Natural Language Models and Interfaces

BSc Artificial Intelligence

Lecturer: Wilker Aziz
Institute for Logic, Language, and Computation

2019, week 5, lecture b

Context-Free Grammars

A **CFG** grammar G is denoted by

- a finite set of **nonterminal** symbols $\mathscr{V}$
- a finite set of **terminal** symbols Σ with $\Sigma \cap \mathscr{V} = \varnothing$
- a finite set $\mathscr{R}$ of **rules** of the form $X \rightarrow \beta$ where
  - $X \in \mathscr{V}$ and $\beta \in (\Sigma \cup \mathscr{V})^*$
- $S \in \mathscr{V}$ a distinguished **start** symbol

Let ε denote an **empty** string

Example CFG

S → NP VP	Vi → sleeps
VP → Vi	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Generative Device

Left-most derivation

- sequence of strings $\alpha_1 \ldots \alpha_n$
  - $\alpha_1 = \langle S \rangle$
  - $\alpha_n \in \Sigma^*$
  - $\alpha_{i \geq 2}$ derived from α_{i-1} by picking the left-most nonterminal X
    - and replacing it by some α such that $X \to \beta \in \mathscr{R}$

Example of Derivation

Example of Derivation

String	Substitution

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the
α_4 =	the NN VP	NN → man

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the
α_4 =	the NN VP	NN → man
α_5 =	the man VP	VP → Vi

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the
α_4 =	the NN VP	NN → man
α_5 =	the man VP	VP → Vi
α_6 =	the man Vi	Vi → sleeps

Example of Derivation

	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the
α_4 =	the NN VP	NN → man
α_5 =	the man VP	VP → Vi
α_6 =	the man Vi	Vi → sleeps
α_7 =	the man sleeps	

Example of Derivation

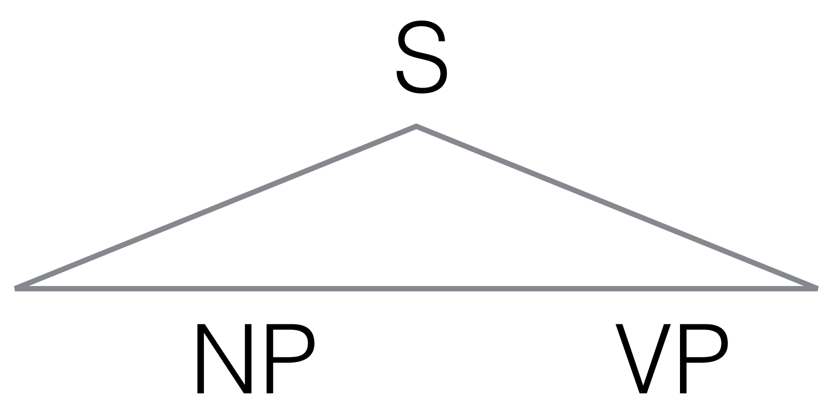
	String	Substitution
α_1 =	S	S → NP VP
α_2 =	NP VP	NP → DT NN
α_3 =	DT NN VP	DT → the
α_4 =	the NN VP	NN → man
α_5 =	the man VP	VP → Vi
α_6 =	the man Vi	Vi → sleeps
α_7 =	the man sleeps	
α_7 =	$S \Rightarrow^*$ the man sleeps	

Example of Generation

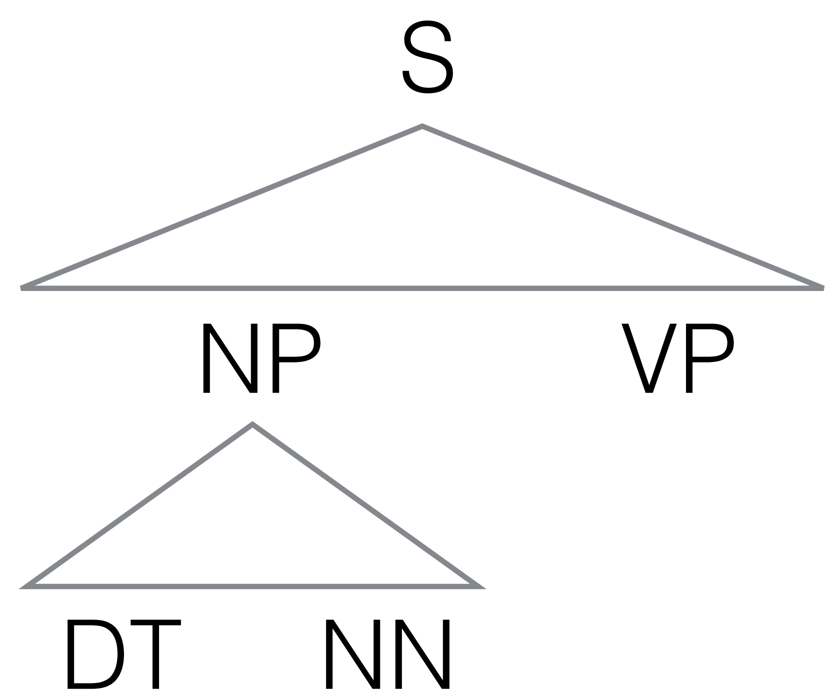
Example of Generation

S

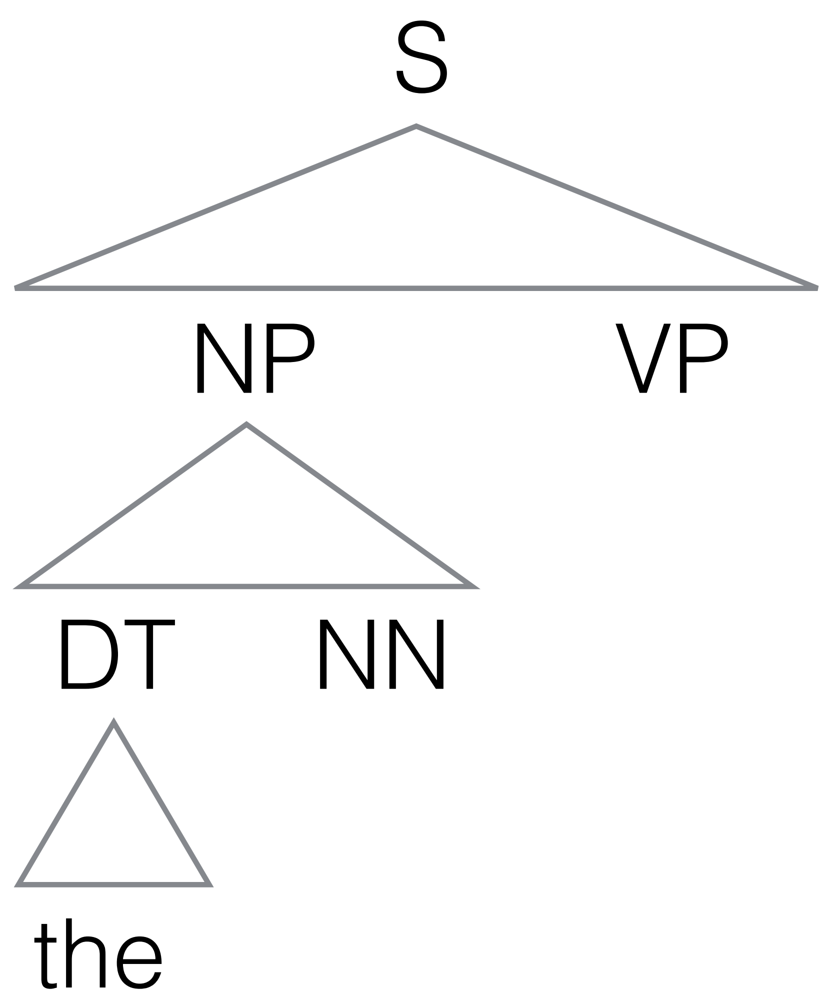
Example of Generation



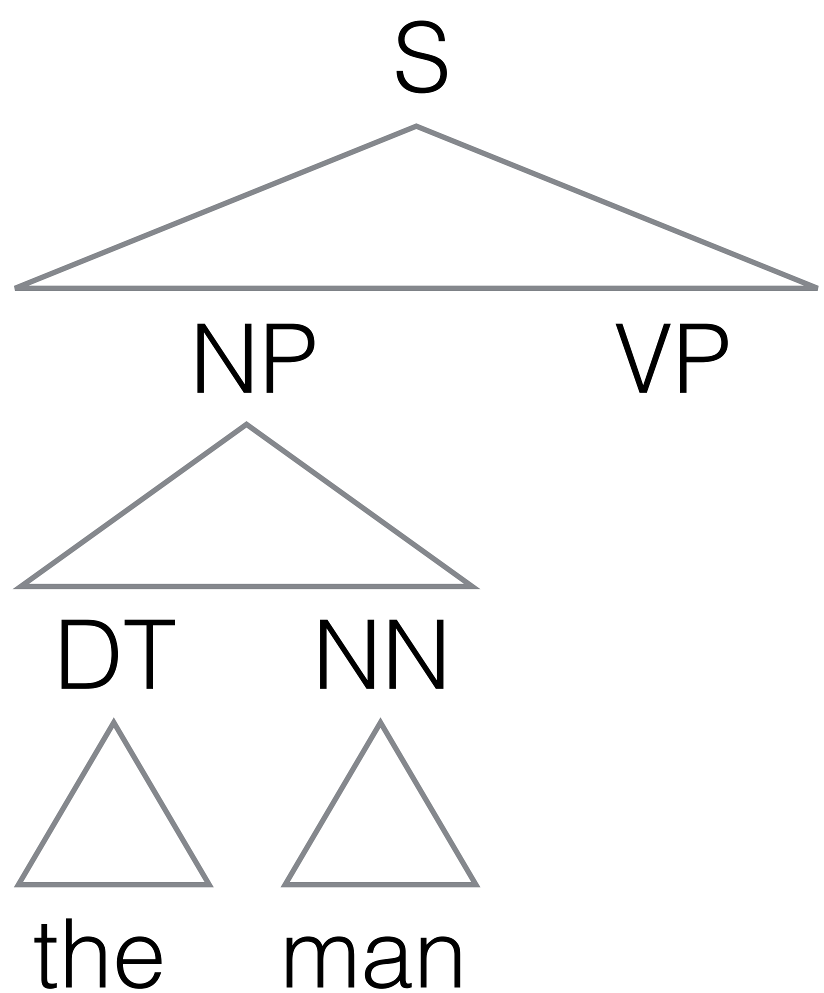
Example of Generation



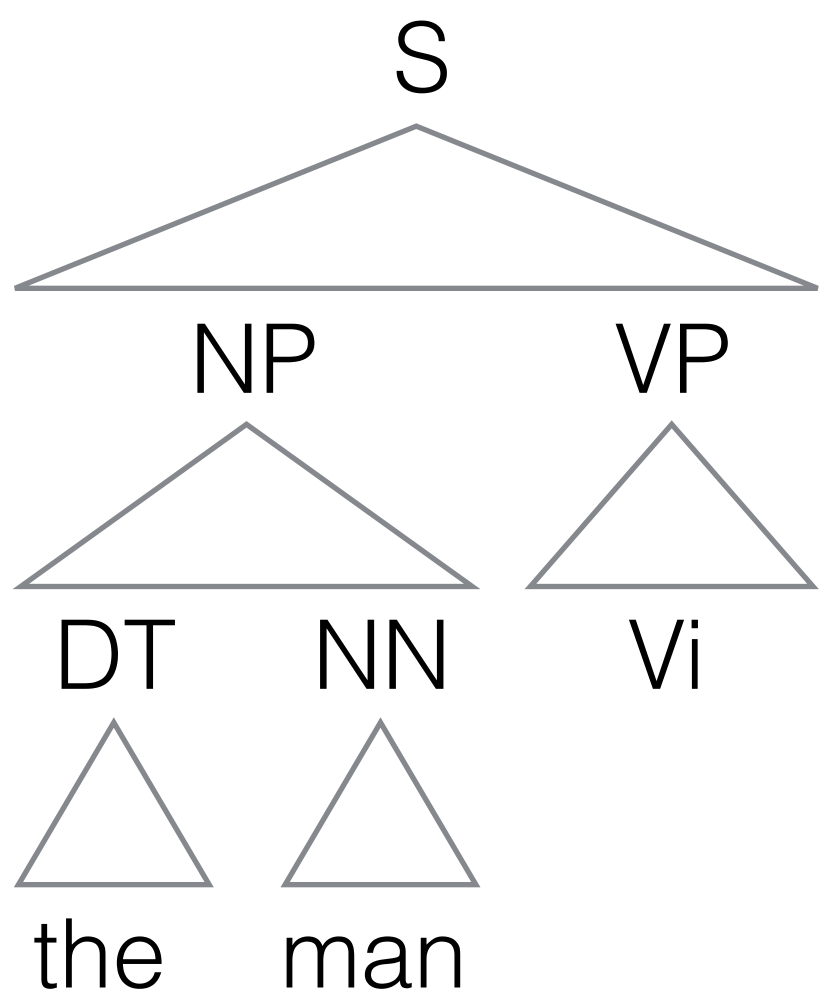
Example of Generation



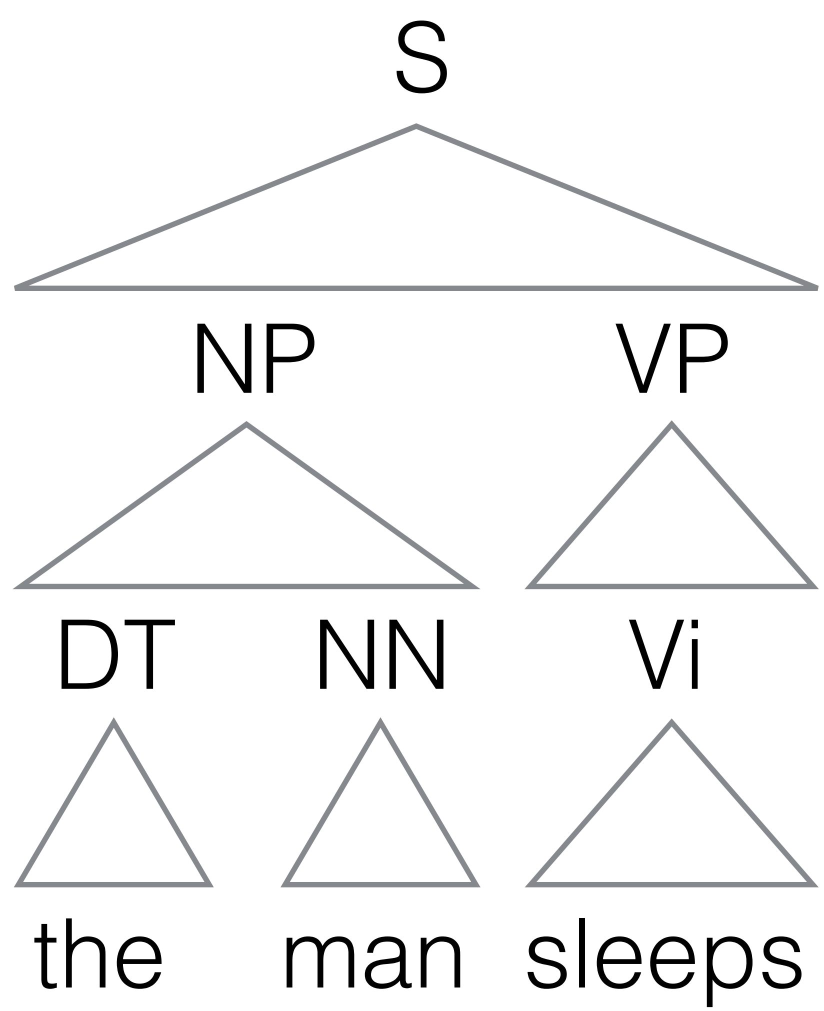
Example of Generation



Example of Generation



Example of Generation



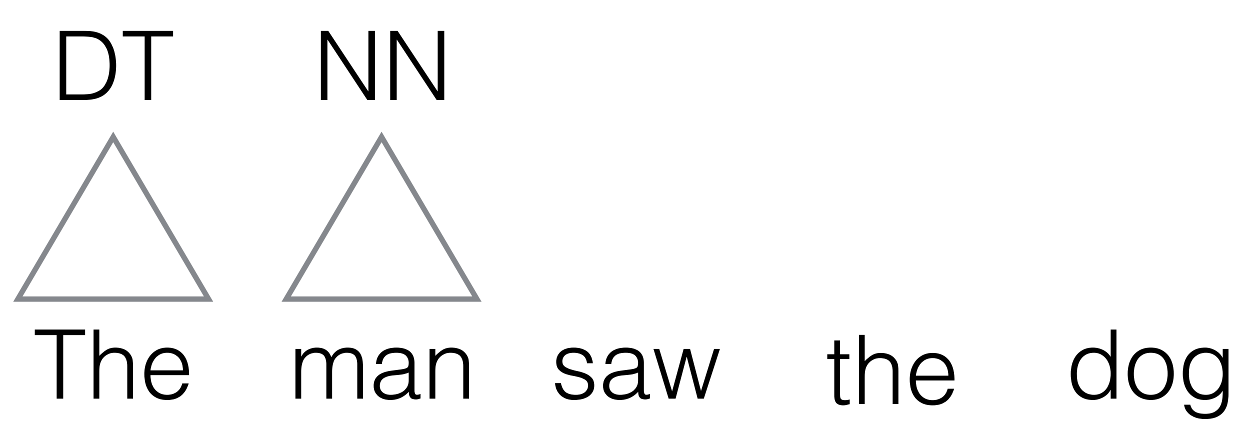
Example of Recognition

Example of Recognition

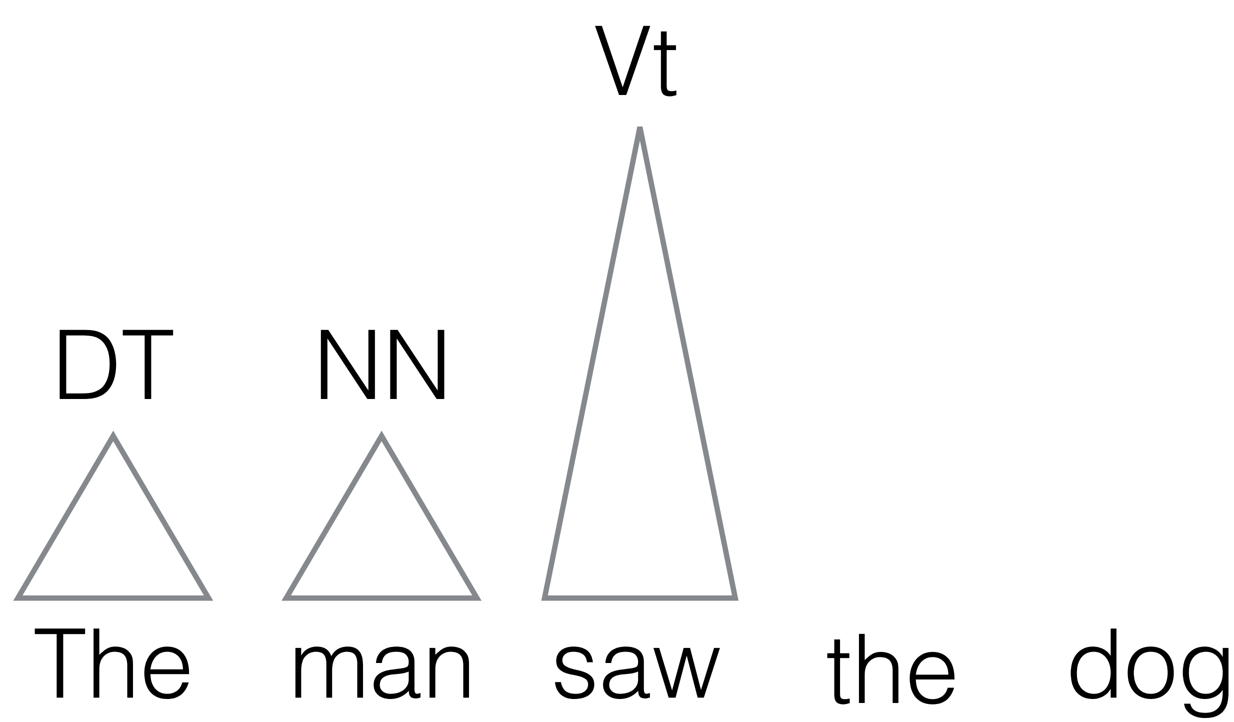
The man saw the dog

Example of Recognition

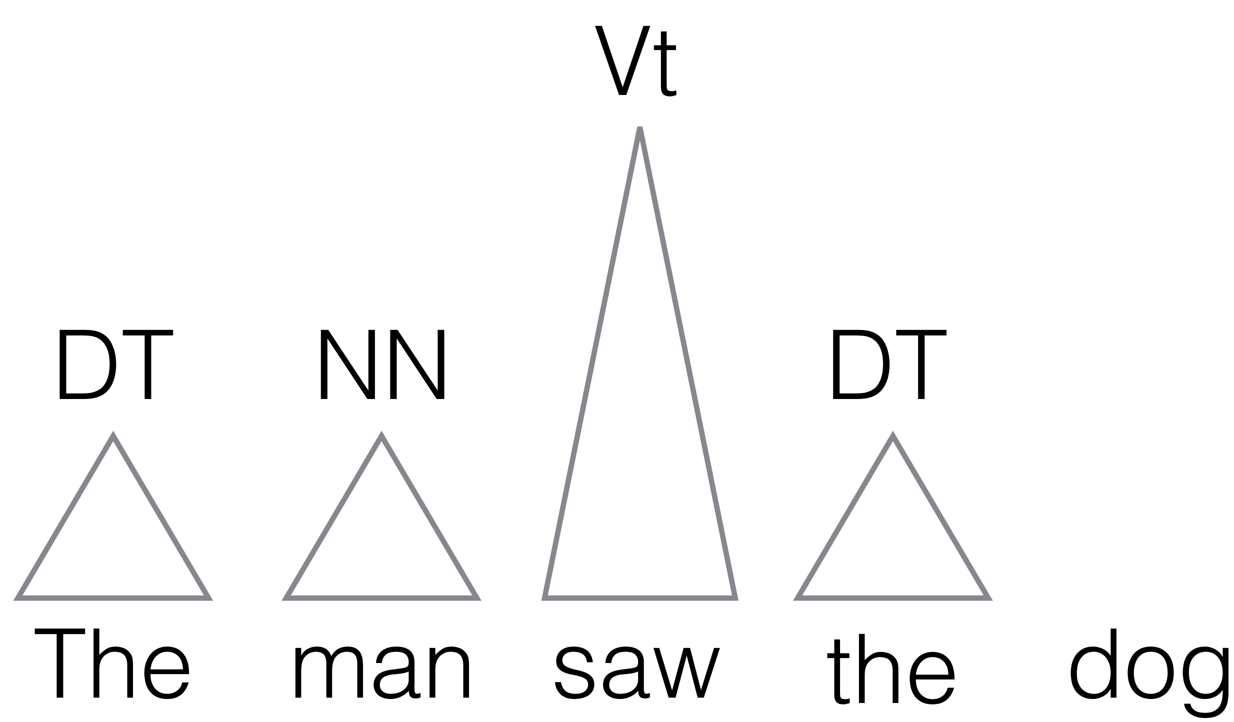
Example of Recognition



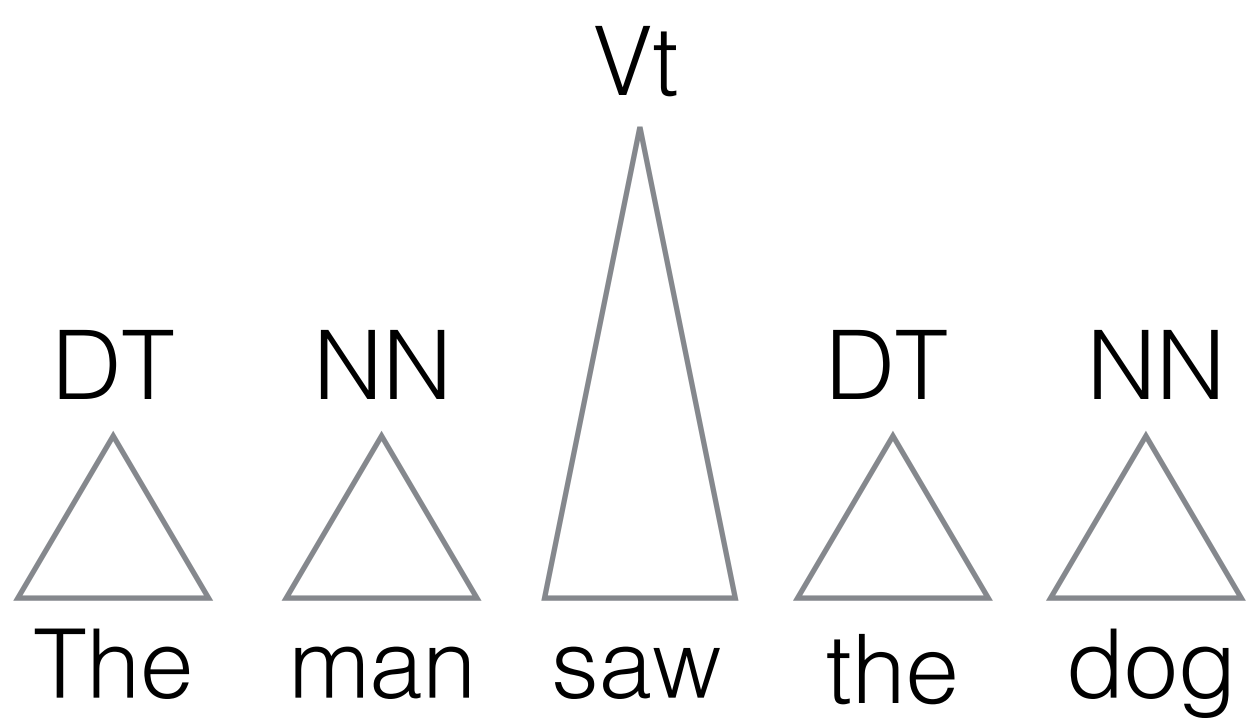
Example of Recognition



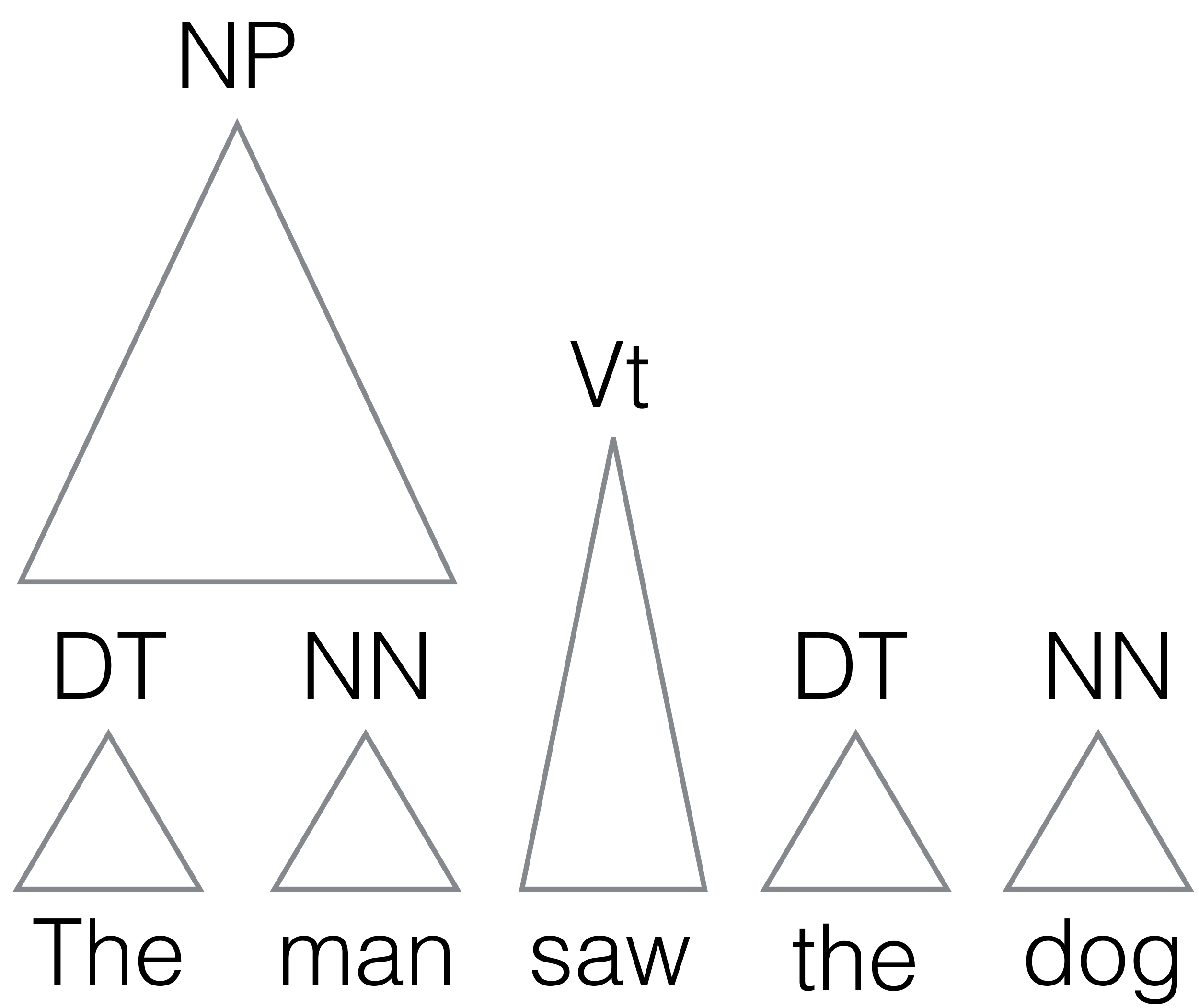
Example of Recognition



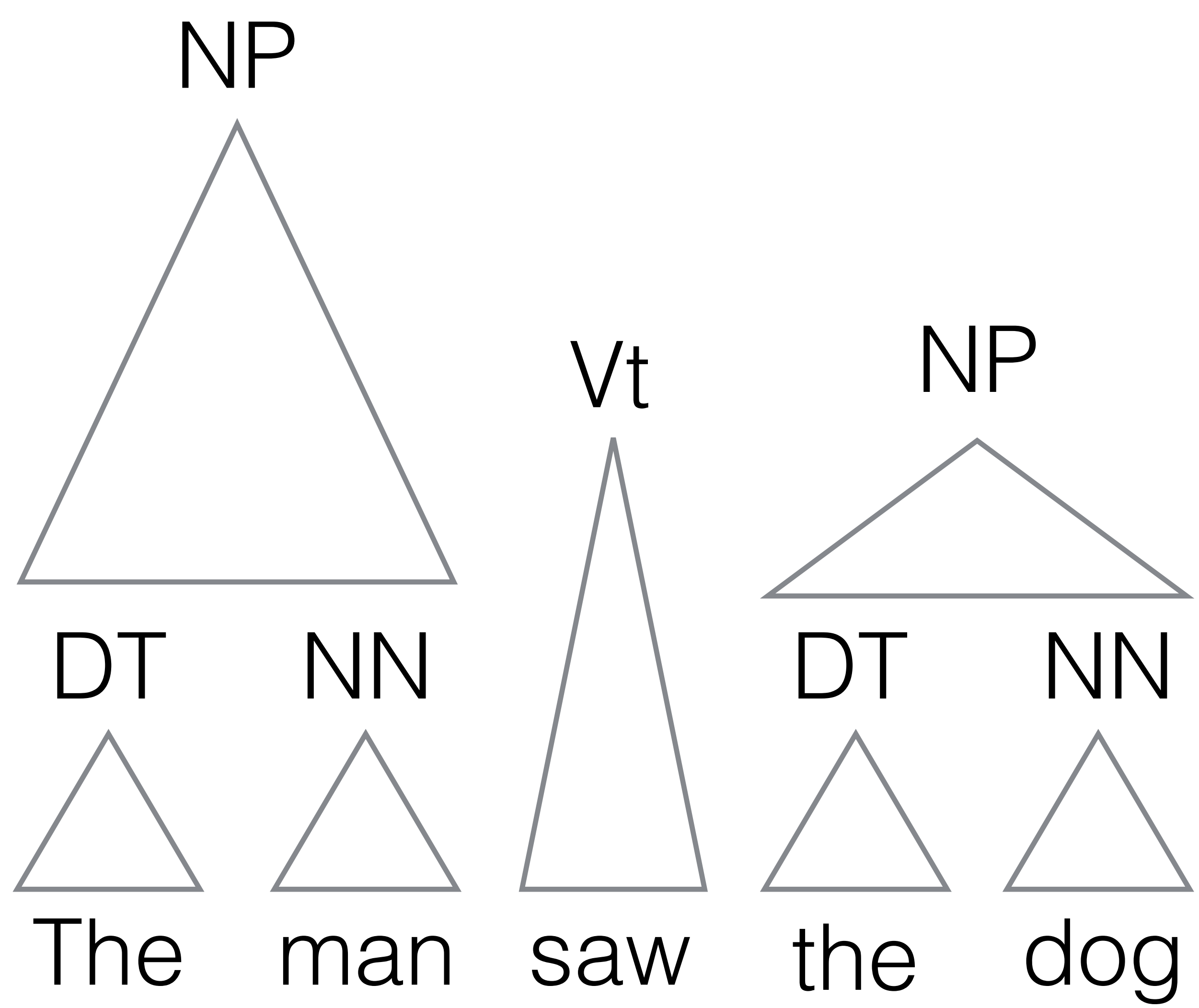
Example of Recognition



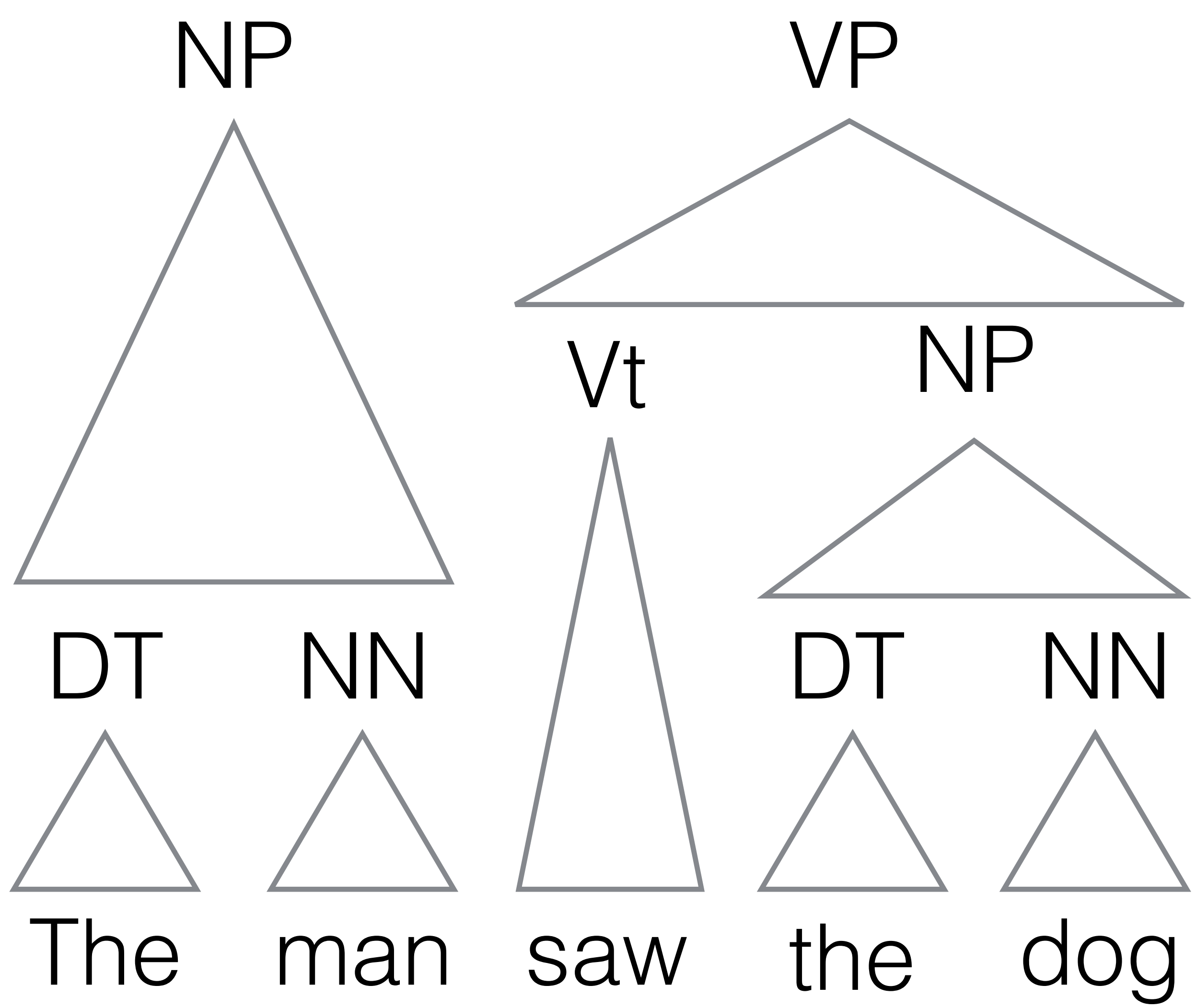
Example of Recognition



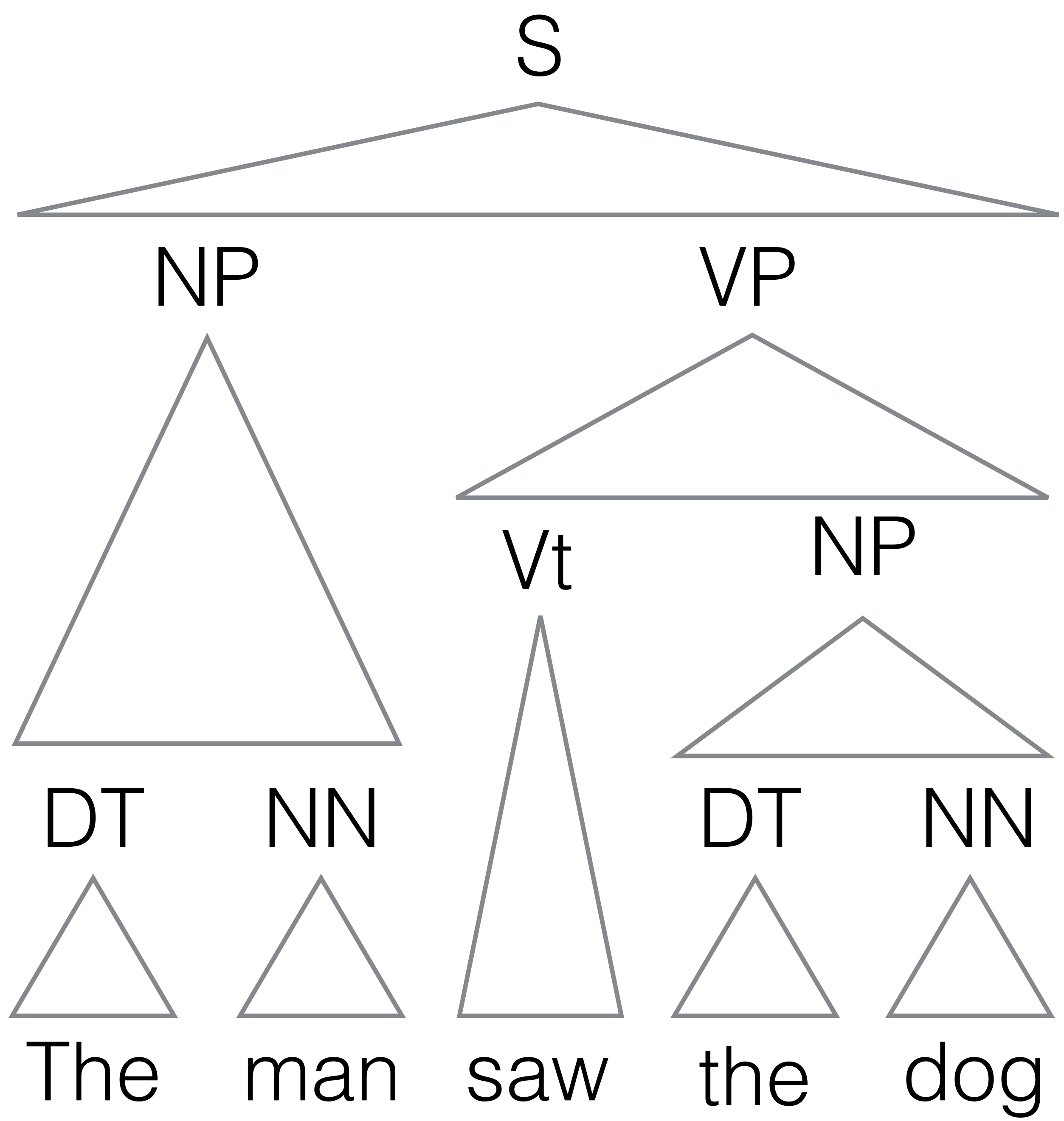
Example of Recognition



Example of Recognition



Example of Recognition



Language

A string $\omega \in \Sigma^*$ is generated/accepted by G if

$$S \Rightarrow^* \omega$$

$\Rightarrow^*$ denotes a sequence of rule applications

Language of G

$$L(G) = \{\omega: S \Rightarrow^* \omega\} \subseteq \Sigma^*$$

Chomsky Normal Form

Every CFG is weakly equivalent to another such that

- $X \rightarrow Y\ Z$ where $X, Y, Z \in \mathscr{V}$
- $X \rightarrow w$ where $w \in \Sigma$
- and possibly $S \rightarrow \varepsilon$

[Hopcroft and Ullman, 1979]

Parsing as Deduction

Parsing as Deduction

Deductive process to prove claims about grammaticality

[Shieber et al., 1995]

Parsing as Deduction

Deductive process to prove claims about grammaticality

[Shieber et al., 1995]

- focus on strategy rather than implementation

Parsing as Deduction

Deductive process to prove claims about grammaticality
[Shieber et al., 1995]

- focus on strategy rather than implementation
- soundness/completeness easier to prove

Parsing as Deduction

Deductive process to prove claims about grammaticality
[Shieber et al., 1995]

- focus on strategy rather than implementation
- soundness/completeness easier to prove
- complexity determined by inspection

Parsing as Deduction

Deductive process to prove claims about grammaticality

[Shieber et al., 1995]

- focus on strategy rather than implementation
- soundness/completeness easier to prove
- complexity determined by inspection
- dynamic program follows directly

Parsing as Deduction

Deductive process to prove claims about grammaticality

[Shieber et al., 1995]

- focus on strategy rather than implementation
- soundness/completeness easier to prove
- complexity determined by inspection
- dynamic program follows directly
- generality

Deductive systems

Item: a statement / intermediate sound result

- formula or schemata expressed with variables

Inference rule: statement derived from existing items

- $$\frac{A_1 \ldots A_m}{B} \text{ (condition)}$$ where A_i and B are items
    - Ai are called antecedents
    - B is called consequent

Deductive program

Axioms: trivial items

- do not depend on previous statements

Goal: states that a proof exists

Proof:

- start from axioms
- exhaustively deduce items
  - never twice under the same premises
- accept if goal is proven

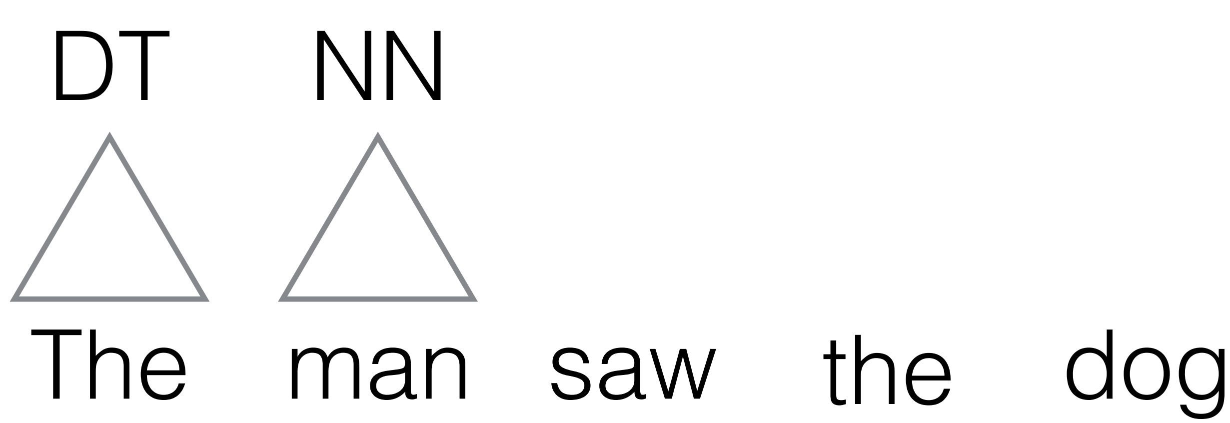
Bottom-up: Shift-Reduce

Bottom-up: Shift-Reduce

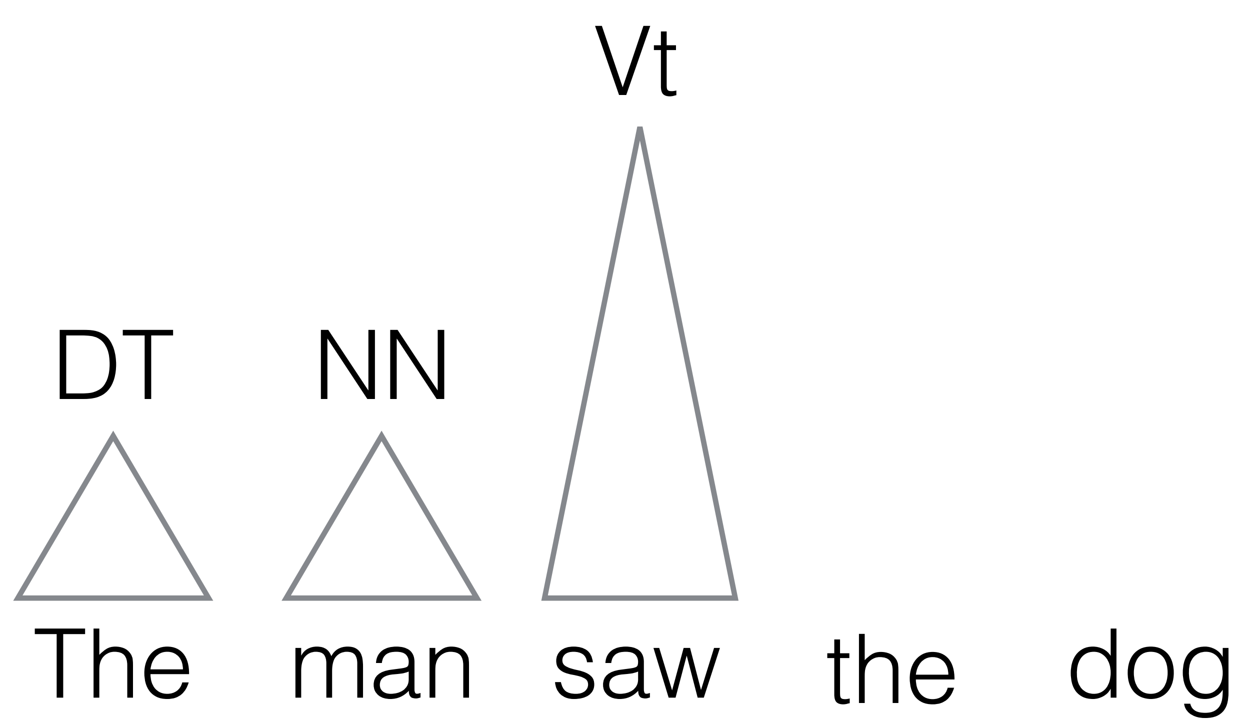
The man saw the dog

Bottom-up: Shift-Reduce

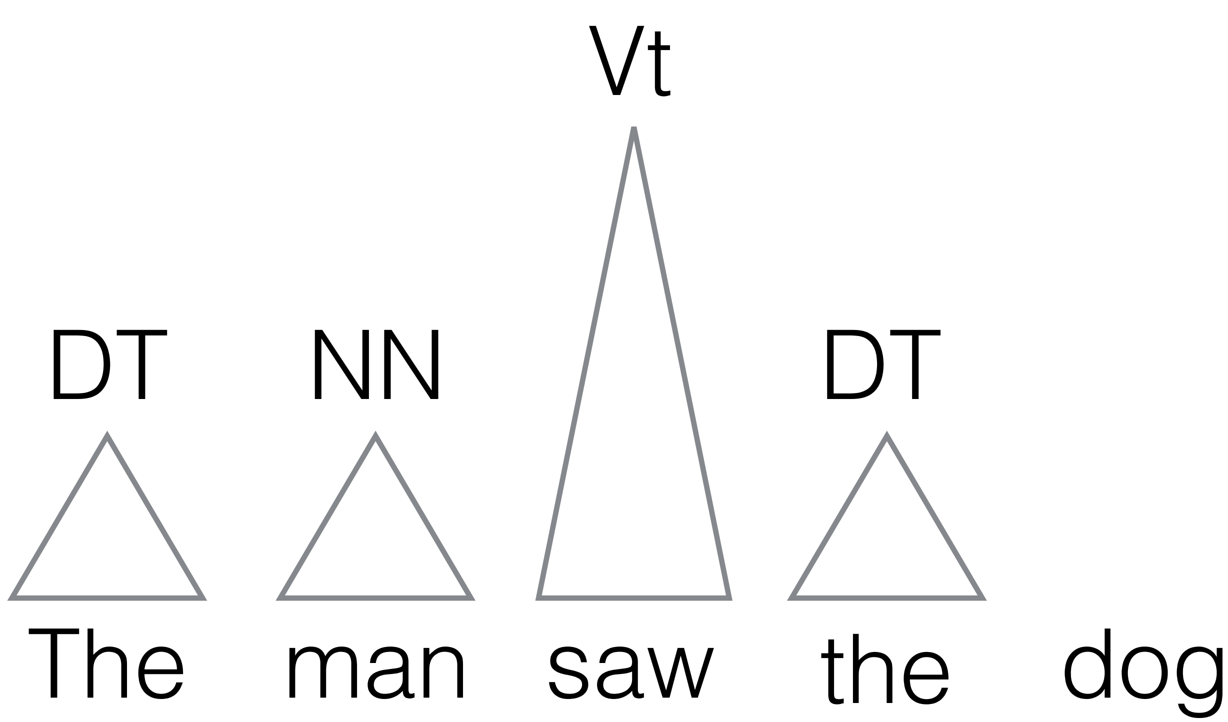
Bottom-up: Shift-Reduce



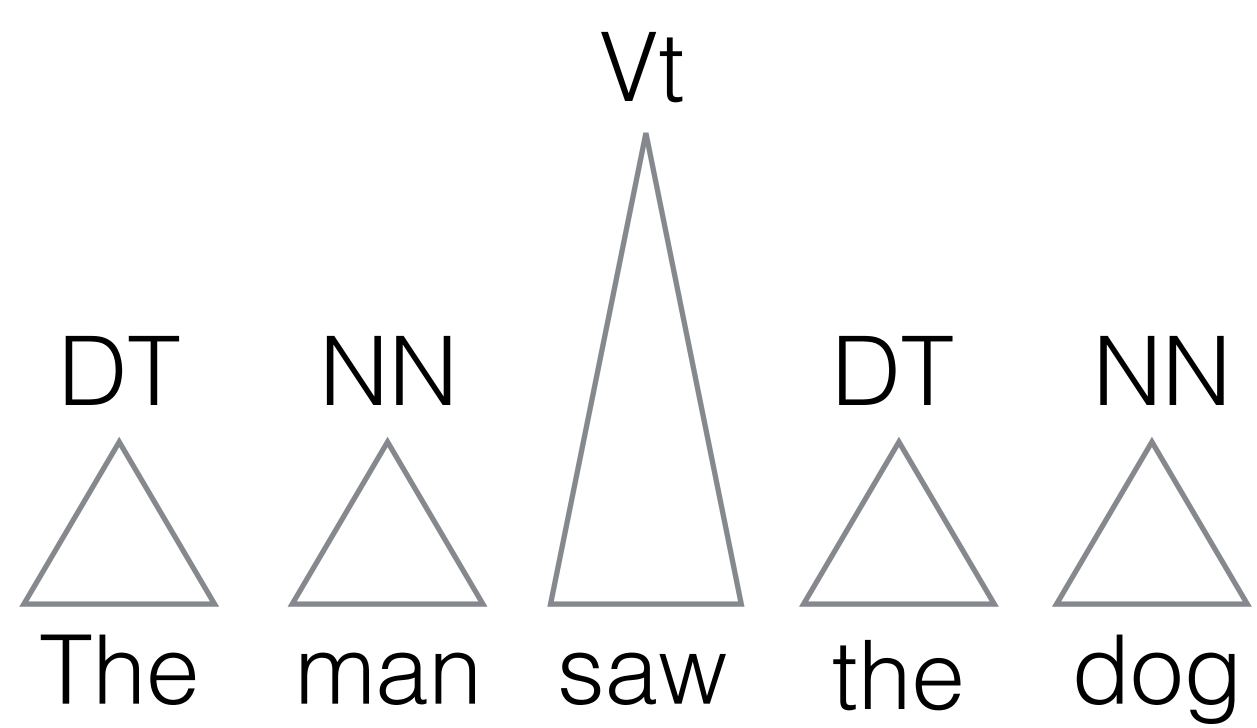
Bottom-up: Shift-Reduce



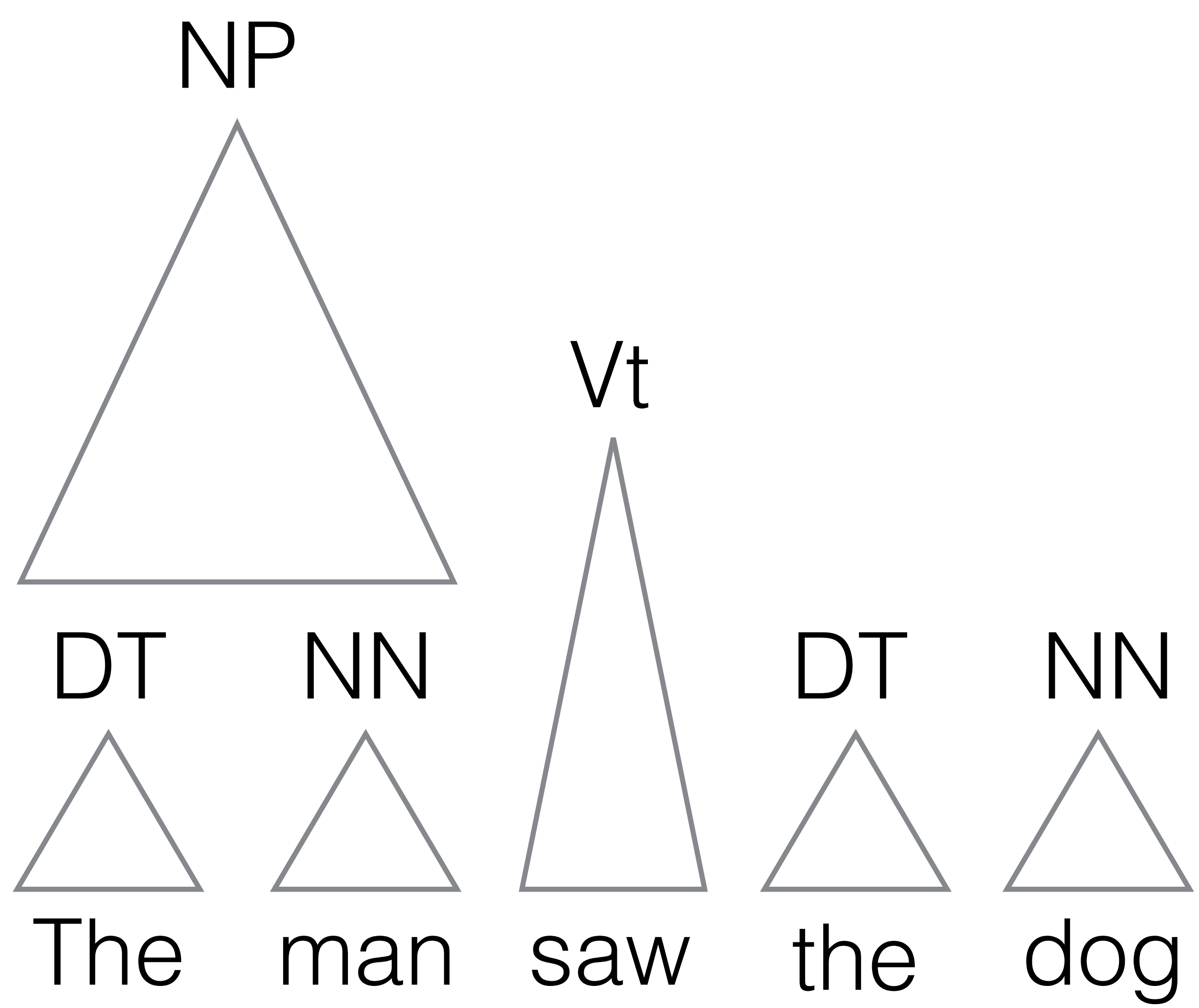
Bottom-up: Shift-Reduce



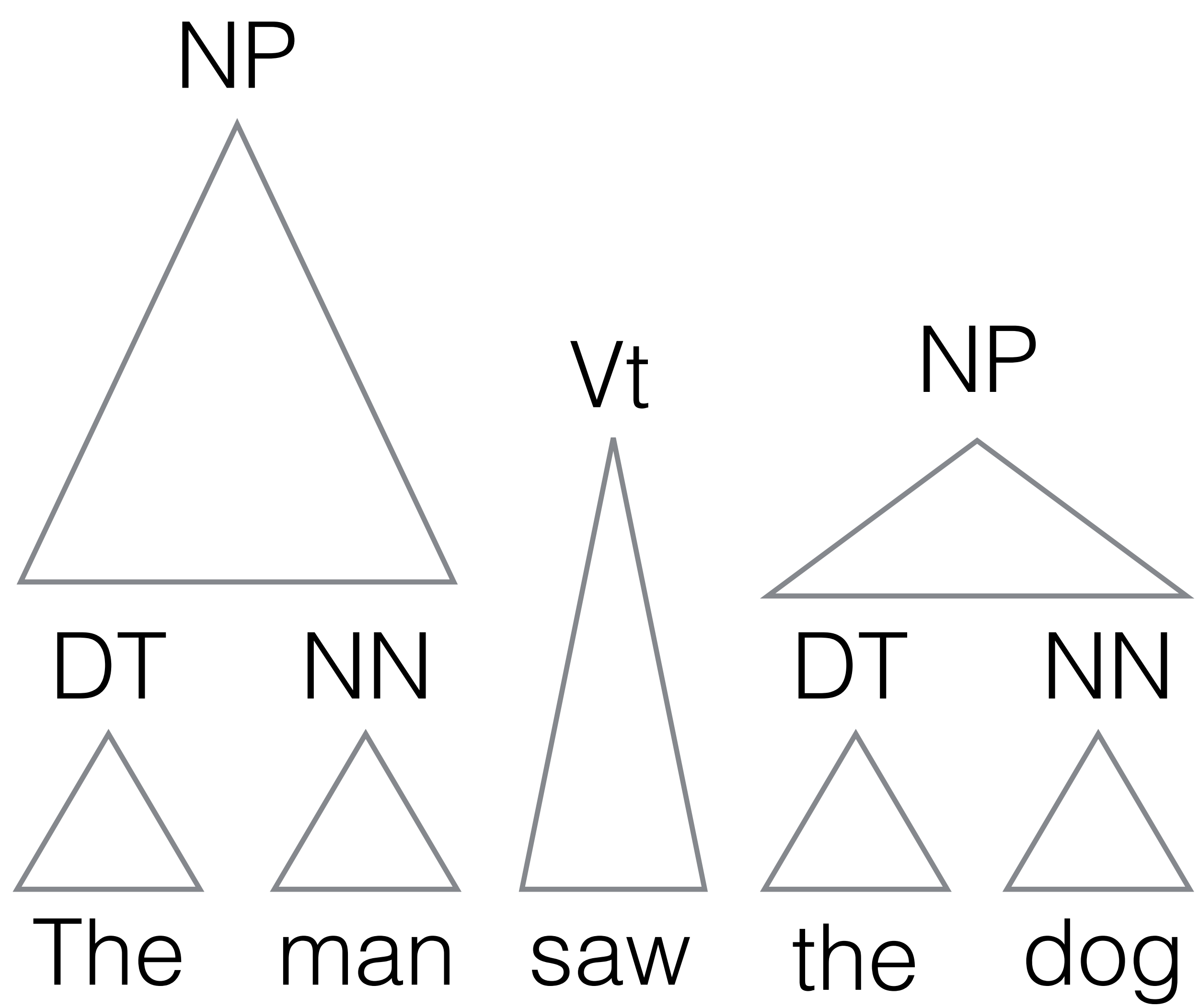
Bottom-up: Shift-Reduce



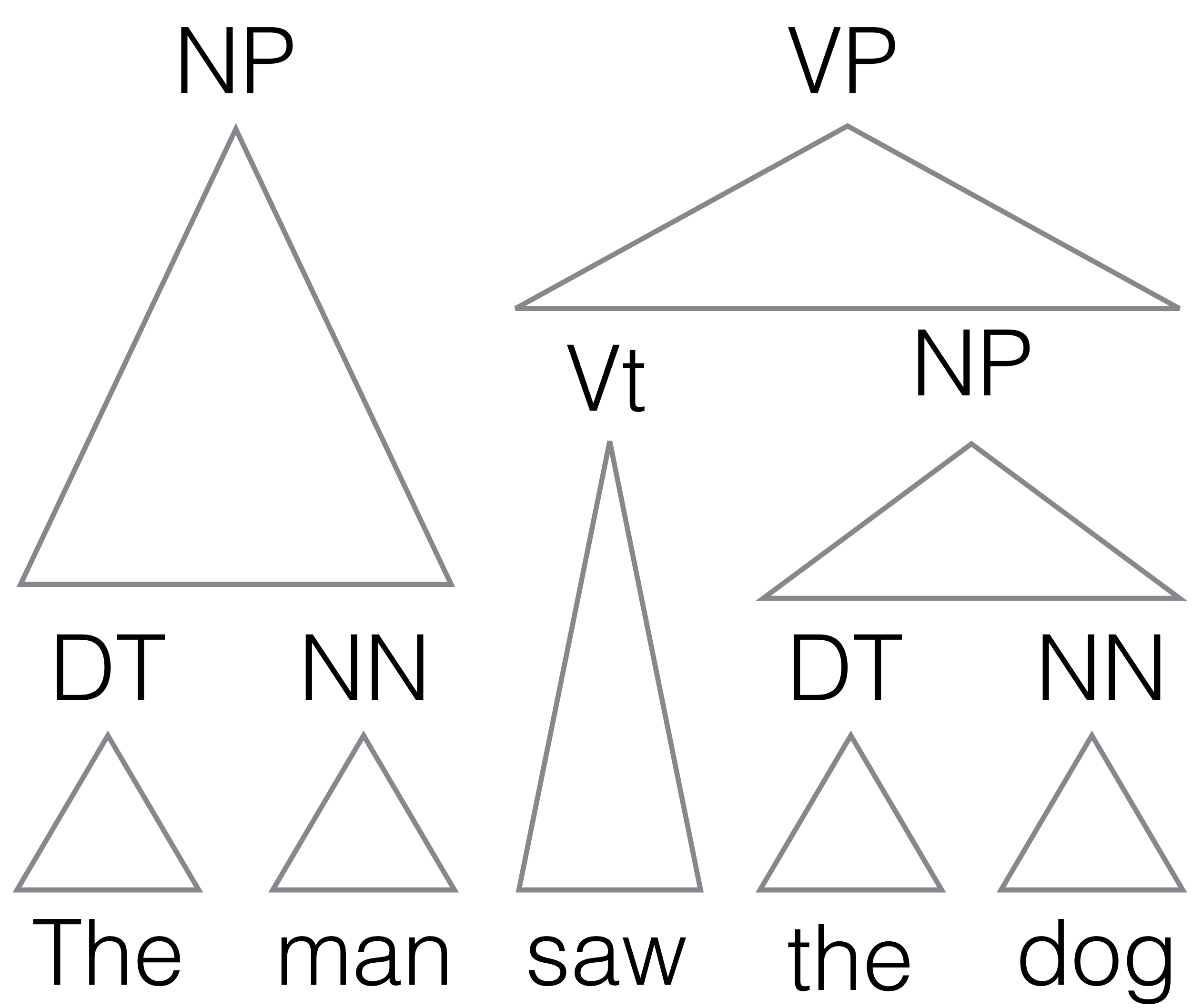
Bottom-up: Shift-Reduce



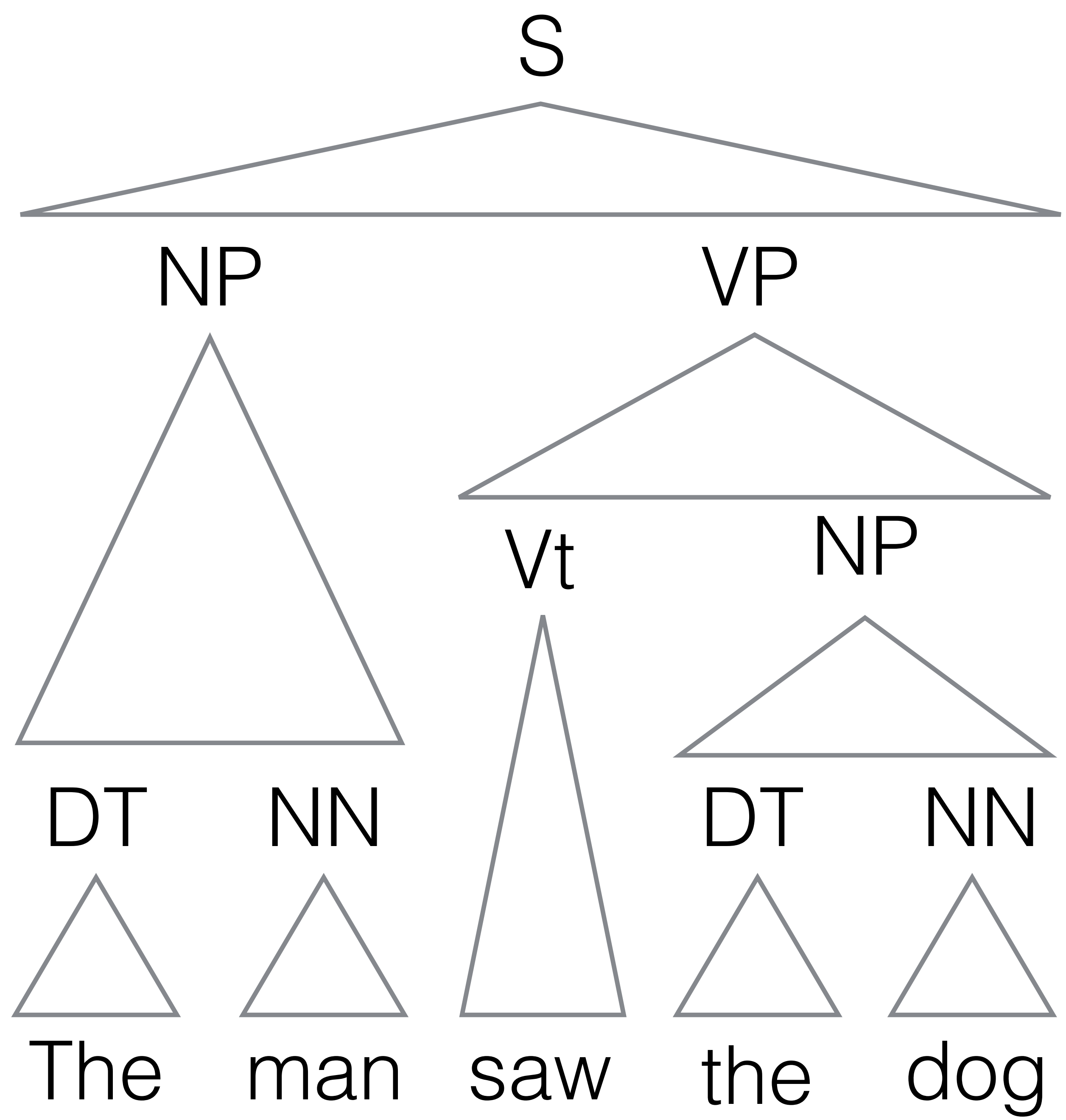
Bottom-up: Shift-Reduce



Bottom-up: Shift-Reduce



Bottom-up: Shift-Reduce



Shift-Reduce Example

Input: *the man sleeps*

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue
Axiom	1	[•,0]	1

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue
Axiom	1	[•,0]	1

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue
Axiom	1	[•,0]	1
Shift: [1]	2	[the•,1]	2

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue
Axiom	1	[•,0]	1
Shift: [1]	2	[the•,1]	2

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8
Reduce: [8]	VP → Vi	9	[NP VP •, 3]	9

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8
Reduce: [8]	VP → Vi	9	[NP VP •, 3]	9

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8
Reduce: [8]	VP → Vi	9	[NP VP •, 3]	9
Reduce: [9]	S → NP VP	10	[S •, 3]	10

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[•,0]	1
Shift: [1]		2	[the•,1]	2
Reduce: [2]	DT → the	3	[DT•,1]	3
Shift: [3]		4	[DT man •, 2]	4
Reduce: [4]	NN → man	5	[DT NN •, 2]	5
Reduce: [5]	NP → DT NN	6	[NP •, 2]	6
Shift: [6]		7	[NP sleeps •, 3]	7
Reduce: [7]	Vi → sleeps	8	[NP Vi •, 3]	8
Reduce: [8]	VP → Vi	9	[NP VP •, 3]	9
Reduce: [9]	S → NP VP	10	[S •, 3]	10
GOAL: [10]				∅

S → NP VP
VP → Vi
VP → Vt NP
VP → VP PP
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Shift-Reduce

Input: G and $x_1 \ldots x_n$

Item form: $[\alpha\bullet, j]$
asserts that $\alpha \Rightarrow^* x_1 \ldots x_j$ or

that $\alpha\, x_{j+1} \ldots x_n \Rightarrow^* x_1 \ldots x_n$

Axiom: $[\bullet, 0]$

Goal: $[S\bullet, n]$

$$\textsc{Shift}\ \frac{[\alpha\bullet, j]}{[\alpha\, x_{j+1}, j+1]}$$

Scan (shift)
asserts that $\alpha\, x_{j+1} \Rightarrow^* x_1 \ldots x_j\, x_{j+1}$

$$\textsc{Reduce}\ \frac{[\alpha\,\beta\bullet, j]}{[\alpha\, B, j]}\ B \rightarrow \beta \in \mathcal{R}$$

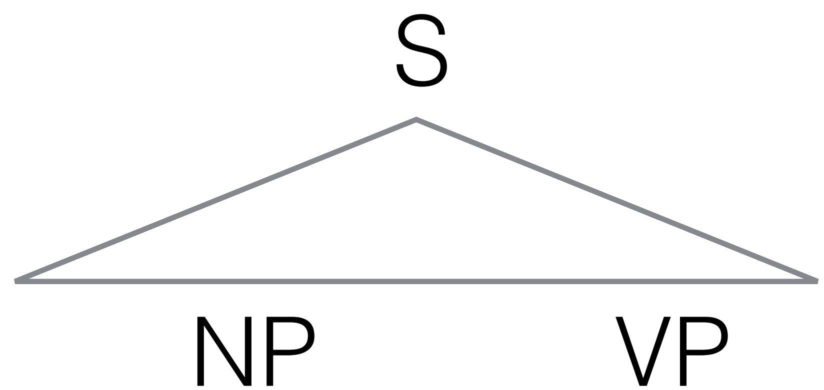
Complete (reduce)
asserts that $\alpha B \Rightarrow^* x_1 \ldots x_j$

Top-down: Predict-Scan

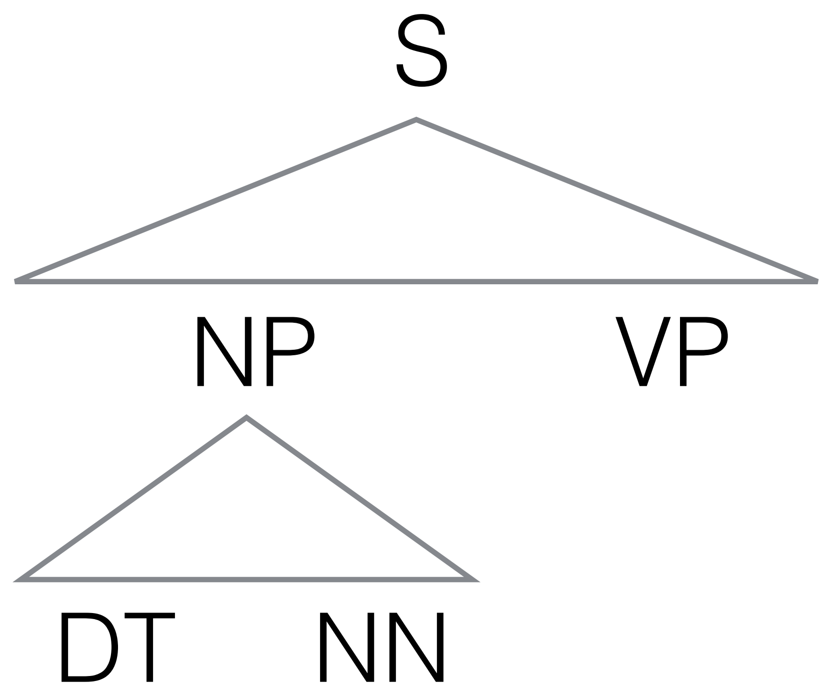
Top-down: Predict-Scan

S

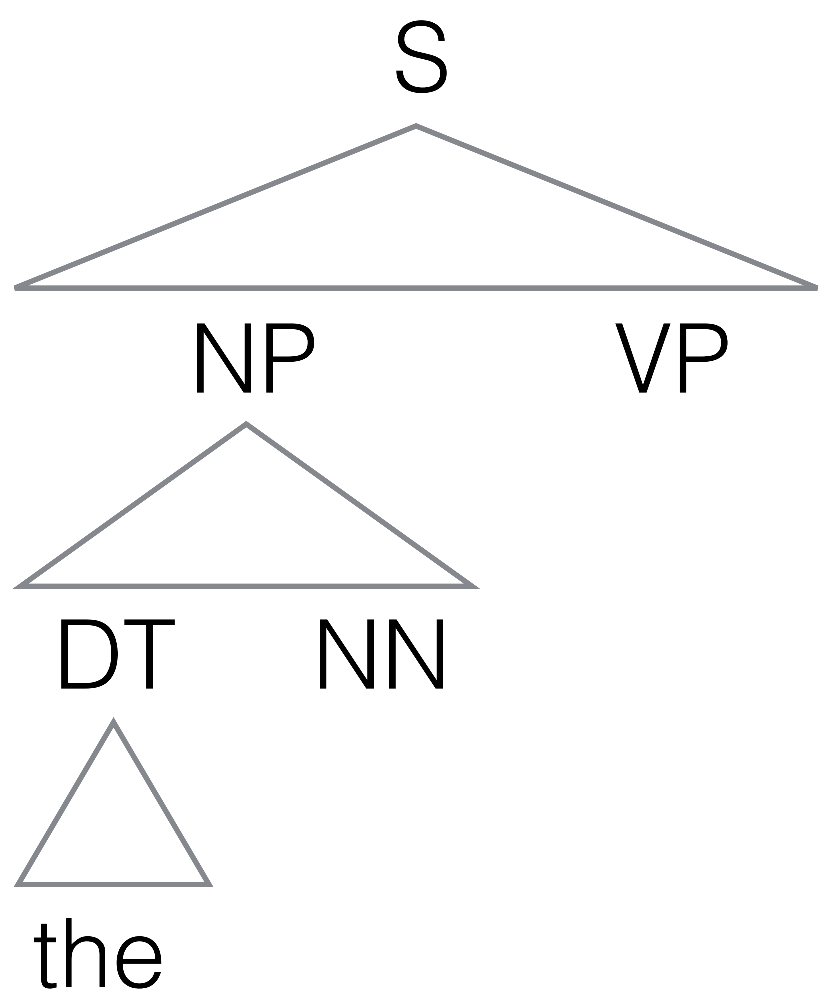
Top-down: Predict-Scan



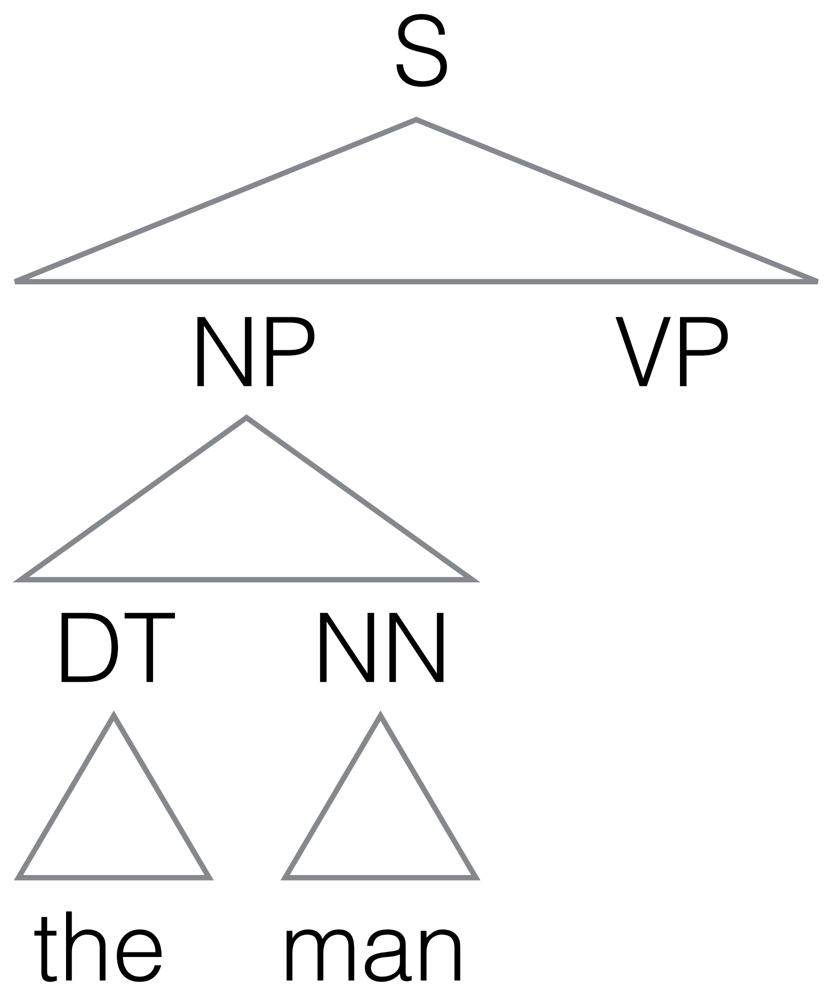
Top-down: Predict-Scan



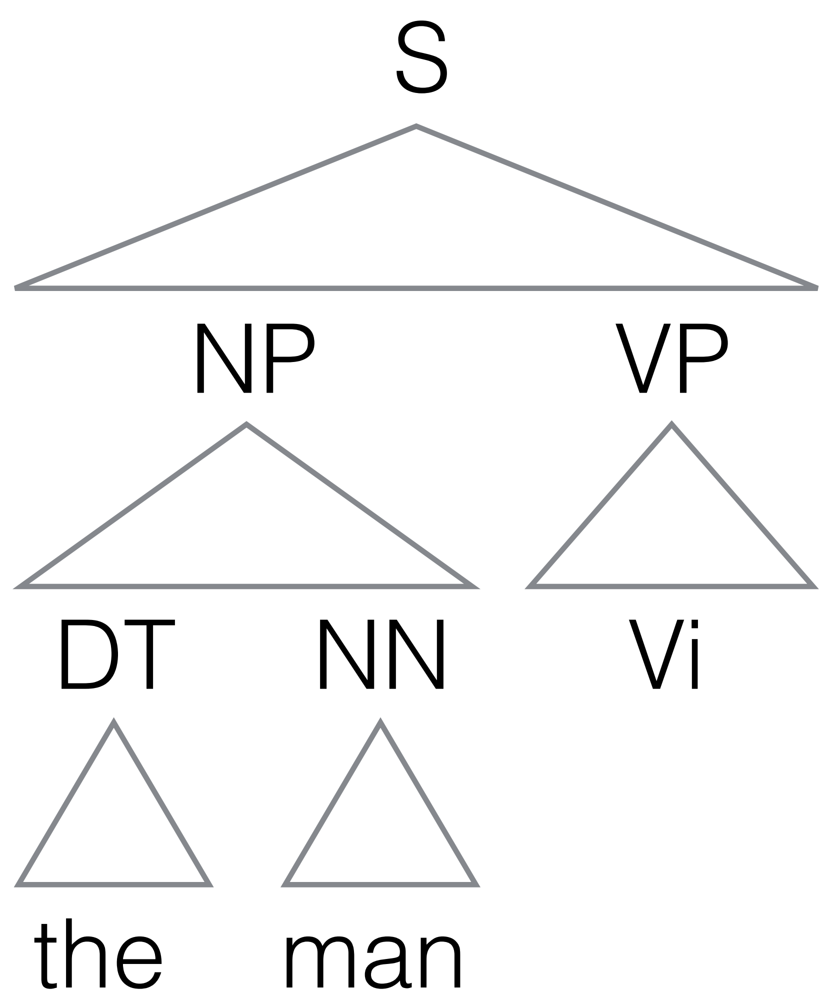
Top-down: Predict-Scan



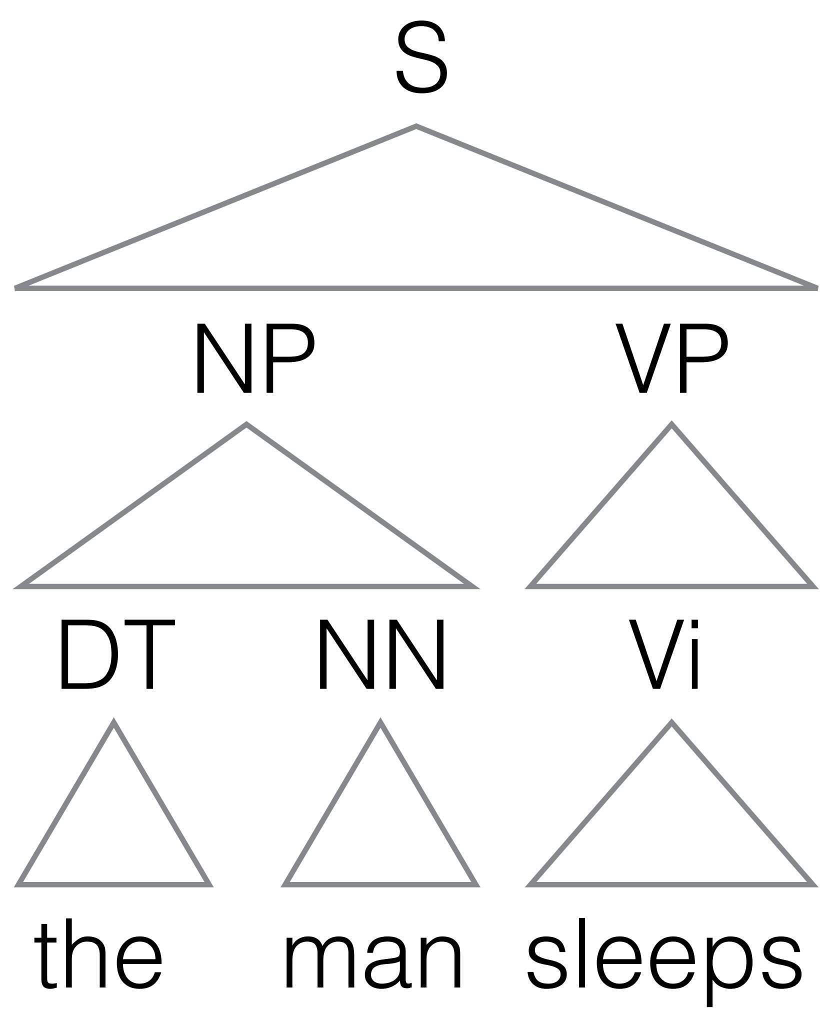
Top-down: Predict-Scan



Top-down: Predict-Scan



Top-down: Predict-Scan



Top-Down Example

Input: *the man sleeps*

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition	Statement	Queue

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10
[9]				10

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10
[9]				10

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10
[9]				10
Scan: [10]		11	[•, 3]	11

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down Example

Input: *the man sleeps*

Rule	Condition		Statement	Queue
Axiom		1	[• S, 0]	1
Predict: [1]	S → NP VP	2	[• NP VP, 0]	2
Predict: [2]	NP → DT NN	3	[• DT NN VP, 0]	3
Predict: [3]	DT → the	4	[• the NN VP,0]	4
Scan: [4]		5	[• NN VP,1]	5
Predict: [5]	NN → man	6	[• man VP, 1]	6
Scan: [6]		7	[• VP, 2]	7
Predict: [7]	VP → Vi	8	[• Vi, 2]	8, 9
	VP → Vt NP	9	[• Vt NP, 2]	
Predict: [8]	Vi → sleeps	10	[• sleeps, 2]	9, 10
[9]				10
Scan: [10]		11	[•, 3]	11
GOAL: [11]				∅

S → NP VP
VP → Vi
VP → Vt NP
~~VP → VP PP~~
NP → DT NN
NP → NP PP
PP → IN NP
Vi → sleeps
Vt → saw
NN → man
NN → dog
NN → telescope
DT → the
IN → with

Top-Down recognition

Input: G and $x_1 \dots x_n$

Item form: [•β, j]
asserts that $S \Rightarrow^* x_1 \dots x_j \; \beta$

Axiom: [•S,0]

Goal: [•,n]

Scan
asserts that $S \Rightarrow^* x_1 \dots x_j \, x_{j+1} \, \beta$

$$\textsc{Scan} \; \frac{[\bullet x_{j+1}\,\beta, j]}{[\bullet \beta, j+1]}$$

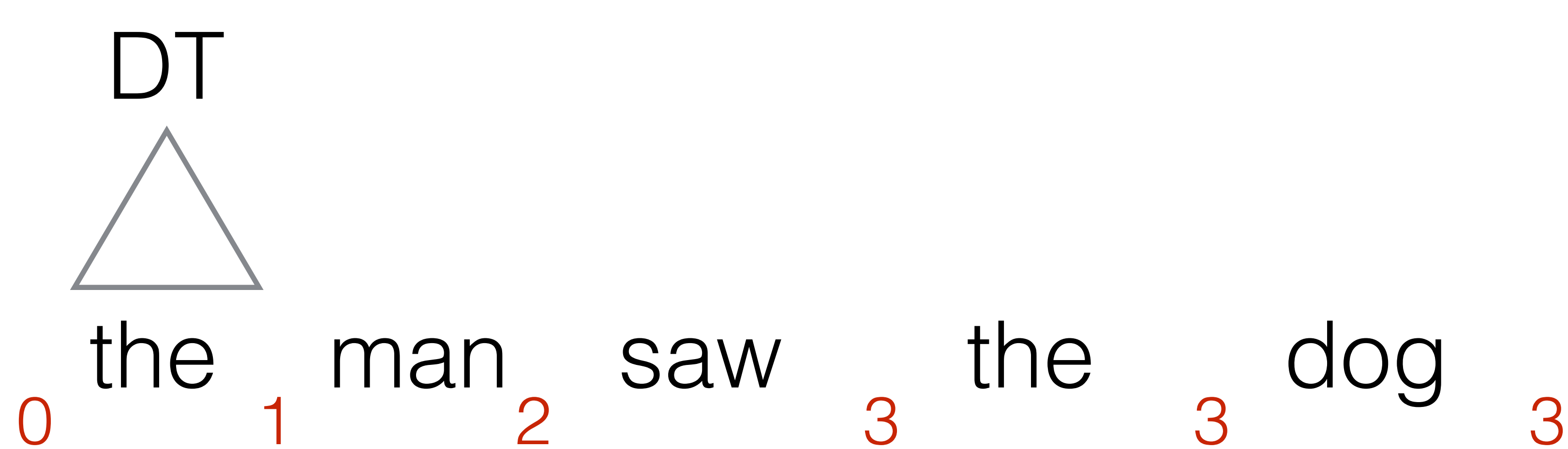
Predict
asserts that $S \Rightarrow^* x_1 \dots x_j \, B \, \beta$

$$\textsc{Predict} \; \frac{[\bullet A\,\beta, j]}{[\bullet \alpha\,\beta, j]} \; A \to \alpha \in \mathcal{R}$$

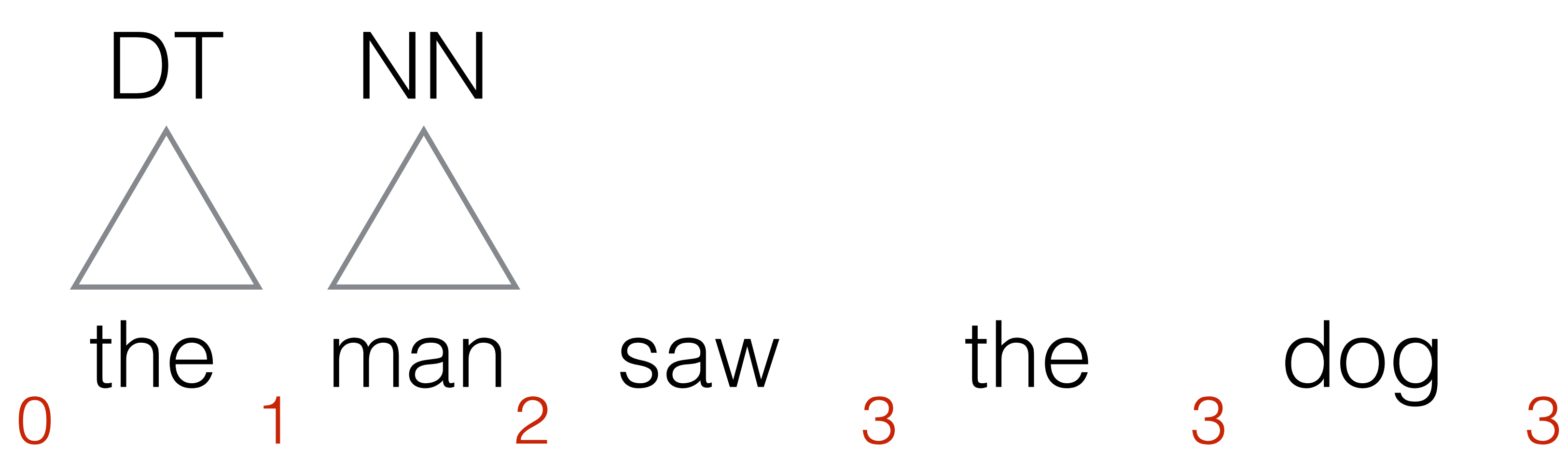
Bottom-Up for CNF: CKY

0 the 1 man 2 saw 3 the 3 dog 3

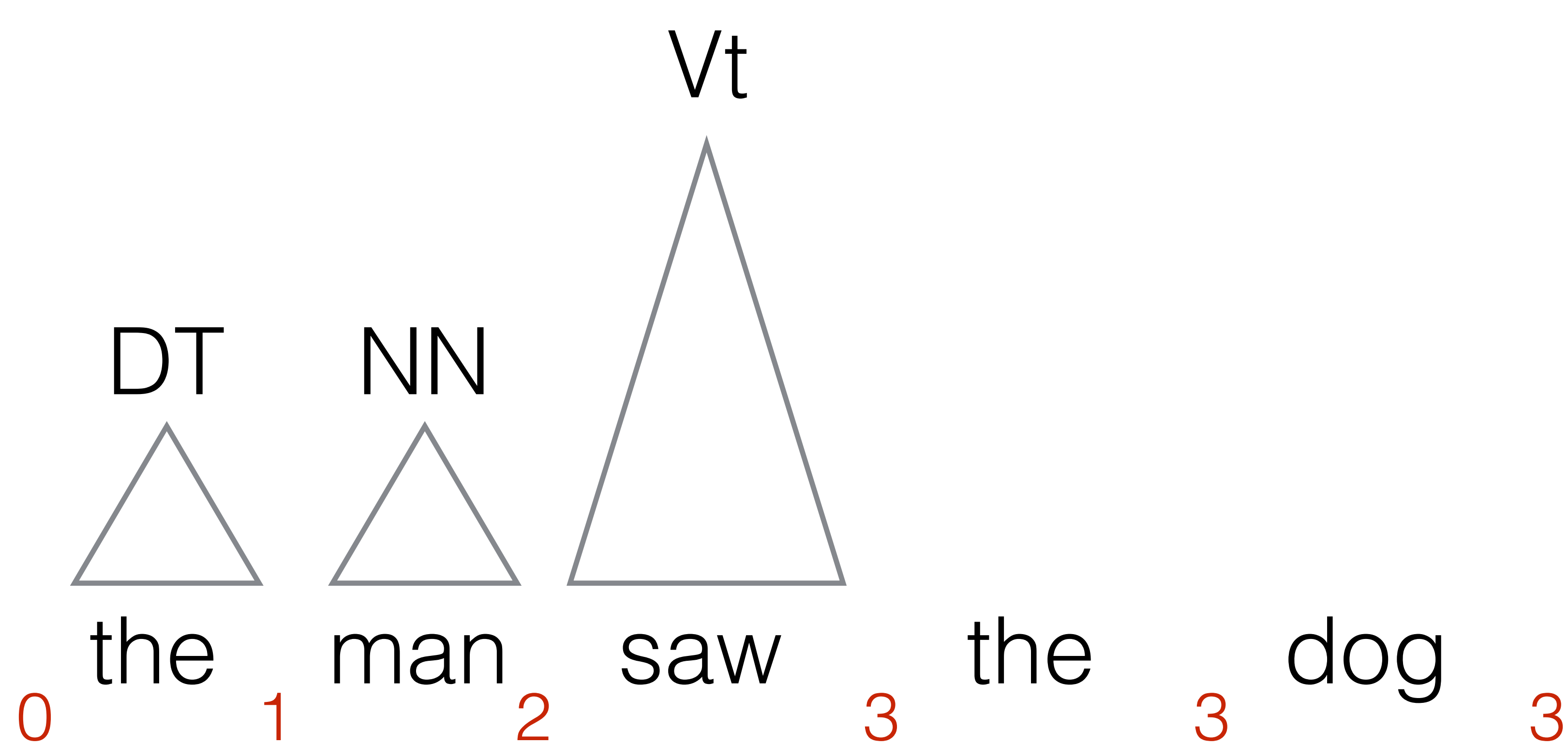
Bottom-Up for CNF: CKY



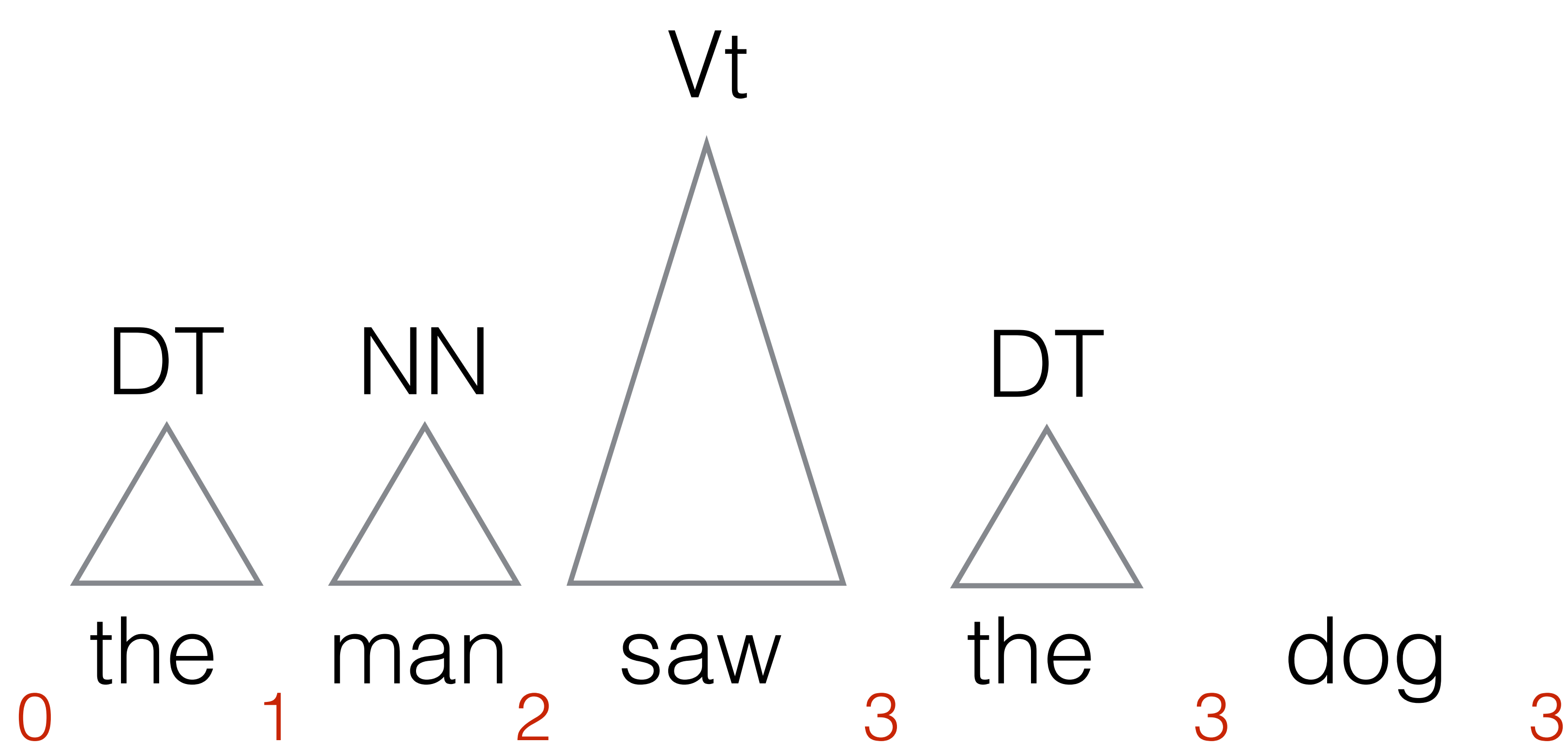
Bottom-Up for CNF: CKY



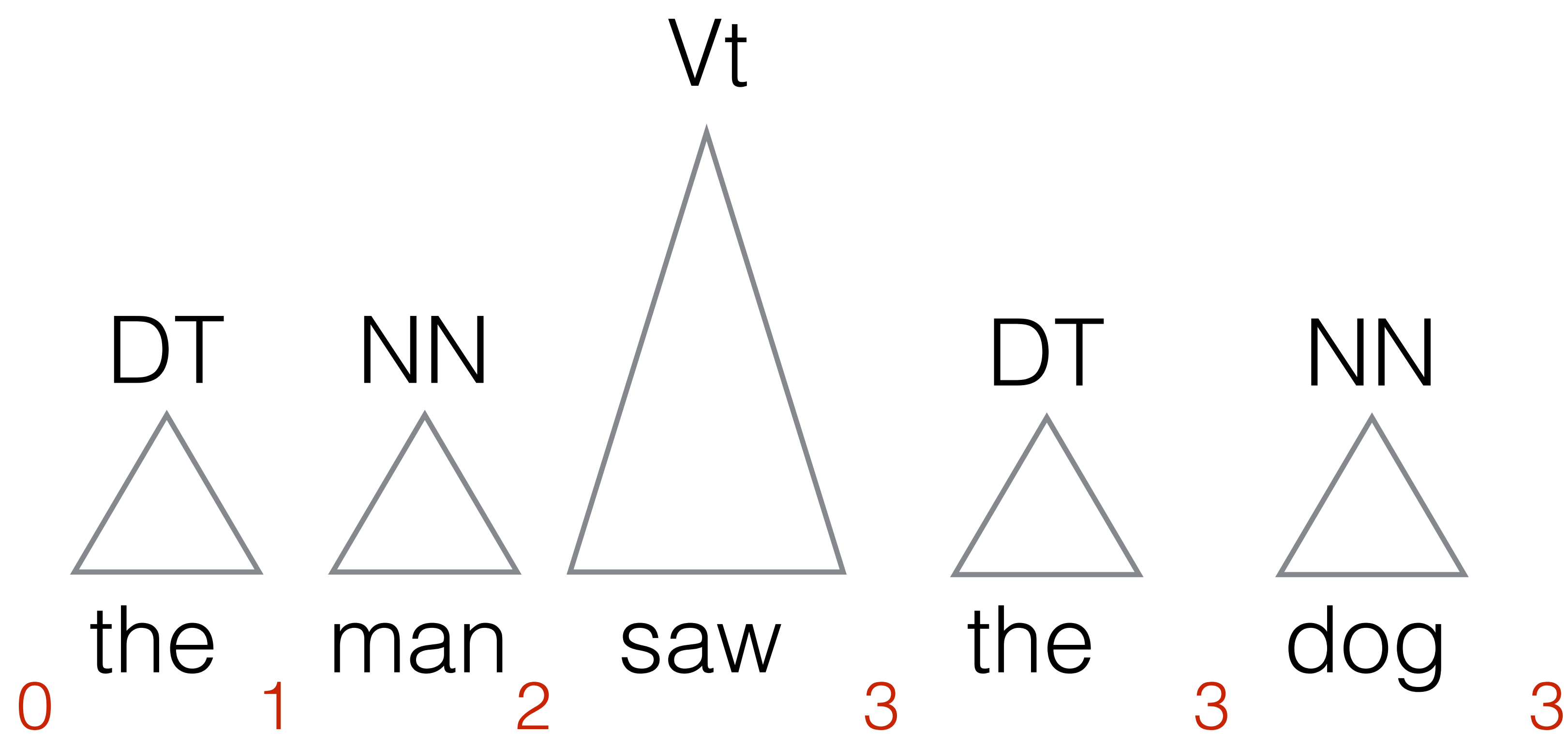
Bottom-Up for CNF: CKY



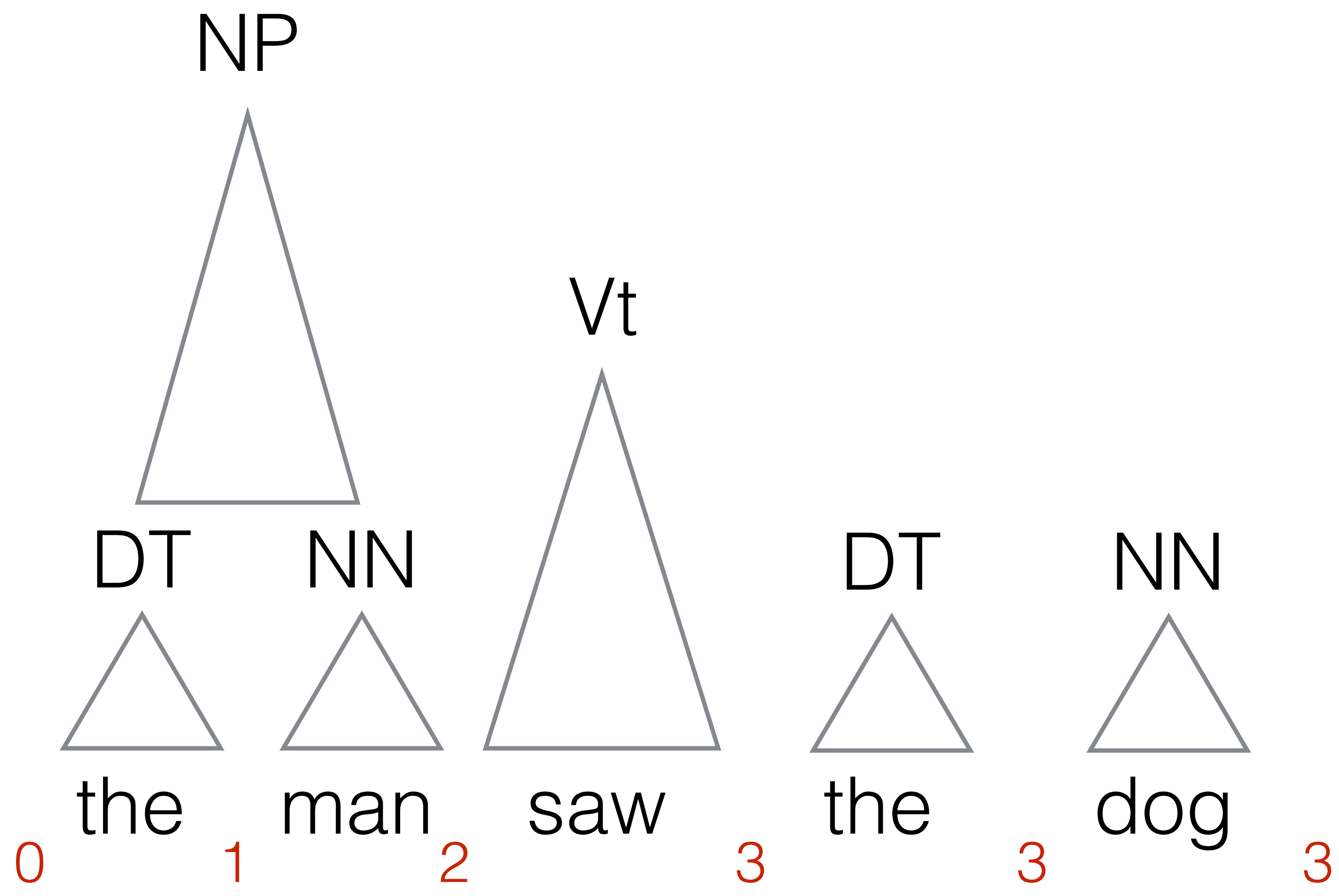
Bottom-Up for CNF: CKY



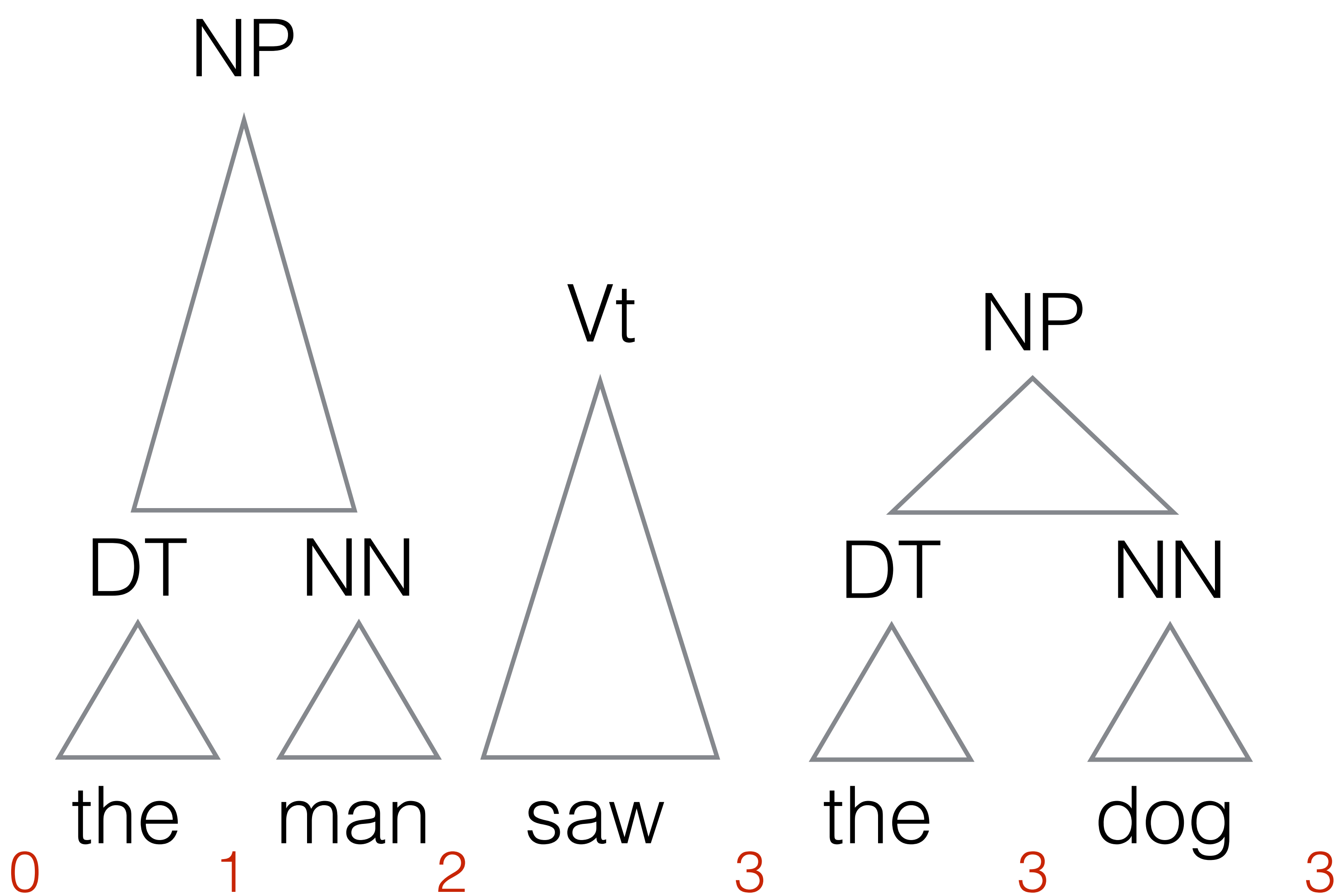
Bottom-Up for CNF: CKY



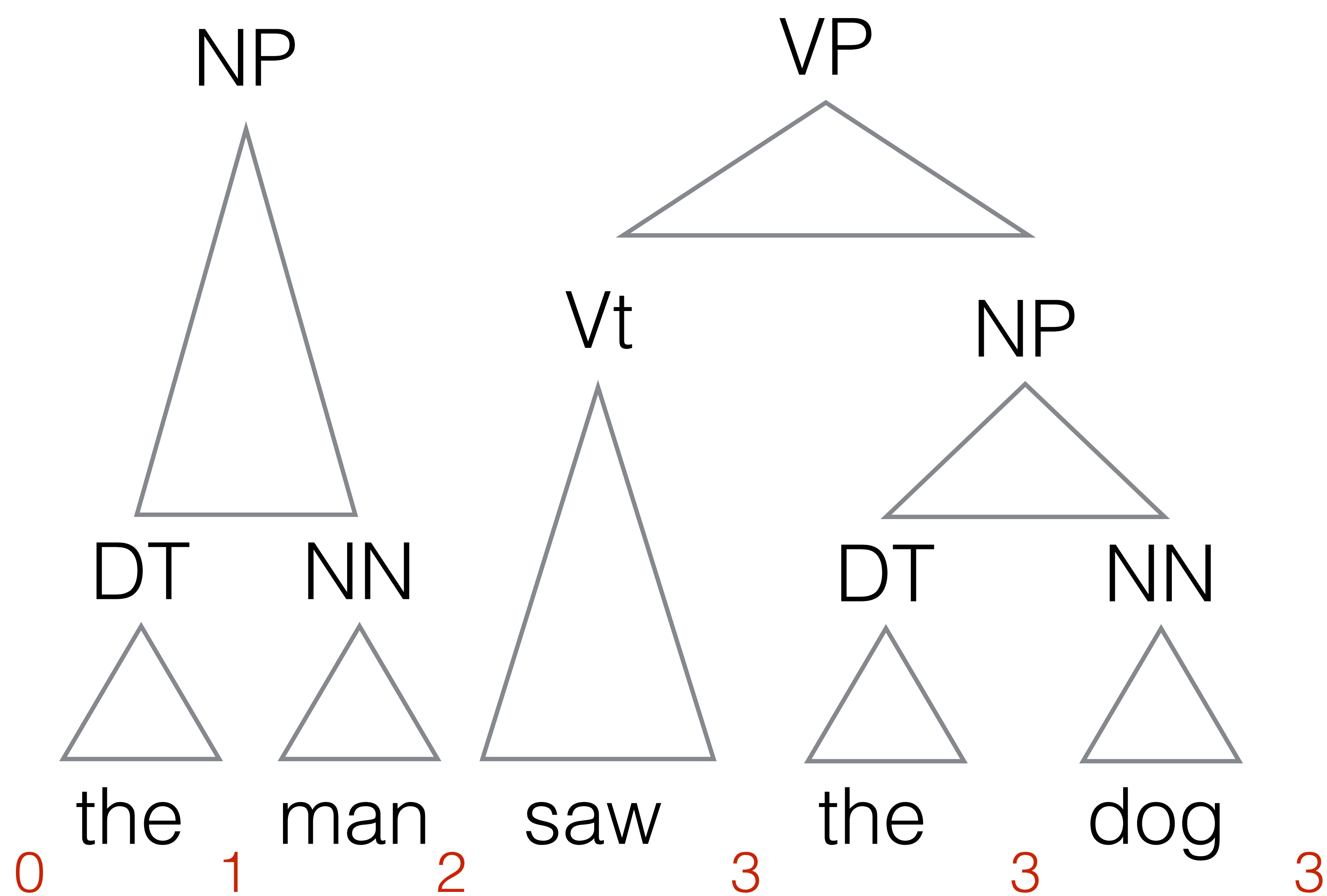
Bottom-Up for CNF: CKY



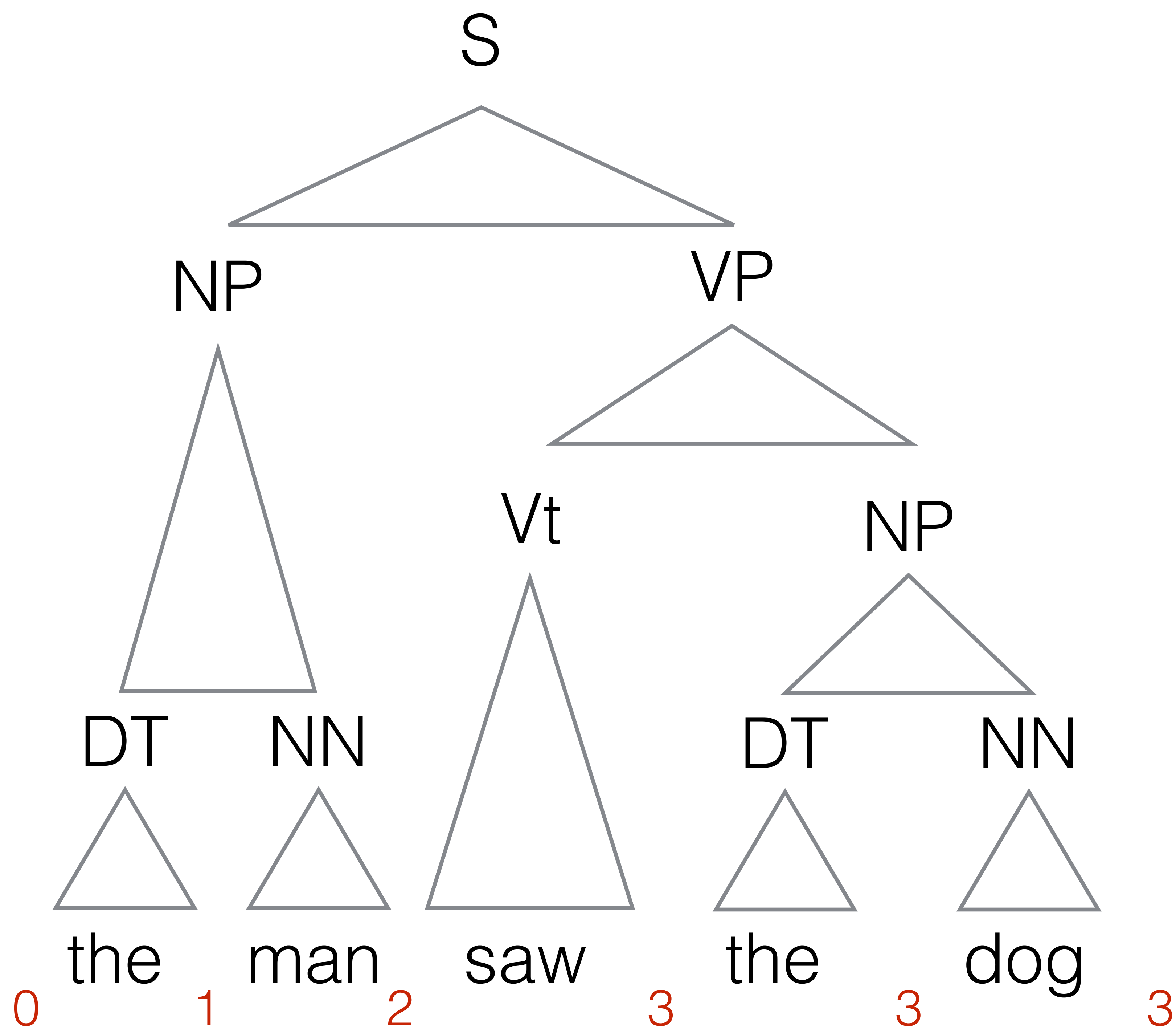
Bottom-Up for CNF: CKY



Bottom-Up for CNF: CKY



Bottom-Up for CNF: CKY



CKY - CNF only

Input: G and $s = x_1 \ldots x_n$

Item form: $[i, X, j]$
asserts that $X \Rightarrow^* x_{i+1} \ldots x_j$

Axioms: $[i, X, i+1]$ $X \rightarrow x_i \in \mathscr{R}$

Goal: $[0, S, n]$

Merge:

$$\frac{[i, A, k][k, B, j]}{[i, C, j]} \quad C \rightarrow A\,B \in \mathcal{R}$$

asserts that
$x_{i+1} \ldots x_k\ x_{k+1} \ldots x_j \Rightarrow^* x_{i+1} \ldots x_j$

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement	Queue	Passive

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition		Statement	Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement	Queue	Passive
Axiom	DT → the	1 [0, DT, 1]	1	
	NN → man	2 [1, NN, 2]	1, 2	
	Vt → saw	3 [2, Vt, 3]	1, 2, 3	
	DT → the	4 [3, DT, 4]	1, 2, 3, 4	
	NN → dog	5 [4, NN, 5]	1, 2, 3, 4, 5	
			2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6 [0, NP, 2]	3, 4, 5, 6	2

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition		Statement	Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4
Merge: [4][5]	NP → DT NN	7	[3, NP, 5]	6, 7	5

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4
Merge: [4][5]	NP → DT NN	7	[3, NP, 5]	6, 7	5
				7	6

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4
Merge: [4][5]	NP → DT NN	7	[3, NP, 5]	6, 7	5
				7	6
Merge: [3] [7]	VP → Vt NP	8	[2, VP, 5]	8	7

CKY Example

Input: *the man saw the dog*

S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4
Merge: [4][5]	NP → DT NN	7	[3, NP, 5]	6, 7	5
				7	6
Merge: [3] [7]	VP → Vt NP	8	[2, VP, 5]	8	7
Merge: [6] [8]	S → NP VP	9	[0, S, 5]	9	8

CKY Example

Input: *the man saw the dog*

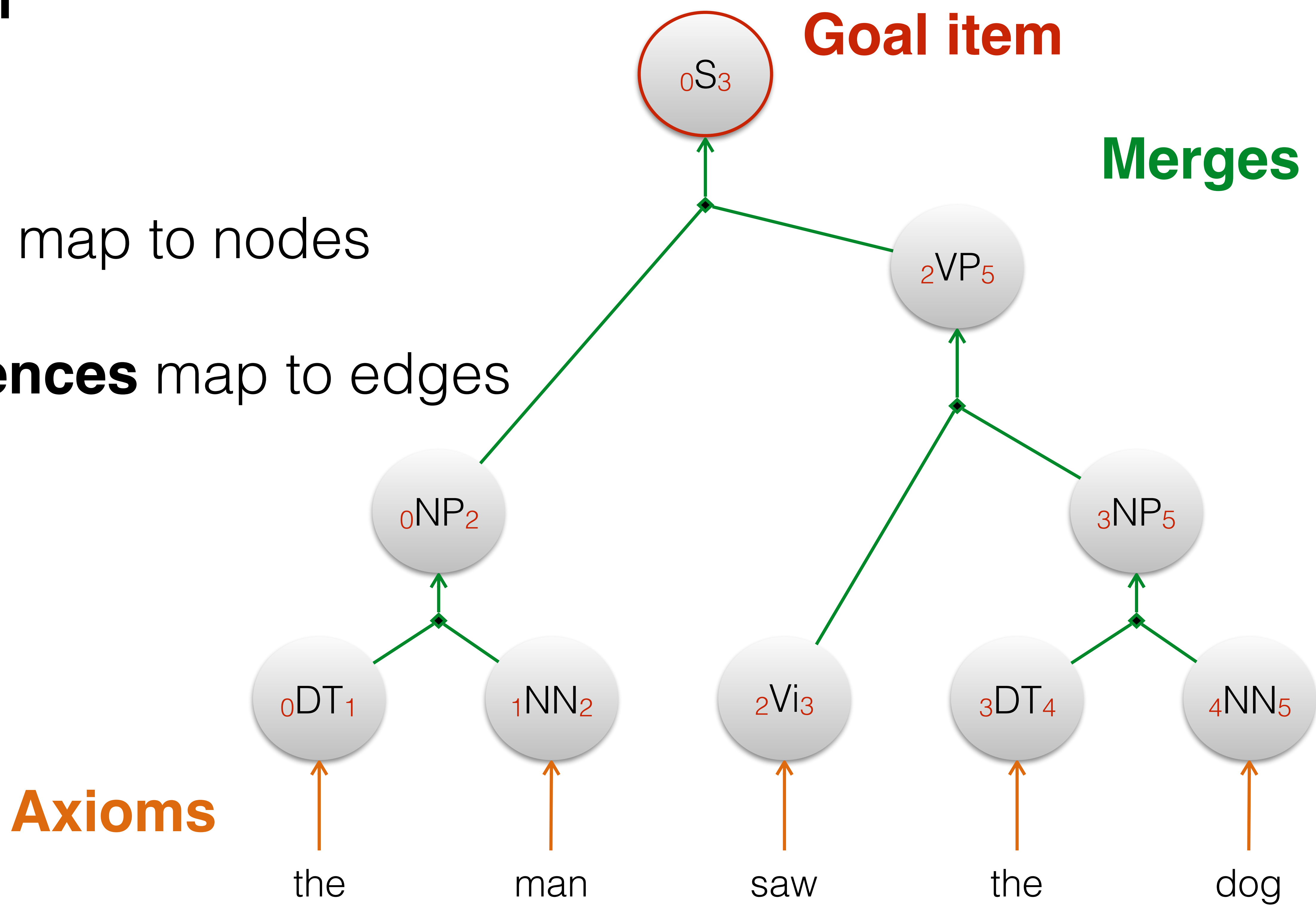
S → NP VP	Vi → sleeps
~~VP → Vi~~	Vt → saw
VP → Vt NP	NN → man
VP → VP PP	NN → dog
NP → DT NN	NN → telescope
NP → NP PP	DT → the
PP → IN NP	IN → with

Rule	Condition	Statement		Queue	Passive
Axiom	DT → the	1	[0, DT, 1]	1	
	NN → man	2	[1, NN, 2]	1, 2	
	Vt → saw	3	[2, Vt, 3]	1, 2, 3	
	DT → the	4	[3, DT, 4]	1, 2, 3, 4	
	NN → dog	5	[4, NN, 5]	1, 2, 3, 4, 5	
				2, 3, 4, 5	1
Merge: [1][2]	NP → DT NN	6	[0, NP, 2]	3, 4, 5, 6	2
				4, 5, 6	3
				5, 6	4
Merge: [4][5]	NP → DT NN	7	[3, NP, 5]	6, 7	5
				7	6
Merge: [3] [7]	VP → Vt NP	8	[2, VP, 5]	8	7
Merge: [6] [8]	S → NP VP	9	[0, S, 5]	9	8
GOAL: [9]				∅	9

Graphical view

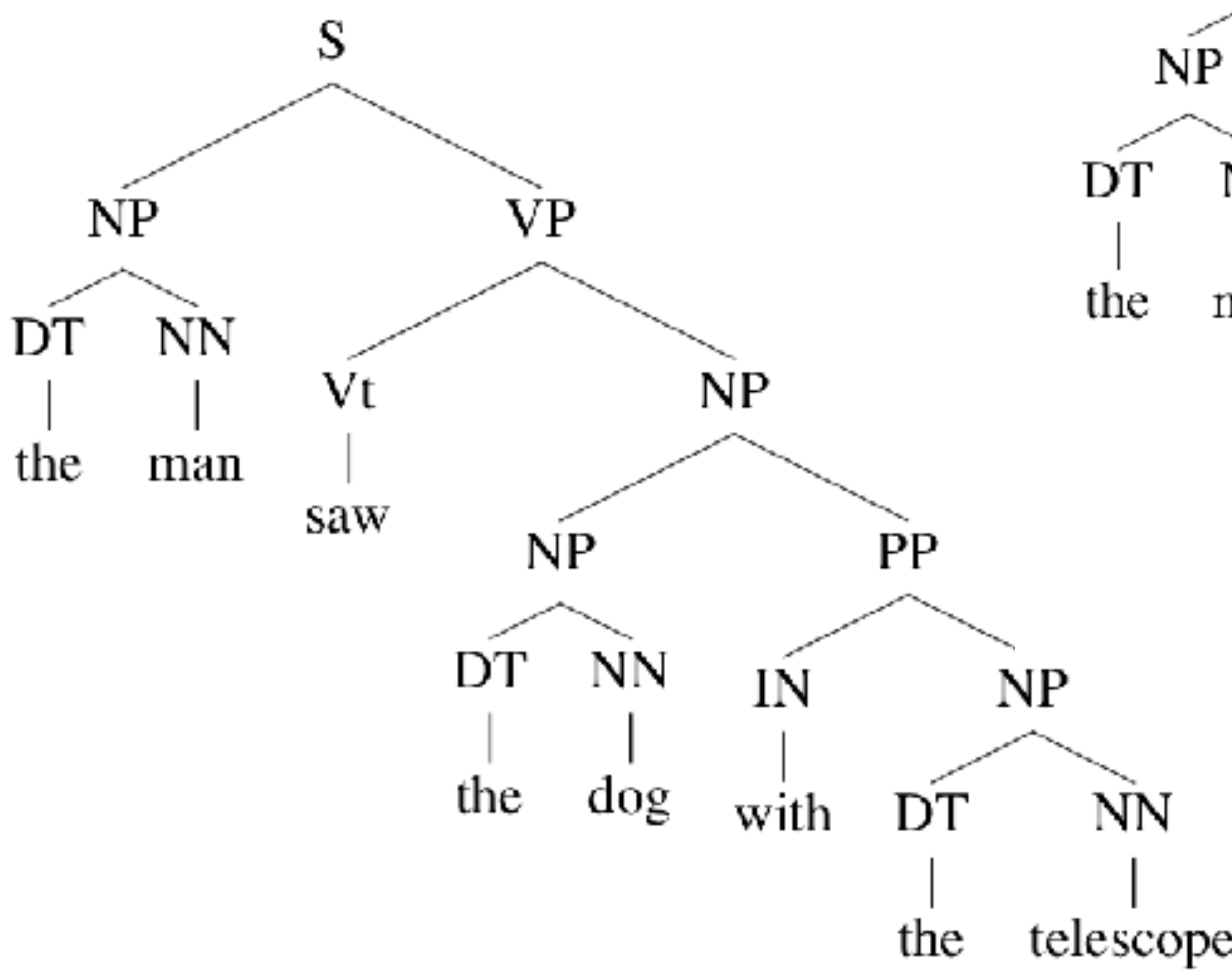
Items map to nodes

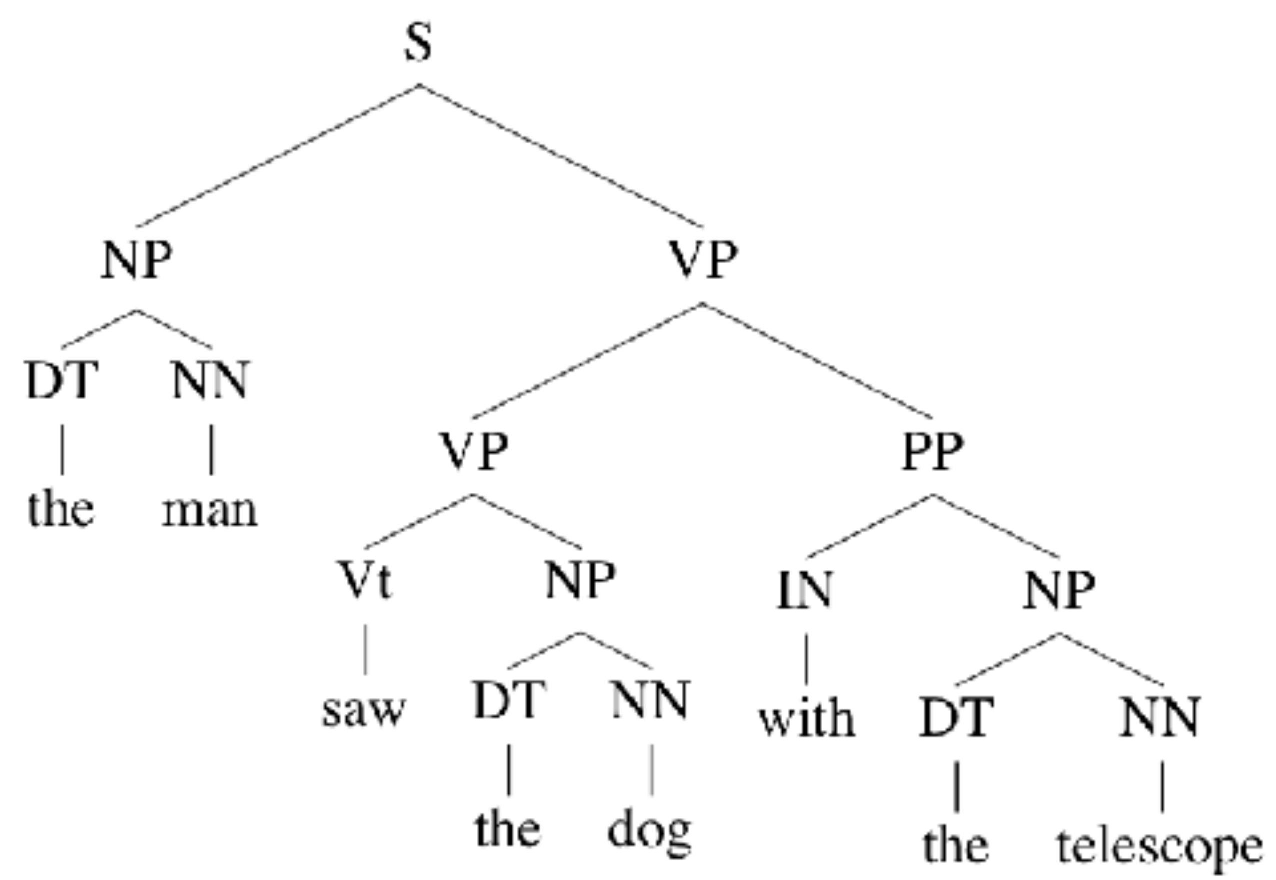
Inferences map to edges



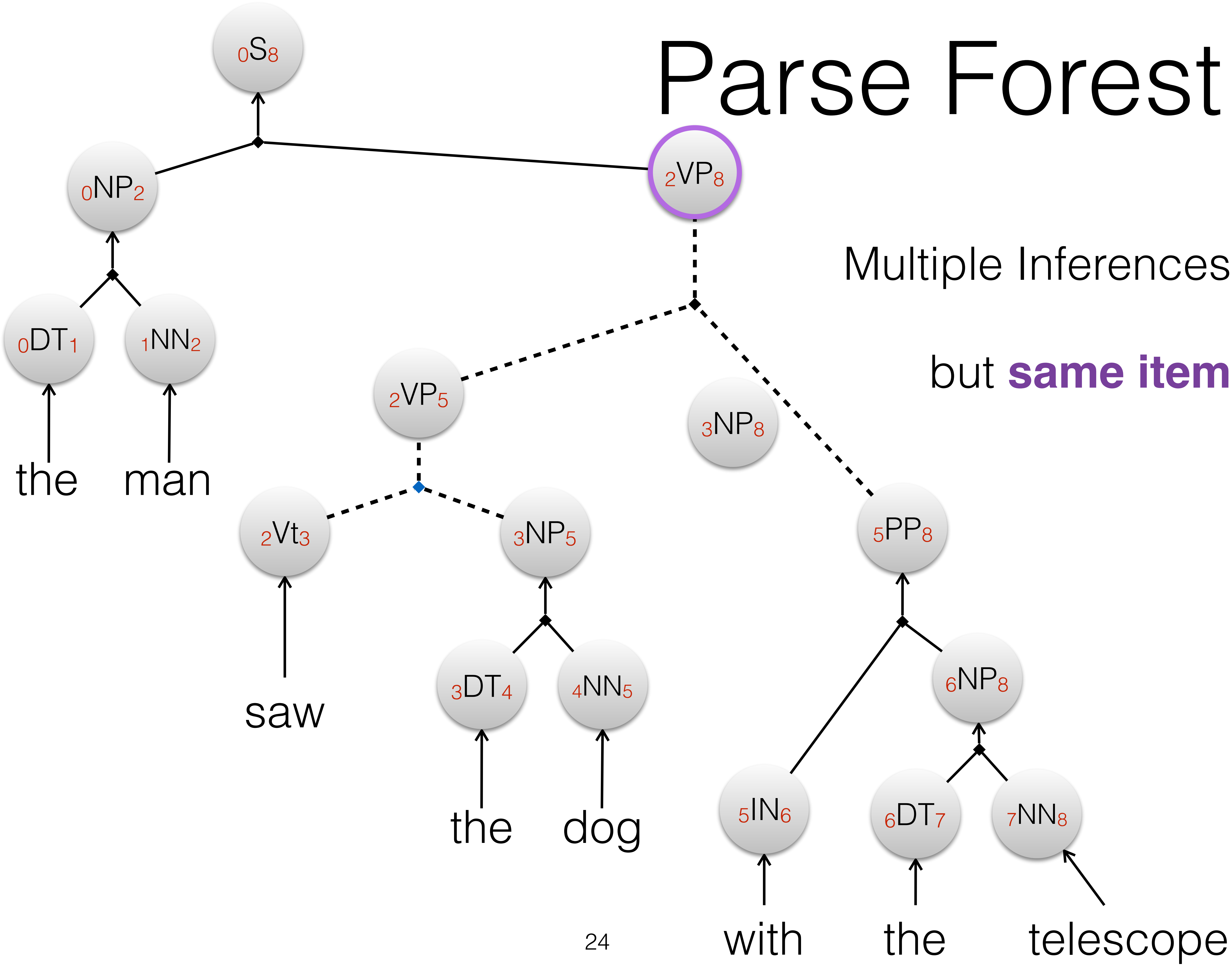
Ambiguity

Some strings may have more than one derivation in G





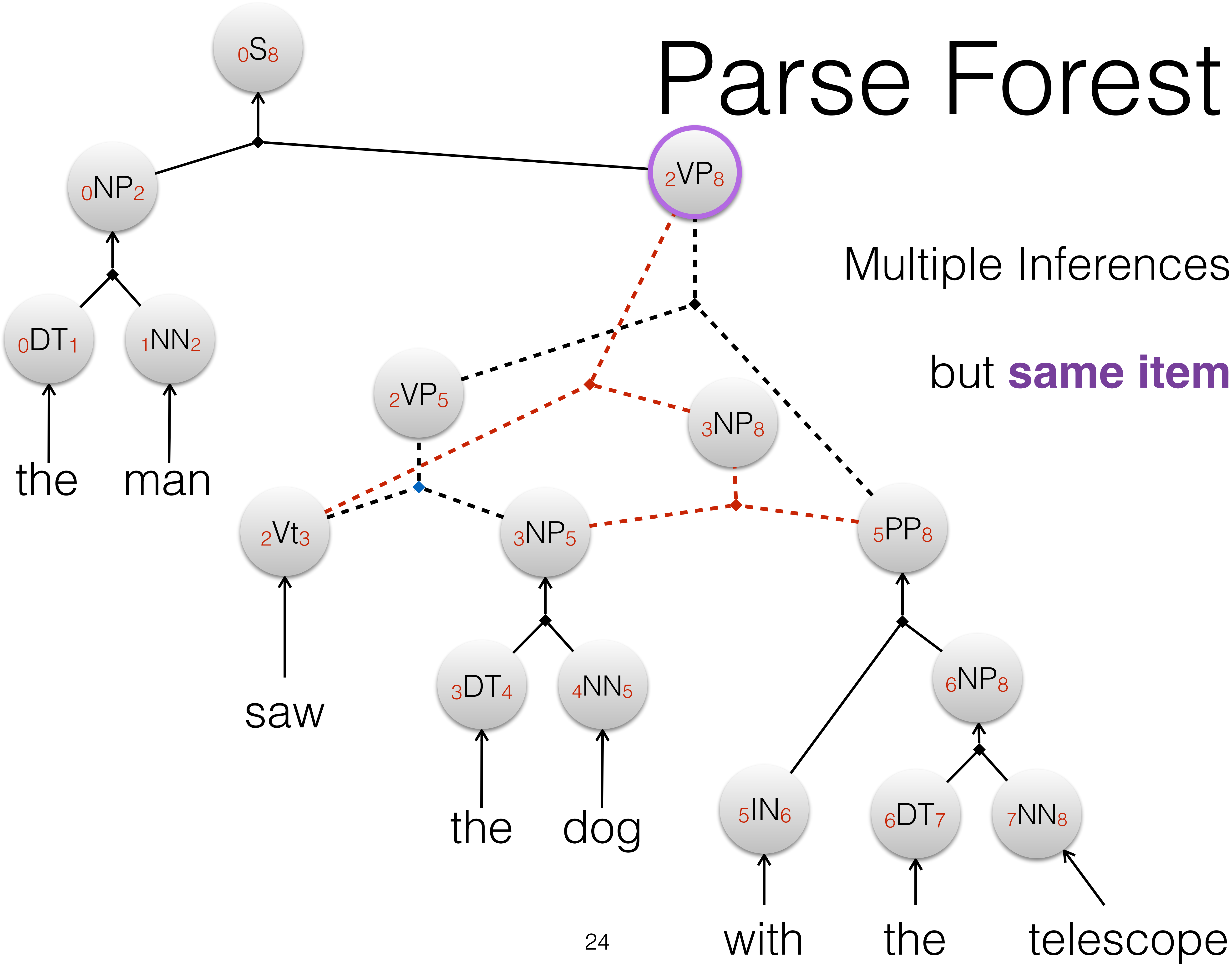
Parse Forest



Multiple Inferences

but **same item**

Parse Forest



Multiple Inferences

but **same item**

Parse Forest

Efficient representation of the whole space $T_G(\omega)$

- each and every possible tree yielding ω

Items (other than the goal) represent partial derivations

- including alternative ones

Dealing with Ambiguity

Statistical model: PCFG

- weight steps in a derivation
- induces a partial ordering over derivations
- can be used to make a decision
    - e.g. best tree under the model

Probabilistic CFG

CFG extended with parameters $0 \leq \theta_r \leq 1$

- where $r \in \mathscr{R}$ and

$$\sum_{\beta : v \rightarrow \beta \in \mathcal{R}} \theta_{v \rightarrow \beta} = 1$$

Probabilistic CFG

Distribution over trees and their yields

$$P_{DS|NM}(R_1^m = r_1^m, X_1^n = \text{yield}(r_1^m)|n, m)$$
$$= \prod_{i=1}^{m} \theta_{r_i} = \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$
$$\text{where } r_i \text{ corresponds to } v_i \to \beta_i$$

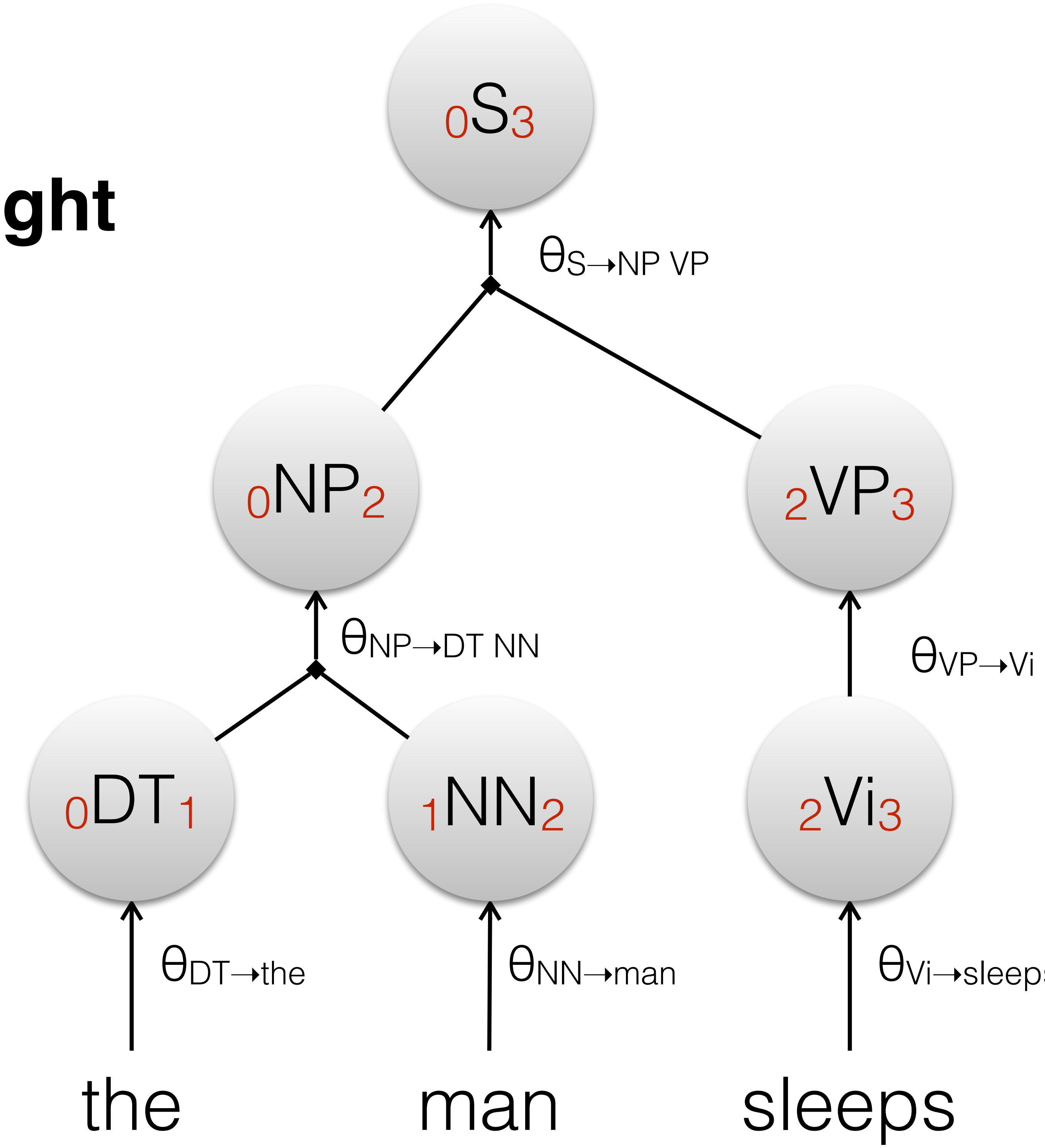
Joint Distribution

Each inference gets a **weight**

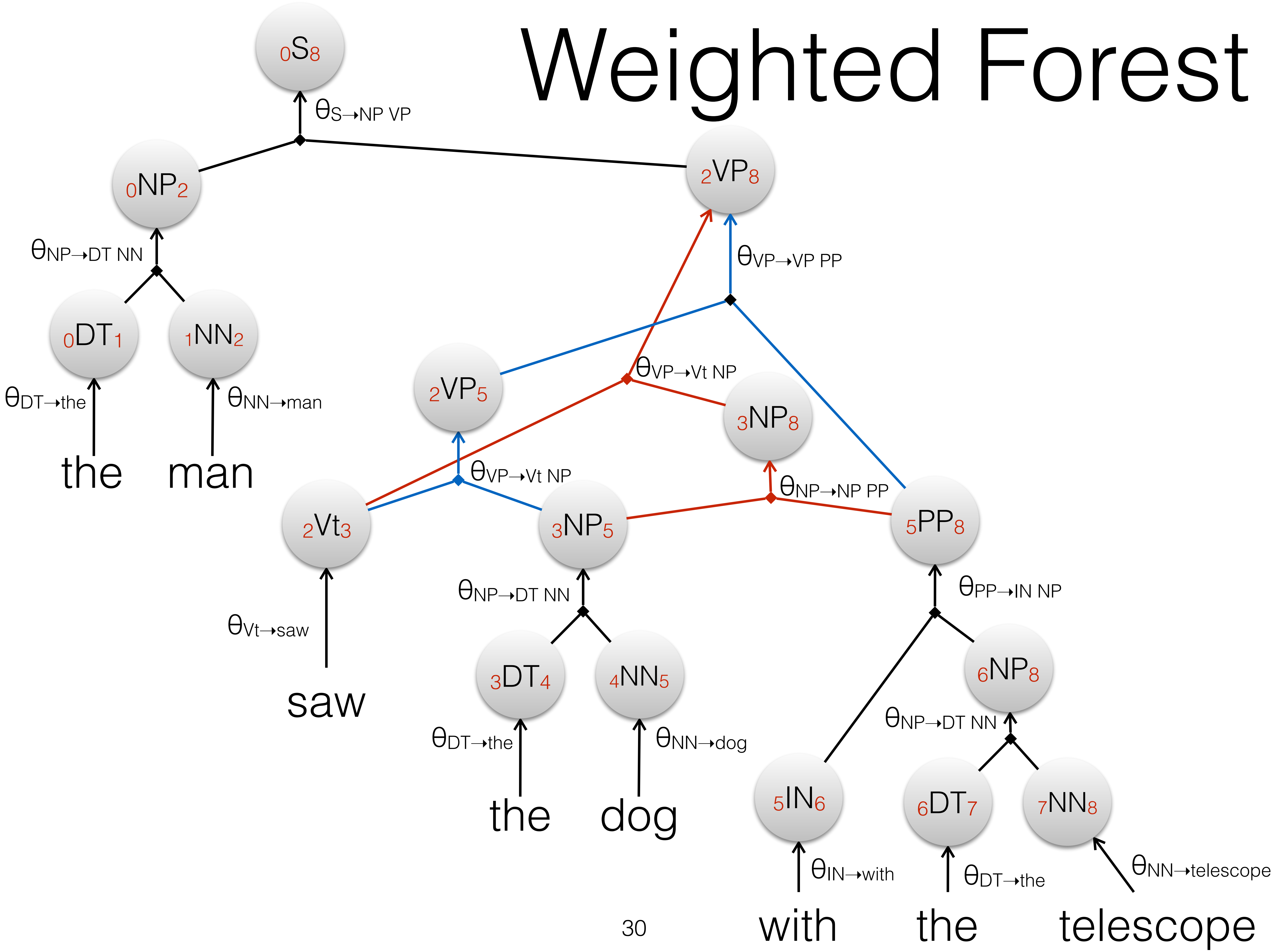
i.e. categorical parameter

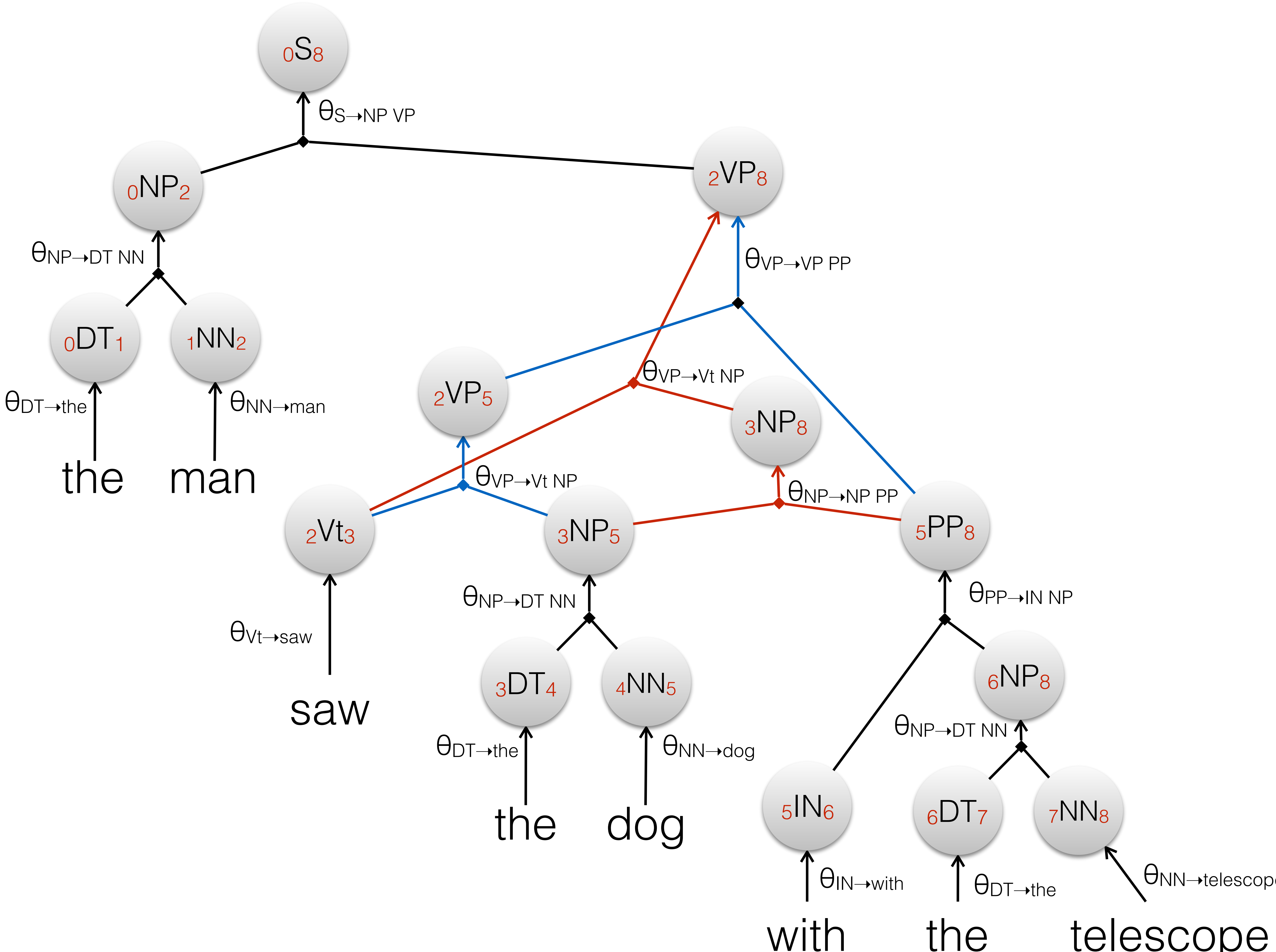
$$\theta_{X \to \beta}$$

of the underlying rule



Weighted Forest





0S8
θS→NP VP
0NP2
2VP8
θNP→DT NN
θVP→VP PP
0DT1
1NN2
θDT→the
θNN→man
the
man
2VP5
θVP→Vt NP
3NP8
θVP→Vt NP
θNP→NP PP
2Vt3
3NP5
5PP8
θNP→DT NN
θPP→IN NP
θVt→saw
saw
3DT4
4NN5
6NP8
θDT→the
θNN→dog
θNP→DT NN
the
dog
5IN6
6DT7
7NN8
θIN→with
θDT→the
θNN→telescope
with
the
telescope

Marginal Probability

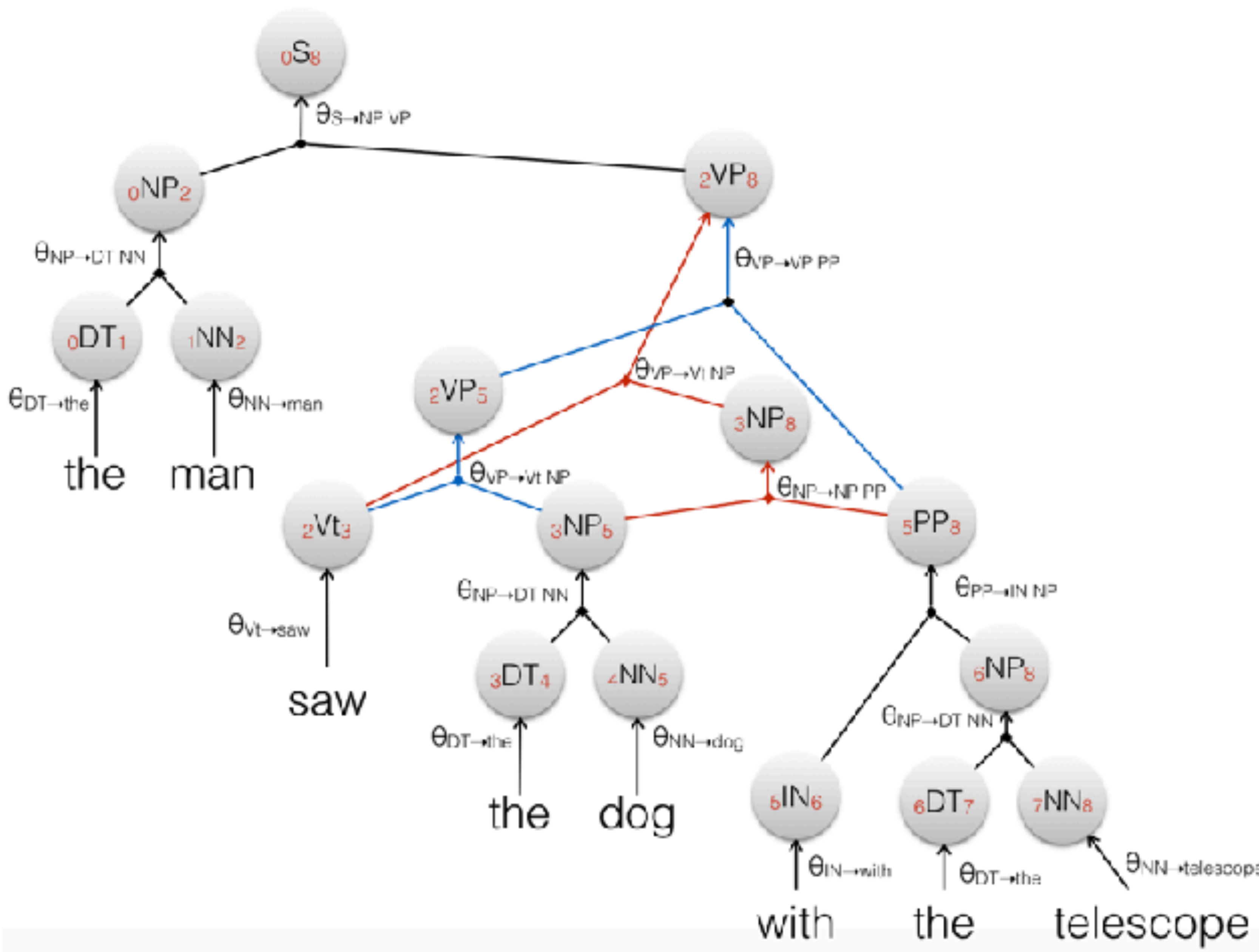
$$P_{S|n}(x_1^n|n) = I({}_0\mathrm{S}_n) = \sum_{r_1^m \in \mathcal{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$

Marginal Probability

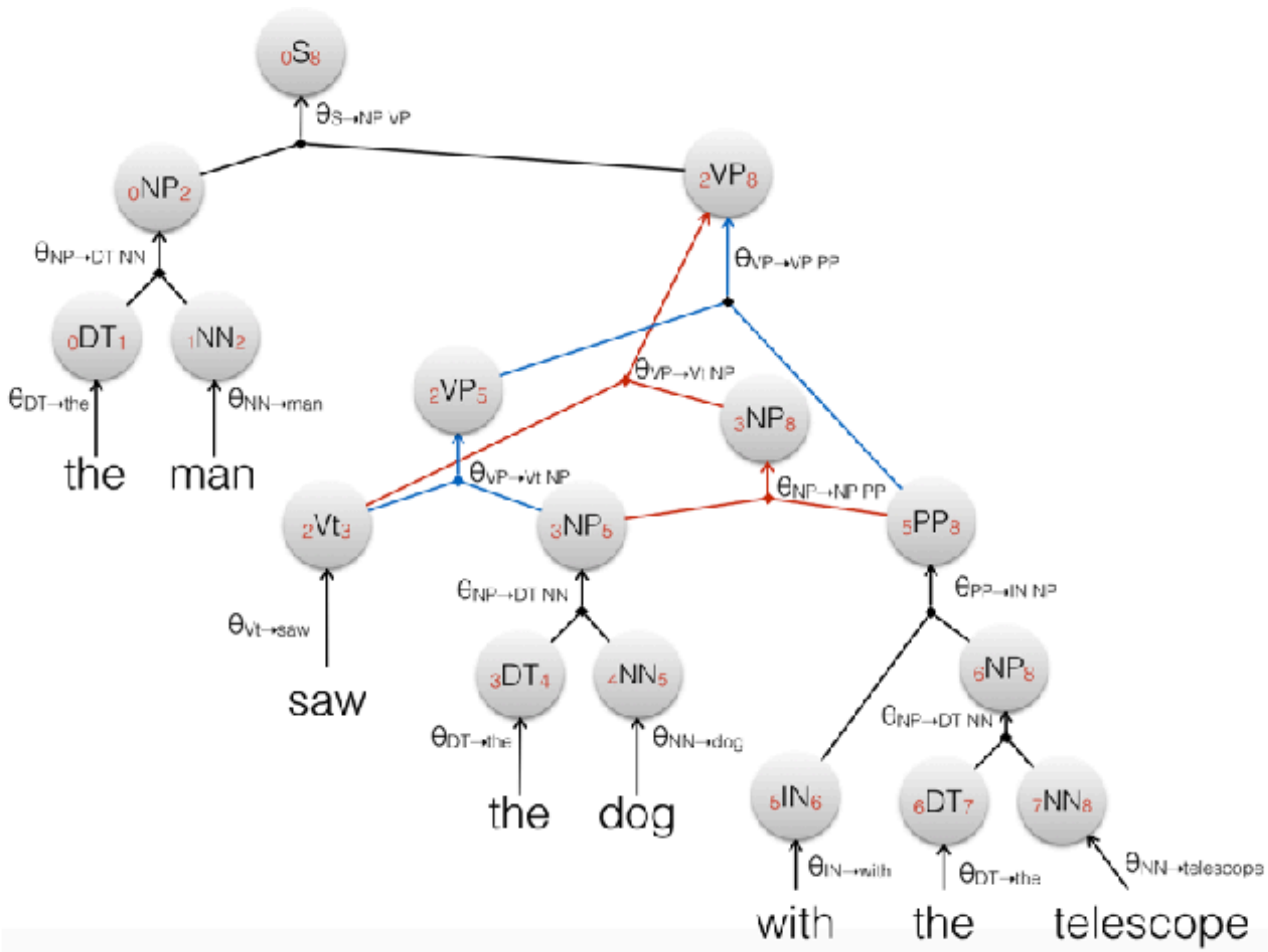
Let the goal item **stand** for the sentence. What's its (marginal/inside) probability $I({}_0S_8)$?

$$P_{S|n}(x_1^n|n) = I({}_0\mathrm{S}_n) = \sum_{r_1^m \in \mathcal{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$

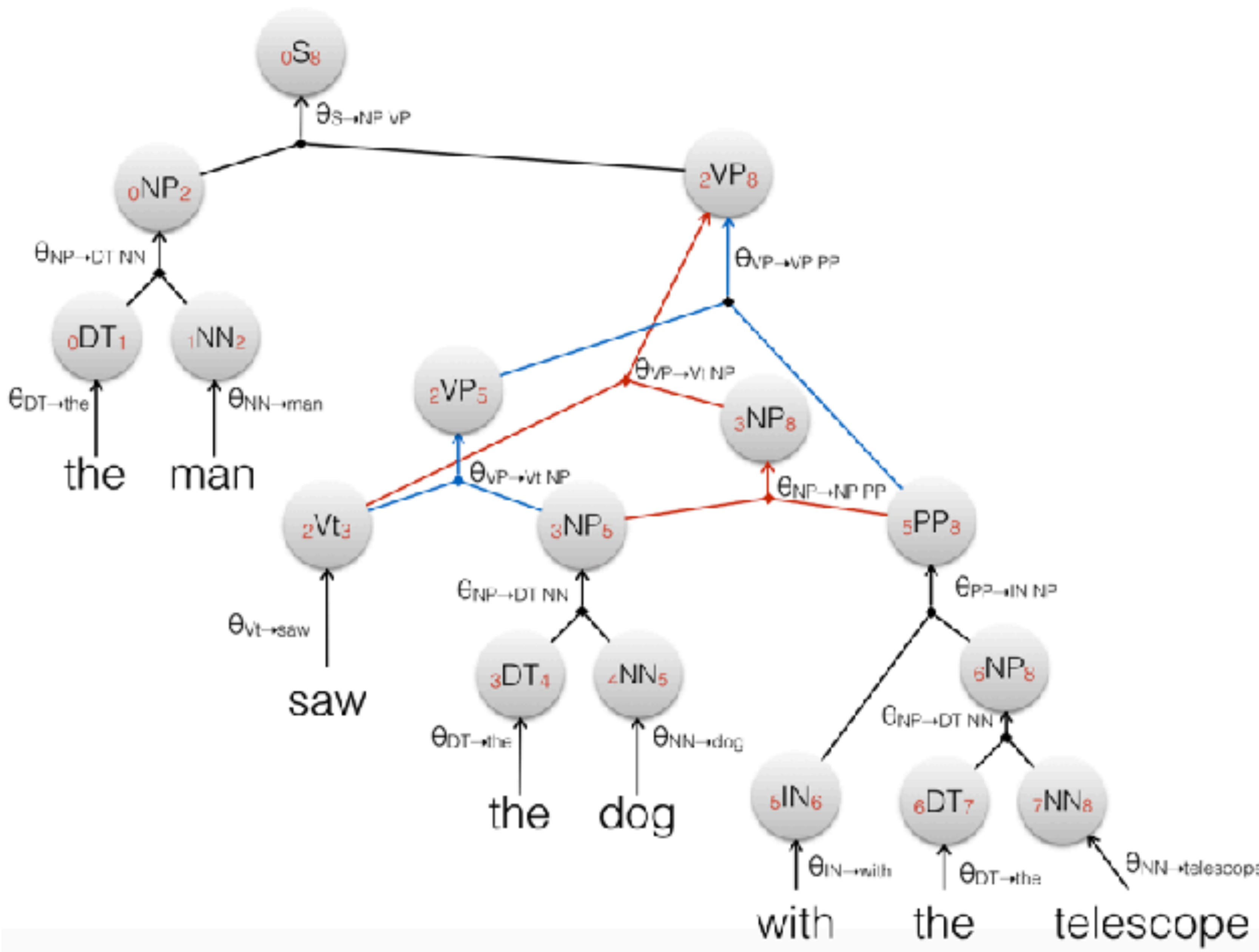
Marginal Probability



Marginal Probability

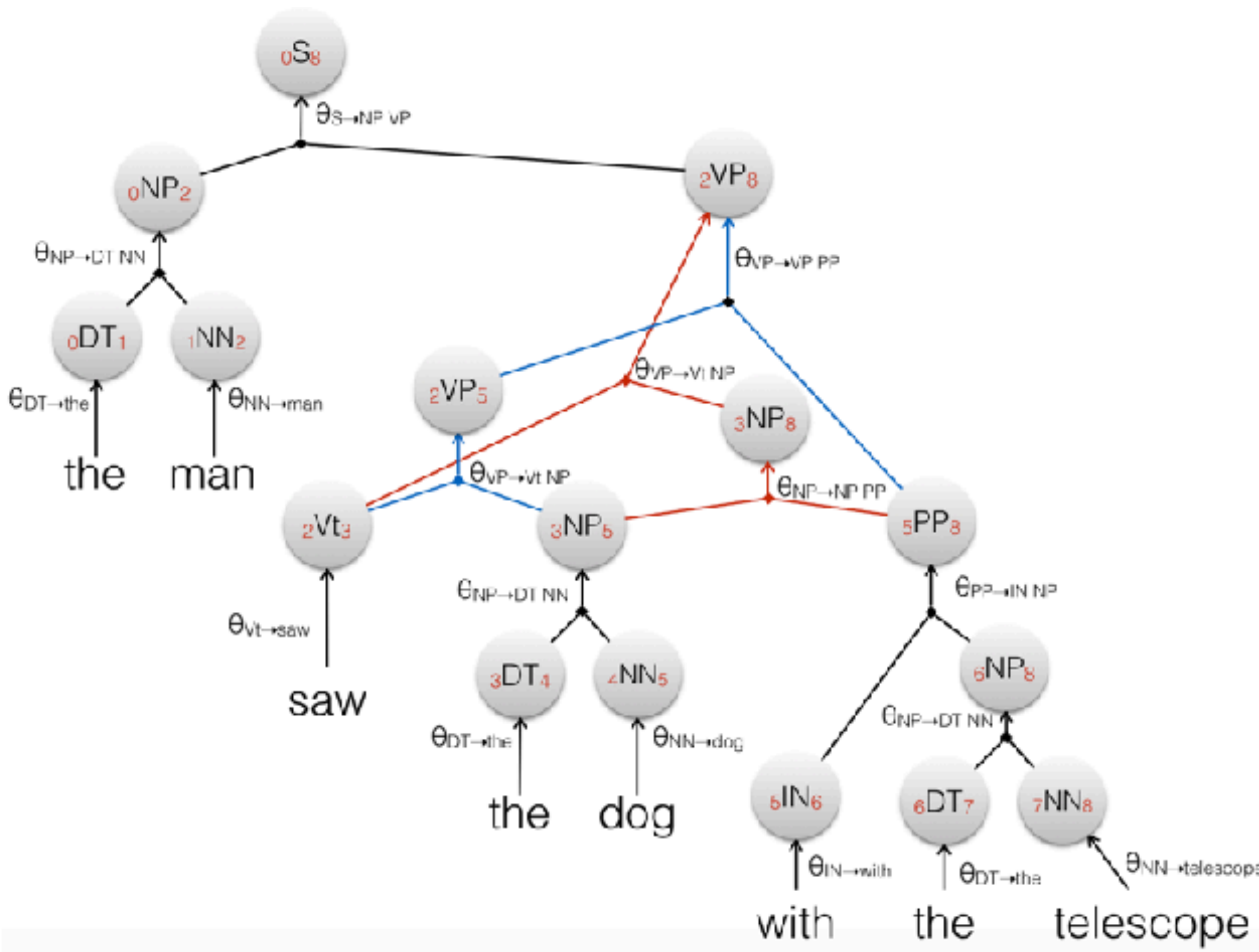


Marginal Probability



- $I({}_0S_8) =$

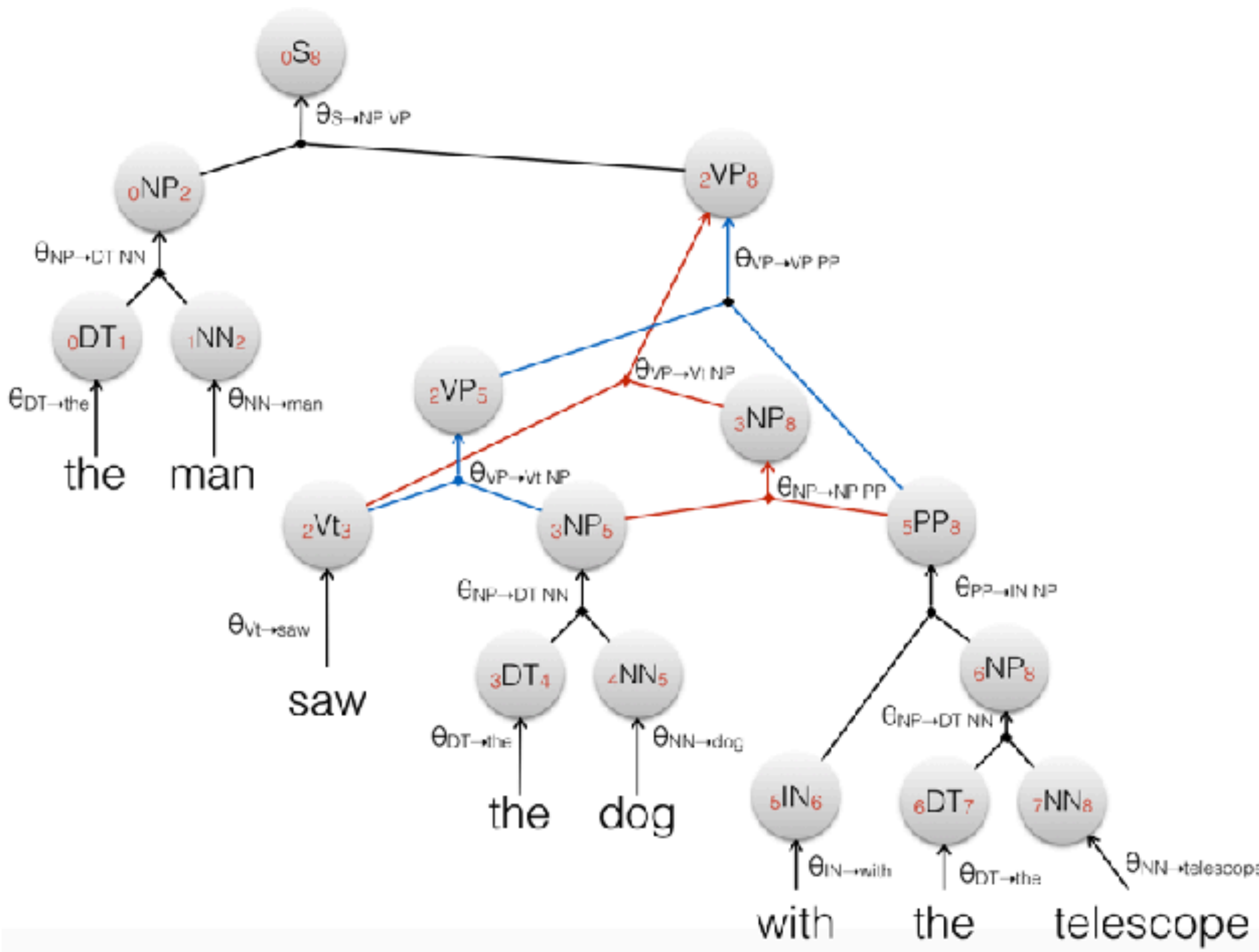
Marginal Probability



- $I({}_0S_8) =$

  $\theta_{S \to NP\ VP}\, I({}_0NP_2)\, I({}_2VP_8)$

Marginal Probability

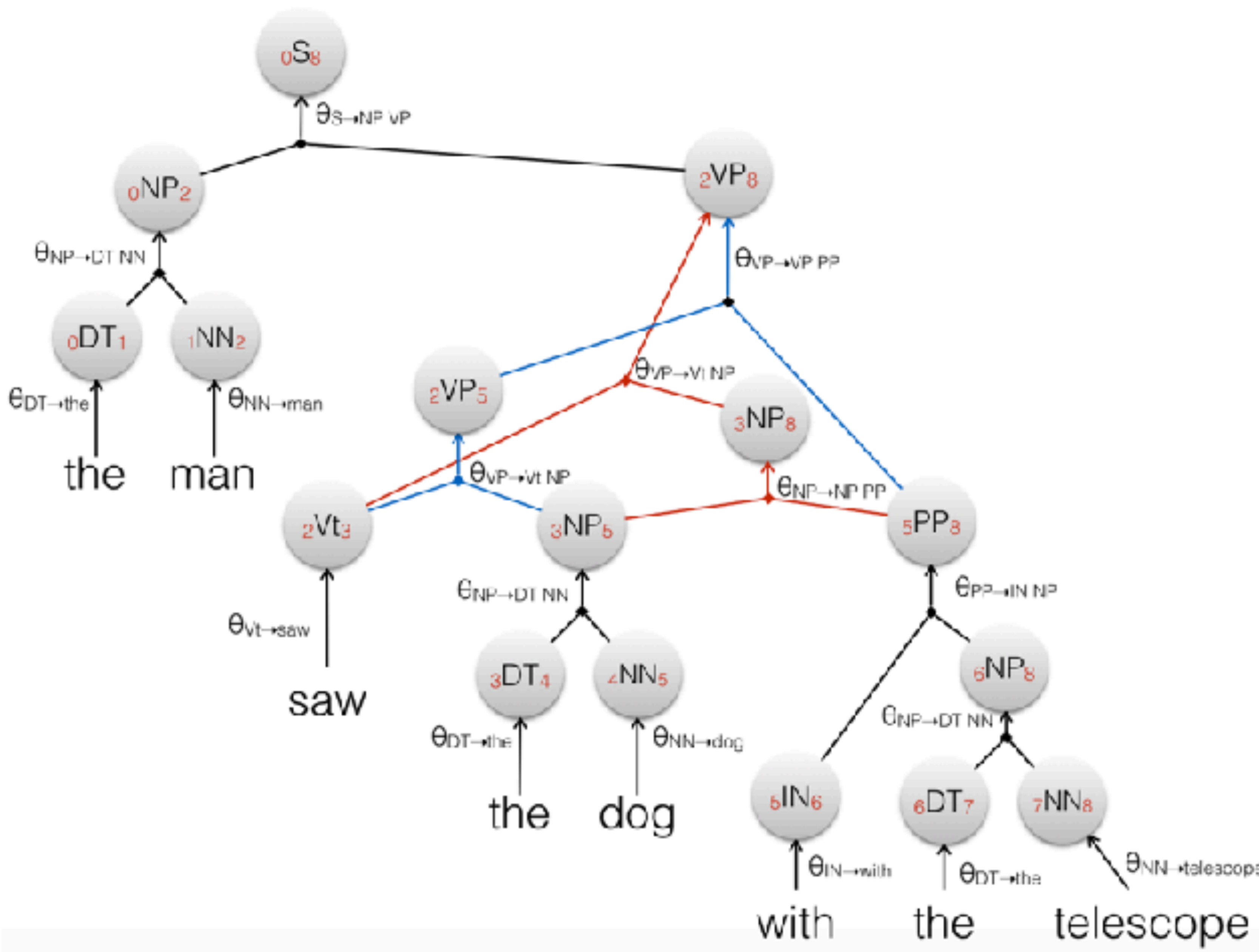


- $I({}_0S_8) =$

  $\theta_{S \to NP\ VP}\ I({}_0NP_2)\ I({}_2VP_8)$

- $I({}_0NP_2) =$

Marginal Probability



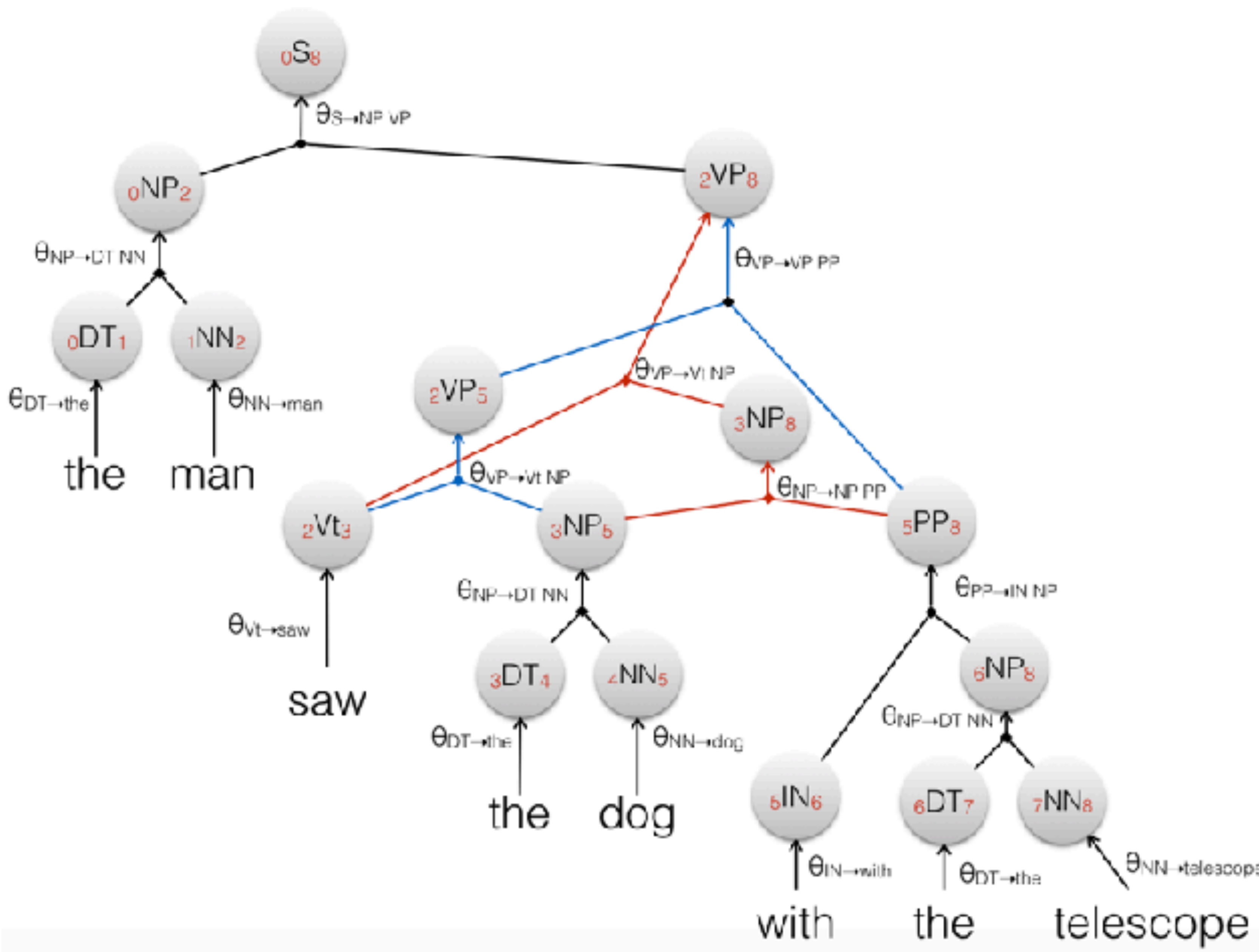
- $I({}_0S_8) =$

  $\theta_{S\rightarrow NP\ VP}\ I({}_0NP_2)\ I({}_2VP_8)$

- $I({}_0NP_2) =$

  $\theta_{NP\rightarrow DT\ NN}\ I({}_0DT_1)\ I({}_1NN_2)$

Marginal Probability



- $I({}_0S_8) =$

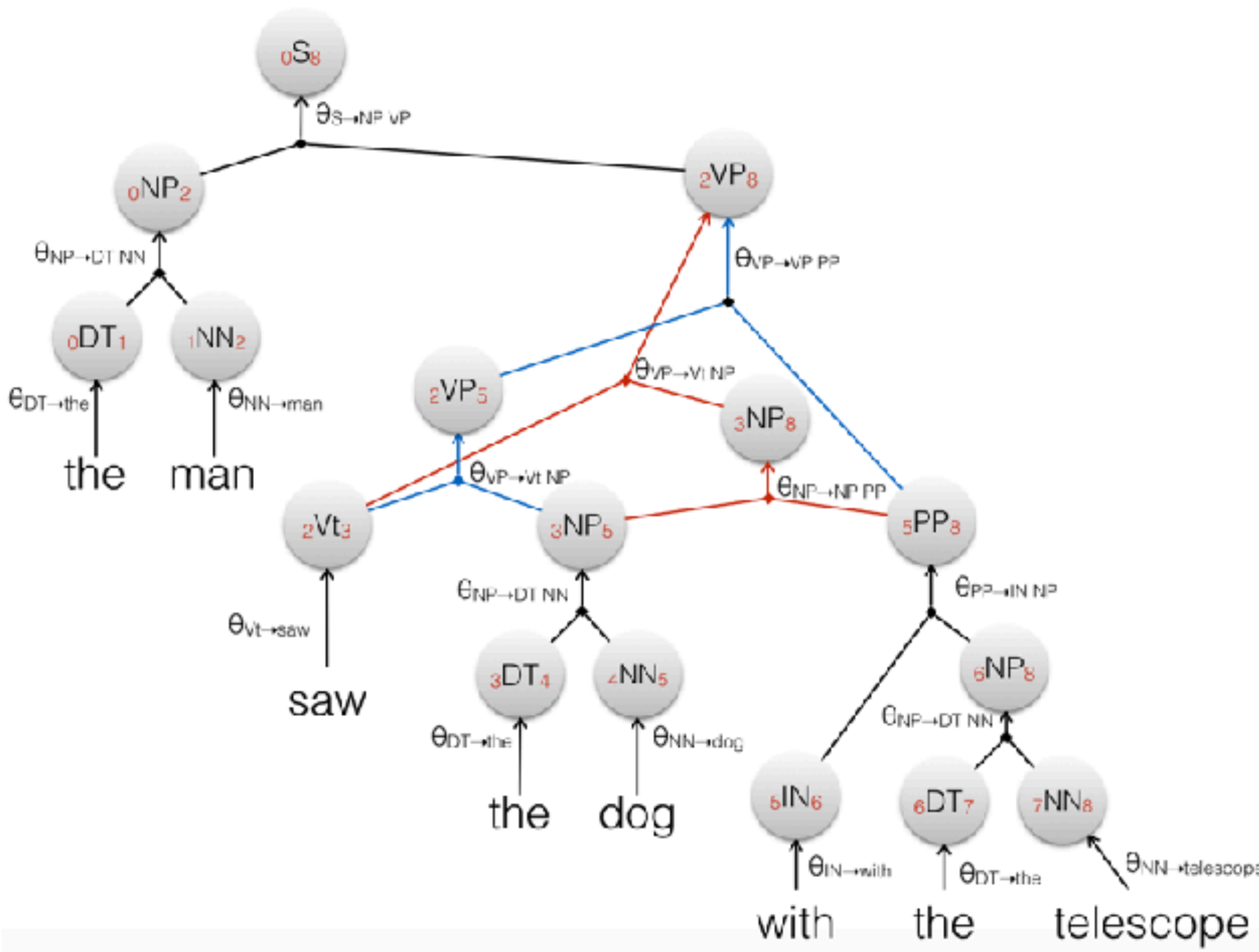
  $\theta_{S\to NP\ VP}\, I({}_0NP_2)\, I({}_2VP_8)$

- $I({}_0NP_2) =$

  $\theta_{NP\to DT\ NN}\, I({}_0DT_1)\, I({}_1NN_2)$

- $I({}_2VP_8) =$

Marginal Probability



- $I({}_0S_8) =$

  $\theta_{S\to NP\ VP}\ I({}_0NP_2)\ I({}_2VP_8)$

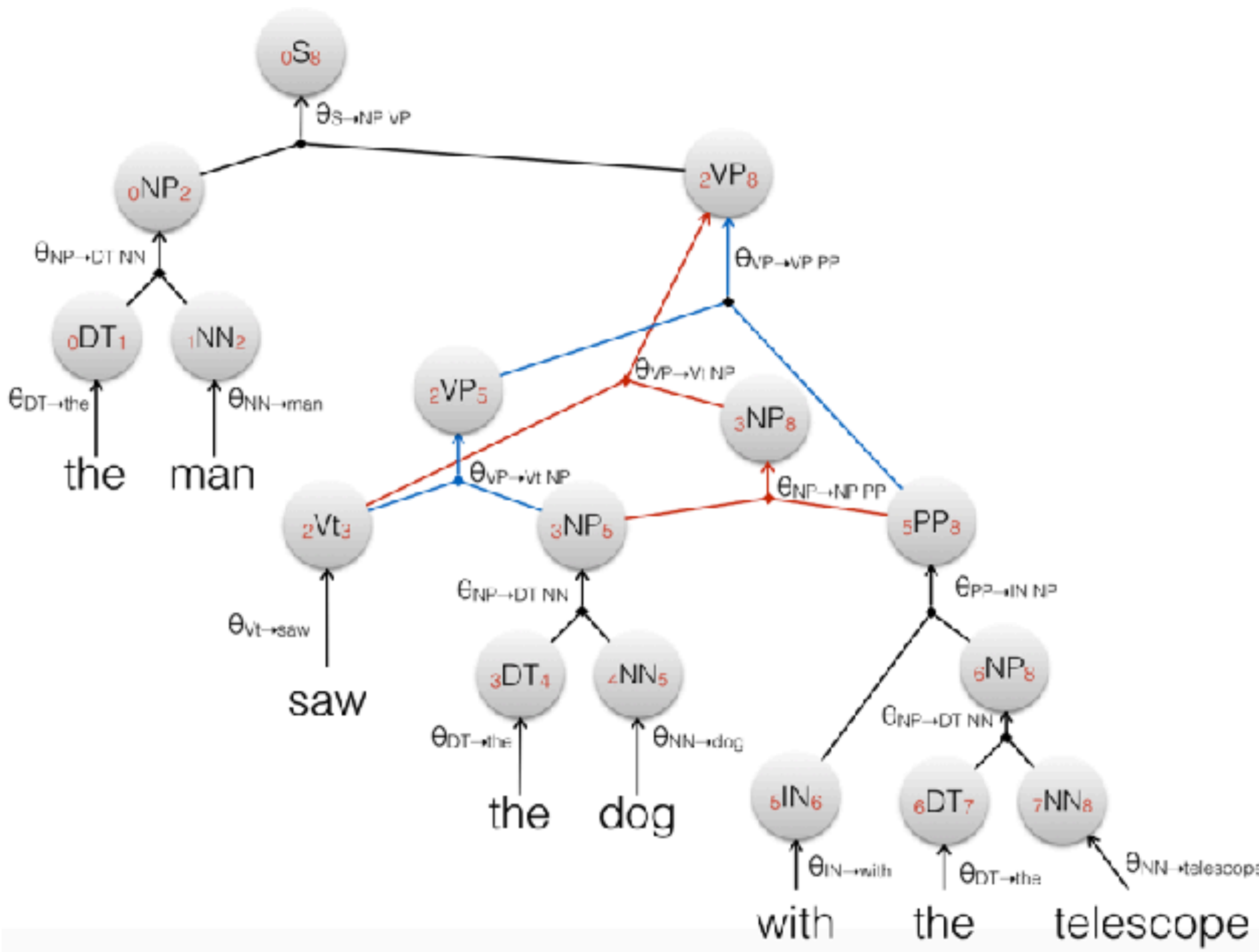
- $I({}_0NP_2) =$

  $\theta_{NP\to DT\ NN}\ I({}_0DT_1)\ I({}_1NN_2)$

- $I({}_2VP_8) =$

  $\theta_{VP\to VP\ PP}\ I({}_2VP_5)\ I({}_5PP_8)$

Marginal Probability



- $I({}_0S_8) =$

  $\theta_{S \to NP\ VP}\ I({}_0NP_2)\ I({}_2VP_8)$

- $I({}_0NP_2) =$

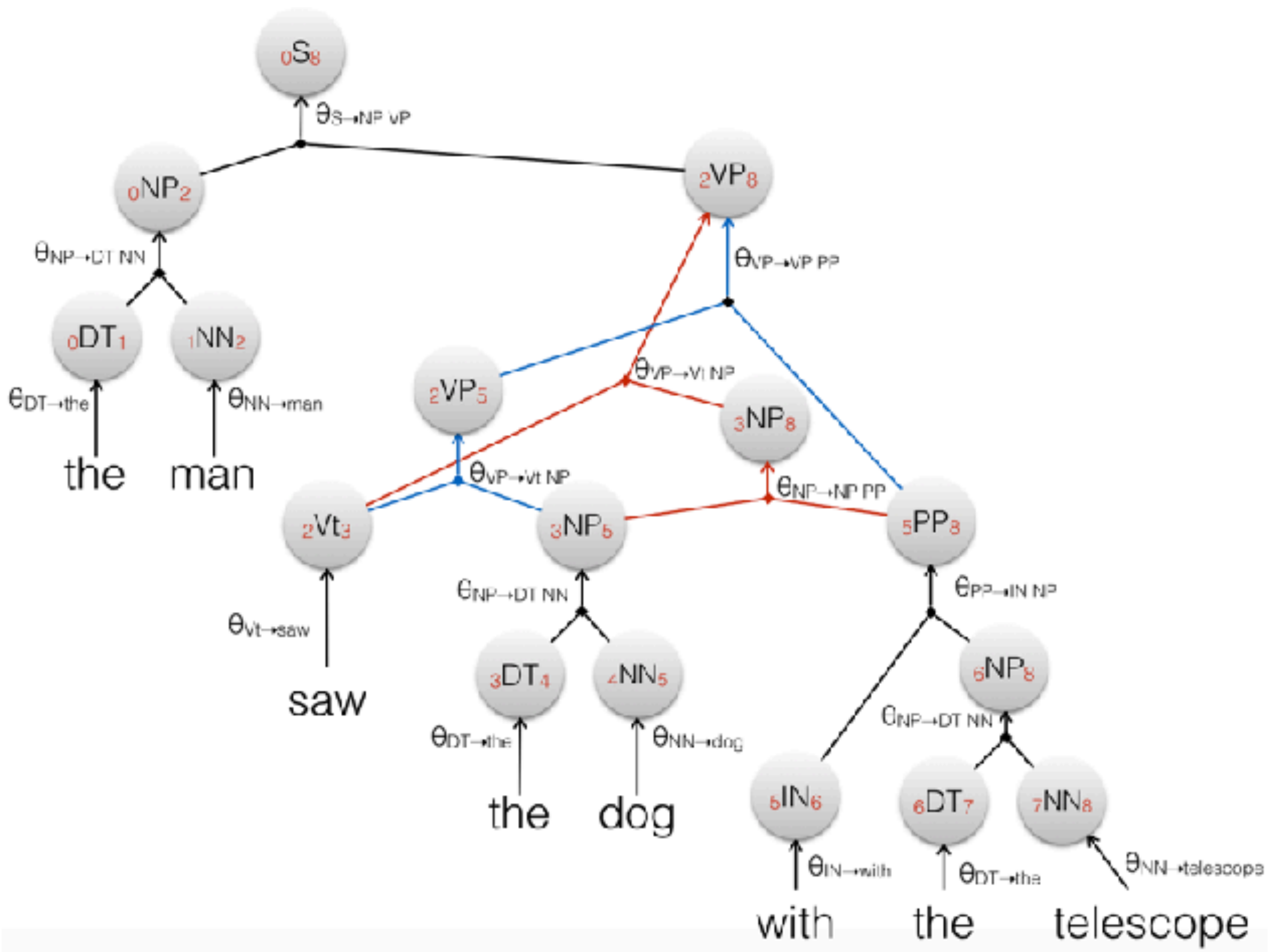
  $\theta_{NP \to DT\ NN}\ I({}_0DT_1)\ I({}_1NN_2)$

- $I({}_2VP_8) =$

  $\theta_{VP \to VP\ PP}\ I({}_2VP_5)\ I({}_5PP_8)$

  $+\ \theta_{VP \to Vt\ NP}\ I({}_2Vt_3)\ I({}_3NP_8)$

Marginal Probability



- $I({}_0S_8) =$

  $\theta_{S \to NP\ VP}\ I({}_0NP_2)\ I({}_2VP_8)$

- $I({}_0NP_2) =$

  $\theta_{NP \to DT\ NN}\ I({}_0DT_1)\ I({}_1NN_2)$

- $I({}_2VP_8) =$

  $\theta_{VP \to VP\ PP}\ I({}_2VP_5)\ I({}_5PP_8)$

  $+\ \theta_{VP \to Vt\ NP}\ I({}_2Vt_3)\ I({}_3NP_8)$

…

Inside Weight

- Let us denote nodes/items by v, a_i
- Let us denote an edge/inference by $$\frac{a_1, \ldots, a_n}{v : \theta}$$
- θ is the weight of the rule underlying the inference
- $B(v)$ is the set of edges *incoming* to a node
  - i.e. *inferences* that prove the node

We call **Inside weight** the sum of weights of all derivations of a certain node

Inside recursion

$$I(v) = \begin{cases} 1 & \text{if } B(v) = \emptyset \\ \displaystyle\sum_{\frac{a_1, \ldots, a_n}{v:\theta} \in B(v)} \theta \times \prod_{i=1}^{n} I(a_i) & \text{otherwise} \end{cases}$$

For a PCFG, the **inside** of the GOAL node corresponds to the **marginal probability** of the sentence

$$P_{S|n}(x_1^n|n) = I({}_0\mathrm{S}_n) = \sum_{r_1^m \in \mathcal{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$

Maximum Probability

$$\max_{r_1^m \in \mathcal{G}(x_1^n)} P_{ST|NM}(x_1^n, r_1^m | n, m) = V({}_0\mathrm{S}_n)$$

$$= \max_{r_1^m \in \mathcal{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$

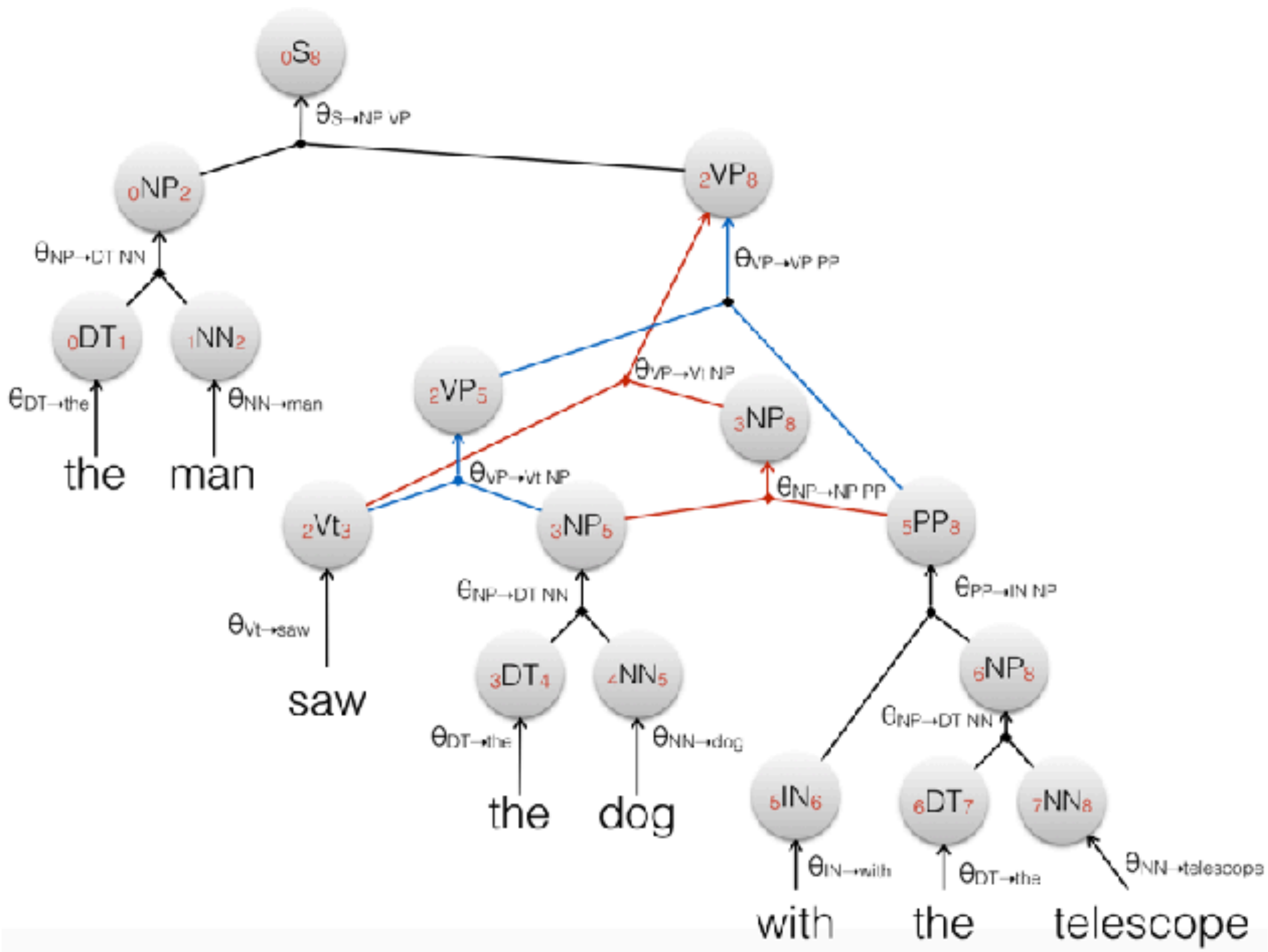
Maximum Probability

Let the goal item **stand** for the sentence. What's the probability of best tree under it V($_0$S$_8$)?

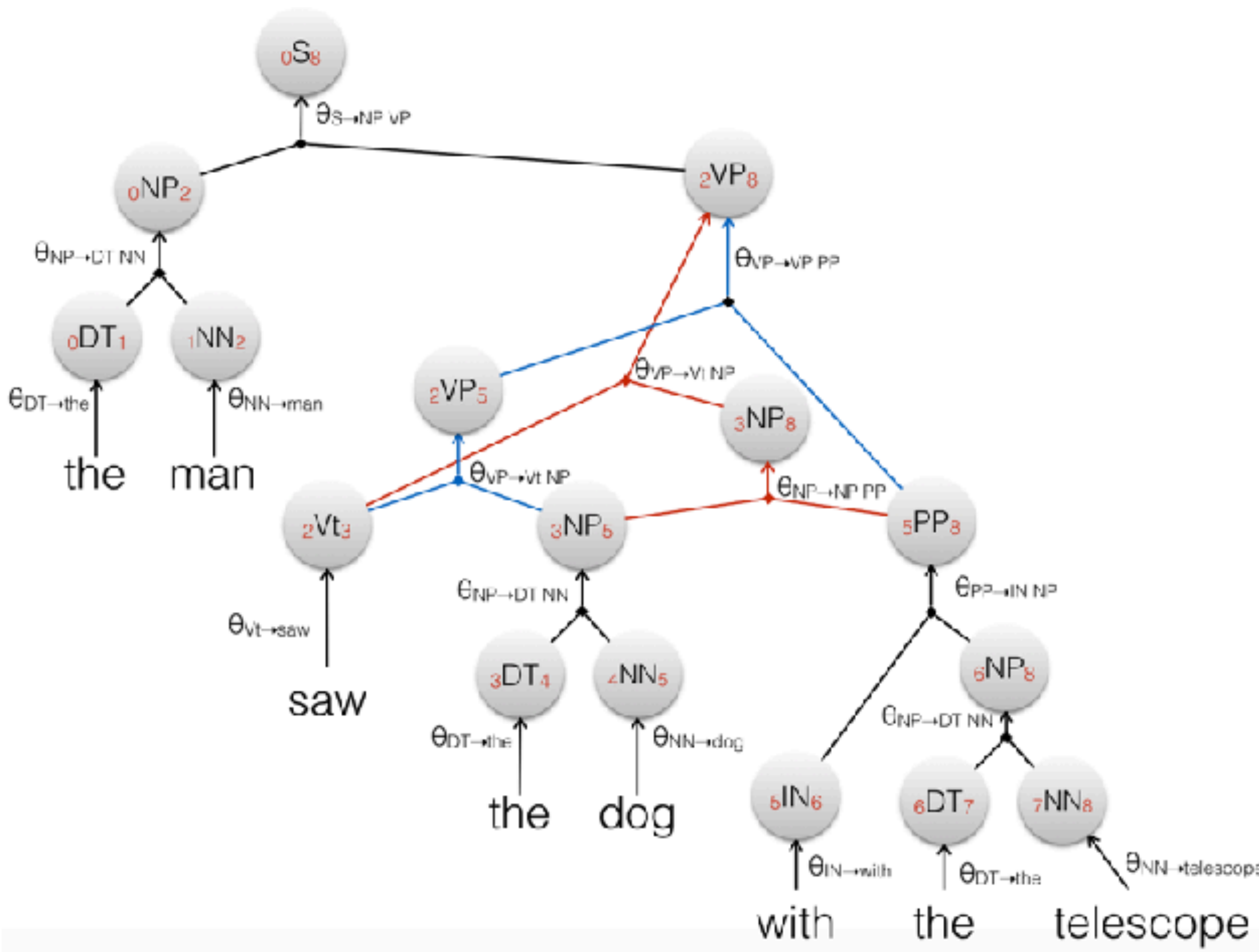
$$\max_{r_1^m \in \mathcal{G}(x_1^n)} P_{ST|NM}(x_1^n, r_1^m | n, m) = V({}_0\mathrm{S}_n)$$

$$= \max_{r_1^m \in \mathcal{G}(x_1^n)} \prod_{i=1}^{m} \theta_{v_i \to \beta_i}$$

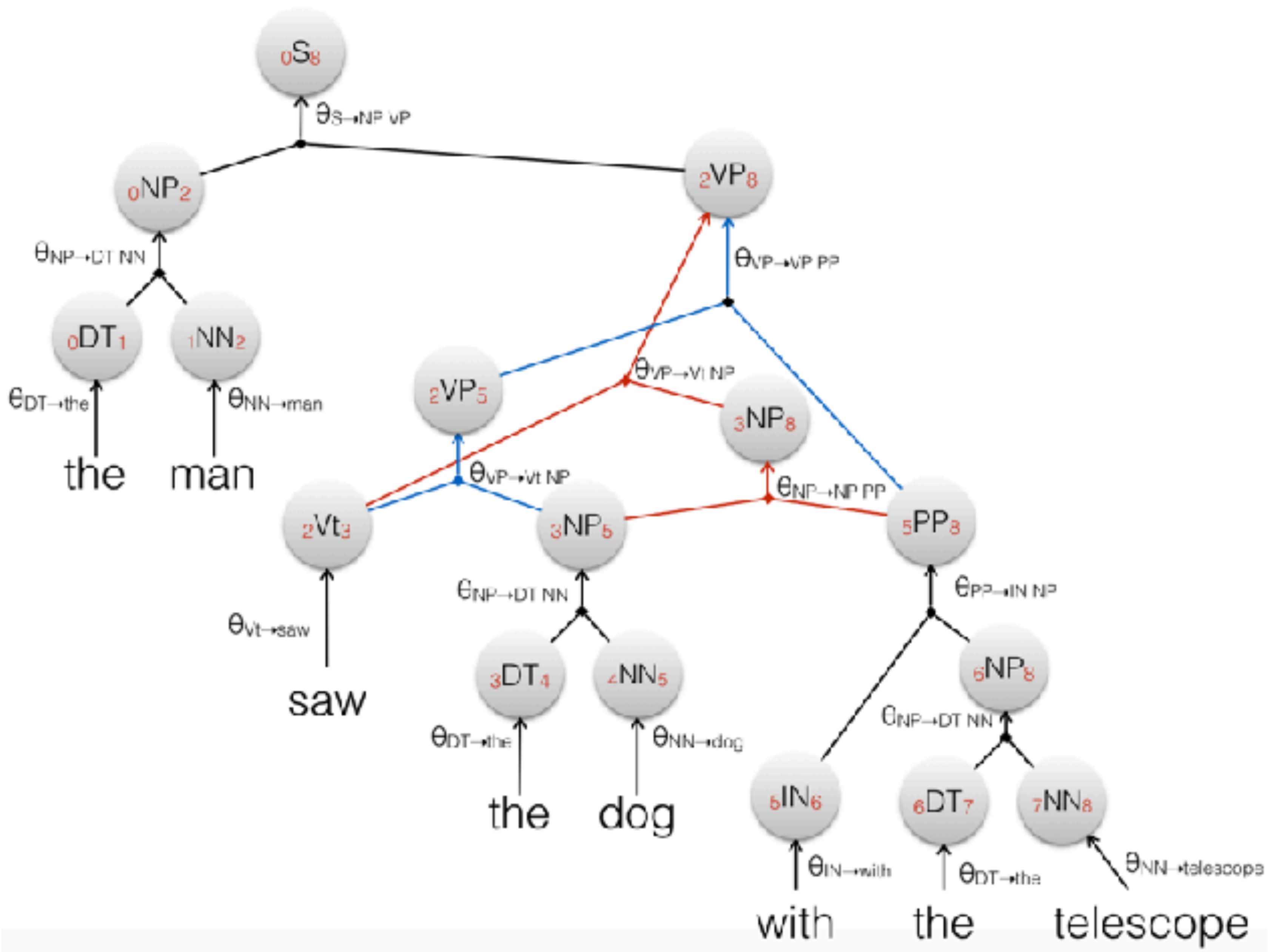
Maximum Probability



Maximum Probability

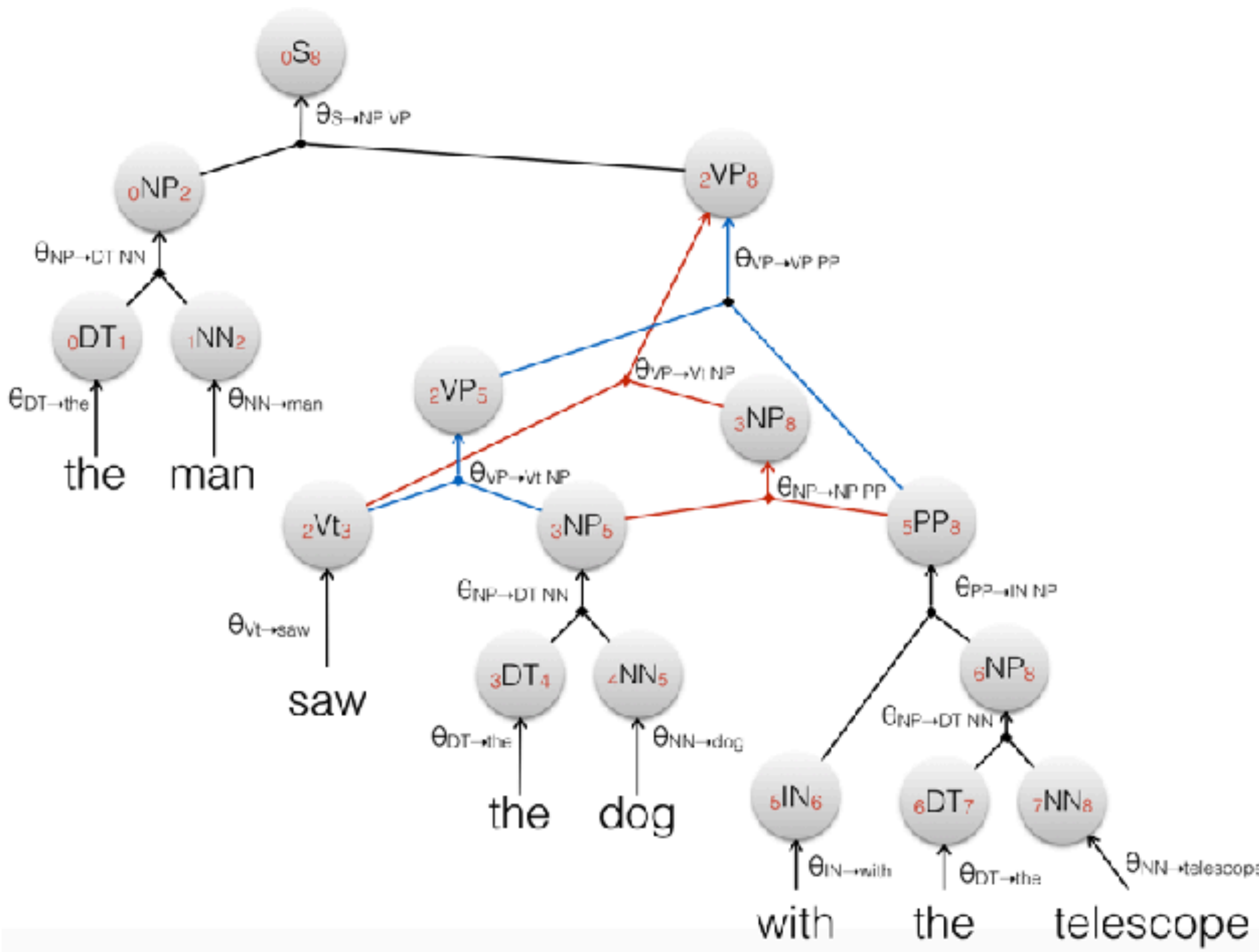


Maximum Probability



- $V({}_0S_8) =$

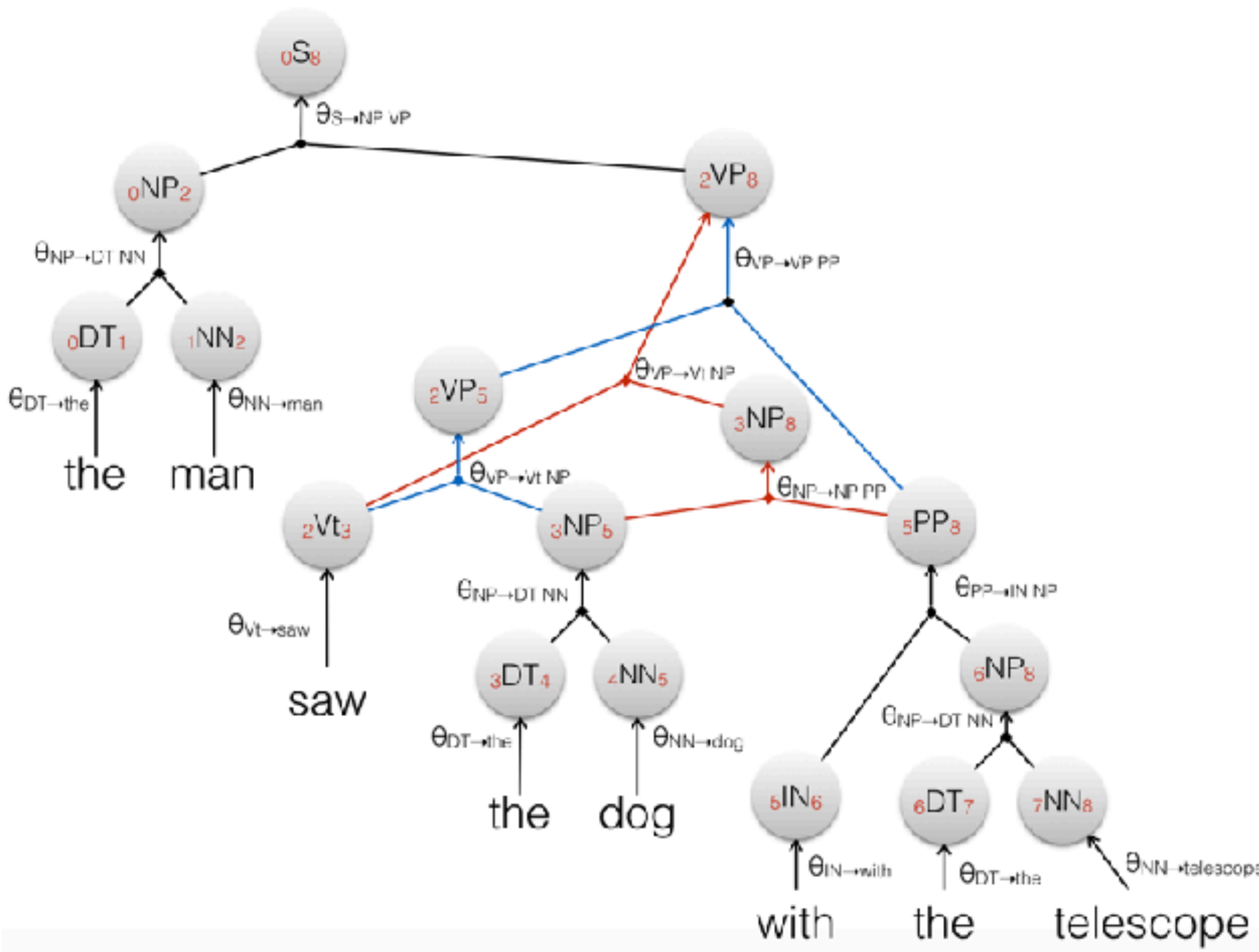
Maximum Probability



- $V({}_0S_8) =$

$\theta_{S\rightarrow NP\ VP}\ V({}_0NP_2)\ V({}_2VP_8)$

Maximum Probability

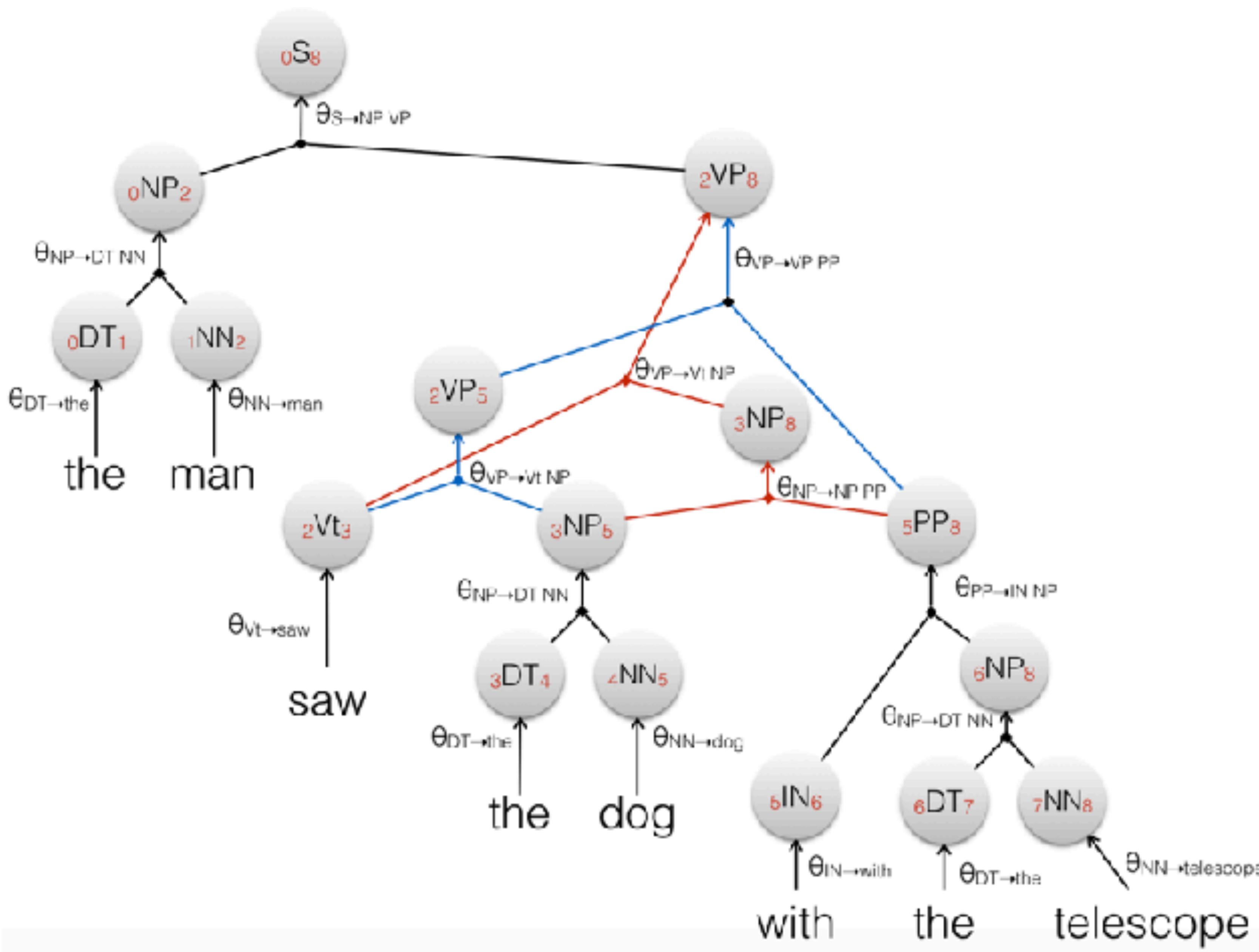


- $V({}_0S_8) =$

  $\theta_{S \to NP\ VP}\, V({}_0NP_2)\, V({}_2VP_8)$

- $V({}_0NP_2) =$

Maximum Probability



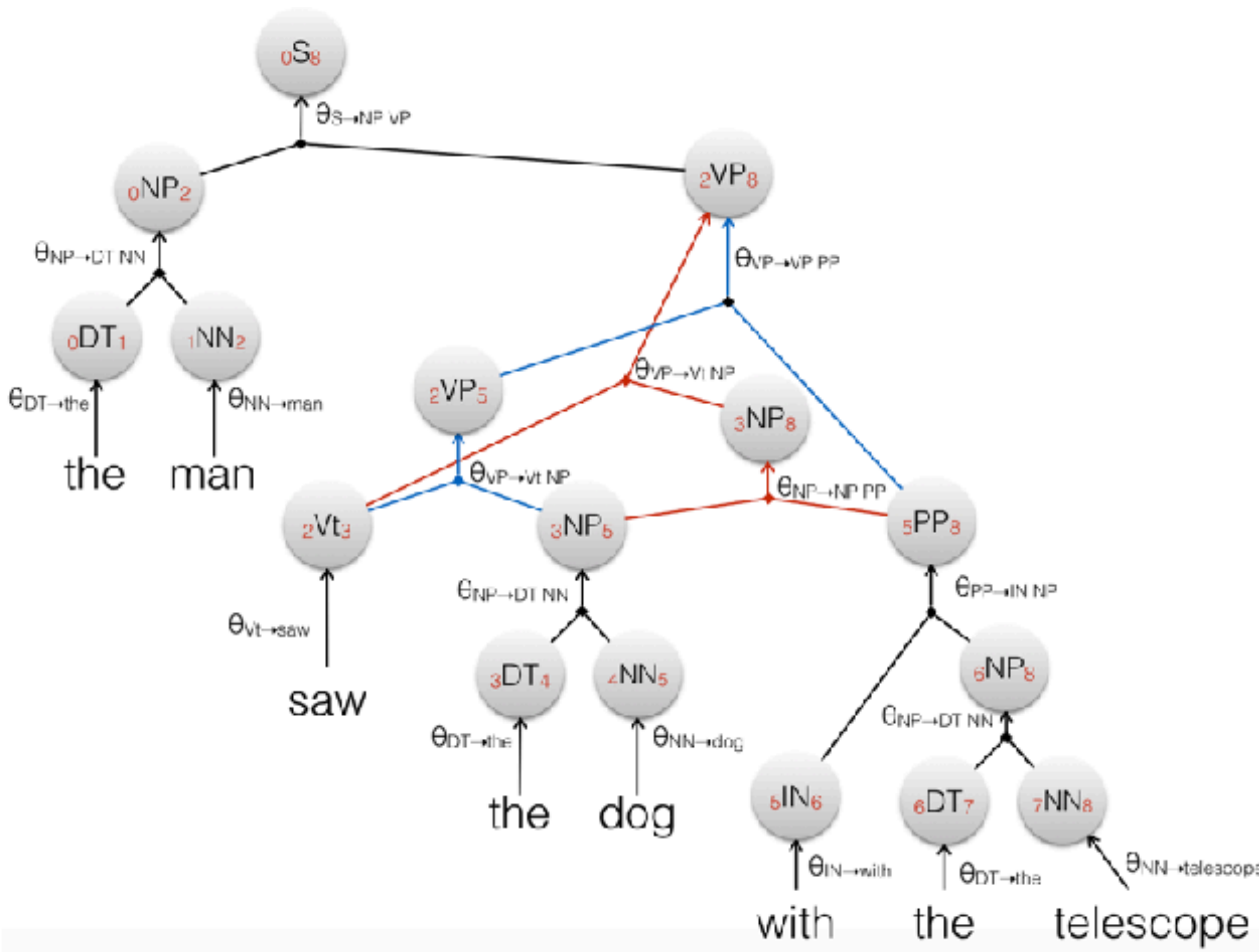
- $V({}_0S_8) =$

  $\theta_{S \rightarrow NP\ VP}\ V({}_0NP_2)\ V({}_2VP_8)$

- $V({}_0NP_2) =$

  $\theta_{NP \rightarrow DT\ NN}\ V({}_0DT_1)\ V({}_1NN_2)$

Maximum Probability



- $V({}_0S_8) =$

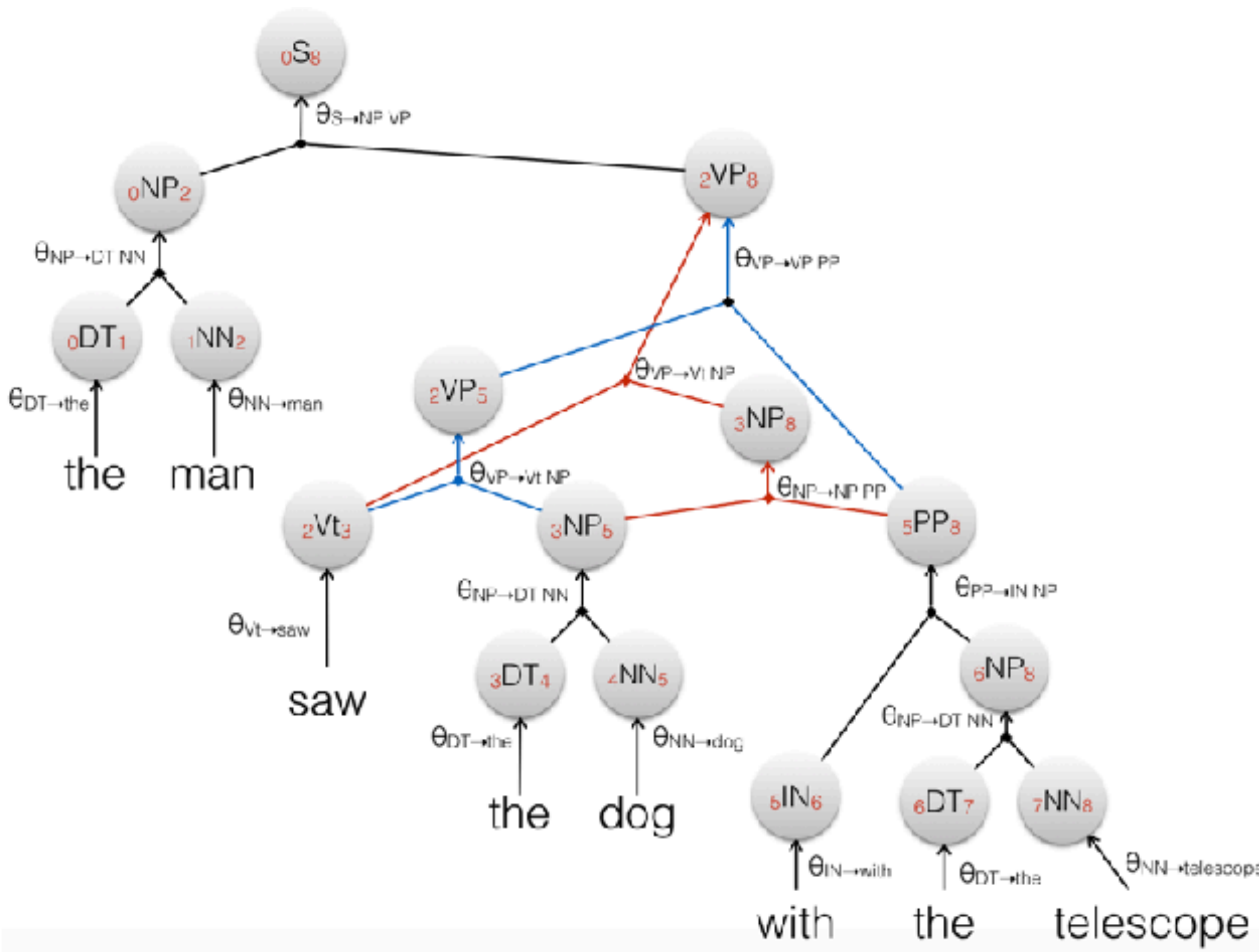
  $\theta_{S \rightarrow NP\ VP}\ V({}_0NP_2)\ V({}_2VP_8)$

- $V({}_0NP_2) =$

  $\theta_{NP \rightarrow DT\ NN}\ V({}_0DT_1)\ V({}_1NN_2)$

- $V({}_2VP_8) =$

Maximum Probability



- $V({}_0S_8) =$

  $\theta_{S \rightarrow NP\ VP}\ V({}_0NP_2)\ V({}_2VP_8)$

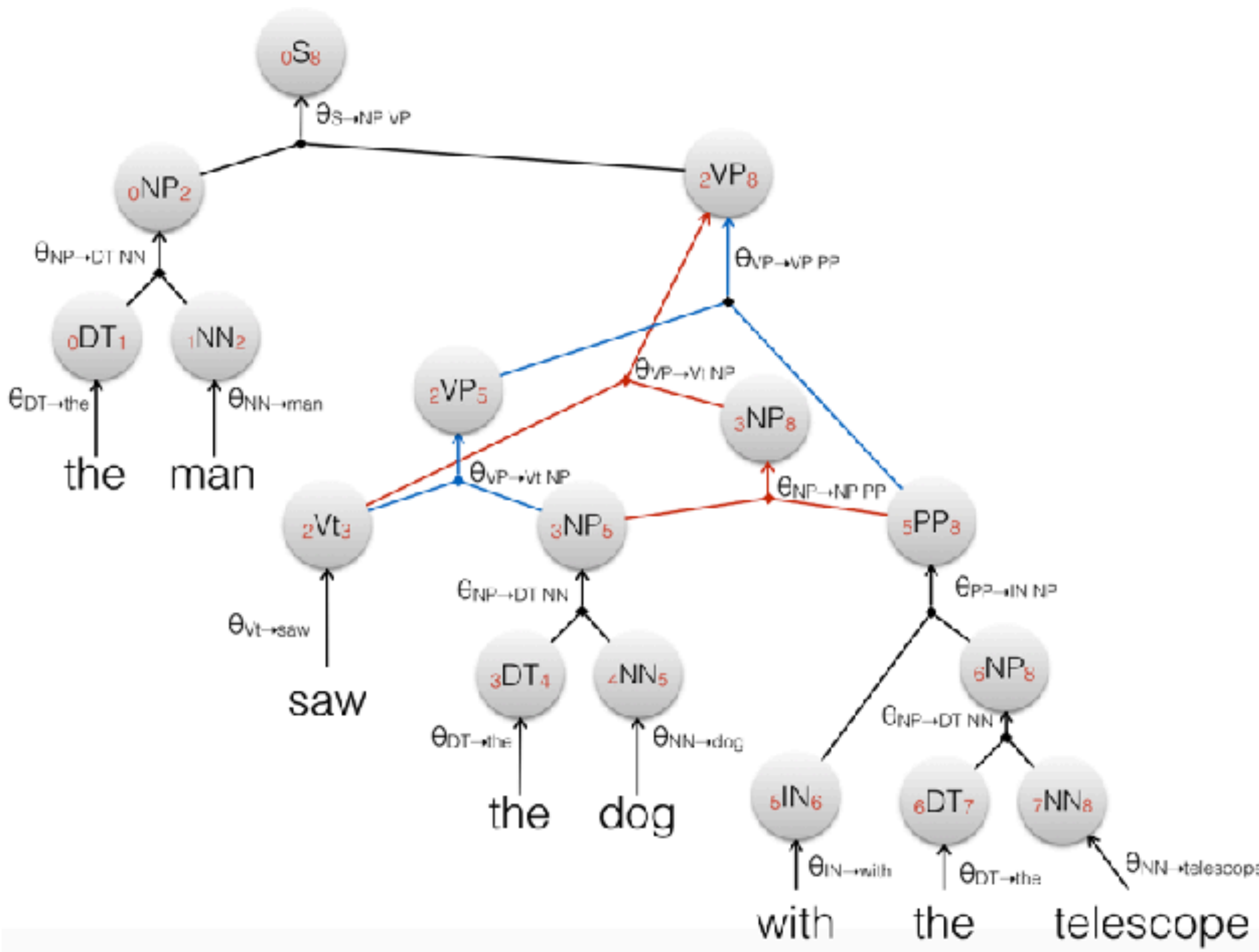
- $V({}_0NP_2) =$

  $\theta_{NP \rightarrow DT\ NN}\ V({}_0DT_1)\ V({}_1NN_2)$

- $V({}_2VP_8) =$

  max {

Maximum Probability



- $V({}_{0}S_{8})$ =

  $\theta_{S \to NP\ VP}\ V({}_{0}NP_{2})\ V({}_{2}VP_{8})$

- $V({}_{0}NP_{2})$ =

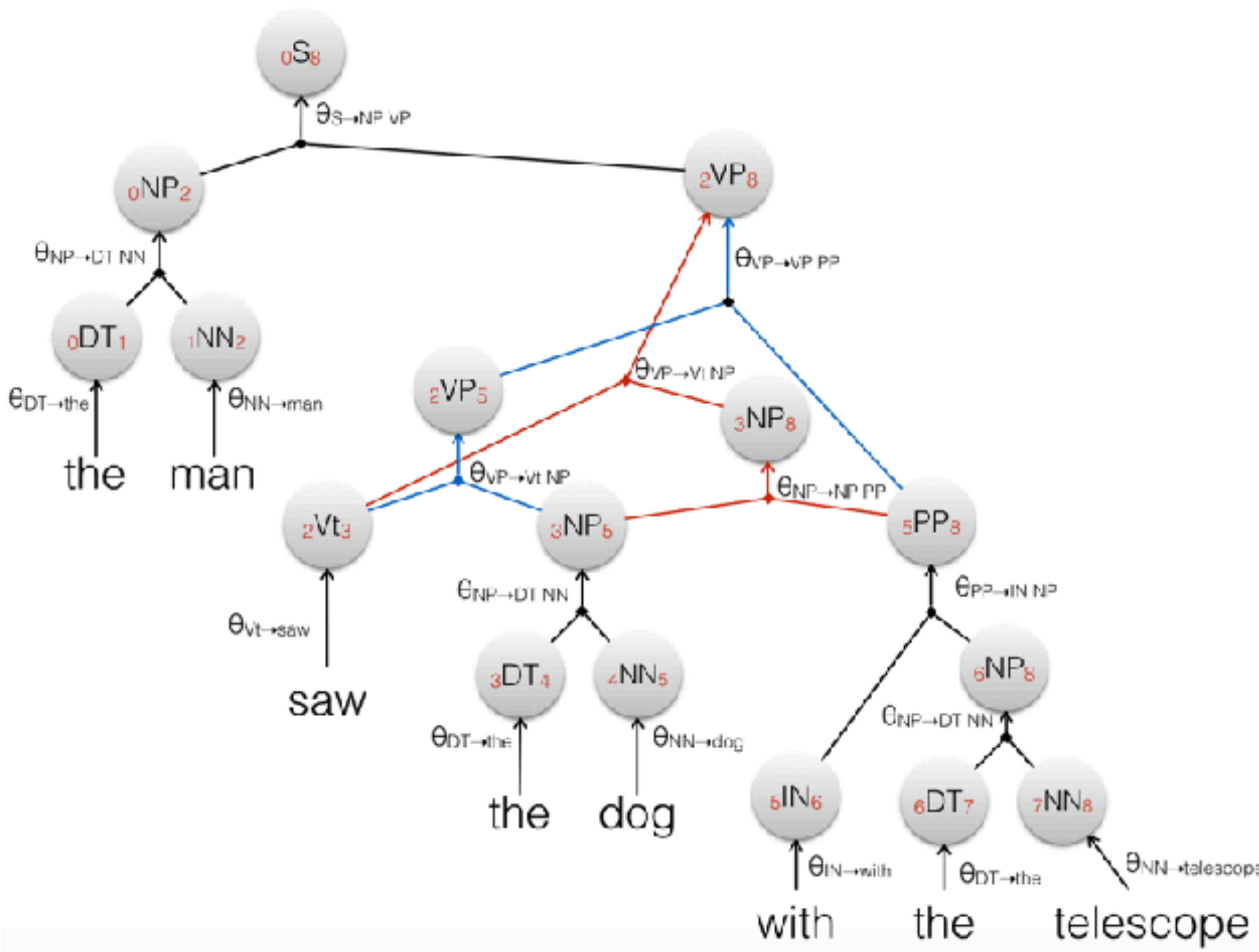
  $\theta_{NP \to DT\ NN}\ V({}_{0}DT_{1})\ V({}_{1}NN_{2})$

- $V({}_{2}VP_{8})$ =

  max {

  $\theta_{VP \to VP\ PP}\ V({}_{2}VP_{5})\ V({}_{5}PP_{8})$**,**

Maximum Probability



- $V({}_{0}S_{8})$ =

  $\theta_{S\rightarrow NP\ VP}\ V({}_{0}NP_{2})\ V({}_{2}VP_{8})$

- $V({}_{0}NP_{2})$ =

  $\theta_{NP\rightarrow DT\ NN}\ V({}_{0}DT_{1})\ V({}_{1}NN_{2})$

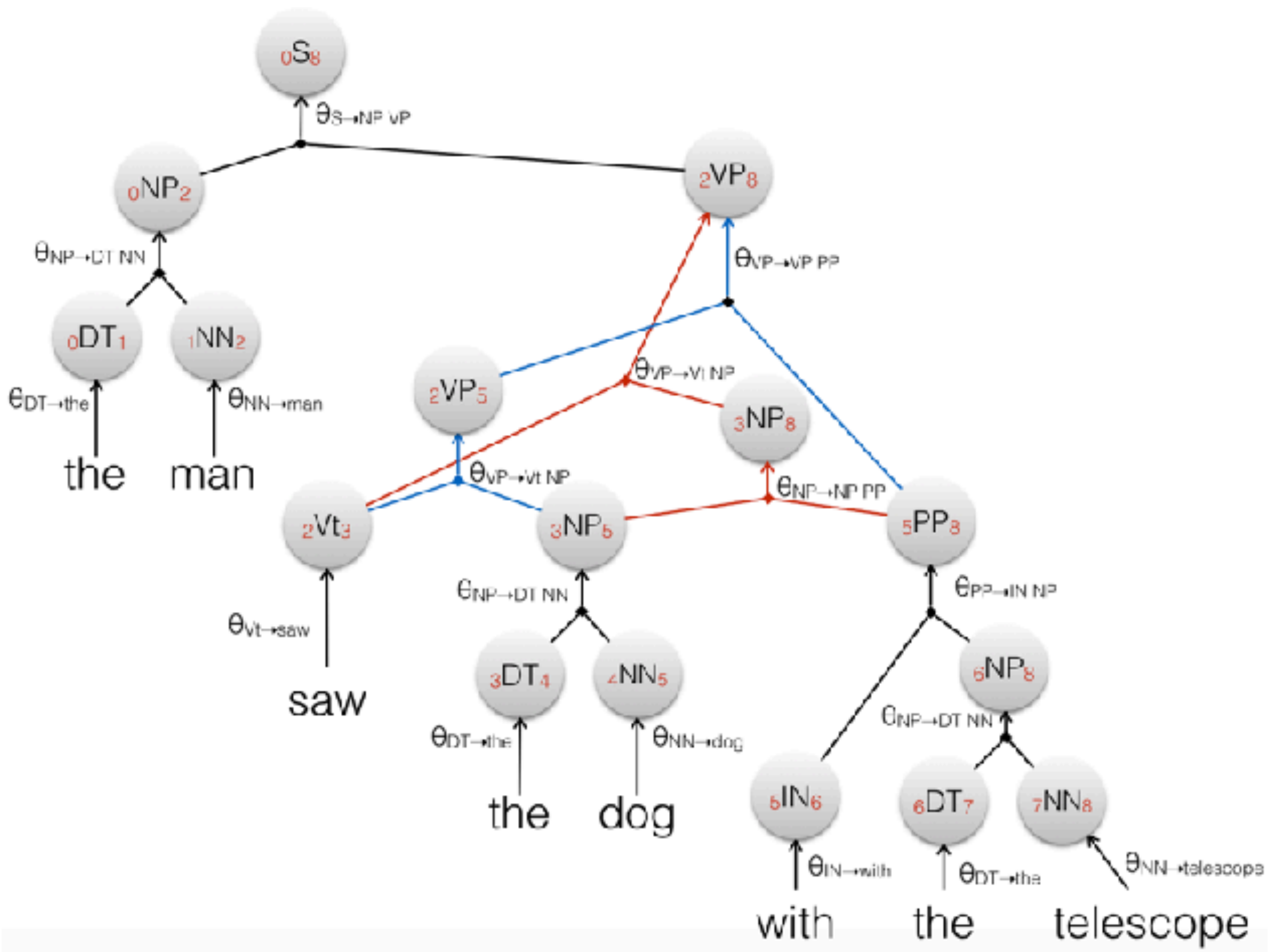
- $V({}_{2}VP_{8})$ =

  max {

  $\theta_{VP\rightarrow VP\ PP}\ V({}_{2}VP_{5})\ V({}_{5}PP_{8})$**,**

  $\theta_{VP\rightarrow Vt\ NP}\ V({}_{2}Vt_{3})\ V({}_{3}NP_{8})$ **}**

Maximum Probability



…

- $V({}_0S_8) =$

  $\theta_{S \rightarrow NP\ VP}\ V({}_0NP_2)\ V({}_2VP_8)$

- $V({}_0NP_2) =$

  $\theta_{NP \rightarrow DT\ NN}\ V({}_0DT_1)\ V({}_1NN_2)$

- $V({}_2VP_8) =$

  max {

  $\theta_{VP \rightarrow VP\ PP}\ V({}_2VP_5)\ V({}_5PP_8)$**,**

  $\theta_{VP \rightarrow Vt\ NP}\ V({}_2Vt_3)\ V({}_3NP_8)$ **}**

Viterbi

$$I_{\max}(v) = \begin{cases} 1 & \text{if } B(v) = \emptyset \\ \max\limits_{\frac{a_1,\ldots,a_n}{v:\theta} \in B(v)} \theta \times \prod_{i=1}^{n} I(a_i) & \text{otherwise} \end{cases}$$

For a PCFG, the **inside algorithm** computed with **max** instead of sum corresponds to the **probability of the best derivation** of the sentence

$$V({}_0\mathrm{S}_n) = I_{\max}({}_0\mathrm{S}_n) = \max_{r_1^m \in \mathcal{G}(x_1^n)} P_{ST|NM}(x_1^n, r_1^m | n, m)$$

Many in One

The inside recursion is very general

- It includes other dynamic programs
  - e.g. Viterbi

Semirings

- Generalise sum and products

Semirings

Marginal (probability)

$$a \oplus b = a + b$$
$$a \otimes b = a \times b$$
$$\bar{1} = 1$$
$$\bar{0} = 0$$

Log-marginal (probability)

$$a \oplus b = \log(\exp a + \exp b)$$
$$a \otimes b = a + b$$
$$\bar{1} = 0$$
$$\bar{0} = -\infty$$

Viterbi (max-probability)

$$a \oplus b = \max(a, b)$$
$$a \otimes b = a \times b$$
$$\bar{1} = 1$$
$$\bar{0} = 0$$

Log-viterbi (max-log-prob)

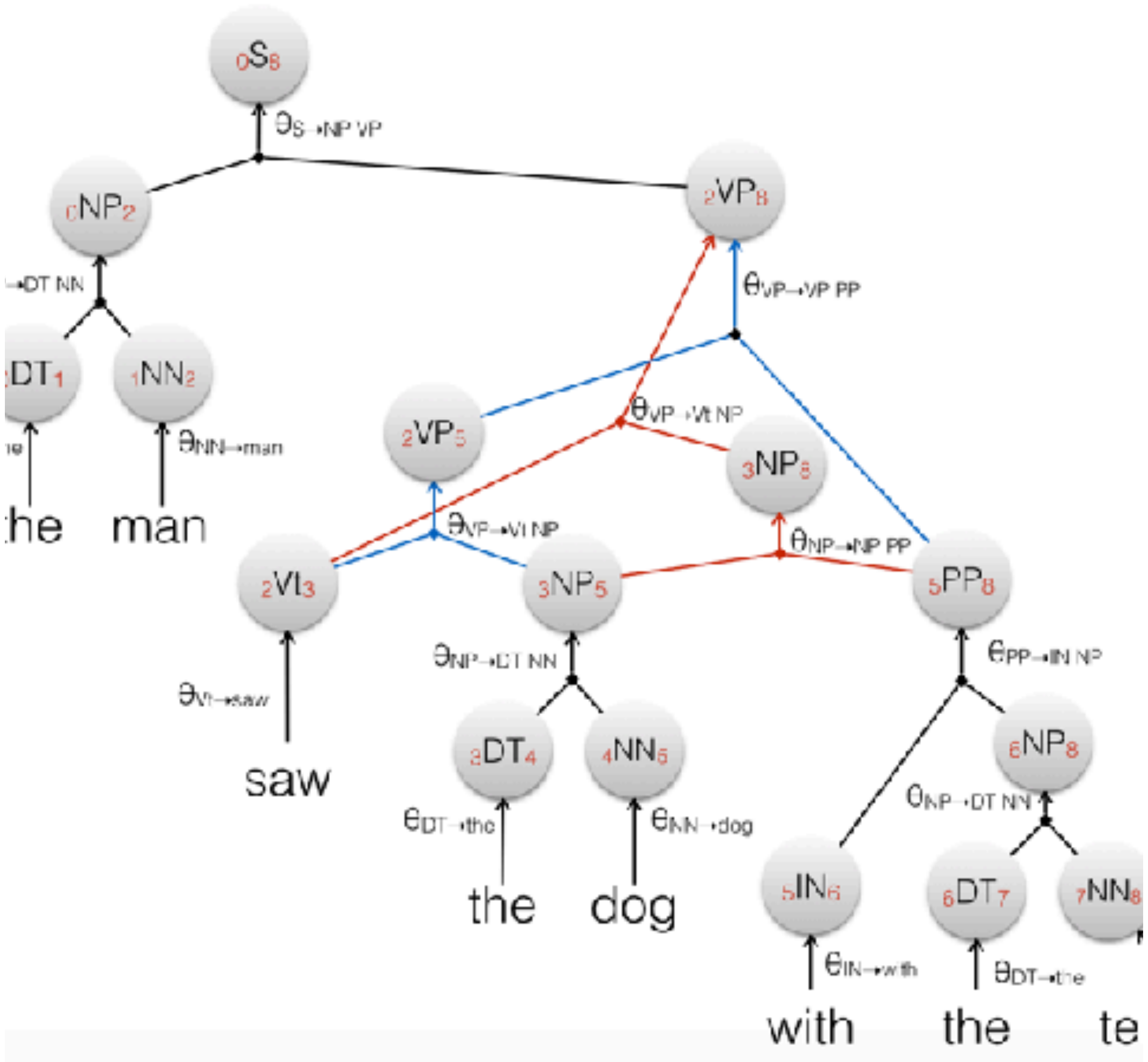
$$a \oplus b = \max(a, b)$$
$$a \otimes b = a + b$$
$$\bar{1} = 0$$
$$\bar{0} = -\infty$$

Inside semiring

With generalised operations

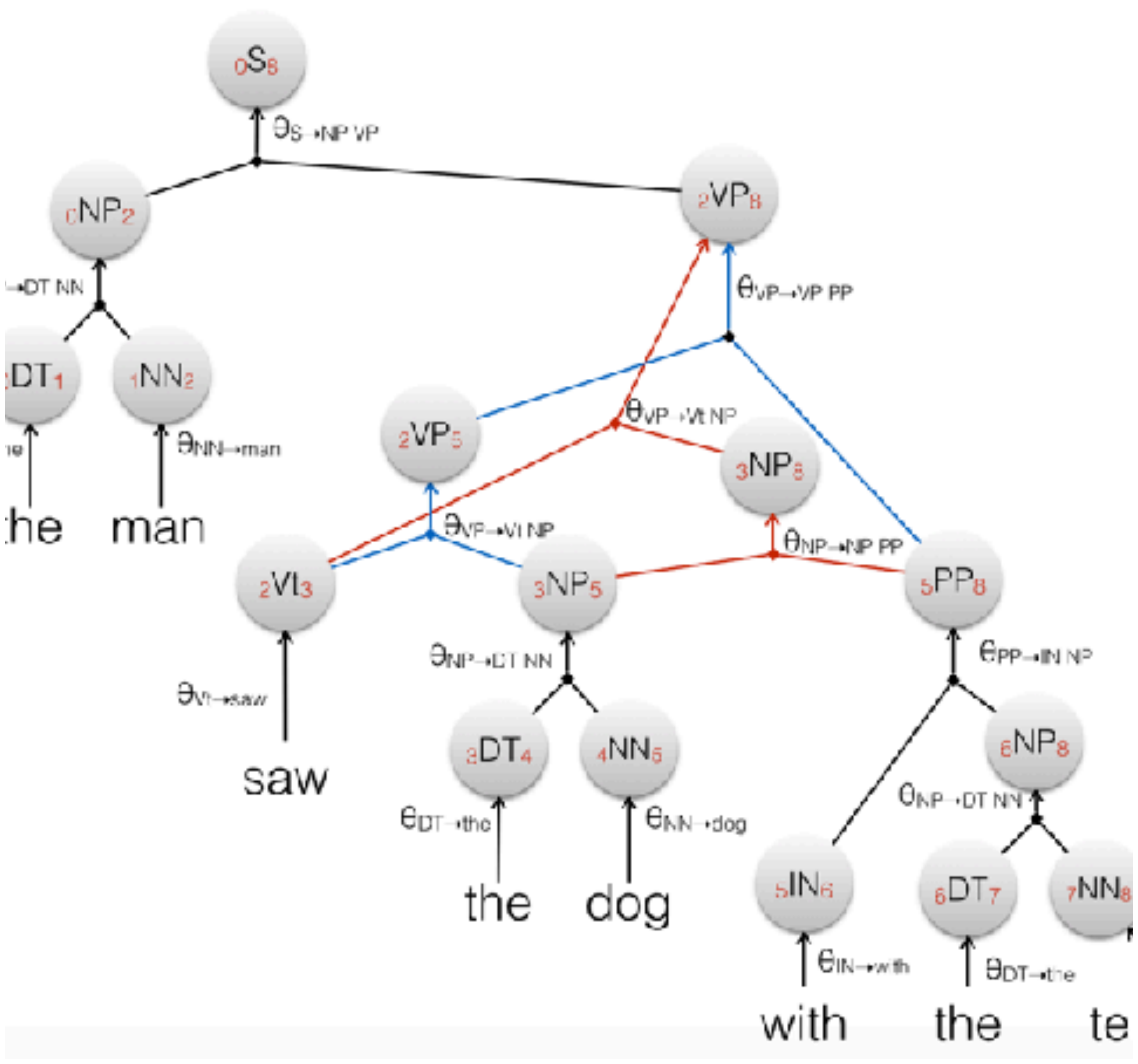
$$I(v) = \begin{cases} \bar{1} & \text{if } B(v) = \emptyset \\ \displaystyle\bigoplus_{\frac{a_1,\ldots,a_n}{v:\theta} \in B(v)} \theta \otimes \bigotimes_{i=1}^{n} I(a_i) & \text{otherwise} \end{cases}$$

Inside example



Inside example

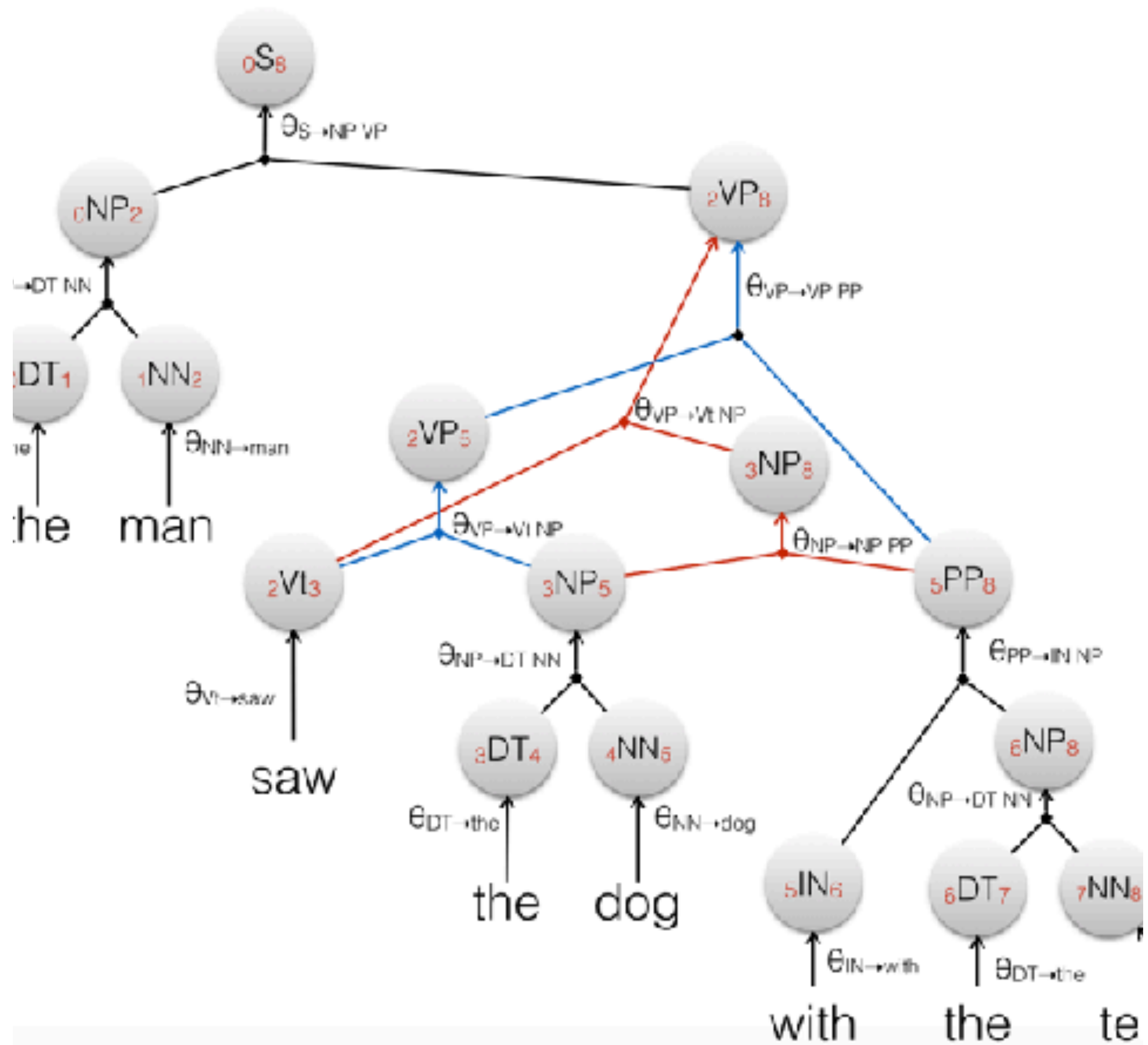
- $I({}_0S_8) =$



Inside example

- $I({}_0S_8) =$

  $\theta_{S\rightarrow NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

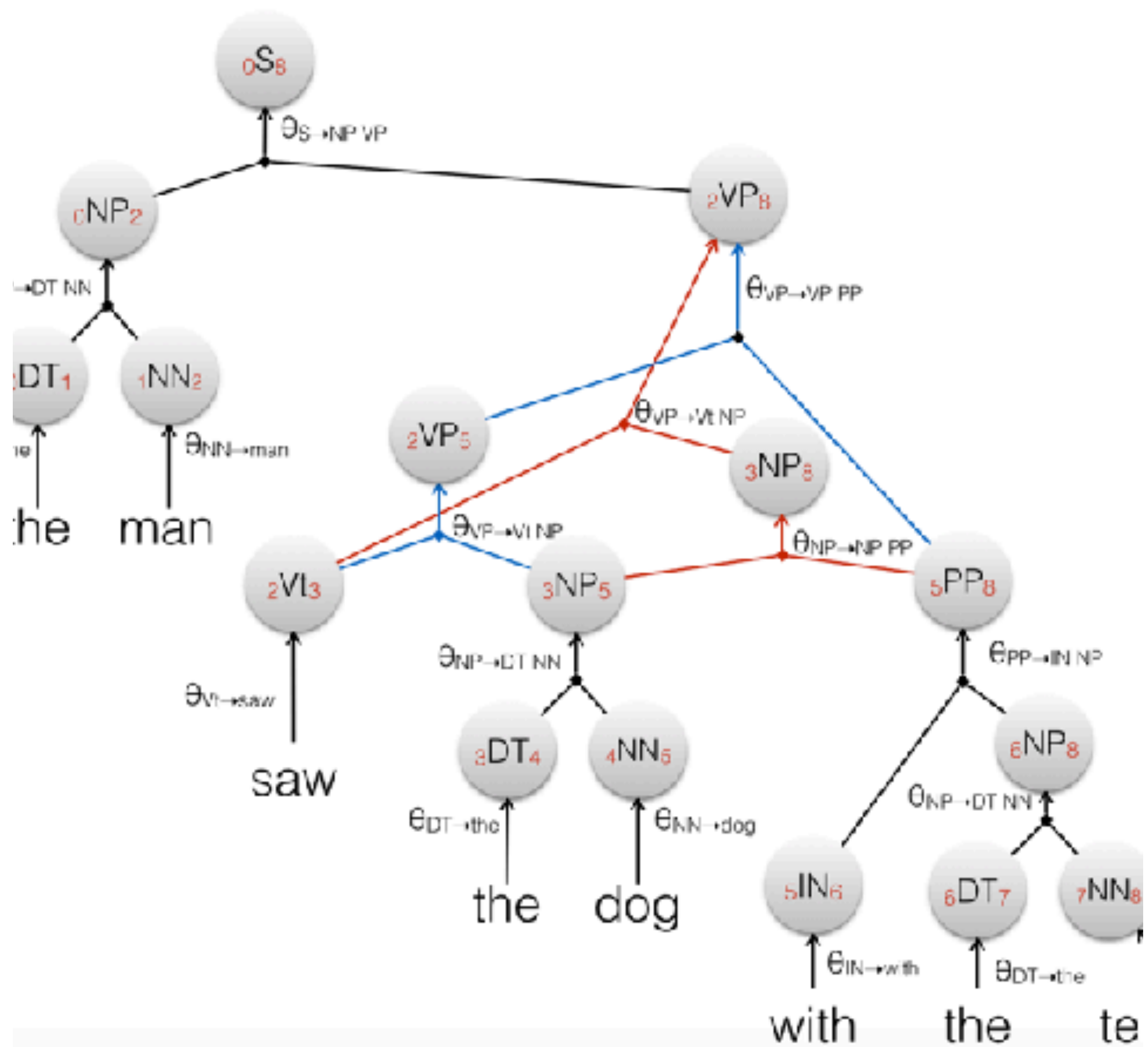


Inside example

- $I({}_0S_8) =$

  $\theta_{S\to NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

- $I({}_0NP_2) =$



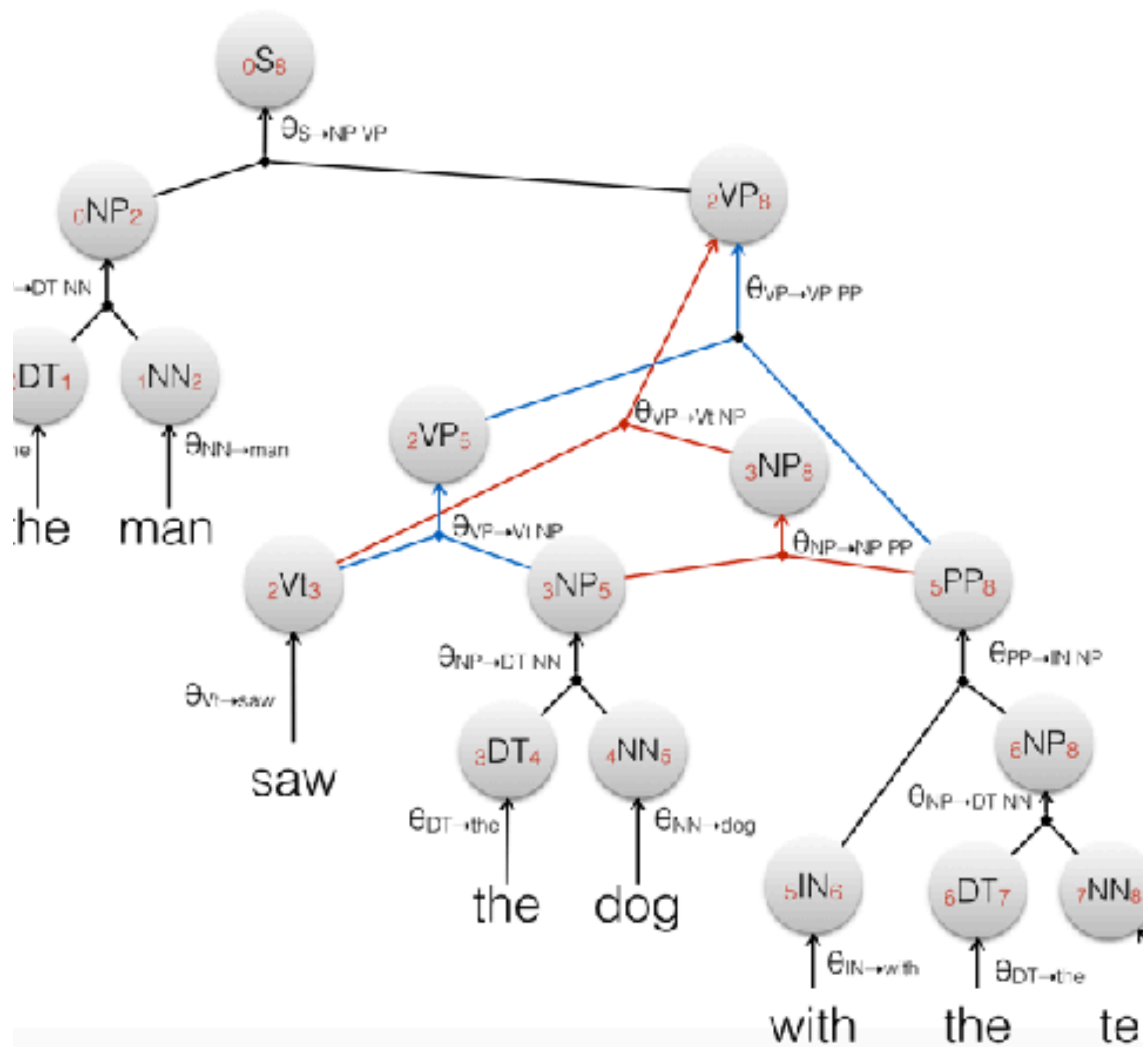
Inside example

- $I({}_0S_8)$ =

  $\theta_{S\to NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

- $I({}_0NP_2)$ =

  $\theta_{NP\to DT\ NN}$
  $\otimes I({}_0DT_1) \otimes I({}_1NN_2)$



Inside example

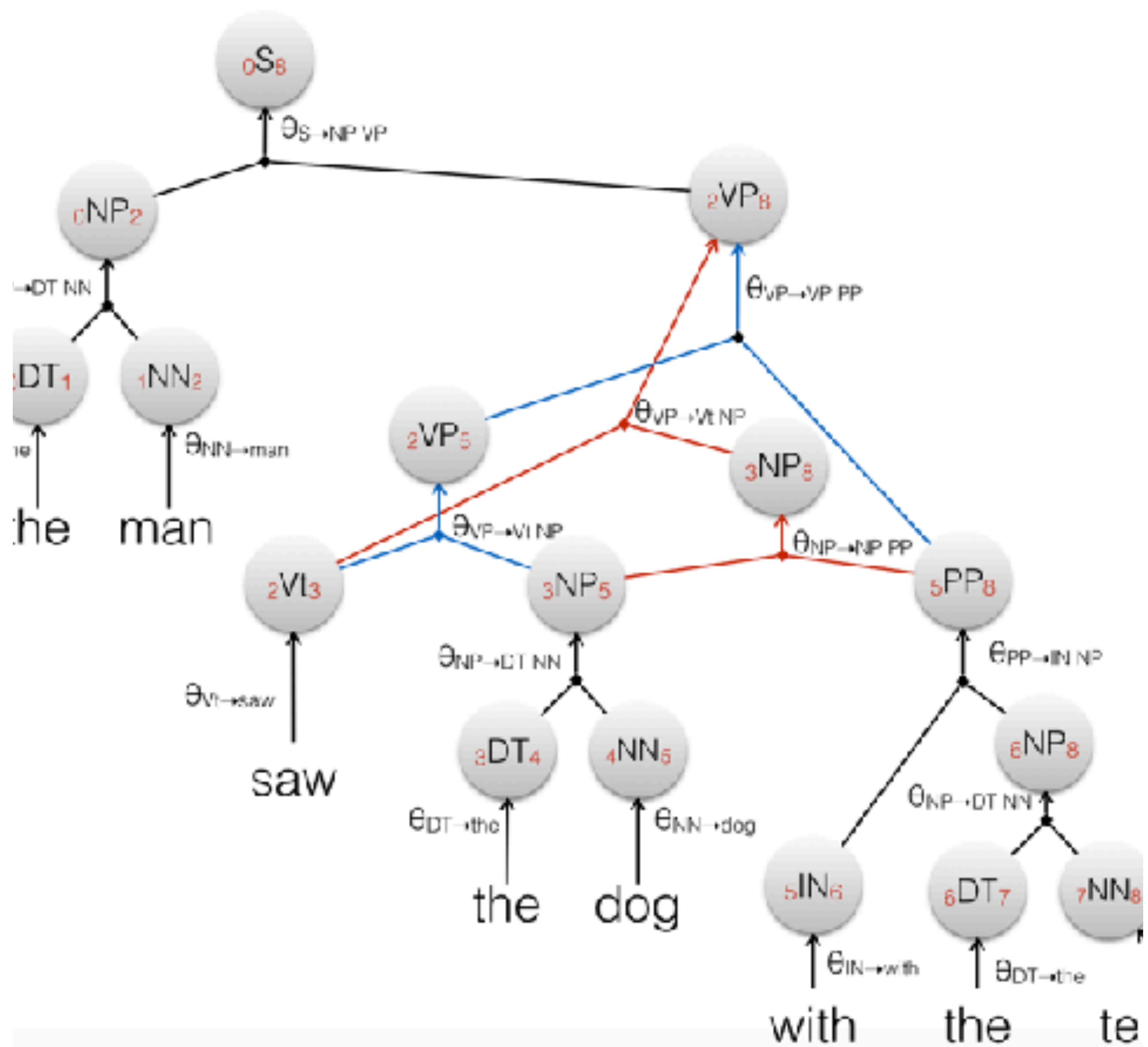
- $I({}_0S_8) =$

  $\theta_{S \rightarrow NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

- $I({}_0NP_2) =$

  $\theta_{NP \rightarrow DT\ NN} \otimes I({}_0DT_1) \otimes I({}_1NN_2)$

- $I({}_0DT_1) =$



Inside example

- $I({}_0S_8) =$

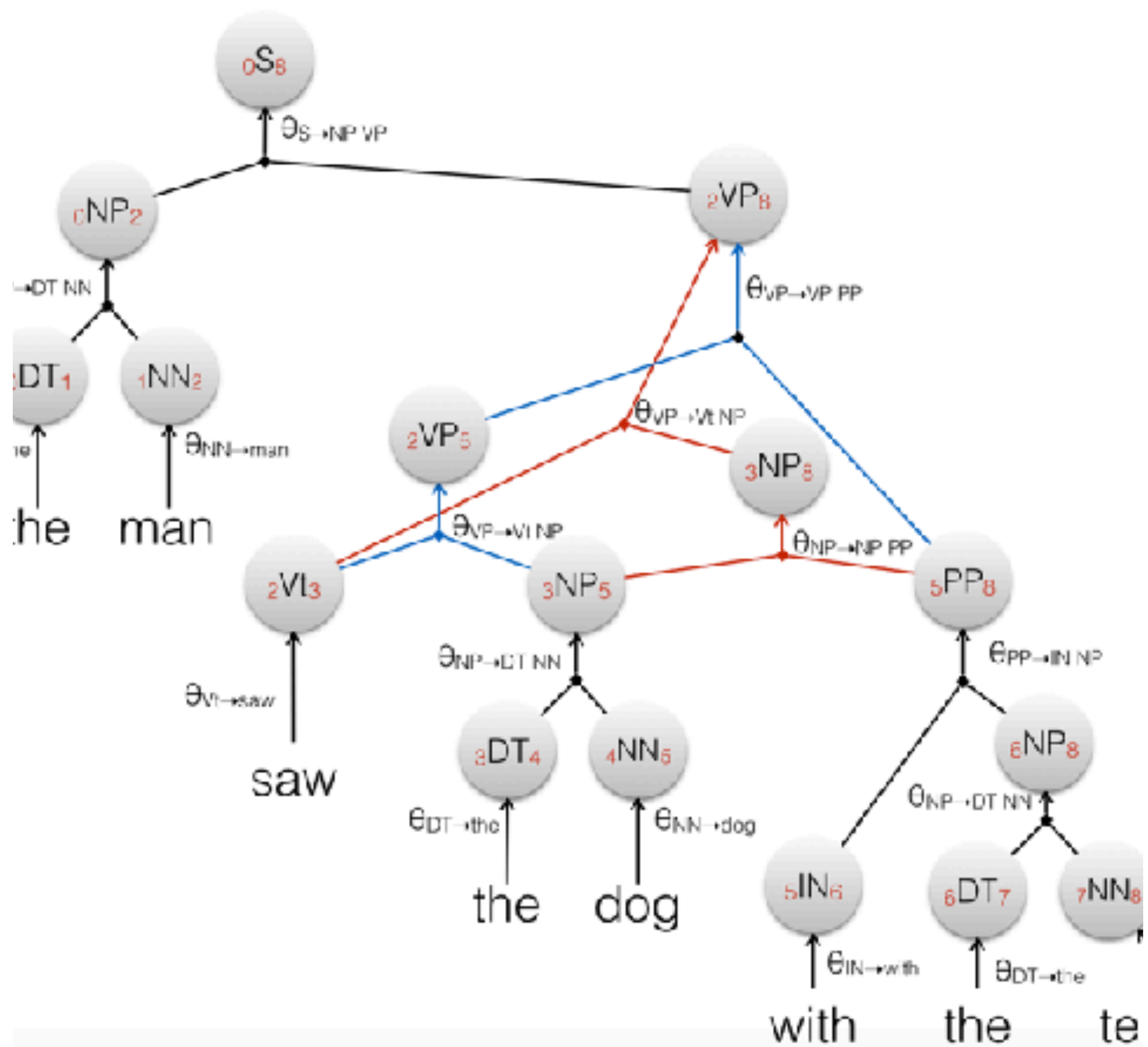
  $\theta_{S\rightarrow NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

- $I({}_0NP_2) =$

  $\theta_{NP\rightarrow DT\ NN} \otimes I({}_0DT_1) \otimes I({}_1NN_2)$

- $I({}_0DT_1) =$

  $\theta_{DT\rightarrow the} \otimes I(the)$



Inside example

- $I({}_0S_8) =$

  $\theta_{S \to NP\ VP} \otimes I({}_0NP_2) \otimes I({}_2VP_8)$

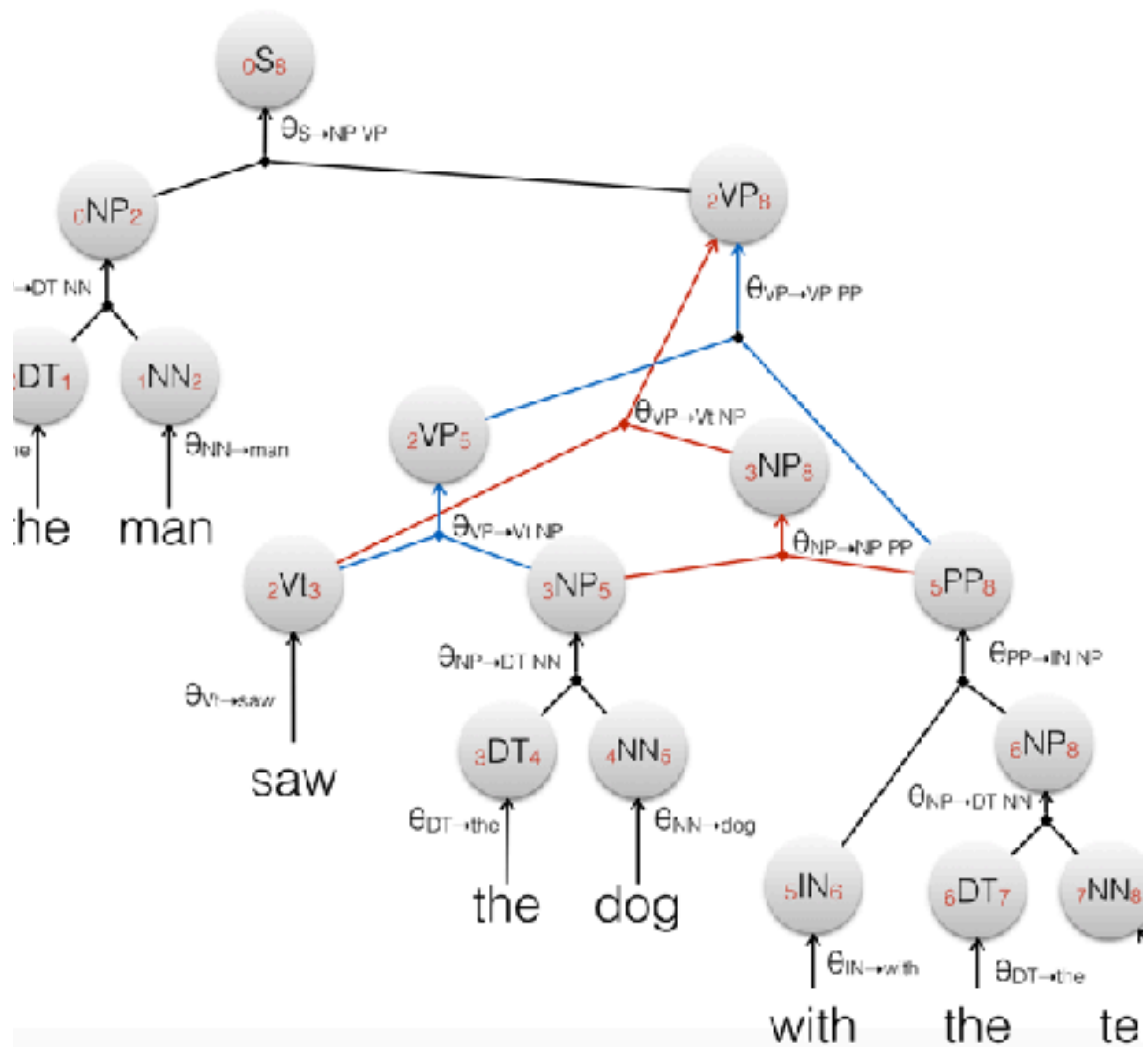
- $I({}_0NP_2) =$

  $\theta_{NP \to DT\ NN}$
  $\otimes I({}_0DT_1) \otimes I({}_1NN_2)$

- $I({}_0DT_1) =$

  $\theta_{DT \to the} \otimes I(the)$

- $I(the) = 1$



Complexity

$$X \to \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \to v(\alpha)\,\beta$$

$$v(\alpha) \to \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

Complexity

Every CFG can be binarised (max arity = 2)

$$X \to \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \to v(\alpha)\,\beta$$

$$v(\alpha) \to \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

Complexity

Every CFG can be binarised (max arity = 2)

- Just pre-process the grammar rules

$$X \rightarrow \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \rightarrow v(\alpha)\,\beta$$

$$v(\alpha) \rightarrow \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

Complexity

Every CFG can be binarised (max arity = 2)

- Just pre-process the grammar rules

$$X \rightarrow \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \rightarrow v(\alpha)\,\beta$$

$$v(\alpha) \rightarrow \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

Complexity

Every CFG can be binarised (max arity = 2)

- Just pre-process the grammar rules

$$X \to \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$
$$\Rightarrow X \to v(\alpha)\, \beta$$
$$v(\alpha) \to \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

Complexity

Every CFG can be binarised (max arity = 2)

- Just pre-process the grammar rules

$$X \to \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \to v(\alpha)\,\beta$$

$$v(\alpha) \to \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

- In total we get up to 3 indices ranging from 0 .. n

Complexity

Every CFG can be binarised (max arity = 2)

- Just pre-process the grammar rules

$$X \rightarrow \alpha \bullet \beta \quad \text{where } |\alpha| > 1 \text{ and } |\beta| = 1$$

$$\Rightarrow X \rightarrow v(\alpha)\,\beta$$

$$v(\alpha) \rightarrow \alpha \quad \text{where } v(\alpha) \text{ turns } \alpha \text{ in a nonterminal}$$

- In total we get up to 3 indices ranging from 0 .. n
- O(n^3) annotated rules

Bibliography

- Hopcroft, John E. and Ullman, Jeffrey D. 1979. Introduction To Automata Theory, Languages, And Computation.

- Shieber, S. and Schabes, Y. and Pereira, F. 1995. Principles and implementation of deductive parsing. In *Journal of Logic Programming*

- Bar-Hillel, Y. and Perles, M. and Shamir, E. 1961. On formal properties of simple phrase structure grammars.

- Billot, S. and Lang, B. 1989. The Structure of Shared Forests in Ambiguous ParsingThe Structure of Shared Forests in Ambiguous Parsing. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*.