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



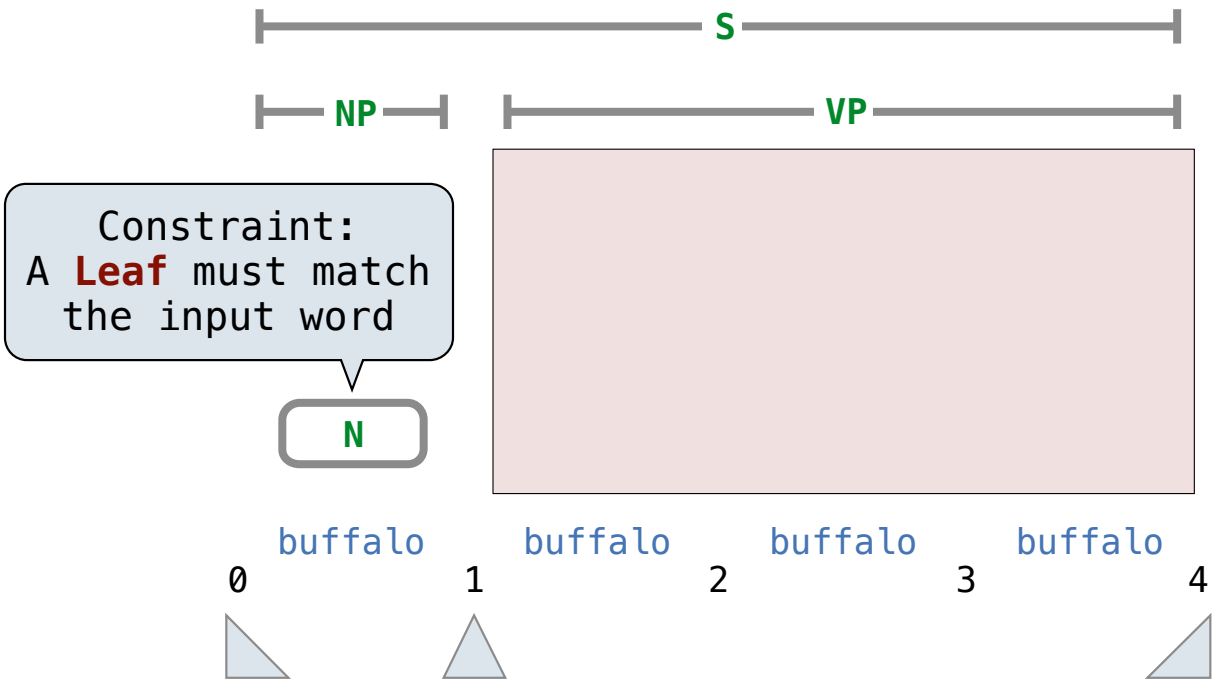
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



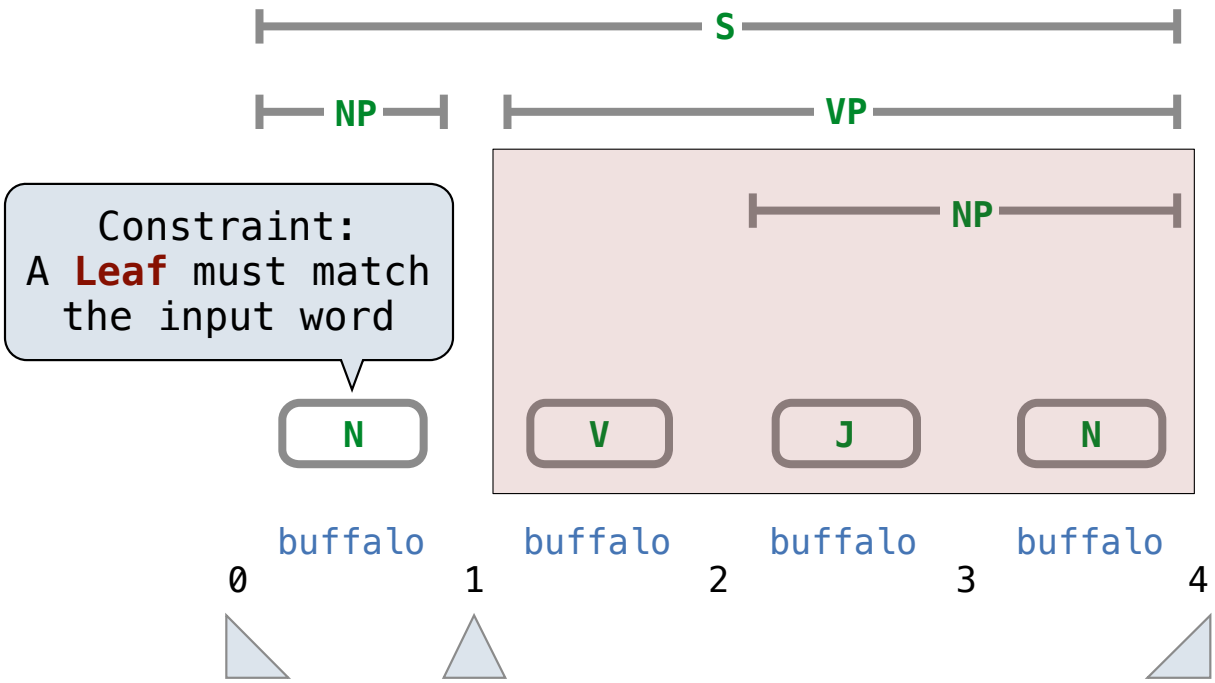
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



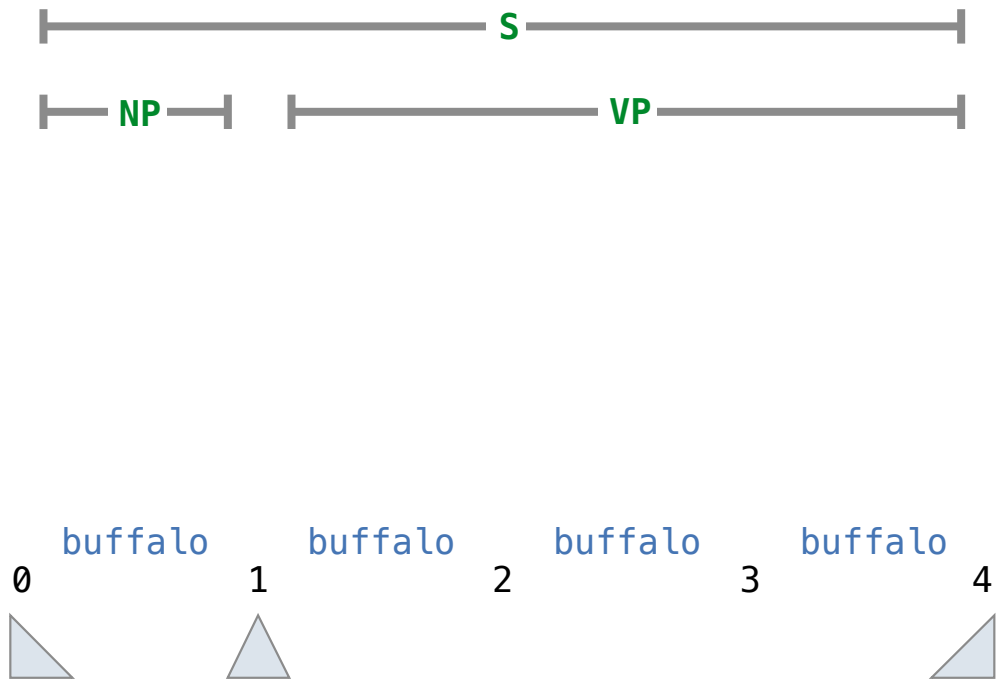
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



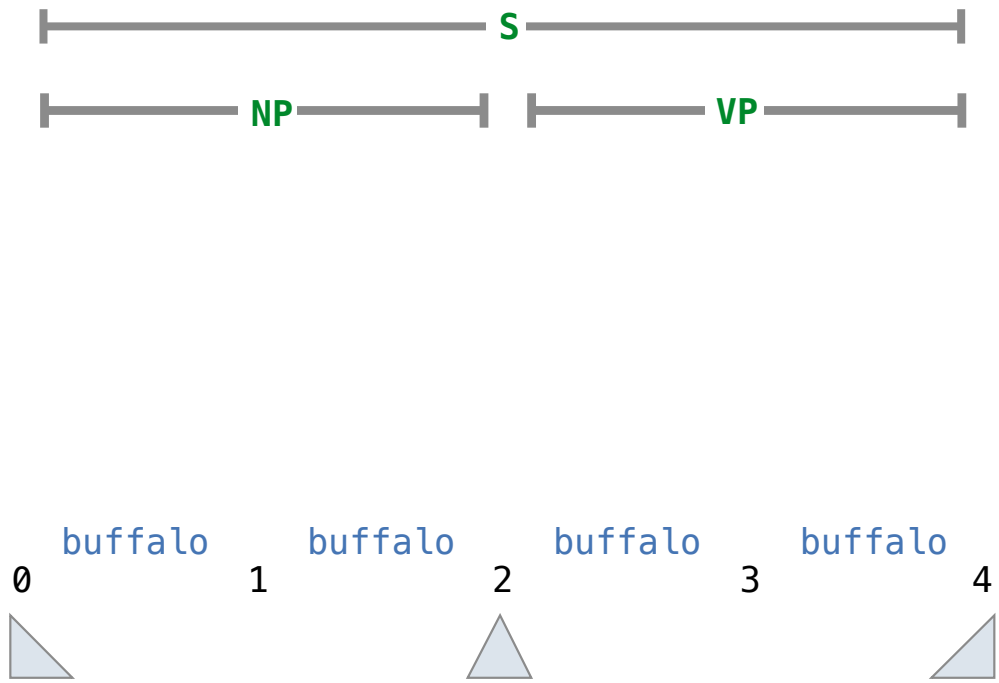
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



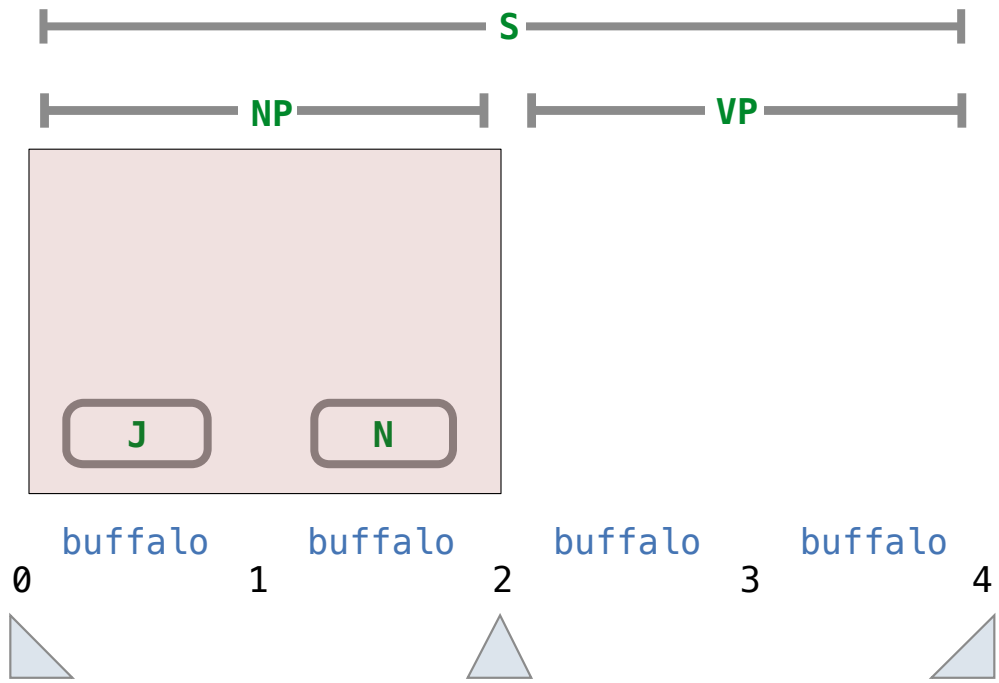
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



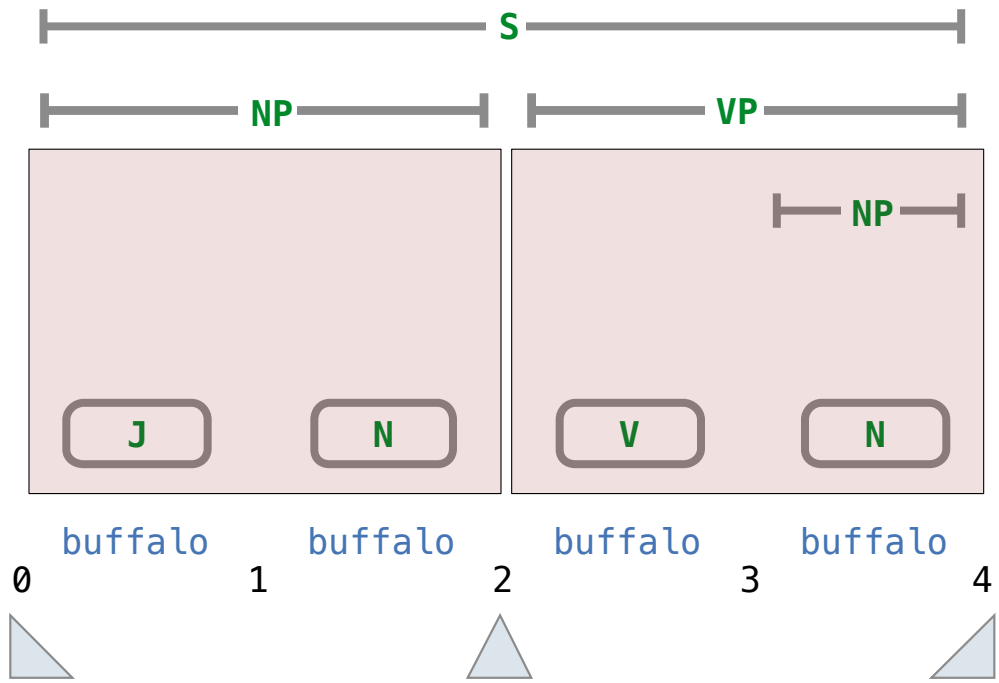
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



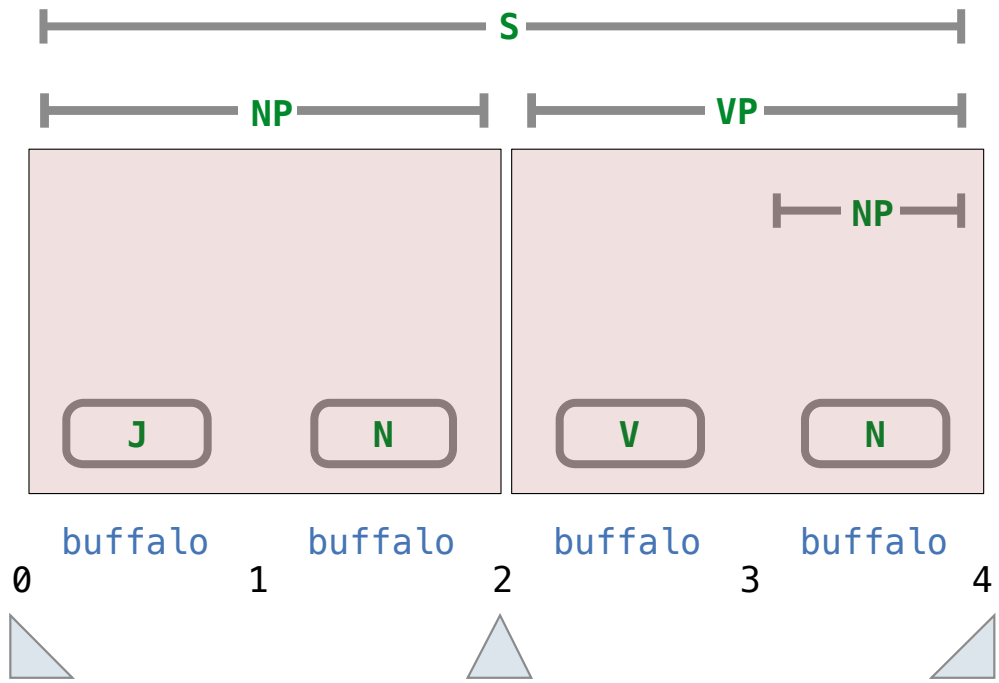
Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



Exhaustive Parsing

Expand all tags recursively, but constrain words to match input



(Demo)

Learning

(Demo)

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

teacher strikes idle kids

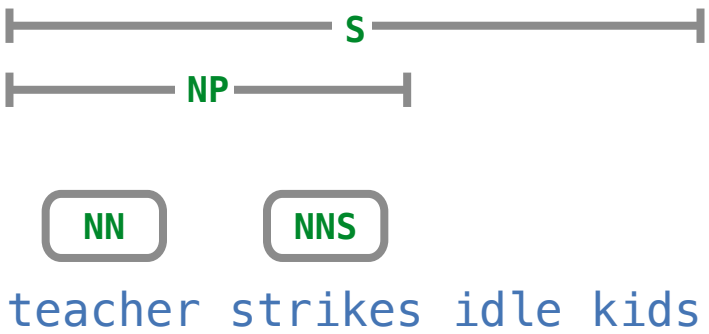
Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

teacher strikes idle kids

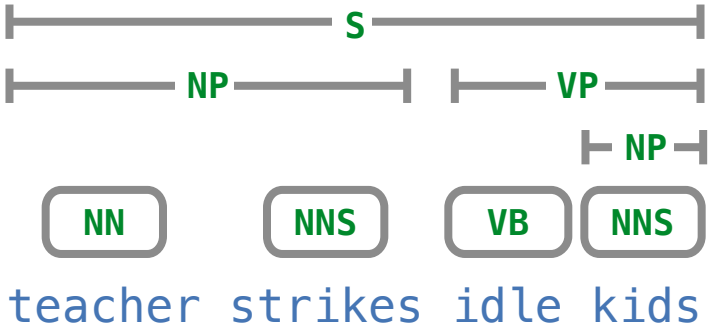
Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common



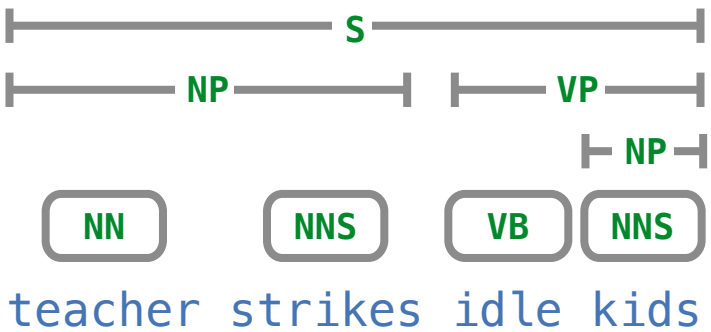
Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common



Scoring a Tree Using Relative Frequencies

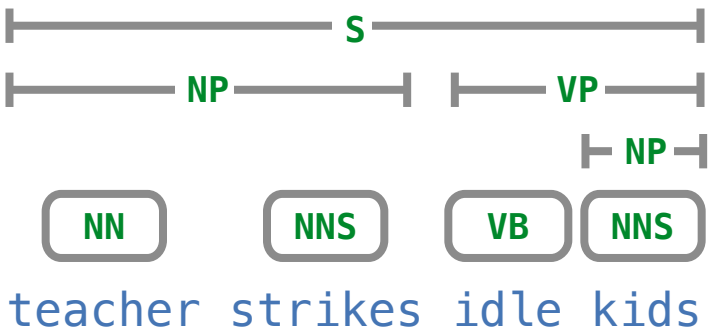
Not all syntactic structures are equally common



S	⟶	NP VP	NN	⟶	teacher
NP	⟶	NN NNS	NNS	⟶	strikes
VP	⟶	VB NP	VB	⟶	idle
NP	⟶	NNS	NNS	⟶	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

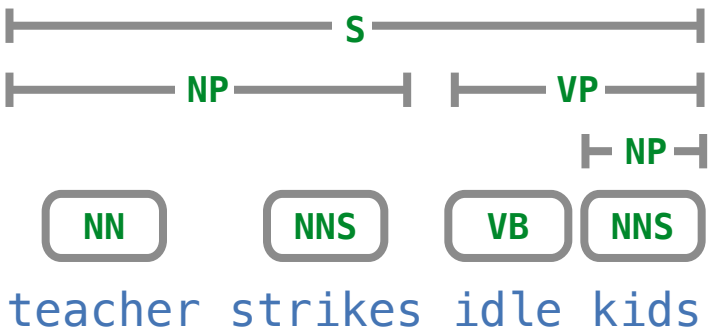


Rule frequency per 100,000 tags

S → NP VP		NN →	teacher
NP → NN NNS		NNS →	strikes
VP → VB NP		VB →	idle
NP → NNS		NNS →	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

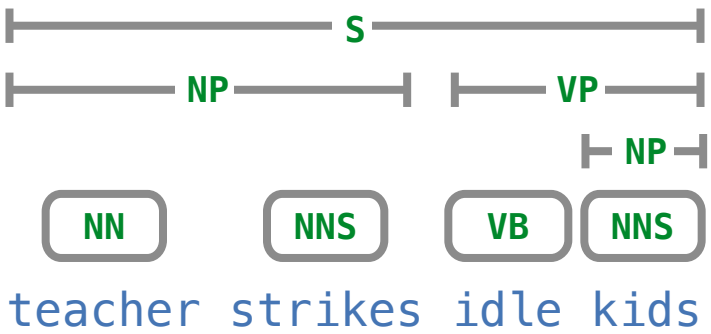


Rule frequency per 100,000 tags

S ⟶ NP VP	25372	NN ⟶	teacher
NP ⟶ NN NNS		NNS ⟶	strikes
VP ⟶ VB NP		VB ⟶	idle
NP ⟶ NNS		NNS ⟶	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

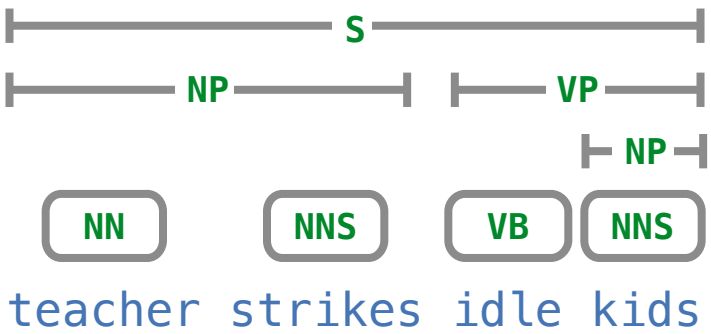


Rule frequency per 100,000 tags

S ⟶ NP VP	25372	NN ⟶	teacher
NP ⟶ NN NNS	1335	NNS ⟶	strikes
VP ⟶ VB NP		VB ⟶	idle
NP ⟶ NNS		NNS ⟶	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

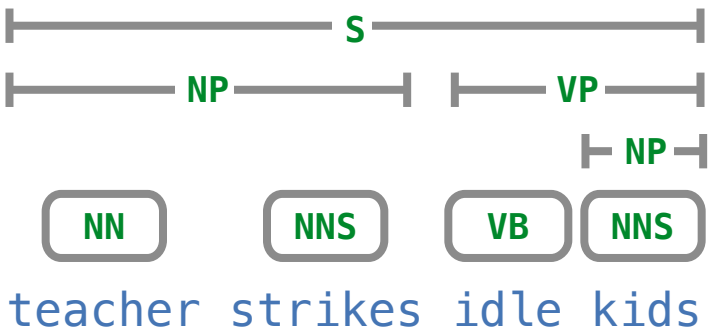


Rule frequency per 100,000 tags

S ⟶ NP VP	25372	NN ⟶	teacher
NP ⟶ NN NNS	1335	NNS ⟶	strikes
VP ⟶ VB NP	6679	VB ⟶	idle
NP ⟶ NNS		NNS ⟶	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

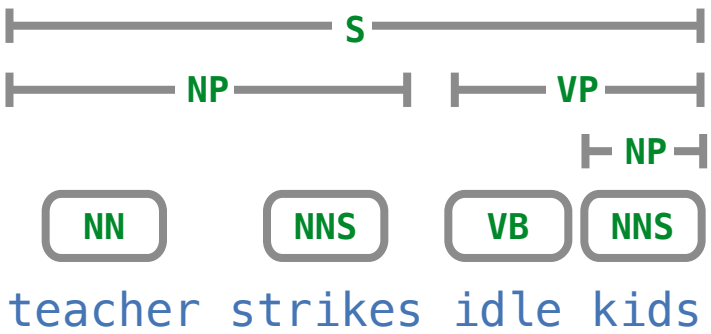


Rule frequency per 100,000 tags

S	⟶ NP VP	25372	NN	⟶	teacher
NP	⟶ NN NNS	1335	NNS	⟶	strikes
VP	⟶ VB NP	6679	VB	⟶	idle
NP	⟶ NNS	4282	NNS	⟶	kids

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

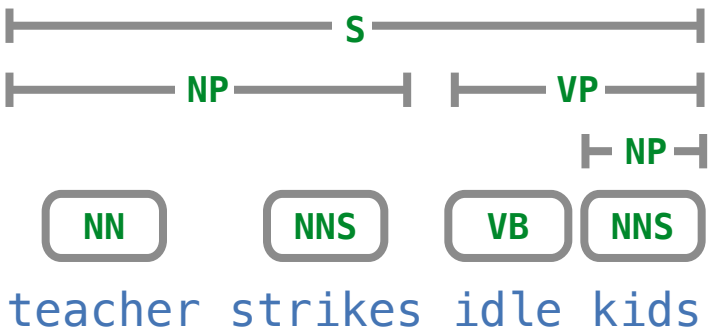


Rule frequency per 100,000 tags

S	⟶	NP VP	25372	NN	⟶	teacher	5
NP	⟶	NN NNS	1335	NNS	⟶	strikes	
VP	⟶	VB NP	6679	VB	⟶	idle	
NP	⟶	NNS	4282	NNS	⟶	kids	

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

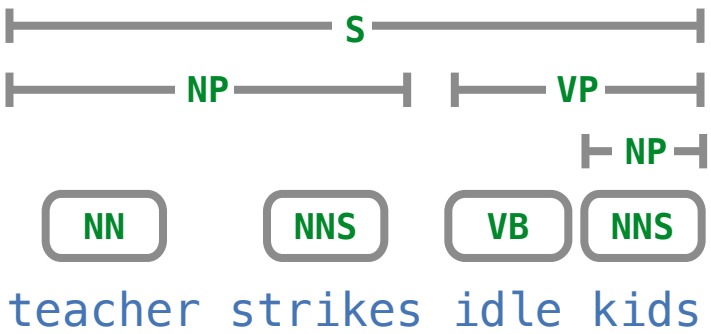


Rule frequency per 100,000 tags

S ⟶ NP VP	25372	NN ⟶	teacher	5
NP ⟶ NN NNS	1335	NNS ⟶	strikes	25
VP ⟶ VB NP	6679	VB ⟶	idle	
NP ⟶ NNS	4282	NNS ⟶	kids	

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

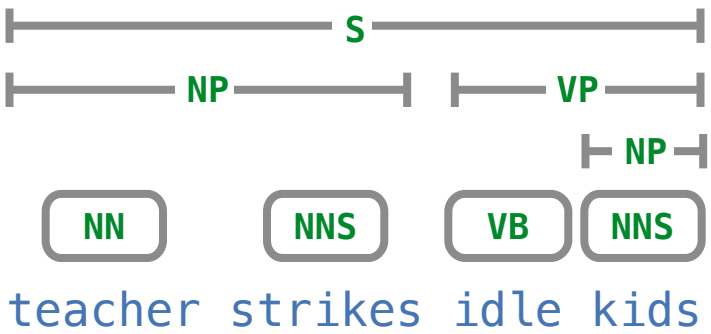


Rule frequency per 100,000 tags

S → NP VP	25372	NN →	teacher	5
NP → NN NNS	1335	NNS →	strikes	25
VP → VB NP	6679	VB →	idle	26
NP → NNS	4282	NNS →	kids	

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

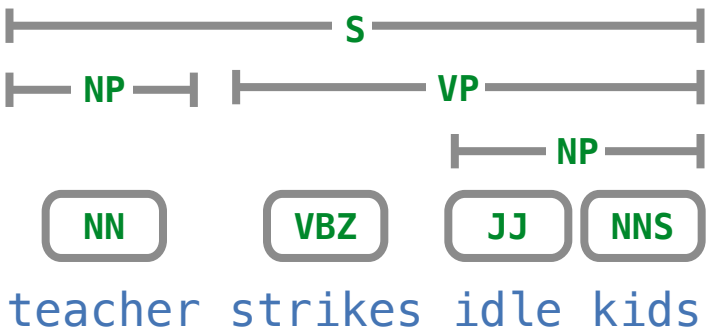


Rule frequency per 100,000 tags

S → NP VP	25372	NN →	teacher	5
NP → NN NNS	1335	NNS →	strikes	25
VP → VB NP	6679	VB →	idle	26
NP → NNS	4282	NNS →	kids	32

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common

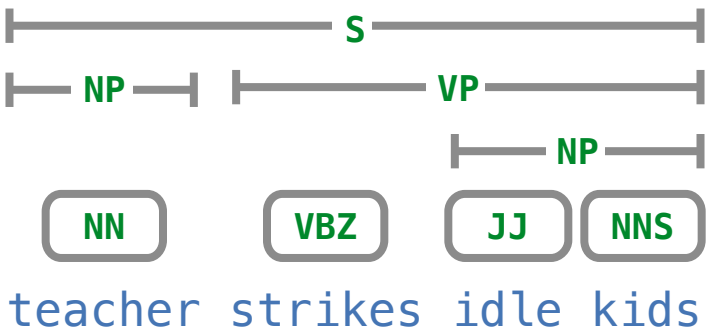


Rule frequency per 100,000 tags

S → NP VP	25372		NN →	teacher	5	
NP → NN	~~1335~~	4358	VBZ →	strikes	~~25~~	19
VP → VBZ NP	~~6679~~	3160	JJ →	idle	~~26~~	18
NP → JJ NNS	~~4282~~	2526	NNS →	kids	32	

Scoring a Tree Using Relative Frequencies

Not all syntactic structures are equally common



Rule frequency per 100,000 tags

S ⟶ NP VP	25372		NN ⟶	teacher	5	
NP ⟶ NN	~~1335~~	4358	VBZ ⟶	strikes	~~25~~	19
VP ⟶ VBZ NP	~~6679~~	3160	JJ ⟶	idle	~~26~~	18
NP ⟶ JJ NNS	~~4282~~	2526	NNS ⟶	kids	32	

(Demo)