知能システム論
自然言語処理(3)
構文解析

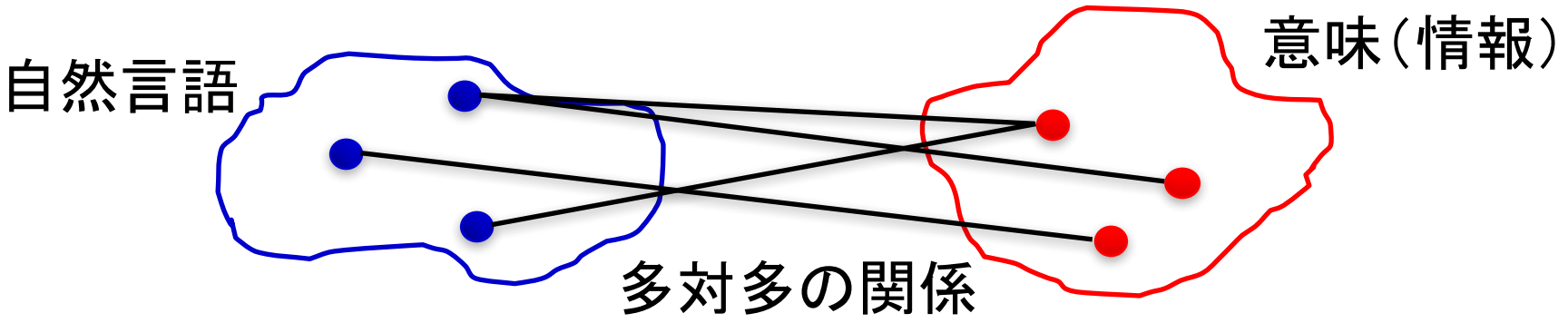
宮尾 祐介

yusuke@is.s.u-tokyo.ac.jp

https://mynlp.github.io/

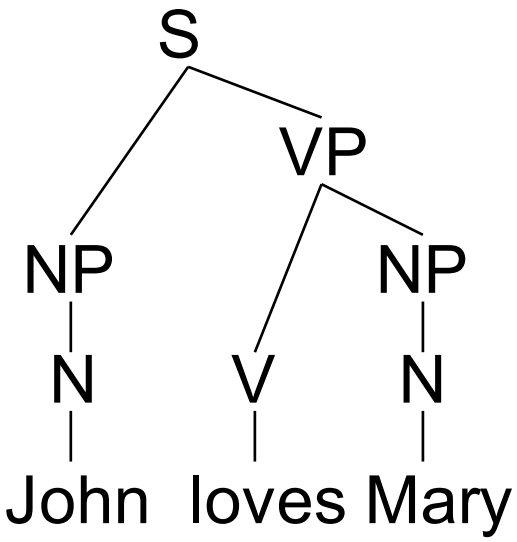
なぜ自然言語処理が必要？

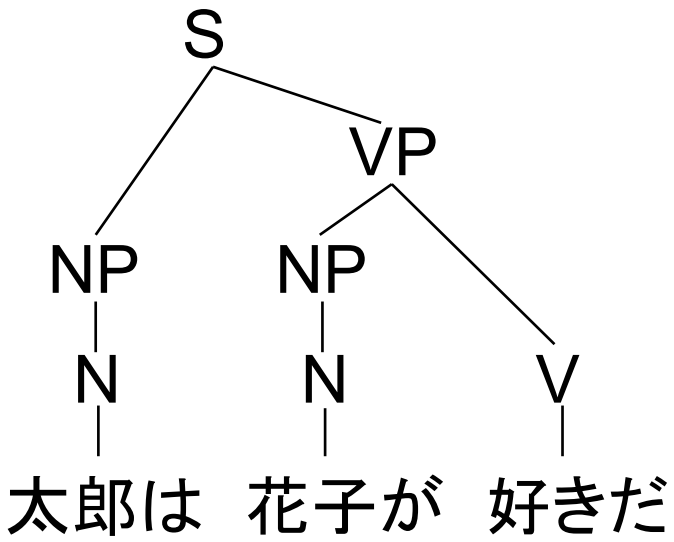
- 自然言語データをそのまま文字列として処理したらよいのでは？
- ポイント：文字列の近さと意味の近さが異なる
  - 友達からケーキをもらった
  - 友達からケヤキをもらった
  - 友達からモンブランをもらった
  - 友達がケーキをもらった
- 文字列そのままではなく、何らかの形で「意味」をとらえる必要がある



構文解析

- 文の構造を計算する技術
- 入力：単語列（＋品詞）、出力：構造木

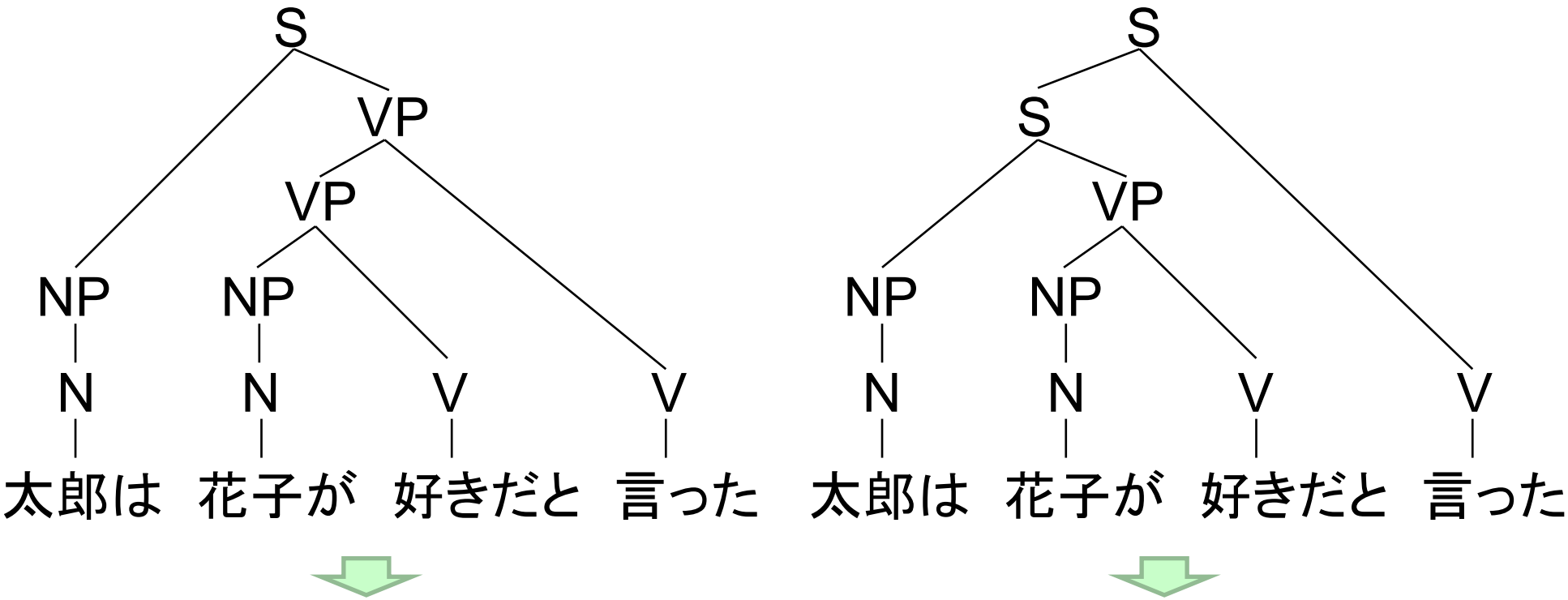




S: 文(sentence)
NP: 名詞句(noun phrase)
VP: 動詞句(verb phrase)
N: 名詞 (noun)
V: 動詞 (verb)

なぜ構文解析が必要？

- 意味を計算する第一歩
  - 文中で単語がどのように組み合わさっているのか（→ 構文木に沿って意味を計算する）



「言った」の主語は「太郎」

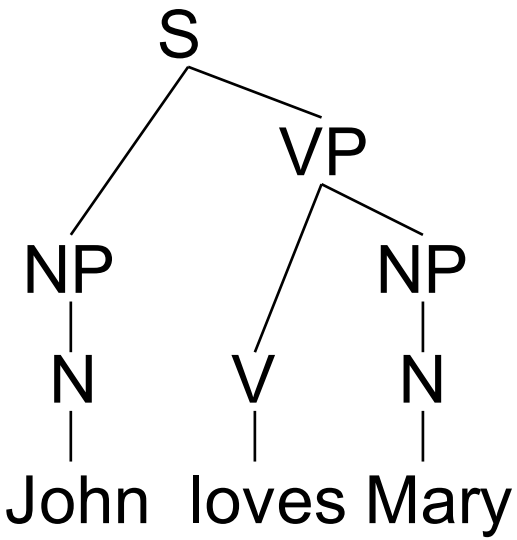
「言った」の主語は「太郎」ではない

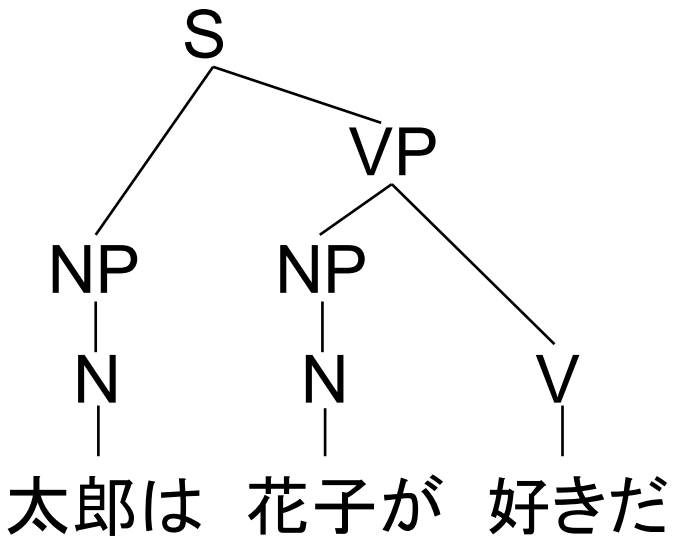
構文解析アルゴリズムは重要

- 構文解析の各種アルゴリズムは「自然言語の構文解析」だけに使われるわけではない
  - 機械翻訳、画像解析、自動証明、遺伝子解析、etc…
- 理論的にも重要
  - 木構造＝動的計画法が適用できる重要なデータ構造
  - 系列ラベリング（HMM, CRF etc.）よりもう一段複雑な動的計画法が必要
  - 構文解析アルゴリズムが理解できれば、世の中の多くの動的計画法は理解できる

構文木

- 単語のまとまり(句)を木構造で表す
- 句構造とも言う

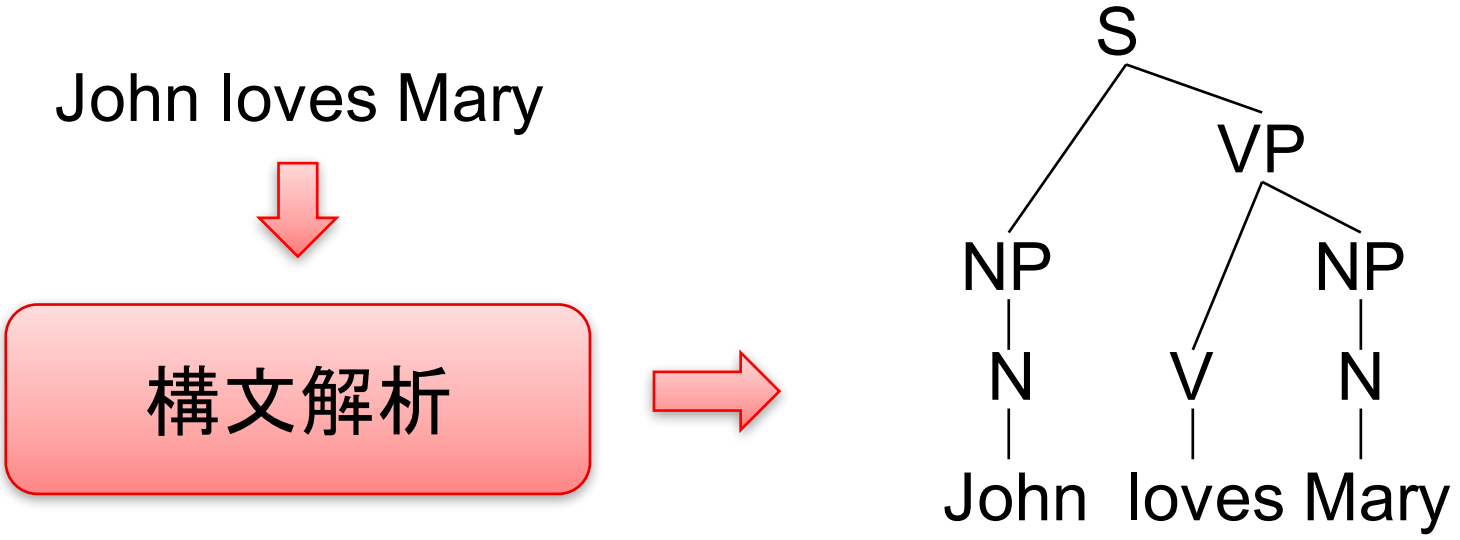




S: 文(sentence)
NP: 名詞句(noun phrase)
VP: 動詞句(verb phrase)
N: 名詞 (noun)
V: 動詞 (verb)

構文解析

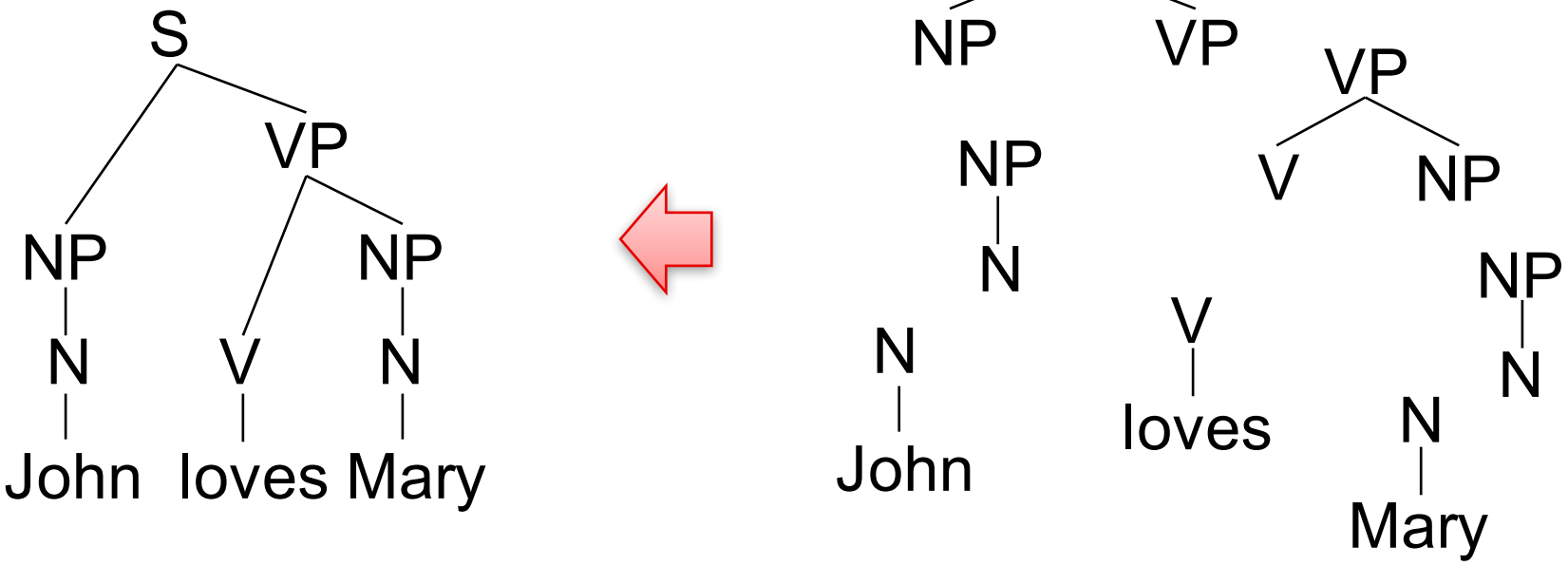
- 入力：単語列（＋品詞）
- 出力：構文木



- 構文木を計算するにはどういうデータ構造とアルゴリズムを用いればよいか？

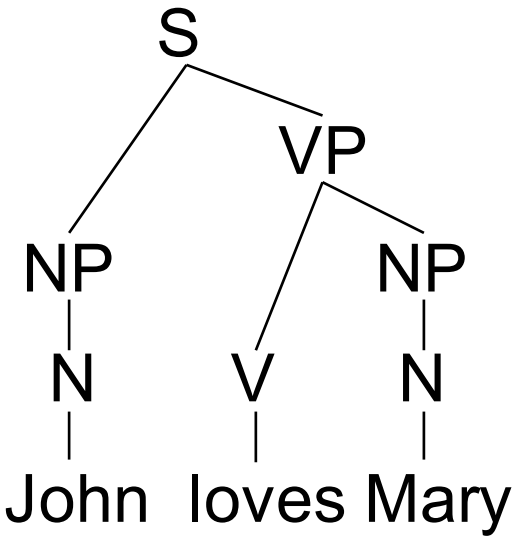
文脈自由文法

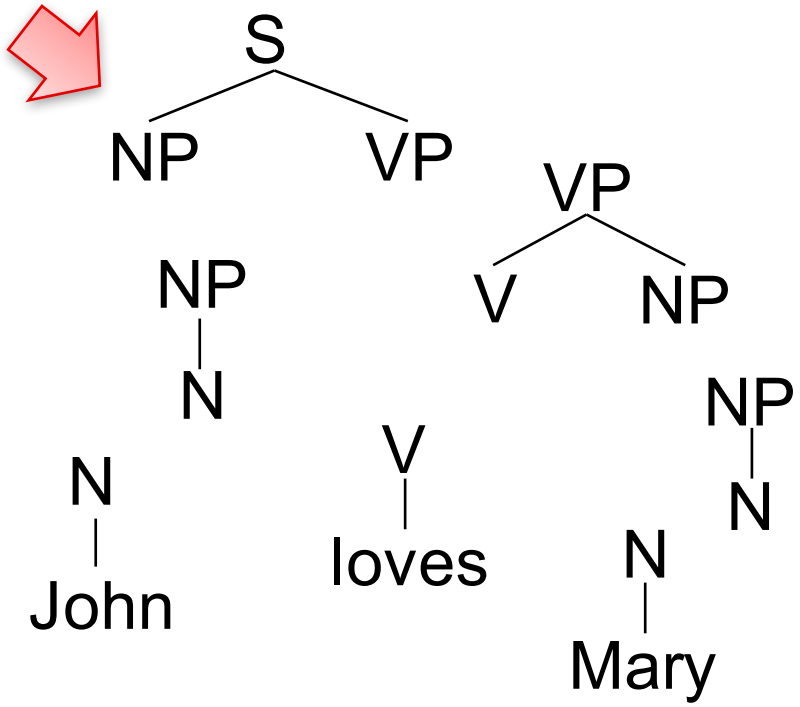
- Context-Free Grammar (CFG)
- 構文木を生成する規則の集合を考える
- 部分木が組み合わさって一つの木になったと考える



文脈自由文法

S → NP VP
NP → N
VP → V NP
N → John
N → Mary
V → loves





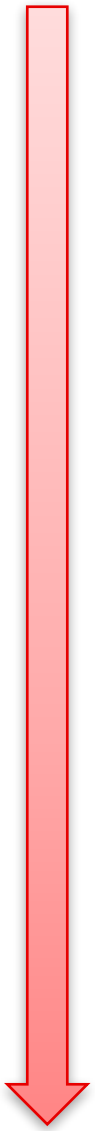
定式化

- 文法 $G = \langle N, \Sigma, R, S \rangle$
  - $N = \{\text{S, NP, VP, ...}\}$: 非終端記号の集合
  - $\Sigma = \{\text{friend, gave, cake, ... }\}$: 終端記号の集合
  - $R = \{\text{S} \rightarrow \text{NP VP, ... }\}$: 生成規則の集合
    - $r \in R$: $lhs \rightarrow rhs$ where $lhs \in N$, $rhs \in (N \cup \Sigma)^*$
  - $S \in N$: 開始記号
- 導出
  - 開始記号からスタートして、非終端記号を書き換えていく
  - 最終的に終端記号列（＝文）が得られる

CFGの例

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP
D → a
D → the
N → girl
N → moon
N → telescope
V → saw
P → with

導出

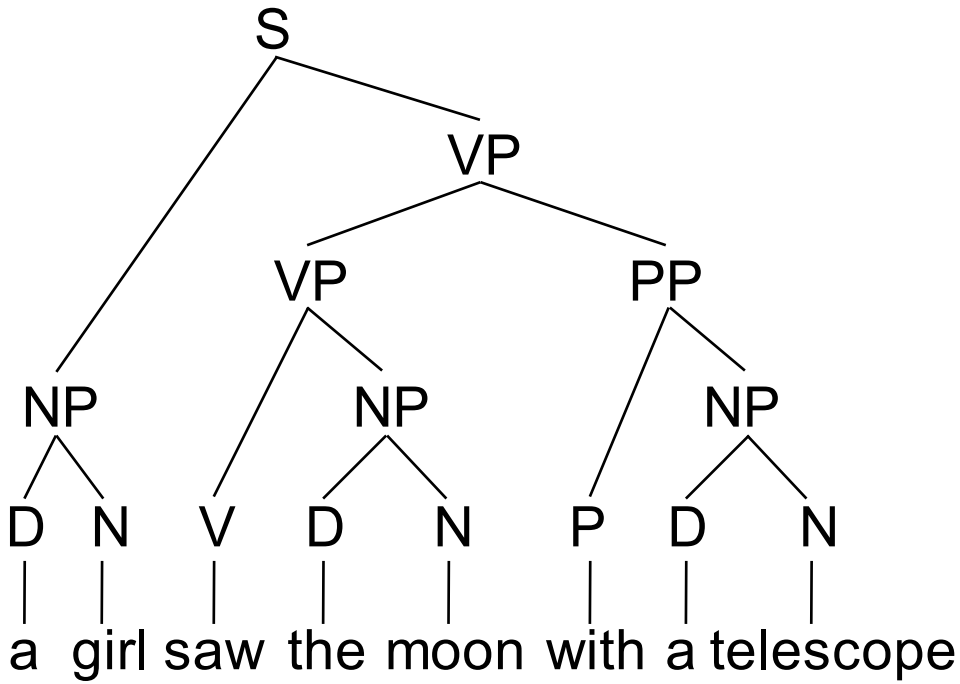


- S ←開始記号
- NP VP
- D N VP
- a N VP
- a girl VP
- a girl VP PP
- a girl V NP PP
- a girl saw NP PP
- a girl saw D N PP
- a girl saw the N PP
- a girl saw the moon PP
- a girl saw the moon P NP
- a girl saw the moon with NP
- a girl saw the moon with D N
- a girl saw the moon with a N
- a girl saw the moon with a telescope

D: 冠詞 (determiner) P: 前置詞 (preposition) PP: 前置詞句 (prepositional phrase)

構文木

構文木＝導出の履歴



- S
- NP VP
- D N VP
- a N VP
- a girl VP
- a girl VP PP
- a girl V NP PP
- a girl saw NP PP
- a girl saw D N PP
- a girl saw the N PP
- a girl saw the moon PP
- a girl saw the moon P NP
- a girl saw the moon with NP
- a girl saw the moon with D N
- a girl saw the moon with a N
- a girl saw the moon with a telescope

D: 冠詞 (determiner) P: 前置詞 (preposition) PP: 前置詞句 (prepositional phrase)

演習1:構文木

- Time flies like an arrow の構文木をすべて挙げよ

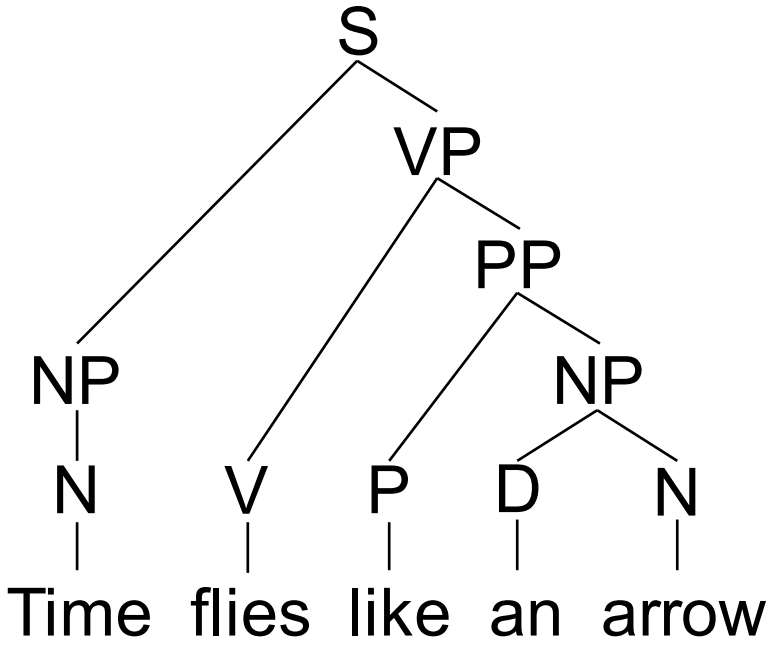
S → NP VP | VP
NP → N | D N | N N | NP PP
VP → V NP | V | VP PP
PP → P NP
D → a | an | the
N → girl | moon | telescope
| time | flies | arrow
V → saw | time | flies | like
A → time | like | happy
P → with | like | in

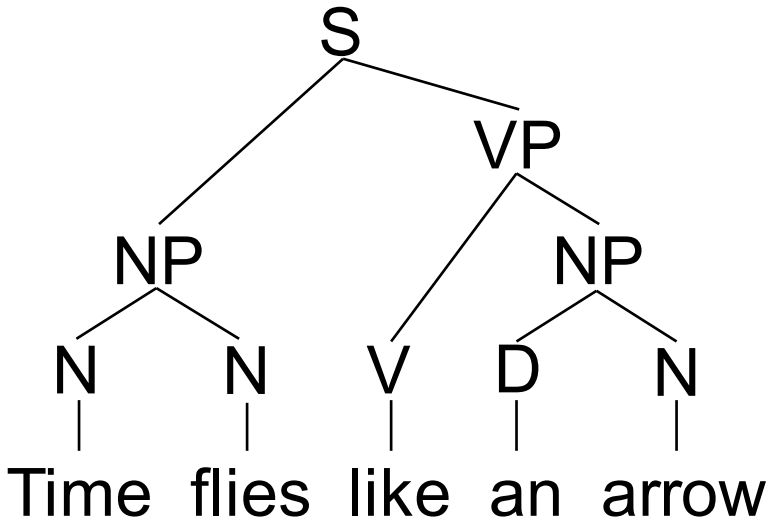
?

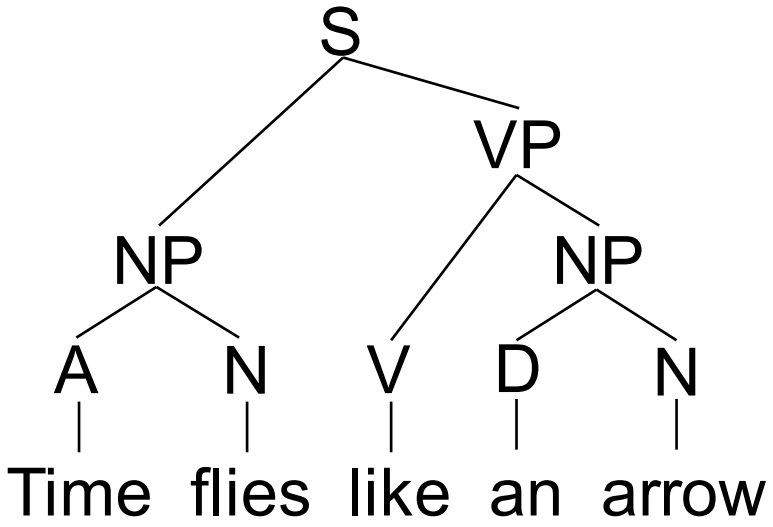
Time flies like an arrow

A: 形容詞 (adjective)

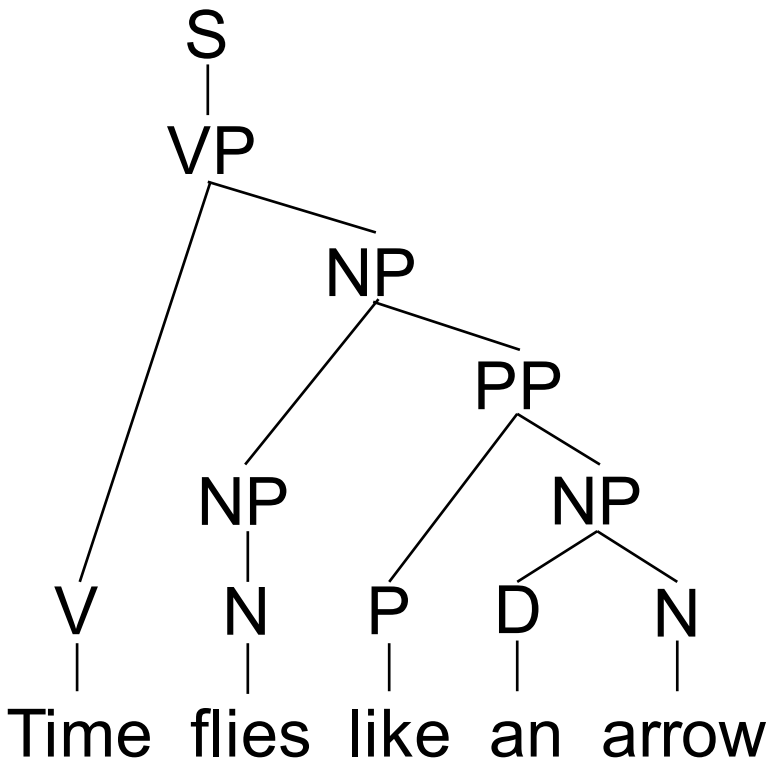
構文木

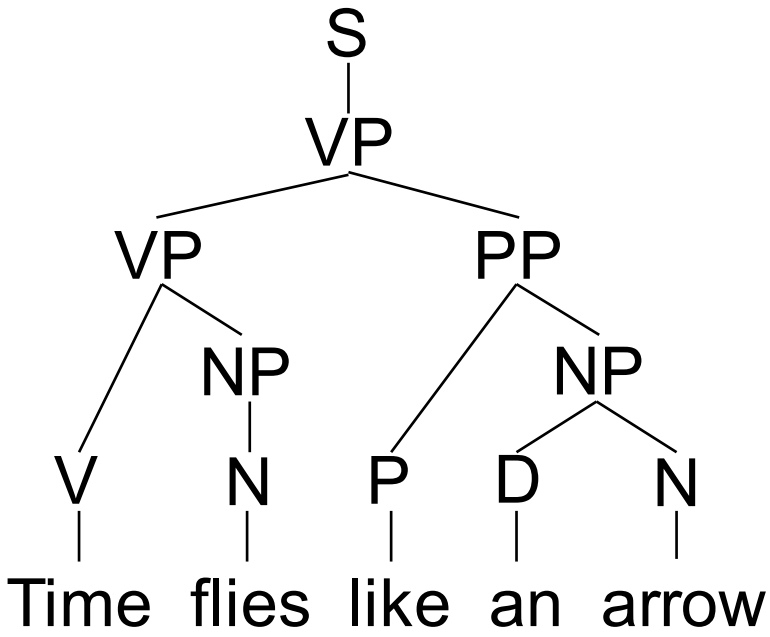

S
VP
PP
NP
NP
N
V
P
D
N
Time
flies
like
an
arrow


S
VP
NP
NP
N
N
V
D
N
Time
flies
like
an
arrow


S
VP
NP
NP
A
N
V
D
N
Time
flies
like
an
arrow

構文木


S
VP
NP
PP
NP
NP
V
N
P
D
N
Time
flies
like
an
arrow

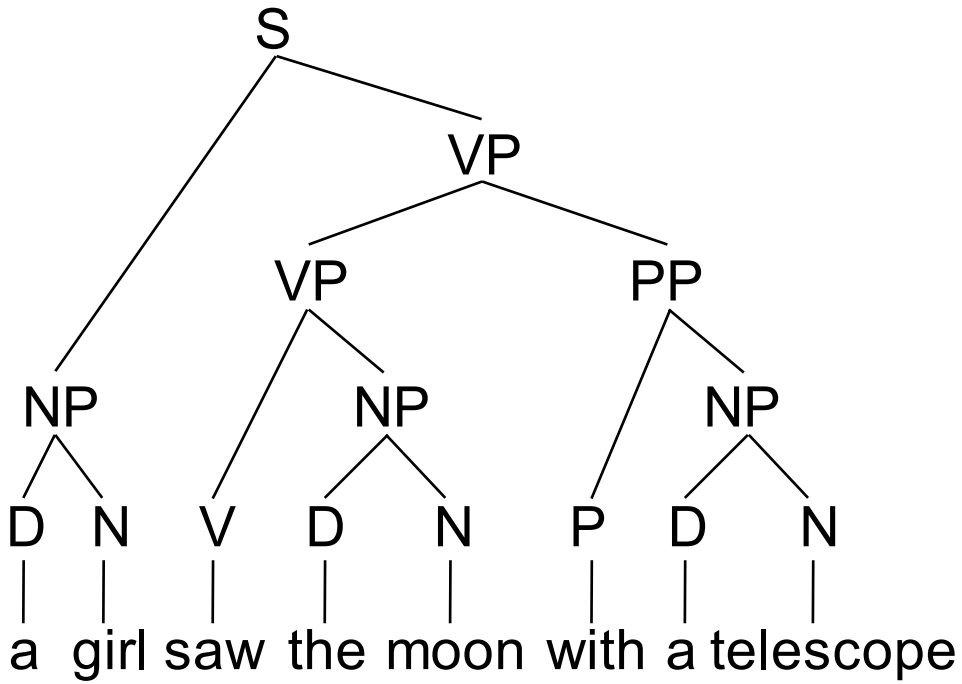

S
VP
VP
PP
NP
NP
V
N
P
D
N
Time
flies
like
an
arrow

構文解析

- 文が与えられた時、その構文木を計算するにはどうしたらよいか？
  - 入力：終端記号列
  - 出力：構文木
- 与えられた文を生成するような導出を計算すればよい
  - つまり、導出の逆向きがしたい
- 基本アイディア
  - ある終端記号列を生成するような規則を見つけて逆向きに適用する（ボトムアップ構文解析）

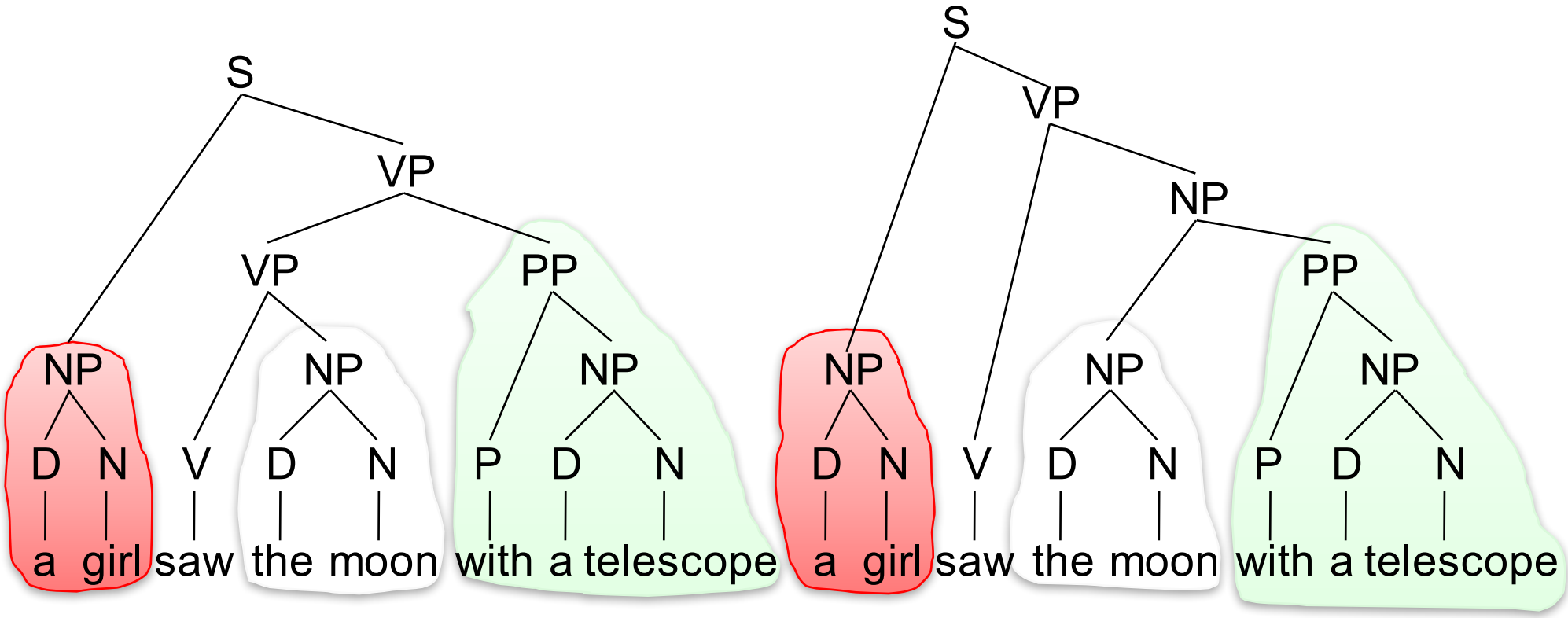
構文解析の例

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP
D → a
D → the
N → girl
N → moon
N → telescope
V → saw
P → with



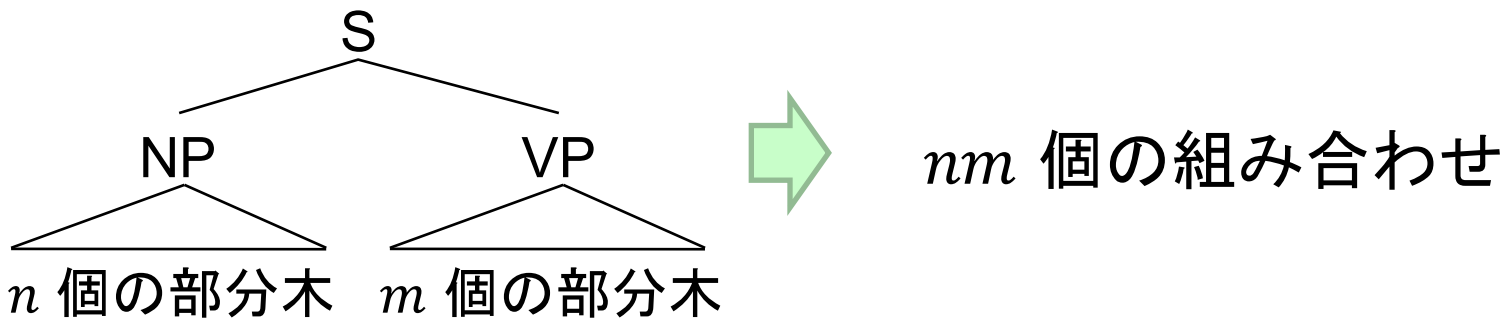
曖昧性の問題

- 同じ終端記号列に対して、複数の構文木が作れる場合がある
- 全ての構文木を列挙すると、同じ部分木を何回も計算する必要がある



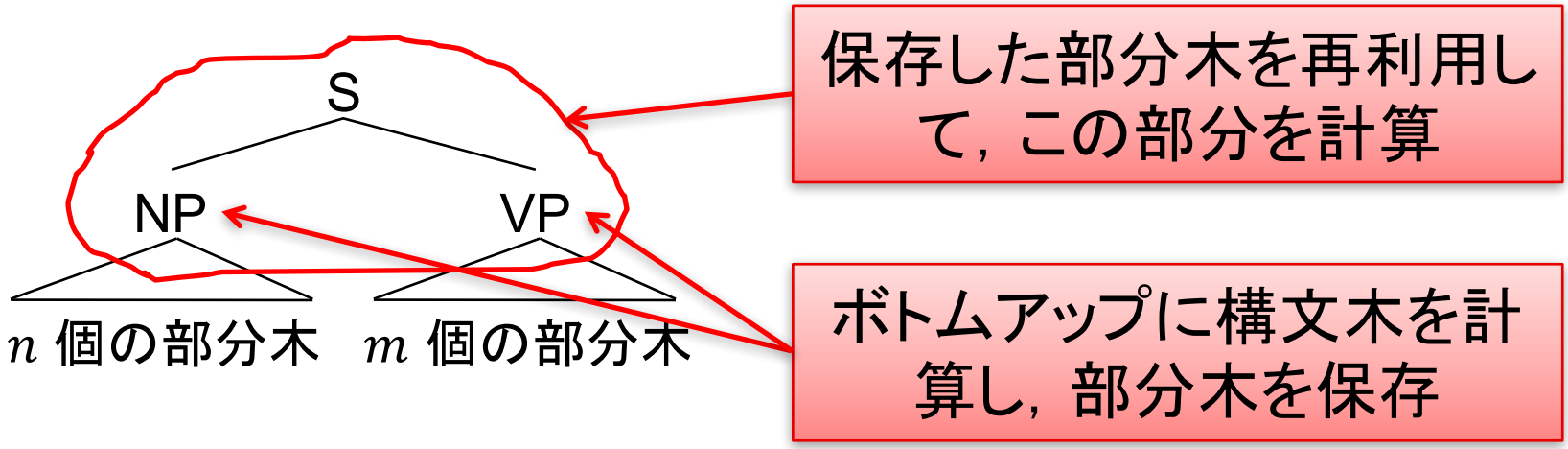
曖昧性の問題

- 全ての構文木を列挙すると指数爆発する
  - c.f. カタラン数



動的計画法

- 構文解析の途中結果（部分木）を保存しながら解析する
- 保存した部分木を再利用することで、同じ部分木を再計算しない
  → 指数爆発しない

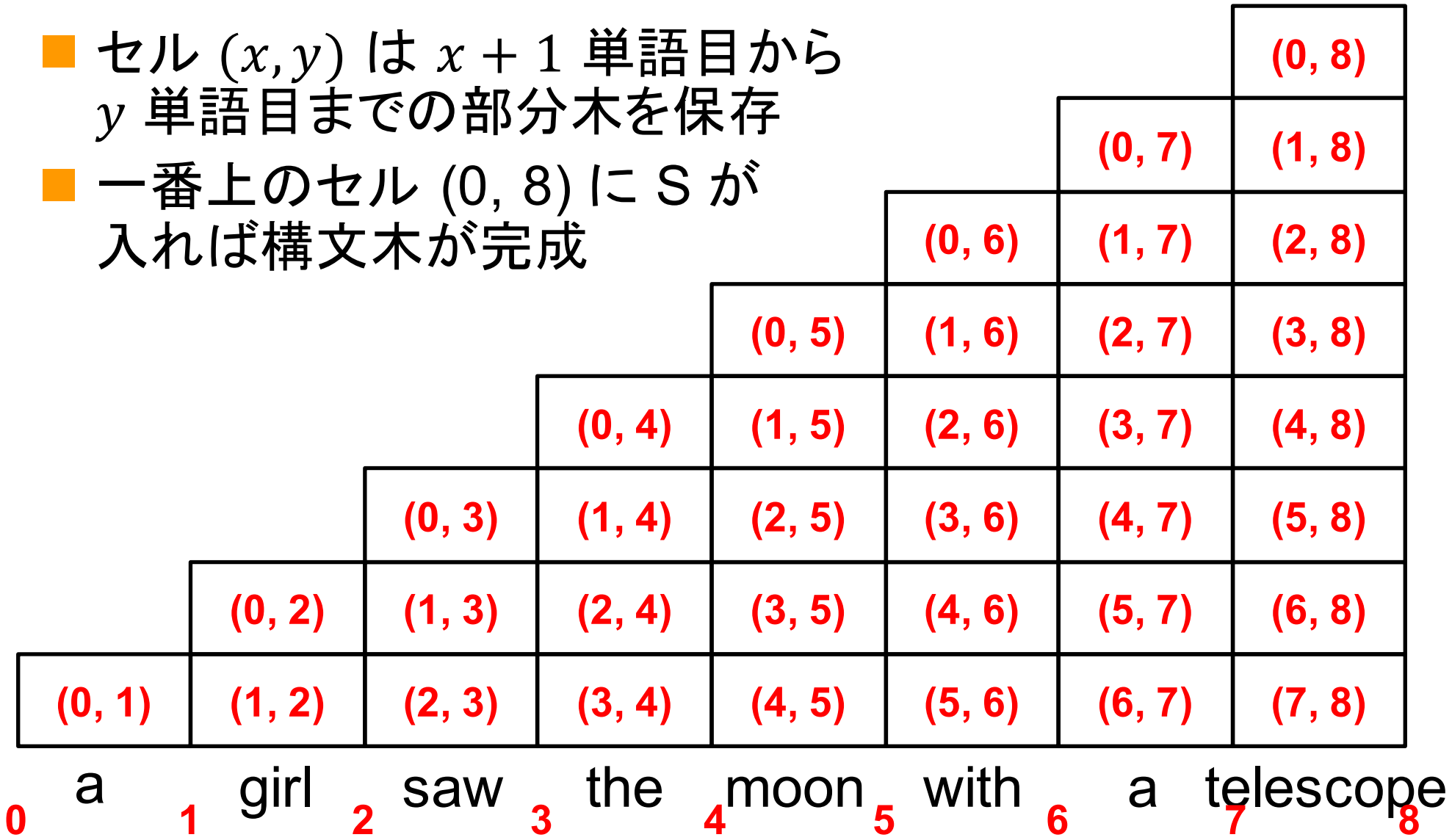


CKY法

- Chomsky標準形(CNF)の CFG の構文解析アルゴリズム
  - CNF: 全ての生成規則の rhs は終端記号1個か、あるいは非終端記号2個
    - A → a
    - A → B C
  - 非終端記号1個の生成規則があっても、簡単な拡張で対応可能
    - A → B

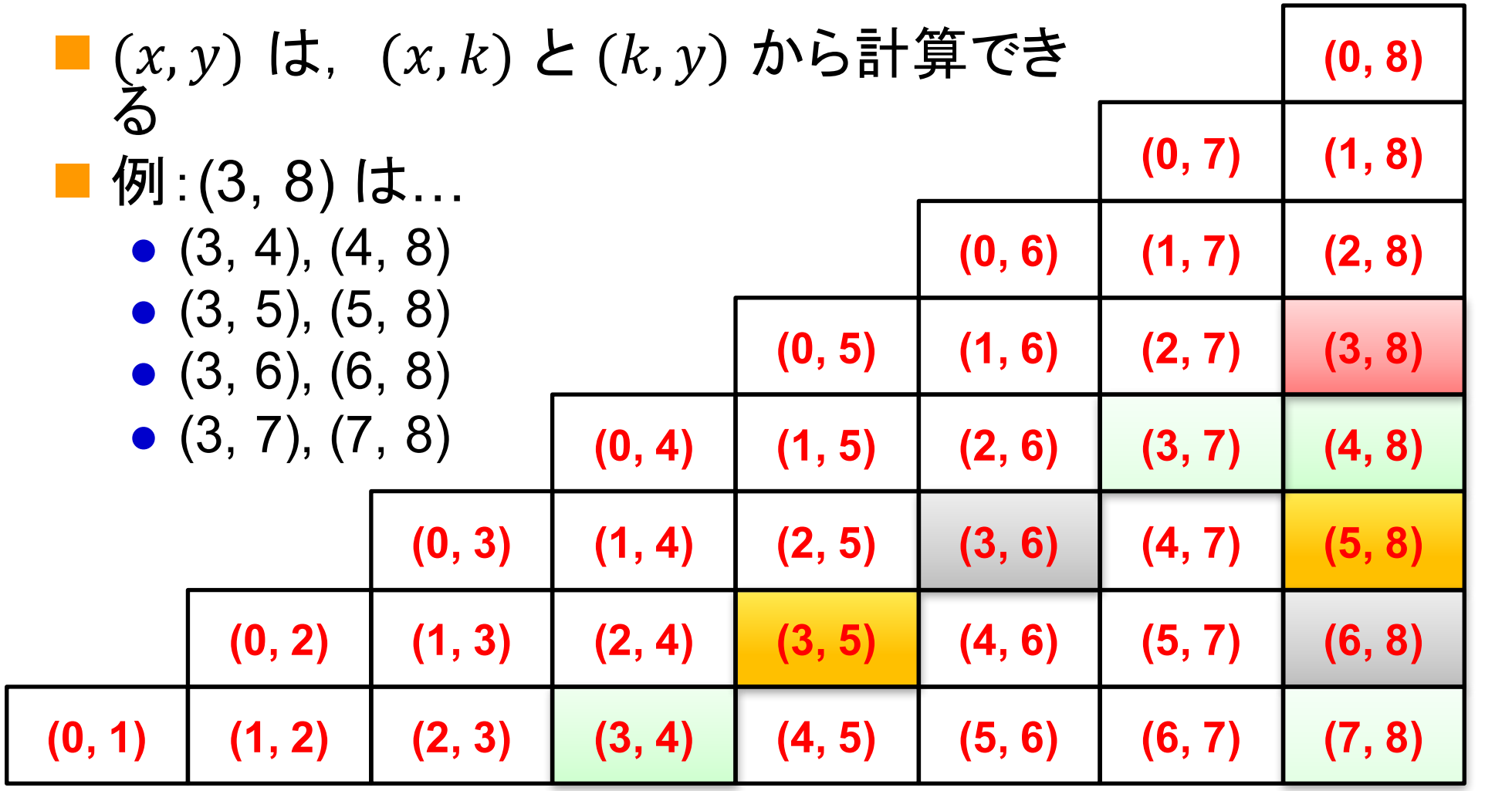
CKY表

- セル (x, y) は $x+1$ 単語目から y 単語目までの部分木を保存
- 一番上のセル (0, 8) に S が入れば構文木が完成



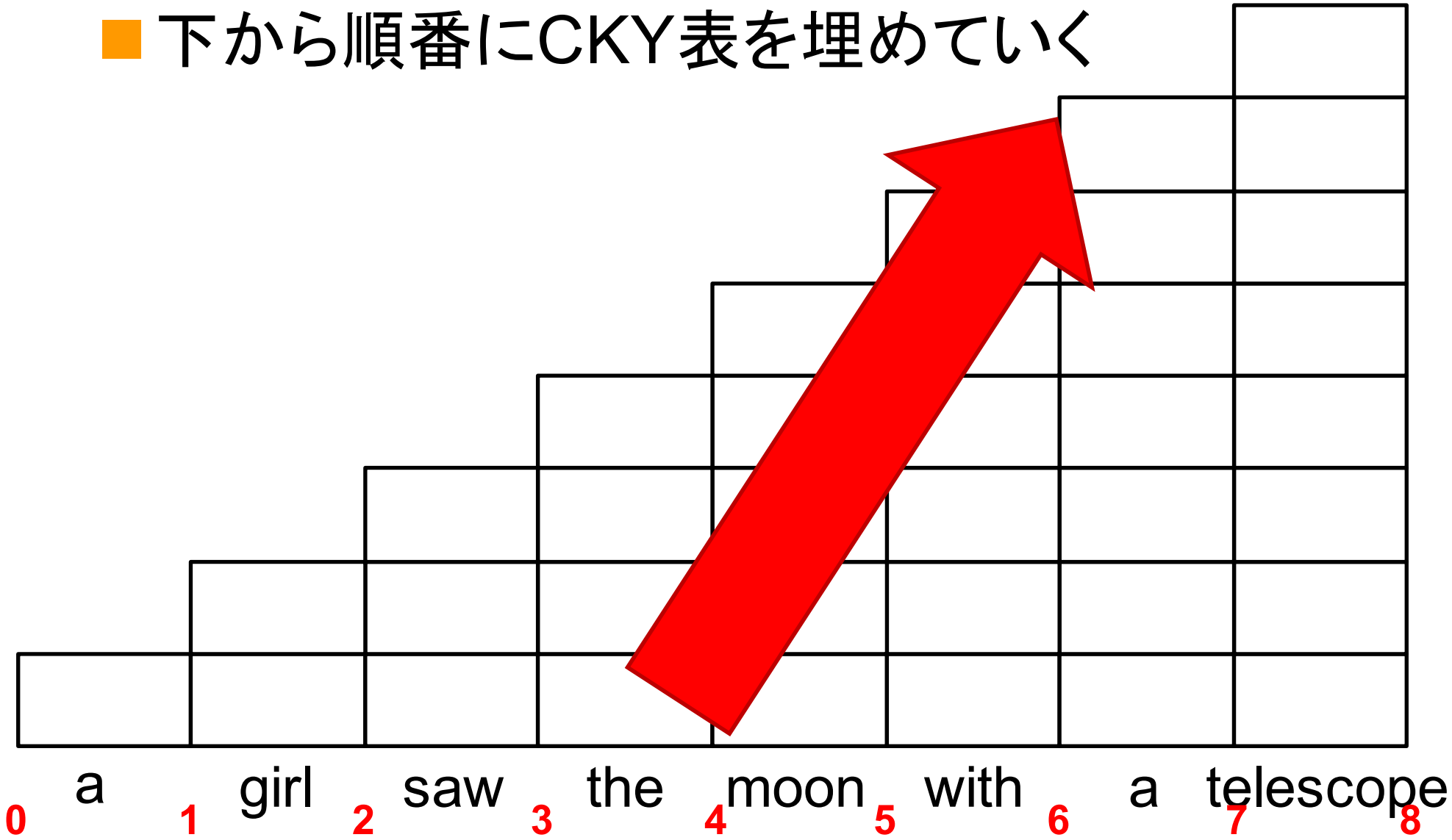
CKY表

- (x, y) は，(x, k) と (k, y) から計算できる
- 例：(3, 8) は…
  - (3, 4), (4, 8)
  - (3, 5), (5, 8)
  - (3, 6), (6, 8)
  - (3, 7), (7, 8)



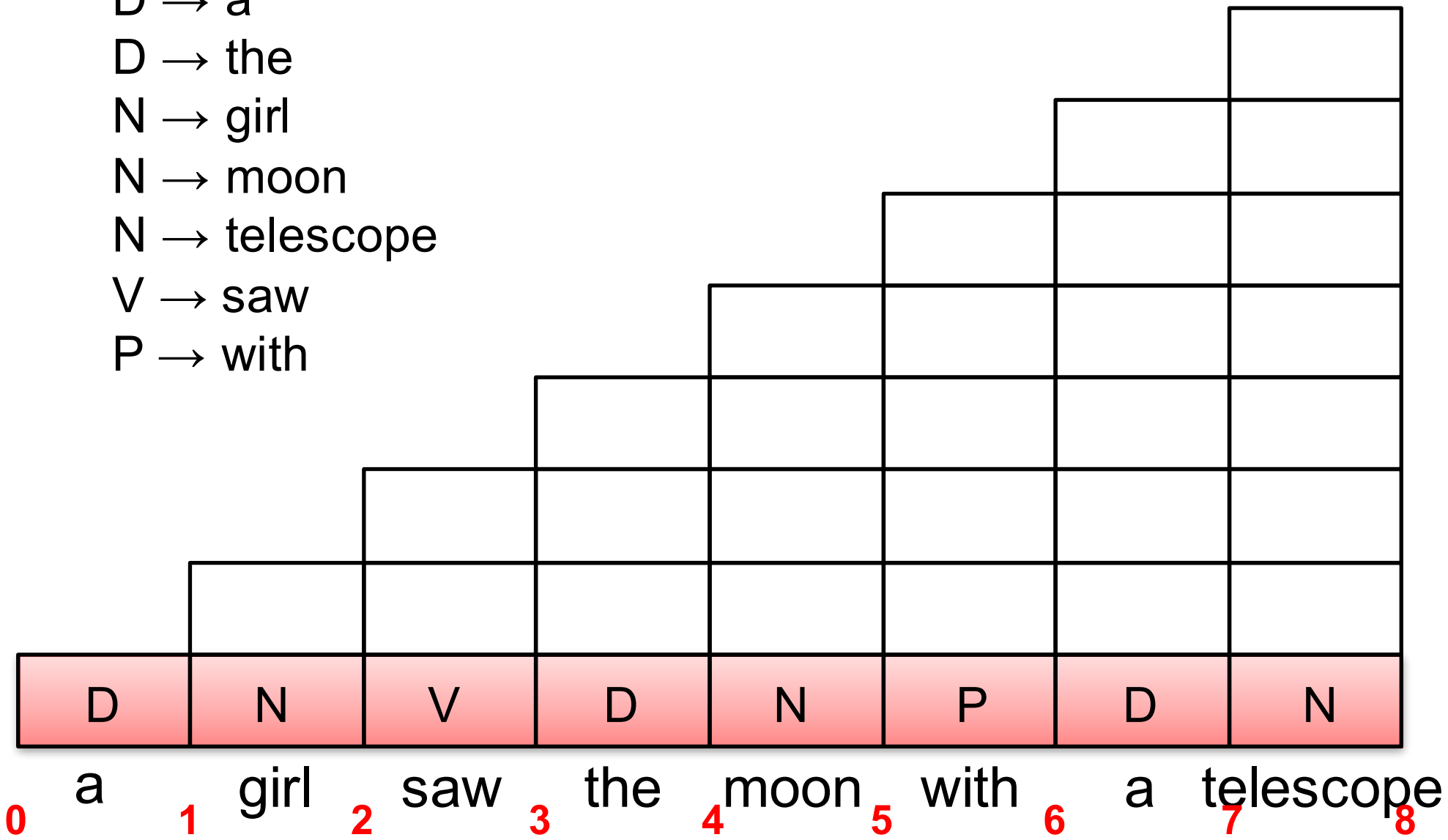
CKY法

- 下から順番にCKY表を埋めていく



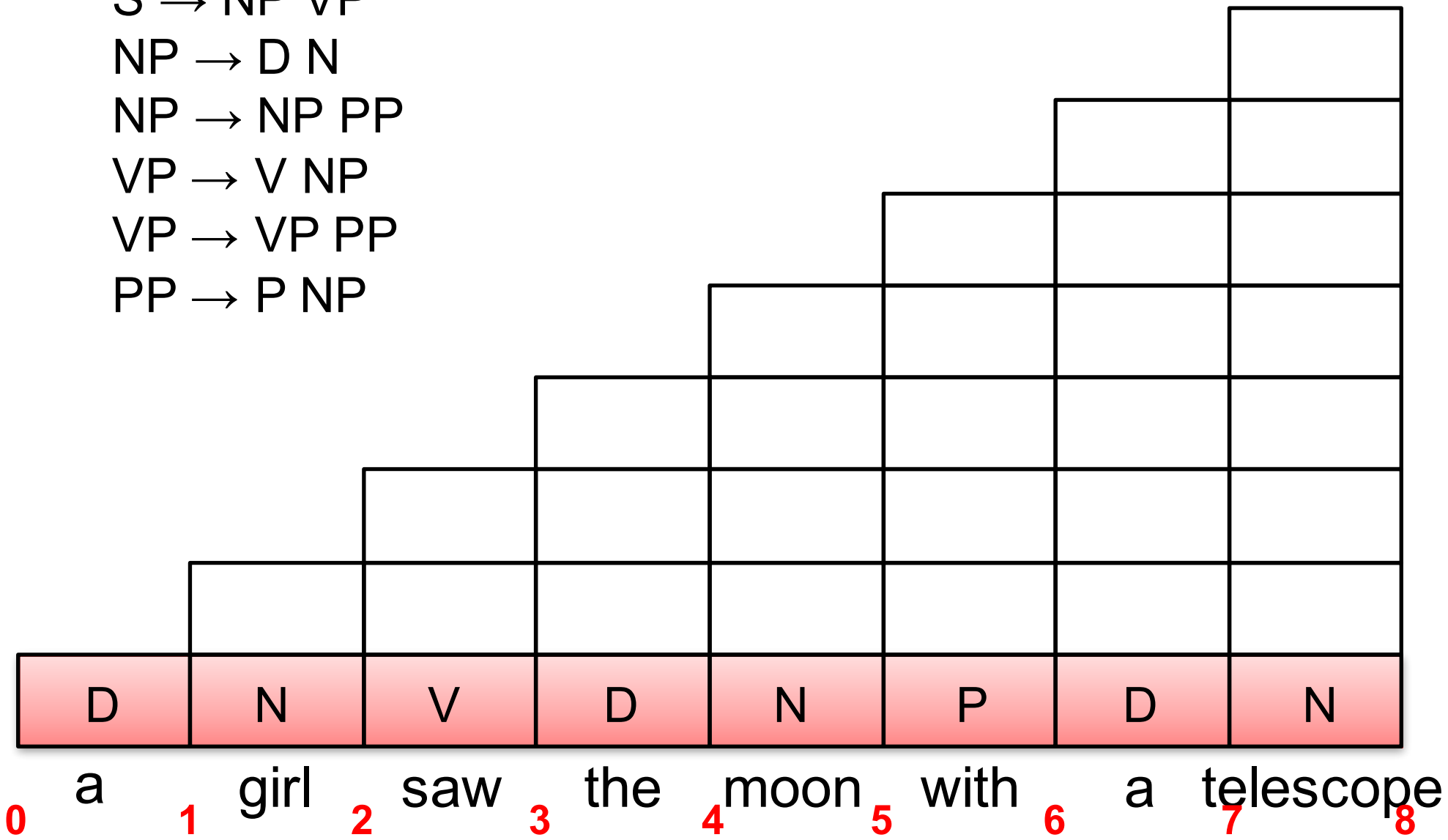
CKY法

D → a
D → the
N → girl
N → moon
N → telescope
V → saw
P → with



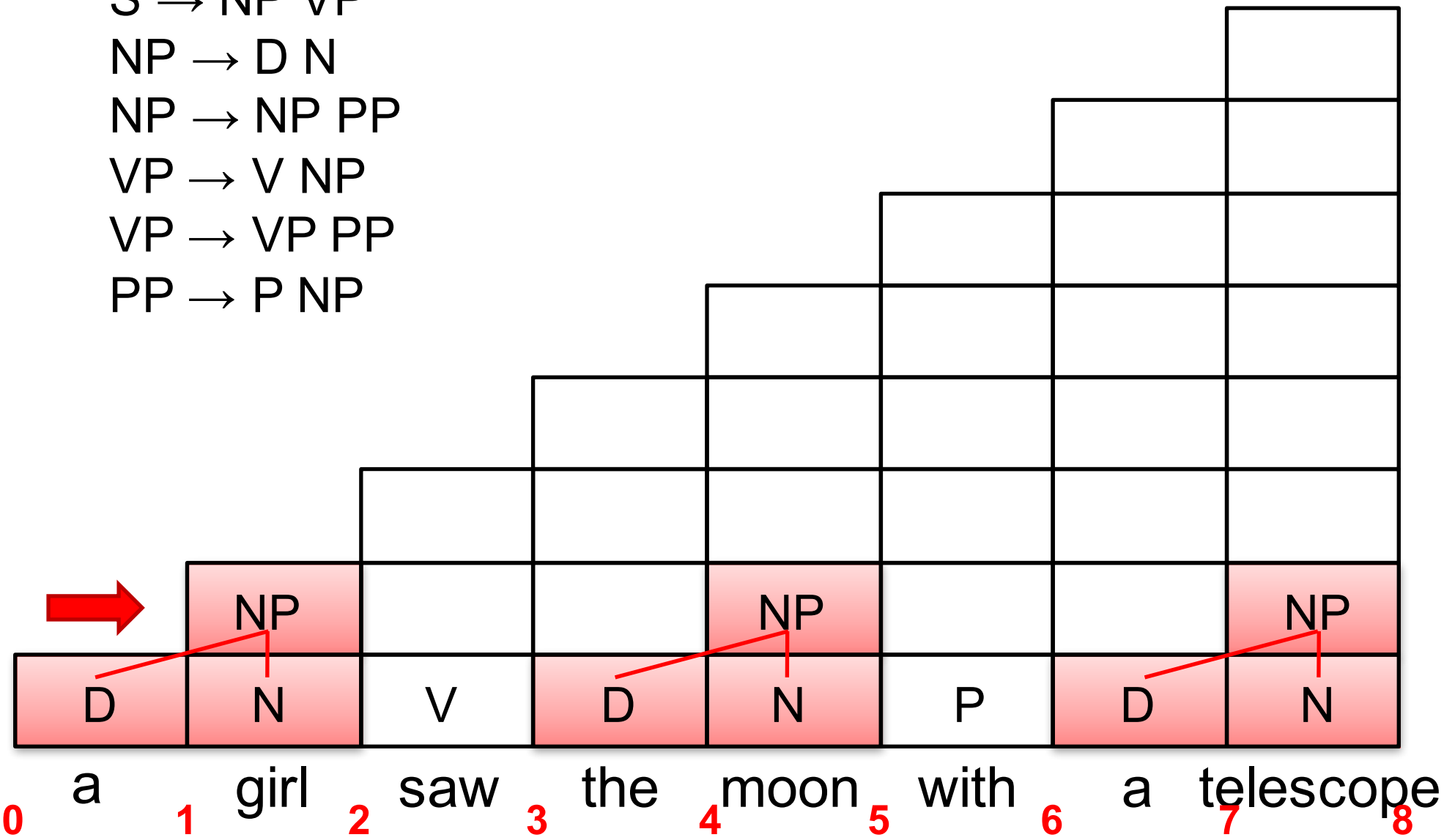
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



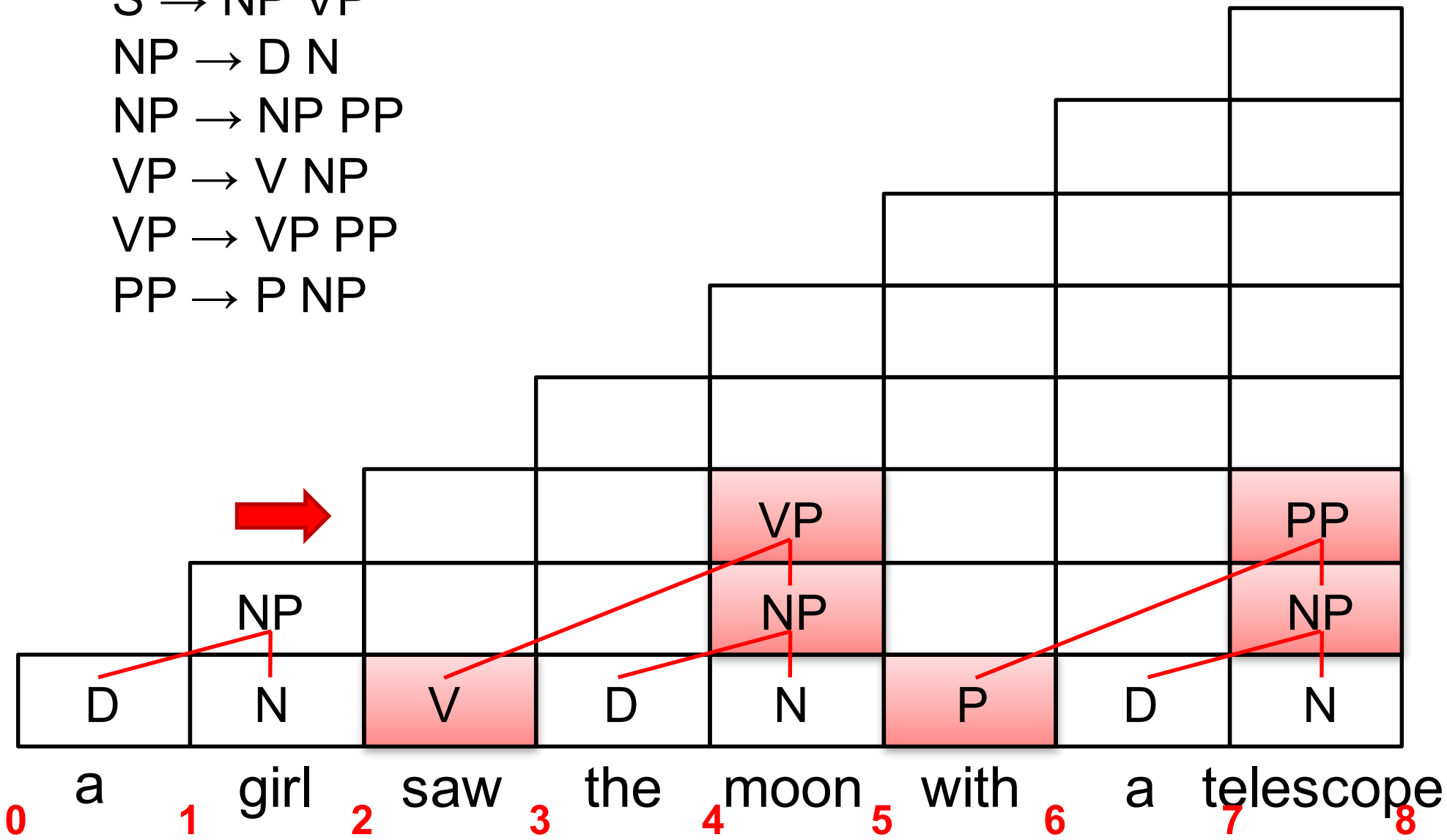
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



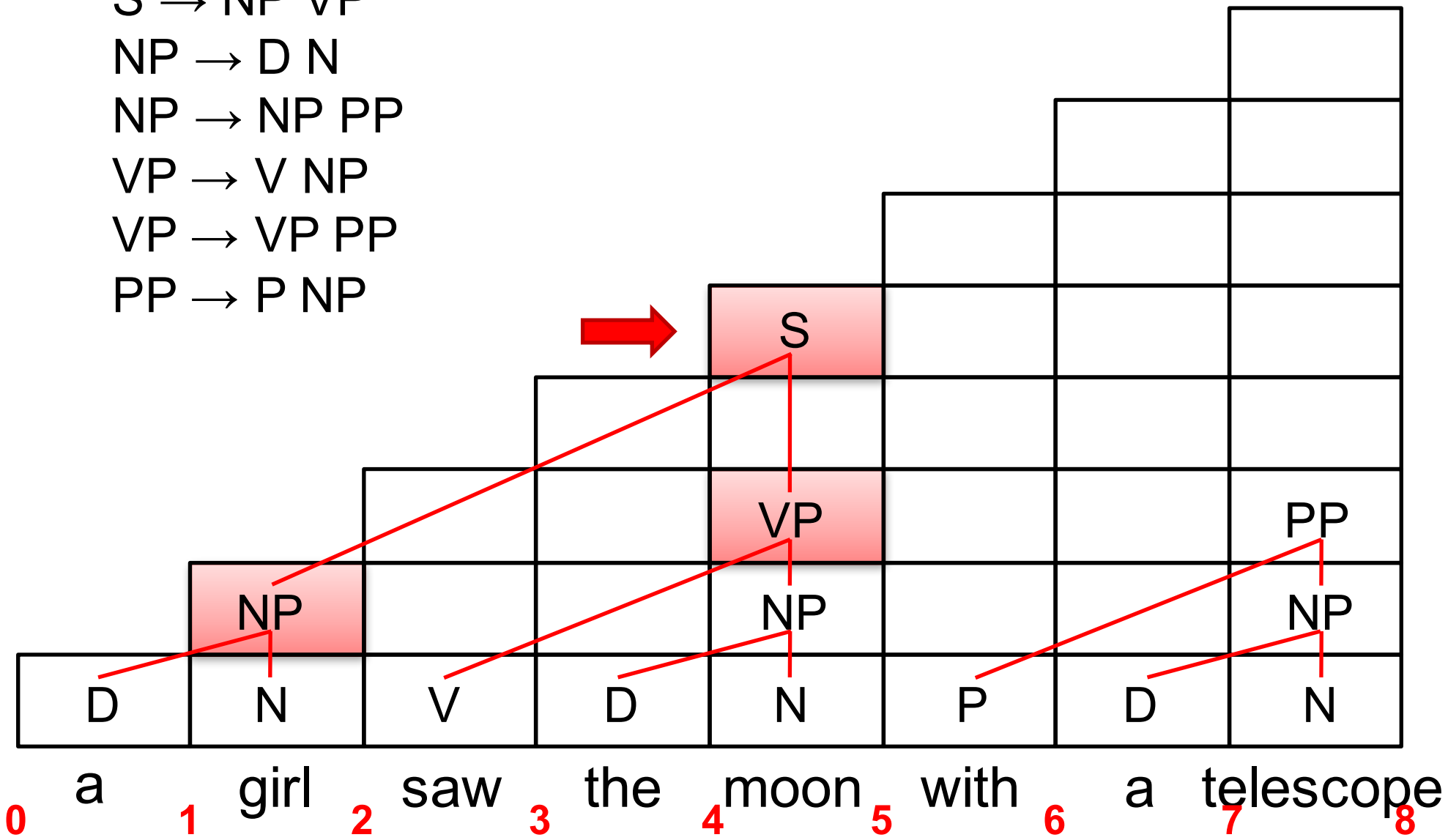
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



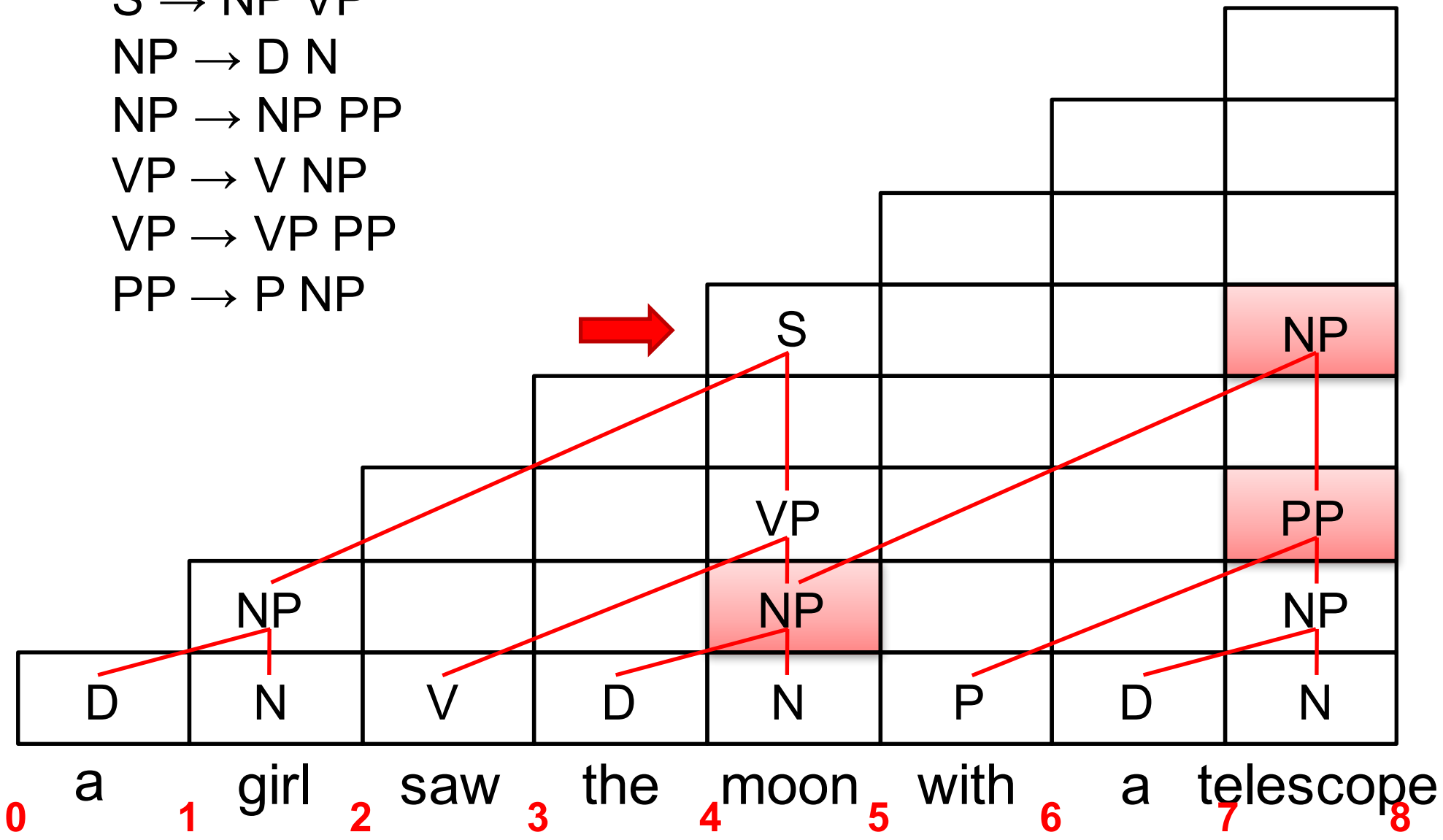
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



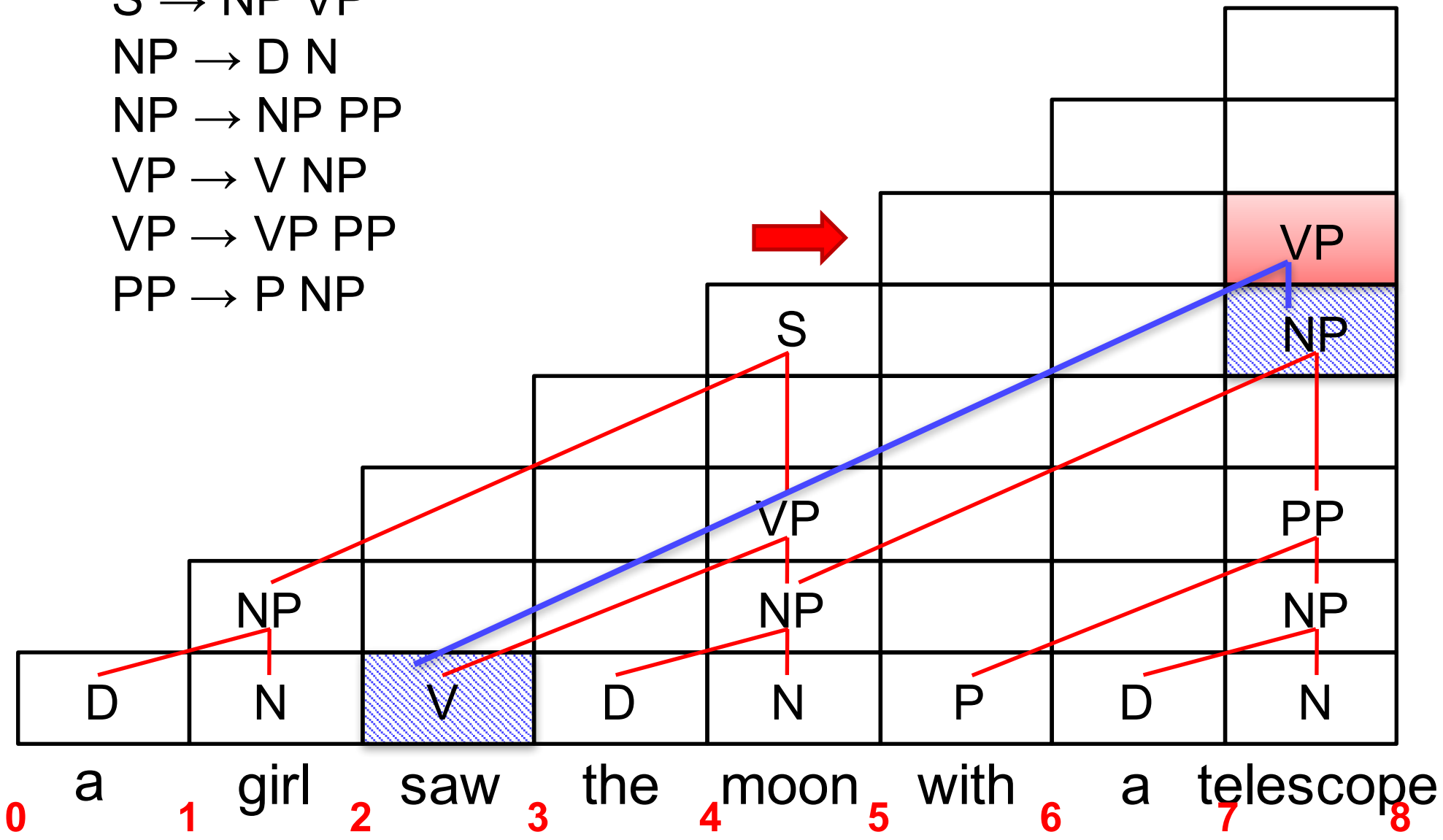
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



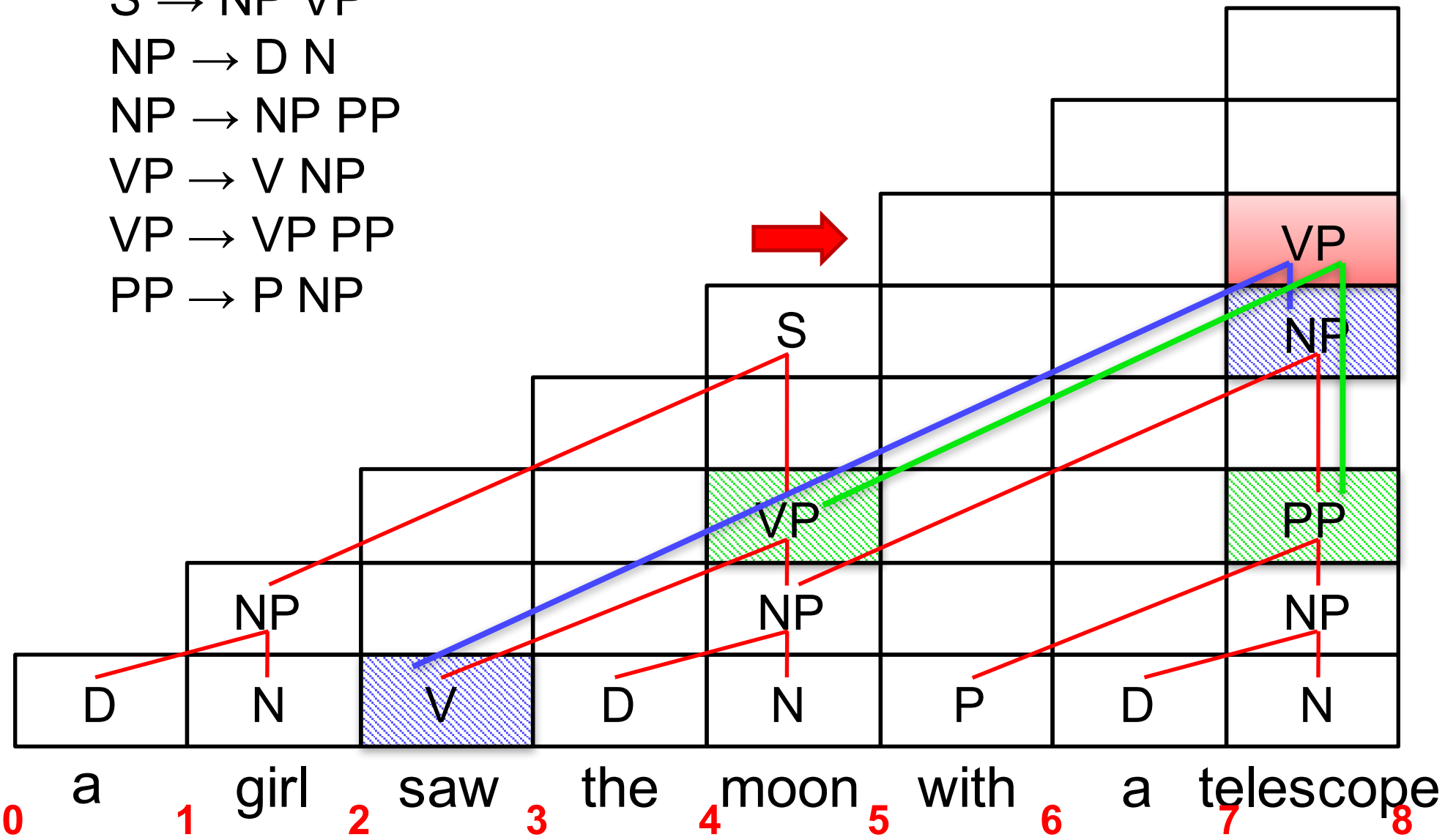
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



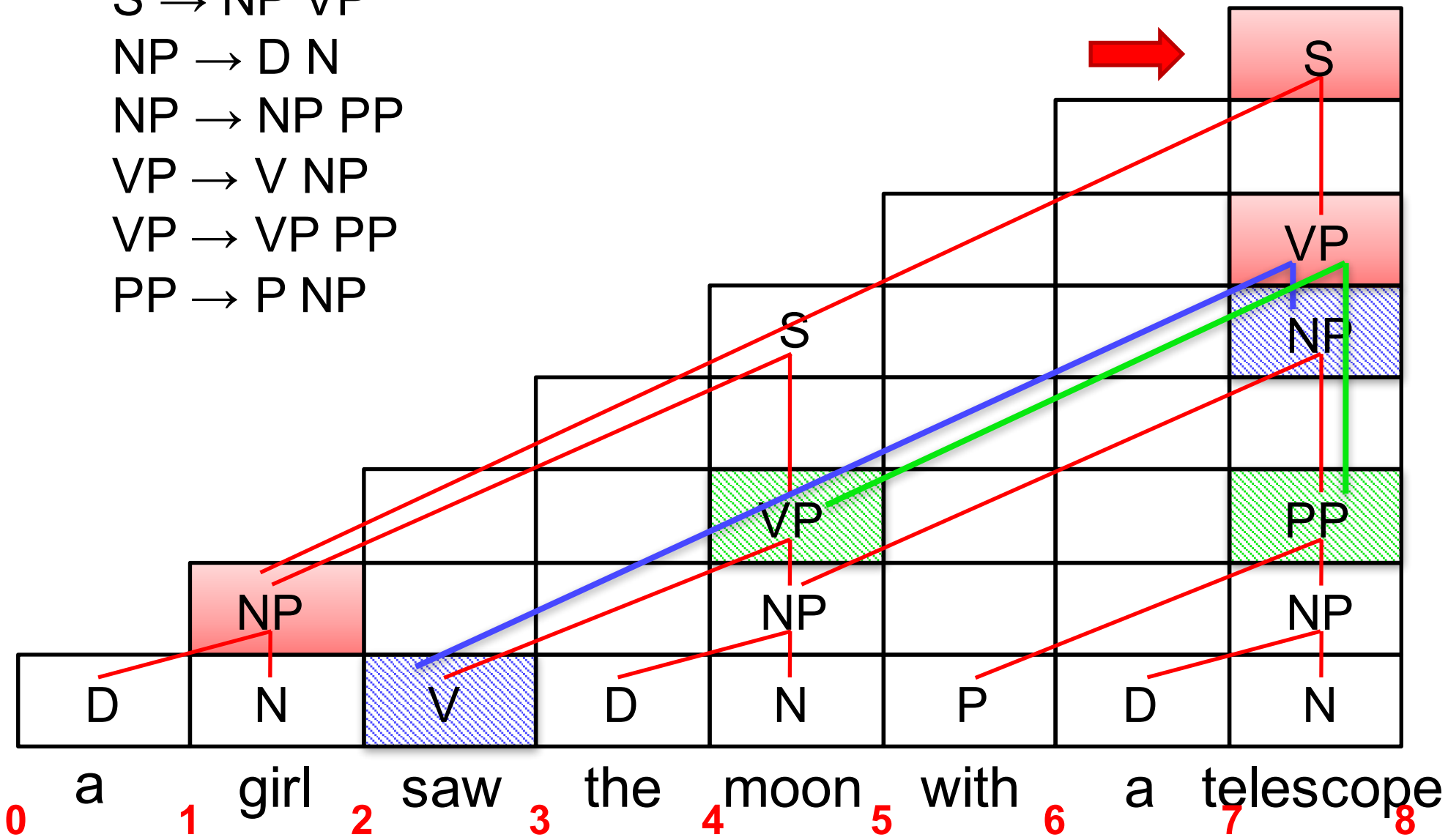
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



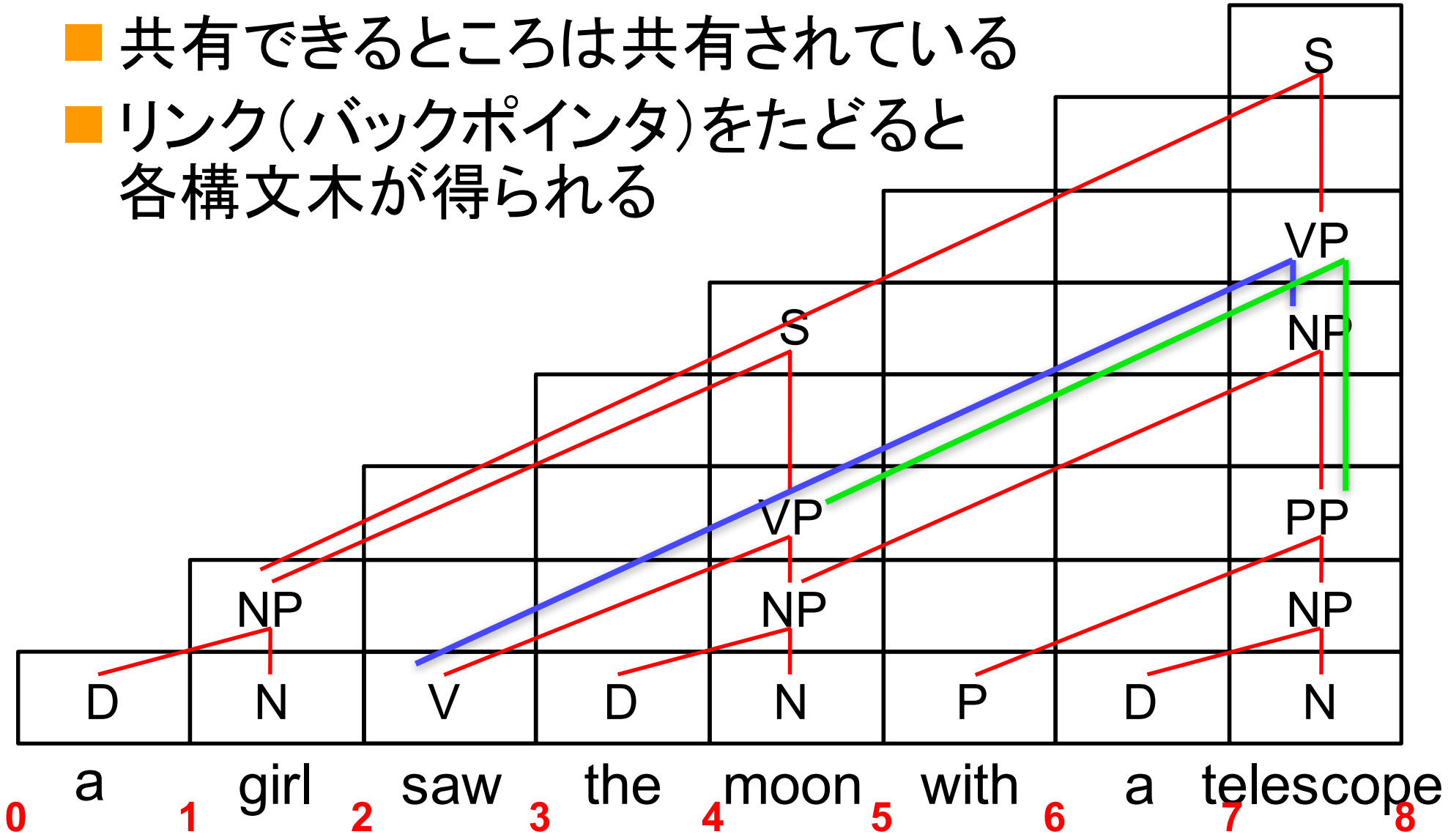
CKY法

S → NP VP
NP → D N
NP → NP PP
VP → V NP
VP → VP PP
PP → P NP



CKY法

- 共有できるところは共有されている
- リンク（バックポインタ）をたどると各構文木が得られる



CKY法

[初期化：CKY表の1段目を埋める]

for $i = 2$ to n ← CKY表の i 段目

  for $x = 0$ to $n - i$

    $y = x + i$

    for $k = x + 1$ to $y - 1$

      for $X \in \text{table}(x, k)$

        for $Y \in \text{table}(k, y)$

          for $Z \in N$

            if $Z \rightarrow X\ Y \in R$ then

              $\text{table}(x, y) \leftarrow \text{table}(x, y) \cup \{Z\}$

              $\text{back}(x, y, Z) \leftarrow \text{back}(x, y, Z) \cup \{(k, X, Y)\}$

左と右のセルに入っている非終端記号

$X\ Y$ を生成するような生成規則があったらCKY表に追加

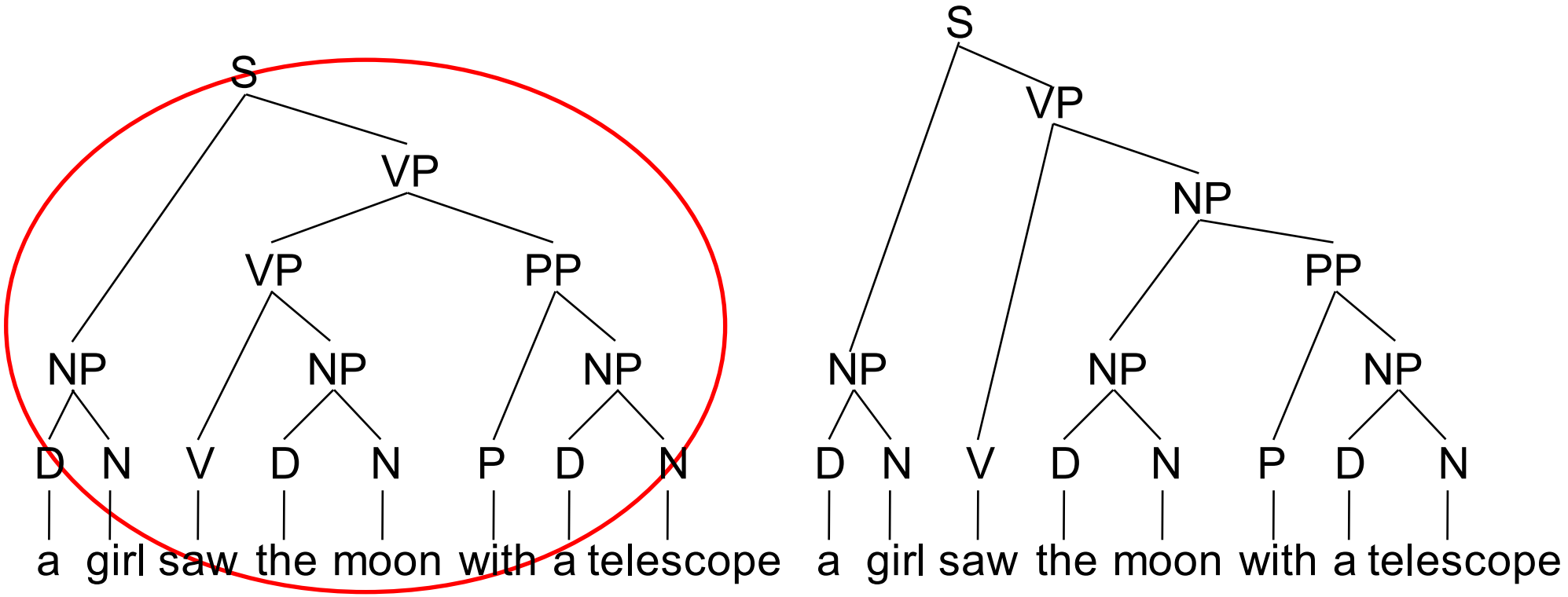
- 計算量は？

その他の構文解析アルゴリズム

- 途中結果を保存して表を順番に埋めていくという方針は同じ
  - アーリー法
  - チャート法
  - 一般化LR法

曖昧性解消

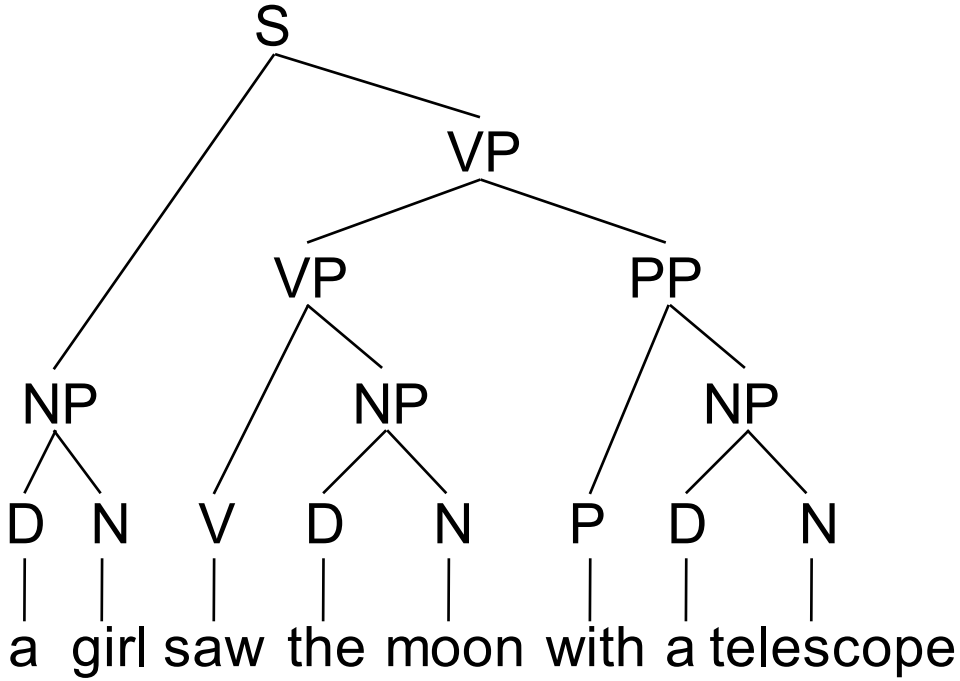
- CFGによる解析では，一つの文に対して複数の構文木が作られる
- 通常，人間の解釈は一つの構文木に対応
- 人間の解釈に相当する構文木を出力するには？



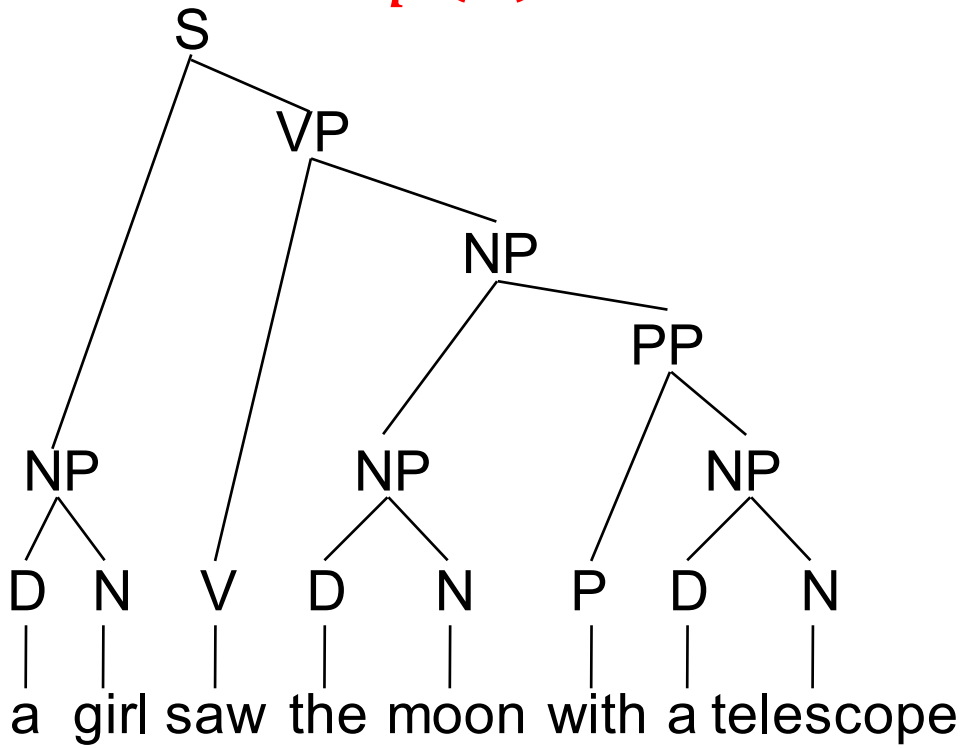
確率による曖昧性解消

- 構文木 T に対して確率 $p(T)$ を求める
- $T^* = \underset{T}{\operatorname{argmax}}\, p(T)$ を出力する

$p(T) = 0.2$



$p(T) = 0.0003$



確率文脈自由文法(PCFG)

- 生成規則に確率を割り当てる
- 文法 $G = \langle N, \Sigma, R, S, p \rangle$
  - N = {S, NP, VP, ...}: 非終端記号の集合
  - Σ = {friend, gave, cake, ... }: 終端記号の集合
  - R = {S → NP VP, ... }: 生成規則の集合
    - $r \in R$: $lhs \rightarrow rhs$ where $lhs \in N$, $rhs \in (N \cup \Sigma)^*$
  - $S \in N$: 開始記号
  - $p(r)$: 生成規則の確率
- 構文木の確率は生成規則の確率の積

$$p(T) = \prod_{r \in T} p(r)$$

確率文脈自由文法(PCFG)

- p の定義
- 左辺が同じ記号の生成規則の確率の和が1
  ＝左辺の記号を条件にした条件付き確率

$p(\text{NP} \rightarrow \text{D N} \mid \text{NP})$ [0.7]
$p(\text{NP} \rightarrow \text{NP PP} \mid \text{NP})$ [0.2]
… …

左辺が NP の規則の確率を足すと1

$p(\text{VP} \rightarrow \text{V NP} \mid \text{VP})$ [0.6]
$p(\text{VP} \rightarrow \text{VP PP} \mid \text{VP})$ [0.3]
… …

左辺が VP の規則の確率を足すと1

演習2:PCFGによる構文解析

- A girl saw the moon with a telescope のすべての構文木の確率を求めよ

S → NP VP	[1.0]
NP → D N	[0.8]
NP → NP PP	[0.2]
VP → V NP	[0.6]
VP → VP PP	[0.4]
PP → P NP	[1.0]
D → a	[0.5]
D → the	[0.5]
N → girl	[0.4]
N → moon	[0.3]
N → telescope	[0.3]
V → saw	[1.0]
P → with	[1.0]

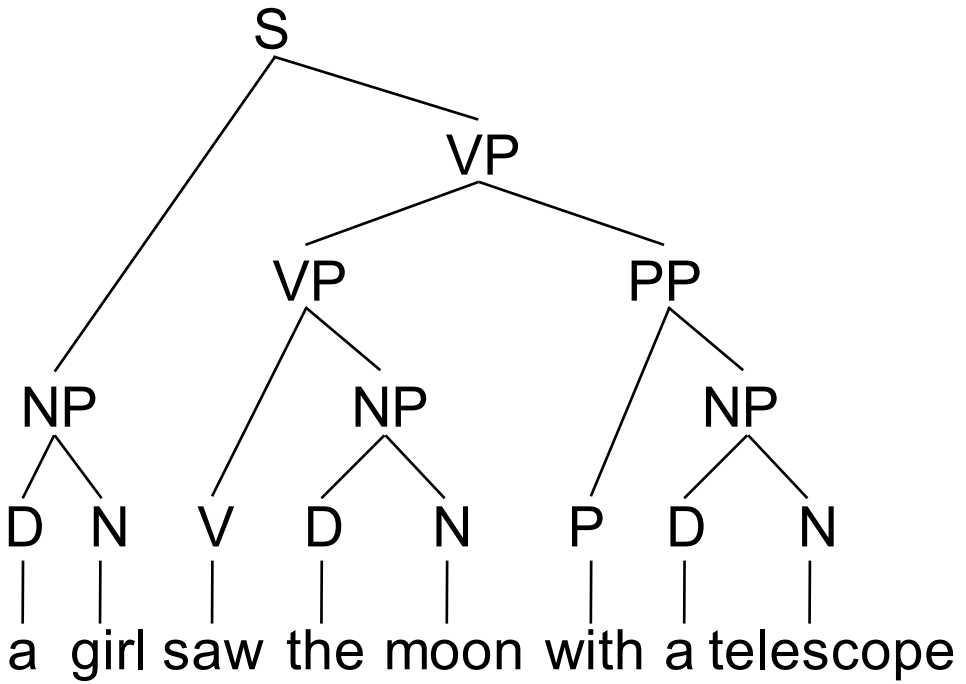
?

a girl saw the moon with a telescope

構文木1

S → NP VP	[1.0]	
NP → D N	[0.8]	×3
NP → NP PP	[0.2]	
VP → V NP	[0.6]	
VP → VP PP	[0.4]	
PP → P NP	[1.0]	
D → a	[0.5]	×2
D → the	[0.5]	
N → girl	[0.4]	
N → moon	[0.3]	
N → telescope	[0.3]	
V → saw	[1.0]	
P → with	[1.0]	

$$p(T_1) = 1.0 \times 0.8^3 \times 0.6 \times 0.4 \times 1.0 \times 0.5^2 \times 0.5 \times 0.4 \times 0.3 \times 0.3 \times 1.0 \times 1.0 = 0.00055296$$



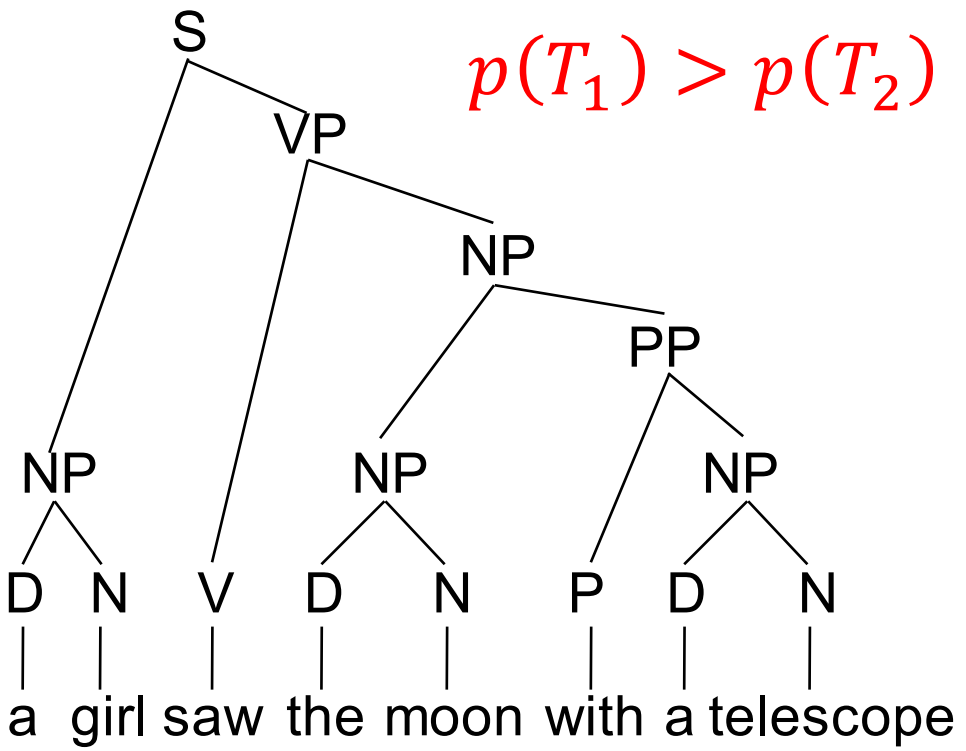
構文木2

S → NP VP	[1.0]	
NP → D N	[0.8]	× 3
NP → NP PP	[0.2]	
VP → V NP	[0.6]	
VP → VP PP	[0.4]	
PP → P NP	[1.0]	
D → a	[0.5]	× 2
D → the	[0.5]	
N → girl	[0.4]	
N → moon	[0.3]	
N → telescope	[0.3]	
V → saw	[1.0]	
P → with	[1.0]	

$p(T_1) = 1.0 \times 0.8^3 \times 0.6 \times 0.4 \times 1.0 \times 0.5^2 \times 0.5 \times 0.4 \times 0.3 \times 0.3 \times 1.0 \times 1.0 = 0.00055296$

$p(T_2) = 1.0 \times 0.8^3 \times 0.2 \times 0.6 \times 1.0 \times 0.5^2 \times 0.5 \times 0.4 \times 0.3 \times 0.3 \times 1.0 \times 1.0 = 0.00027648$

$p(T_1) > p(T_2)$



PCFG の諸問題

- 学習データを用意する

- 文法 $G = \langle N, \Sigma, R, S, p \rangle$ をデータから学習する
  - 教師付き学習：構文木の正解データから学習
  - 教師なし学習：単語列のみから学習
    → EMアルゴリズムを適用（本講義では割愛）

- 文法 $G = \langle N, \Sigma, R, S, p \rangle$ が与えられた時、入力文に対して確率が最大となる構文木を推定する

PCFGによる構文解析

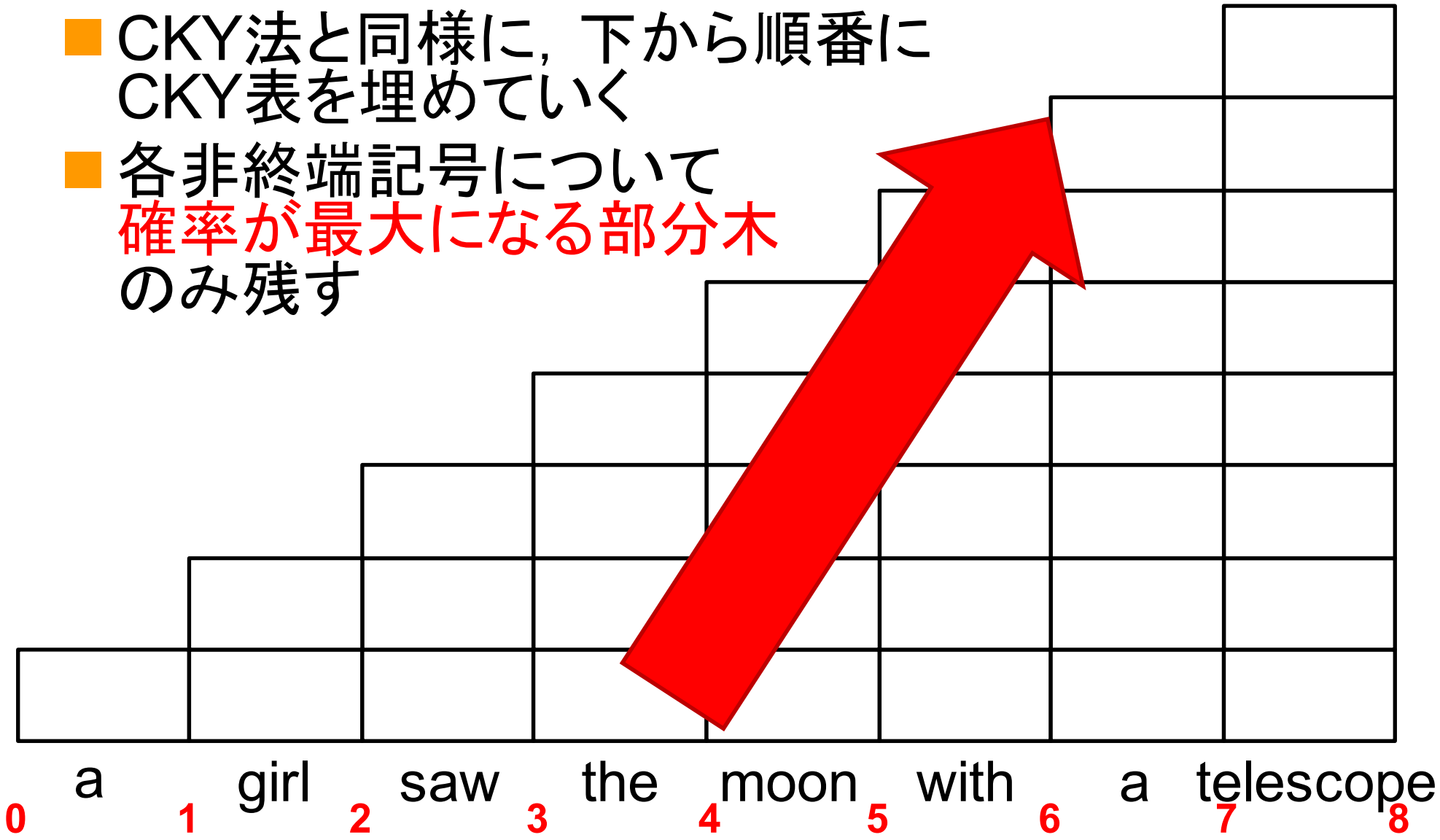
- PCFGによる構文解析＝入力文に対して、確率最大の構文木を計算する

$$T^* = \underset{T}{\operatorname{argmax}} \prod_{r \in T} p(r)$$

- すべての構文木を列挙して確率を計算
  → 指数爆発
- 動的計画法：部分構文木の確率値を表に保存しながらボトムアップに確率を計算
- CKY アルゴリズムの拡張
  - CKY表を埋める時に、確率が最大の部分構文木を求める

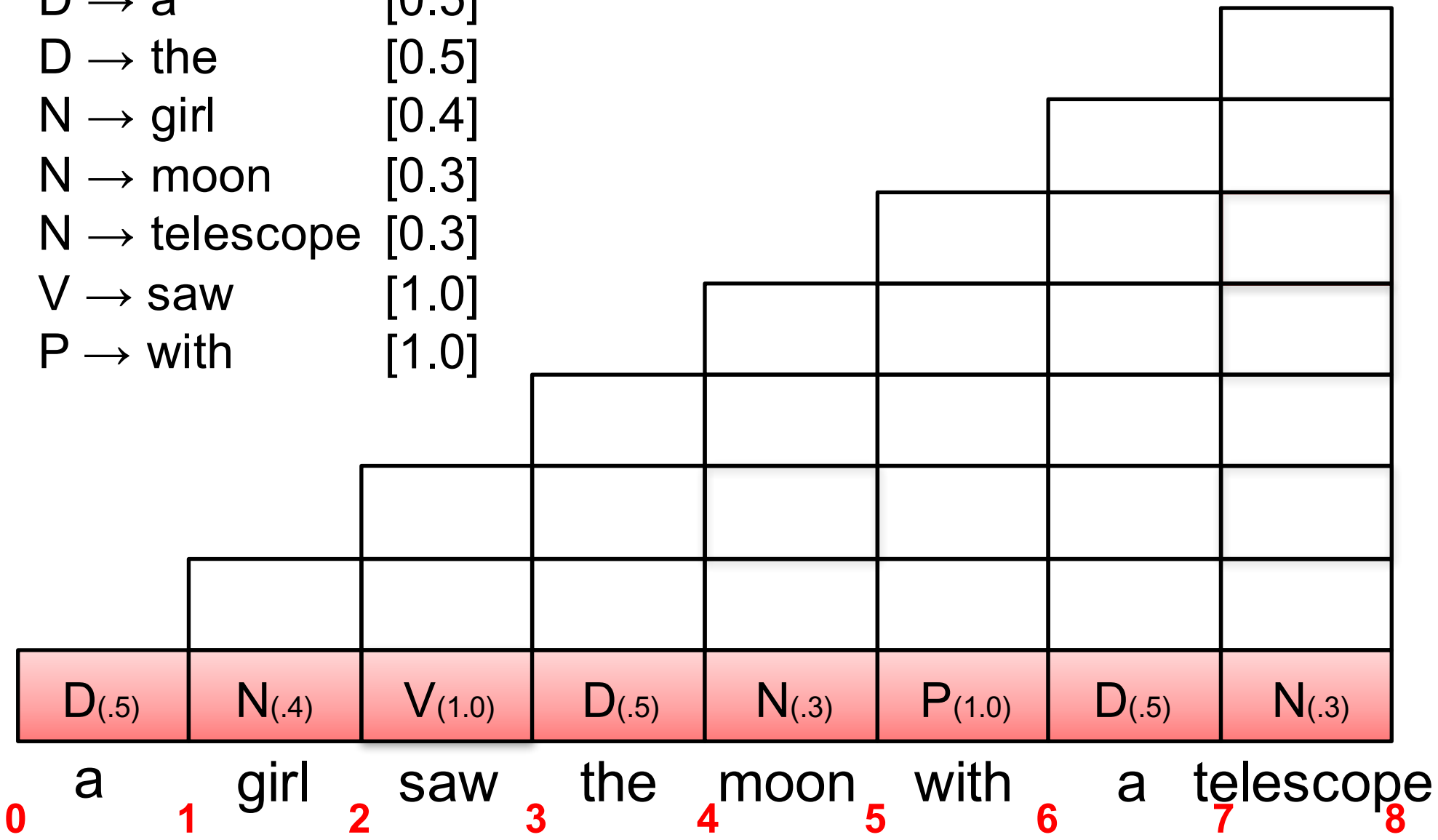
ビタビアルゴリズム

- CKY法と同様に，下から順番にCKY表を埋めていく
- 各非終端記号について確率が最大になる部分木のみ残す



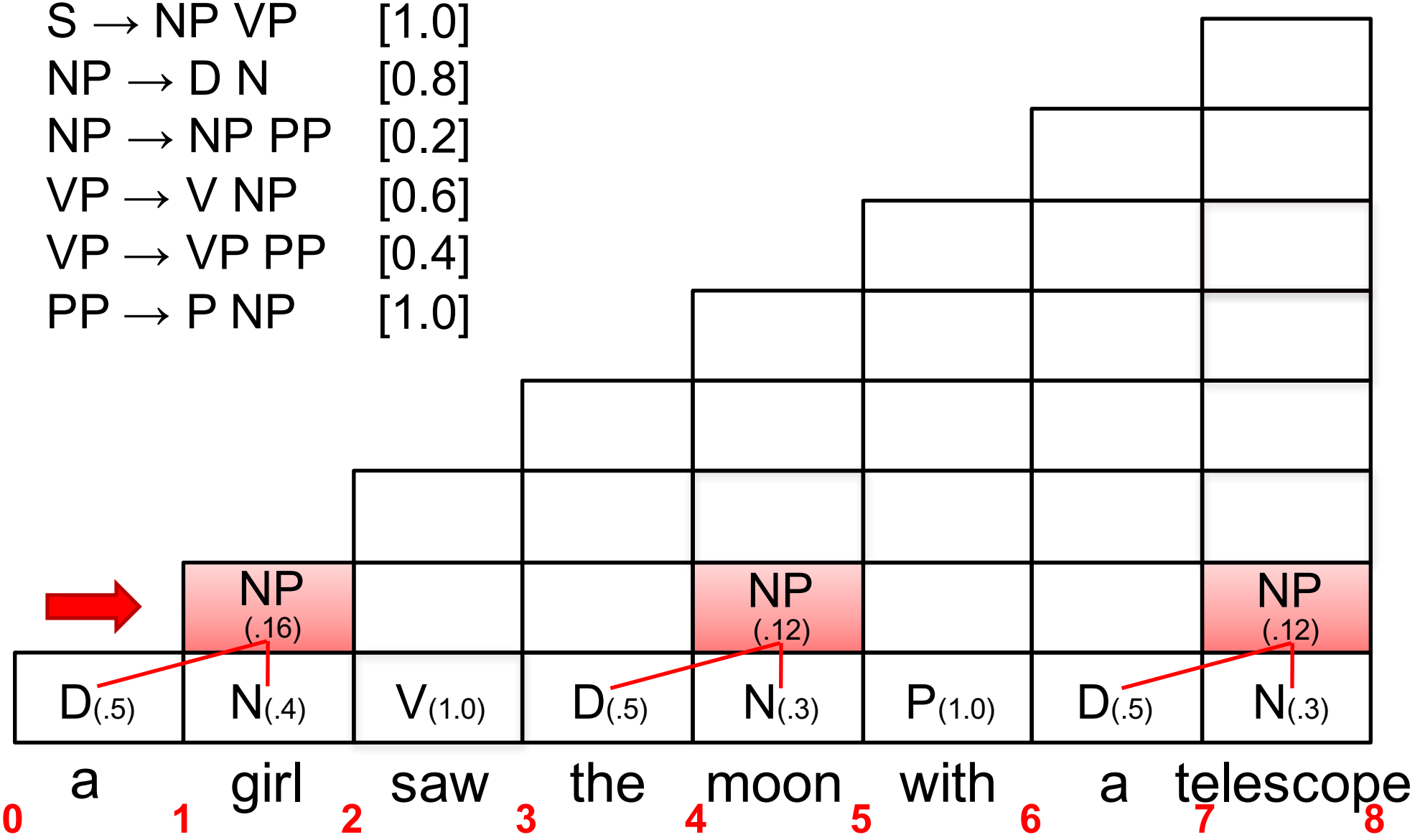
ビタビアルゴリズム

D → a [0.5]
D → the [0.5]
N → girl [0.4]
N → moon [0.3]
N → telescope [0.3]
V → saw [1.0]
P → with [1.0]



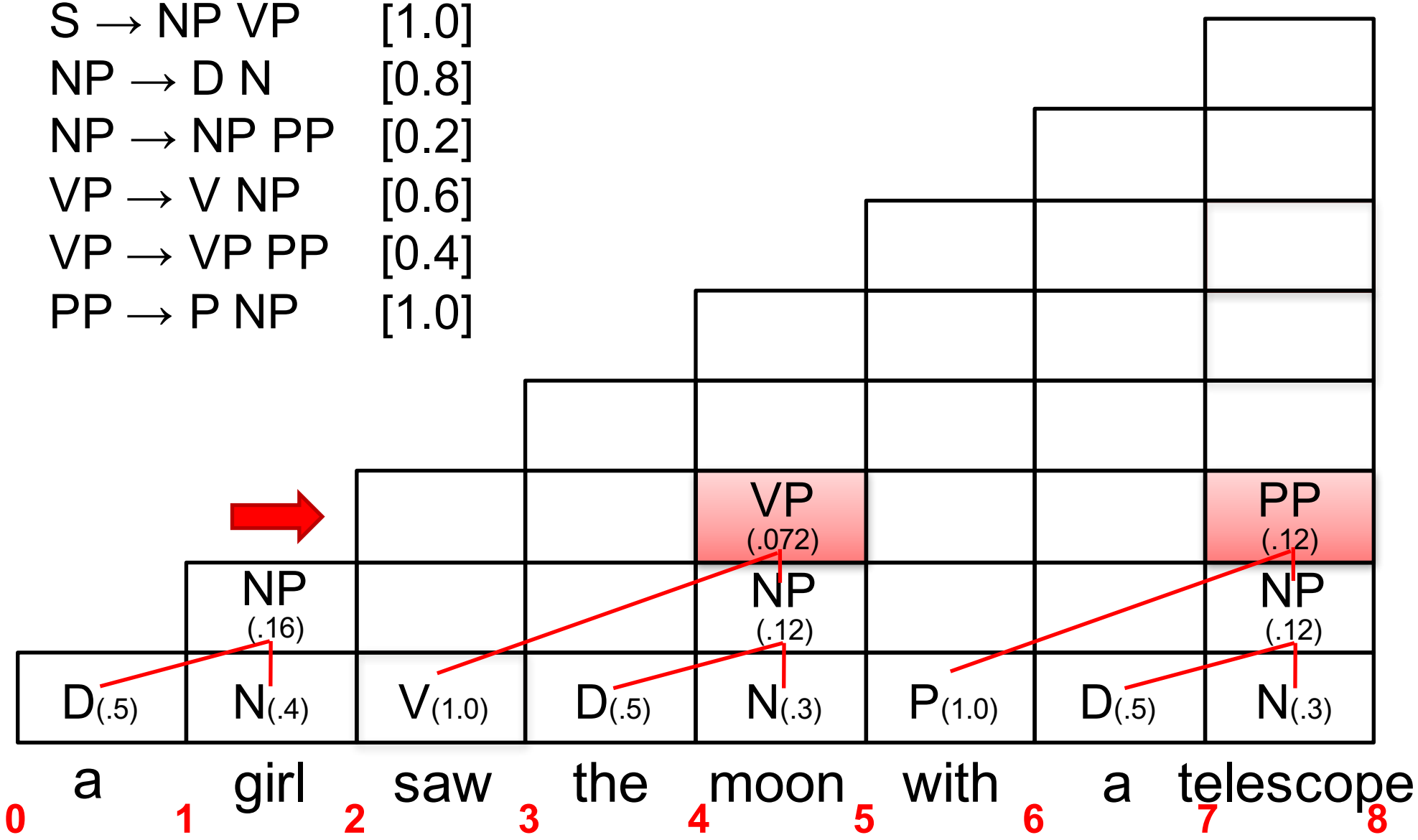
ビタビアルゴリズム

S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]

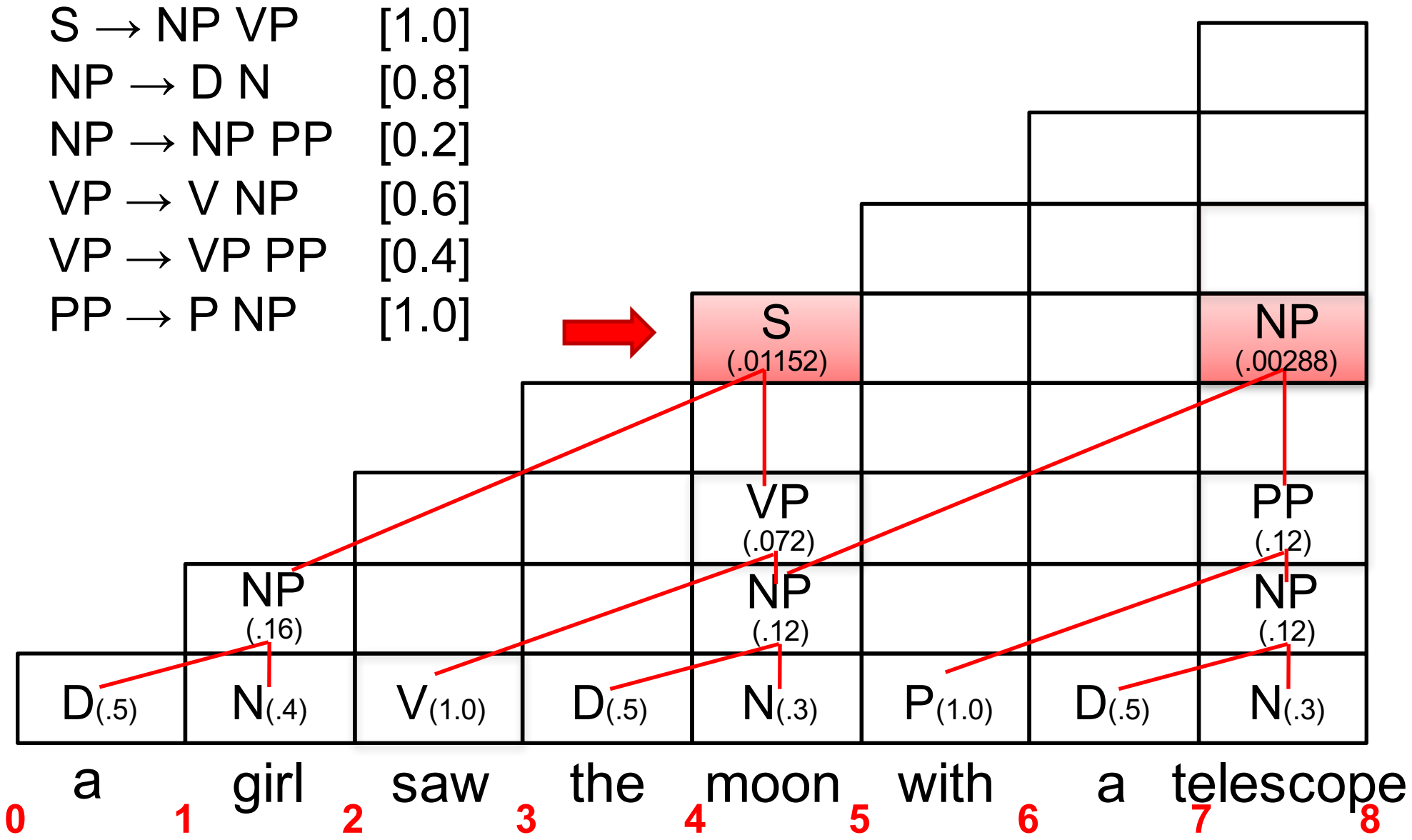


ビタビアルゴリズム

S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]



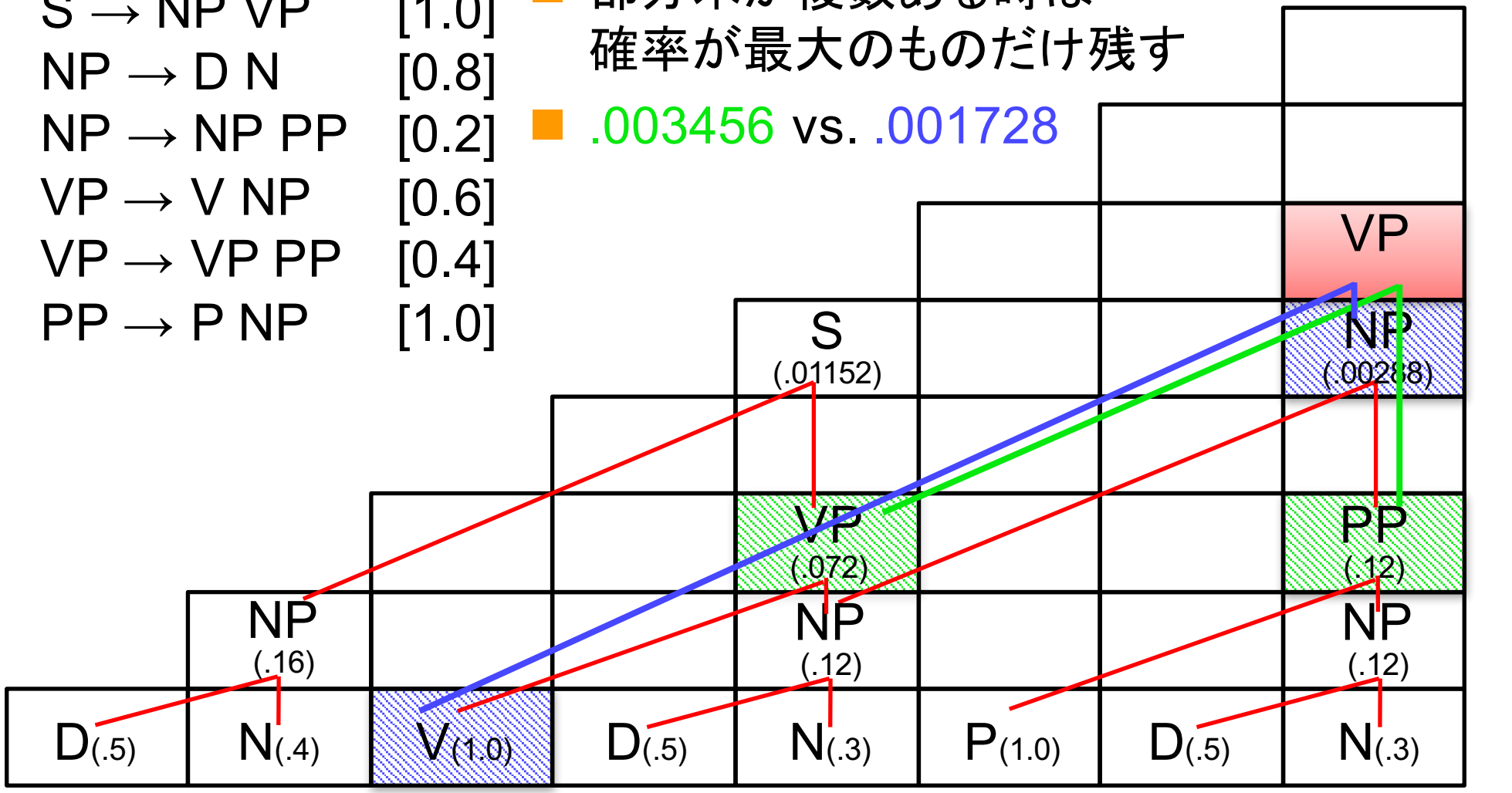
ビタビアルゴリズム


S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]
S (.01152)
NP (.00288)
VP (.072)
PP (.12)
NP (.16)
NP (.12)
NP (.12)
D(.5) N(.4) V(1.0) D(.5) N(.3) P(1.0) D(.5) N(.3)
a girl saw the moon with a telescope
0 1 2 3 4 5 6 7 8

ビタビアルゴリズム

S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]

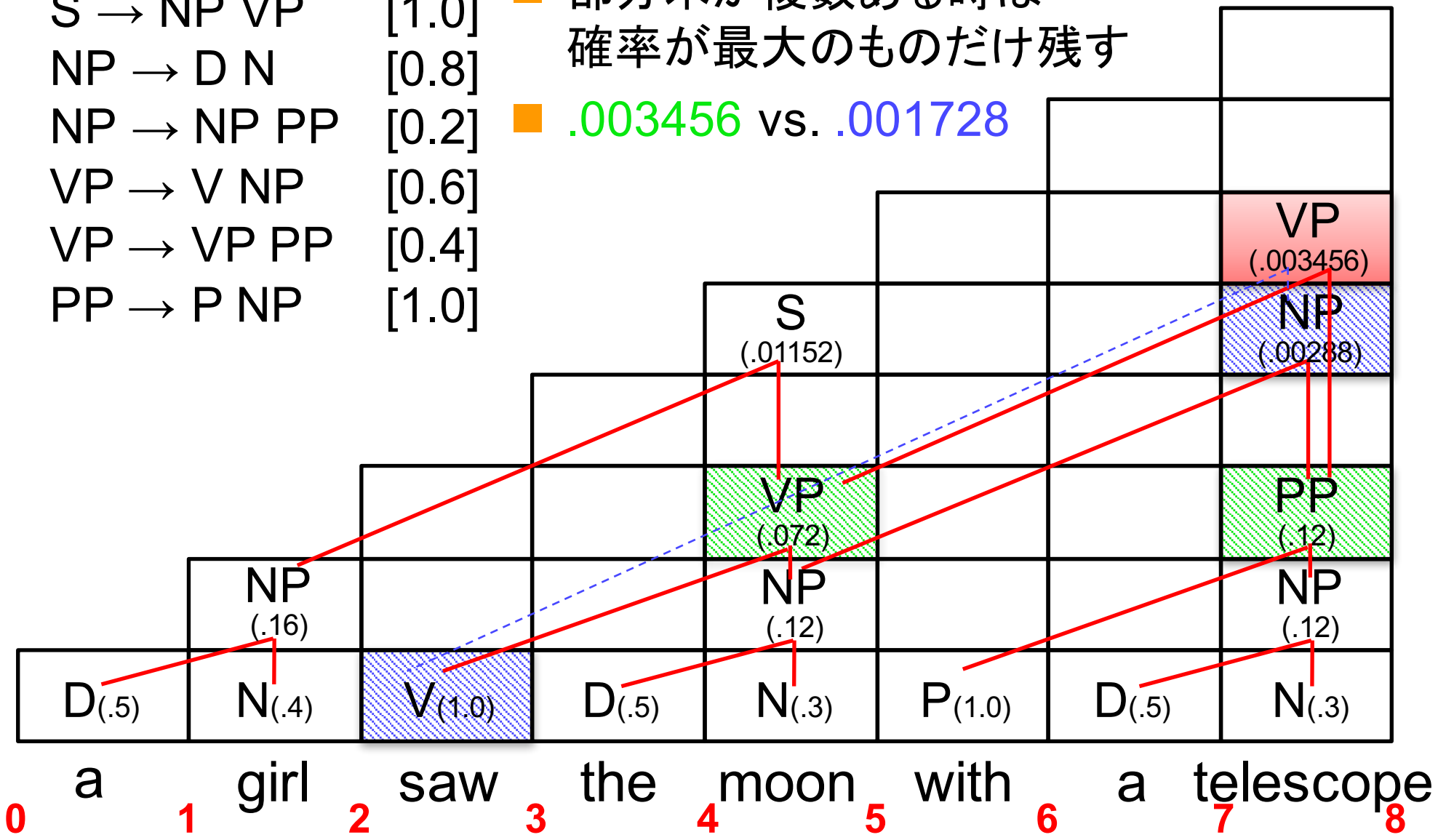
- 部分木が複数ある時は
  確率が最大のものだけ残す
- .003456 vs. .001728



ビタビアルゴリズム

S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]

- 部分木が複数ある時は
  確率が最大のものだけ残す
- .003456 vs. .001728

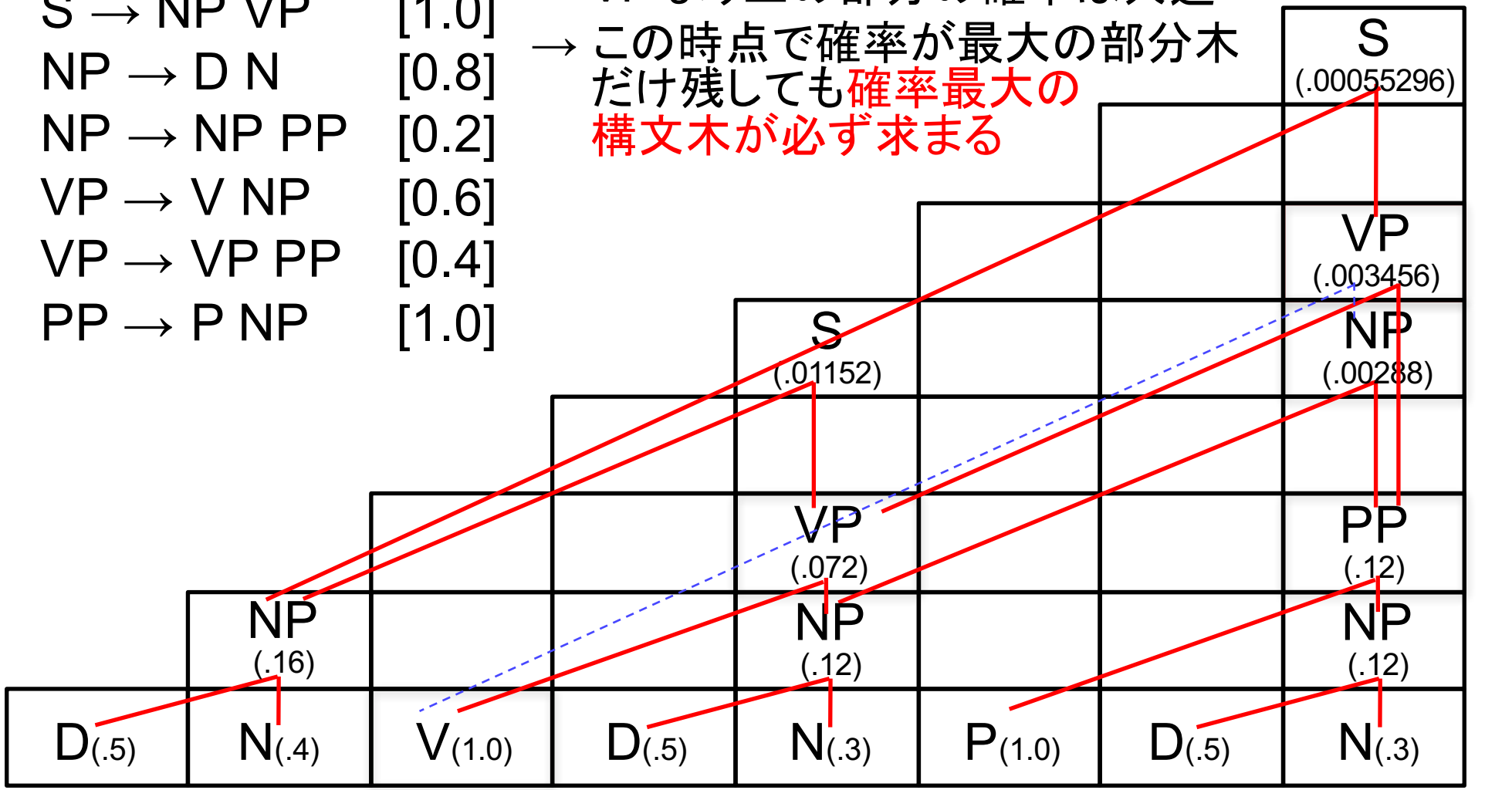


ビタビアルゴリズム

S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]

- VP より上の部分の確率は共通

→ この時点で確率が最大の部分木だけ残しても確率最大の構文木が必ず求まる



ビタビアルゴリズム

[初期化：CKY表の1段目を埋める]

for $i = 1$ to n

  for $x = 0$ to $n - i$

    $y = x + i$

    for $k = x + 1$ to $y - 1$

      for $X \in \mathrm{table}(x, k)$

        for $Y \in \mathrm{table}(k, y)$

          for $Z \in N$

            if $Z \rightarrow X\ Y \in R$ then　X　Y を生成する生成規則があったら

              $p = \mathrm{prob}(x, k, X)\mathrm{prob}(k, y, Y)p(Z \rightarrow X\ Y)$　確率値を計算

              if $p > \mathrm{prob}(x, y, Z)$ then

                $\mathrm{table}(x, y) \leftarrow \mathrm{table}(x, y) \cup \{Z\}$

                $\mathrm{prob}(x, y, Z) \leftarrow p$

                $\mathrm{back}(x, y, Z) \leftarrow (k, X, Y)$　} 確率最大の部分木だけ残す

PCFG の諸問題

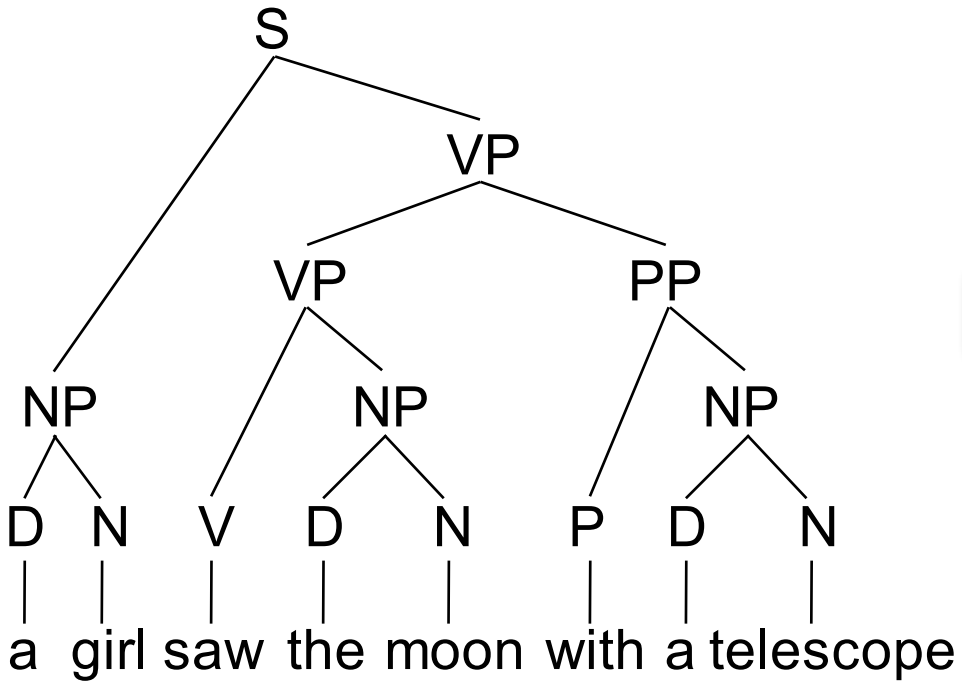
- 学習データを用意する

- 文法 $G = \langle N, \Sigma, R, S, p \rangle$ をデータから学習する
  - 教師付き学習：構文木の正解データから学習
  - 教師なし学習：単語列のみから学習
    → EMアルゴリズムを適用（本講義では割愛）

- 文法 $G = \langle N, \Sigma, R, S, p \rangle$ が与えられた時、入力文に対して確率が最大となる構文木を推定する

文法と確率の学習

- 文法を人手で作るのは難しい（確率パラメータはほぼ無理）
  → データから学習する
- 文法・確率を自分で作るのは大変だが、学習データを作ることはできる（データ主導のアプローチ）



S → NP VP [1.0]
NP → D N [0.8]
NP → NP PP [0.2]
VP → V NP [0.6]
VP → VP PP [0.4]
PP → P NP [1.0]
… …

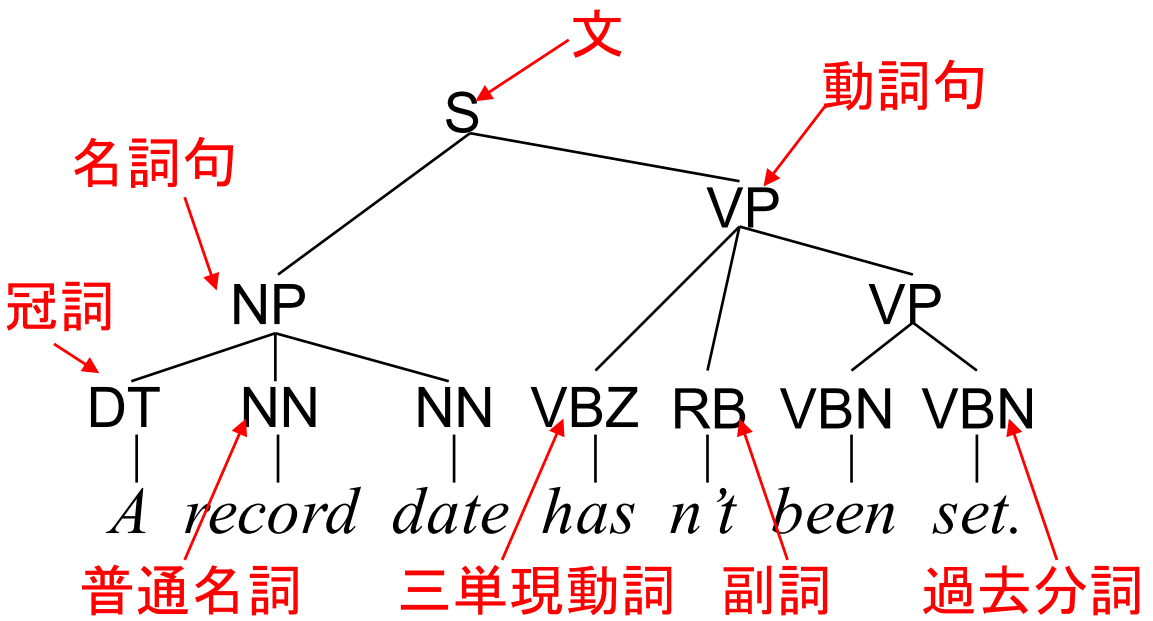
学習データ＝ツリーバンク

- 構文木の正解データ
- 英語を中心にいろいろな言語でツリーバンクが開発されている
  - Penn Treebank [Marcus et al. 1993]
  - SUSANNE [Sampson 1995]
  - TIGER Treebank [Brants et al. 2002]
  - Prague Dependency Treebank [Hajic 1998]
  - Verbmobil [Hinrichs et al. 2000]
  - Universal Dependencies [Nivre et al. 2016]
  - EDRコーパス [EDR 1995]
  - 京都大学テキストコーパス [黒橋ら 1997]
  - 日本語話し言葉コーパス(CSJ) [前川ら 2000]
  - 現代日本語書き言葉均衡コーパス(BCCWJ) [前川ら 2010]

Penn Treebank

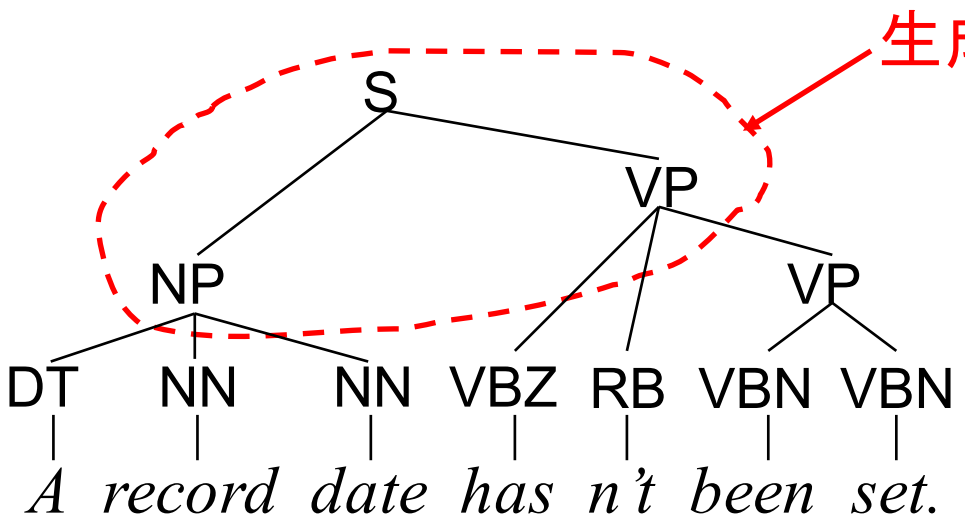
- 最初の大規模ツリーバンク [Marcus et al. 1993]
- Wall Street Journal から抽出した約5万文に品詞と構文木が付いている

```
(S (NP A/DT
       record/NN
       date/NN)
   (VP has/VBZ
       n't/RB
       (VP been/VBN
           set/VBN))
   ./.)
```



生成規則の学習

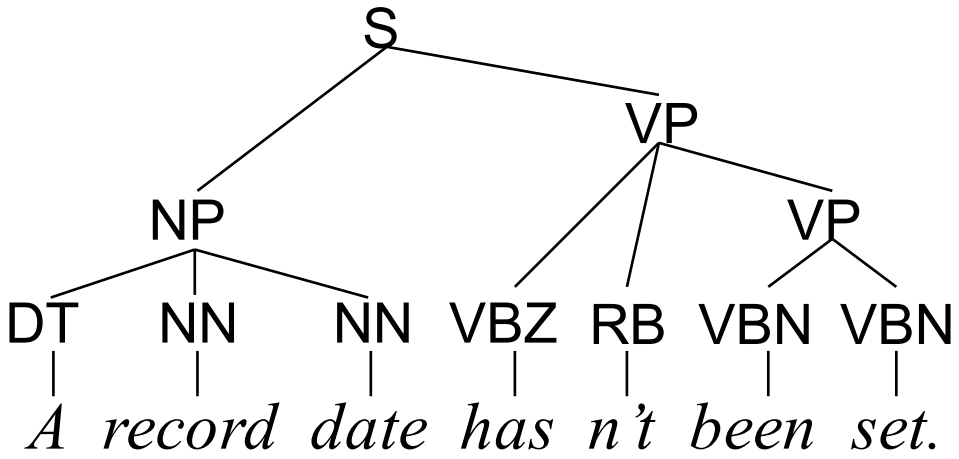
- ツリーバンクの各分岐を生成規則だと思って生成規則を抽出



S → NP VP
NP → DT NN NN
VP → VBZ RB VP
VP → VBN VBN

パラメータの学習

- ツリーバンクでの出現頻度から確率を推定（最尤推定）

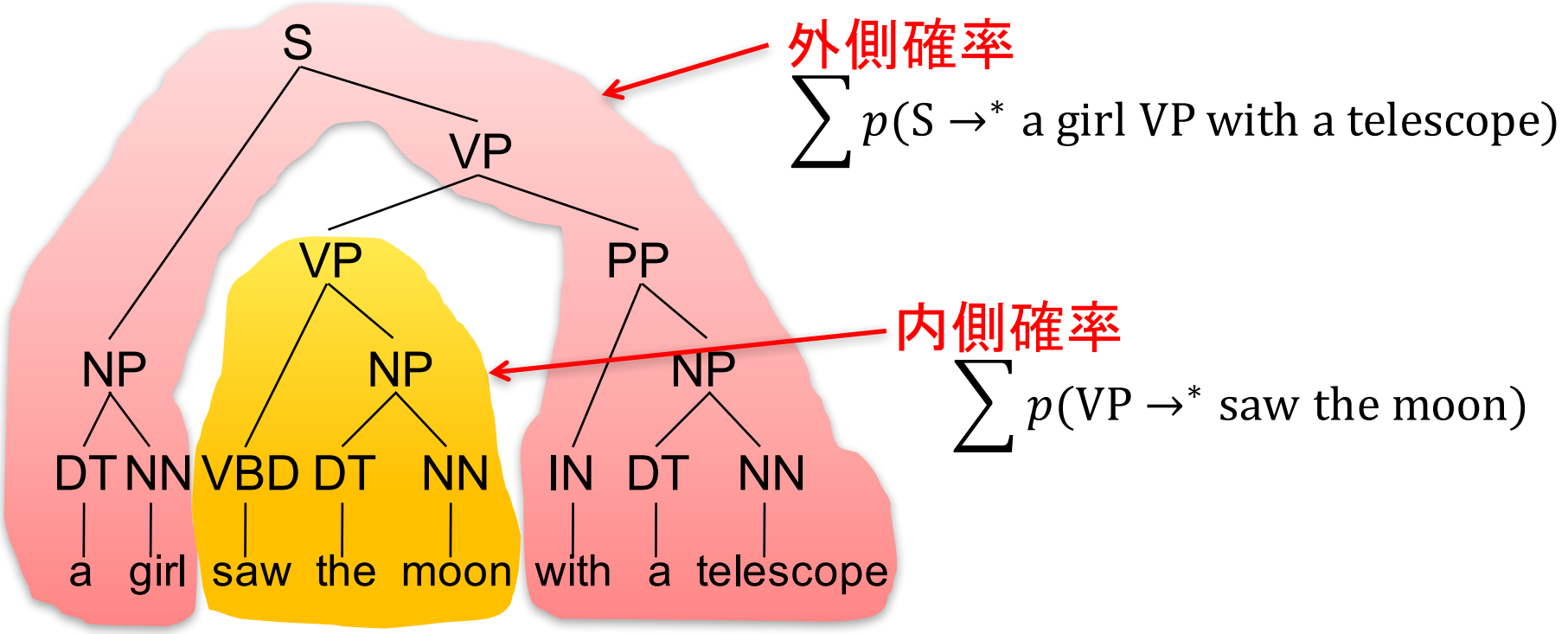


S → NP VP 0.5
NP → DT NN NN 0.03
VP → VBZ RB VP 0.02
VP → VBN VBN 0.1

$$p(lhs \rightarrow rhs) = \frac{C(lhs \rightarrow rhs)}{\sum_{rhs'} C(lhs \rightarrow rhs')}$$

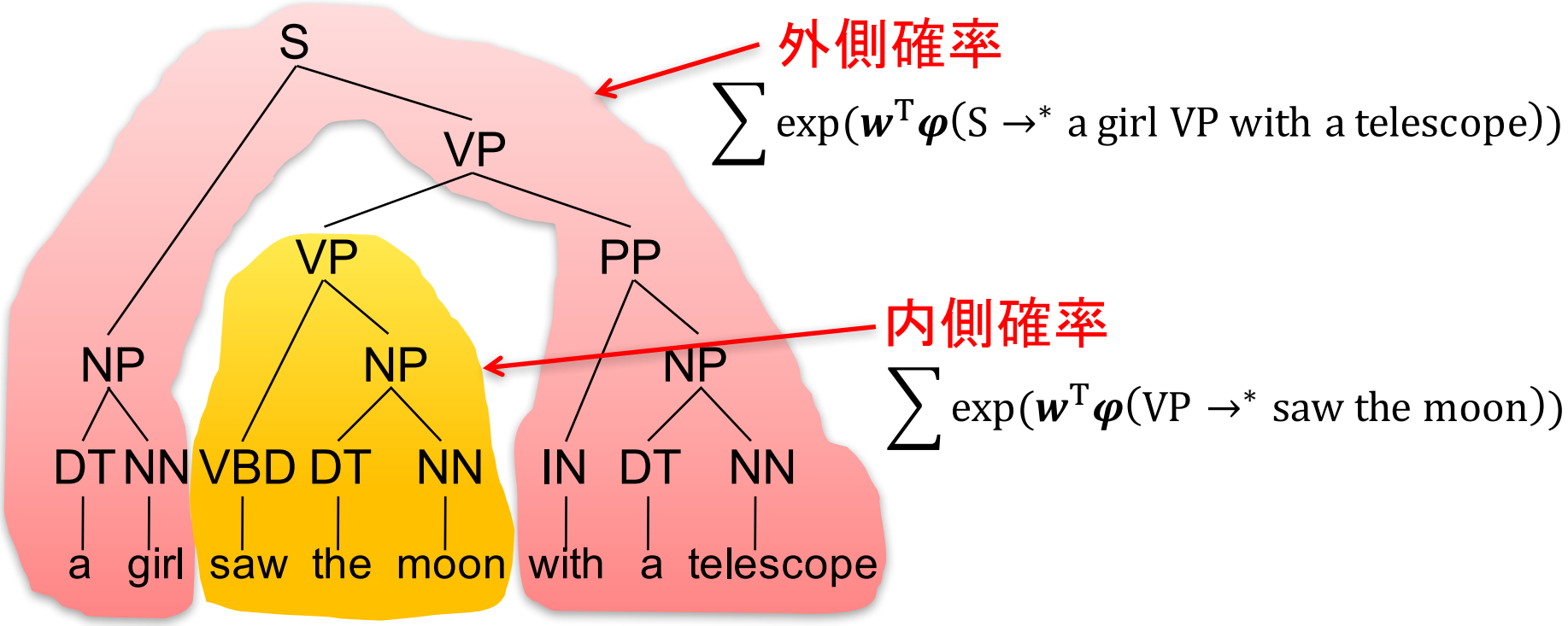
教師なし学習

- HMMのBaum-Welchアルゴリズムと同様の方法
- 内側・外側アルゴリズム：内側確率と外側確率を計算し、各生成規則の期待値を計算



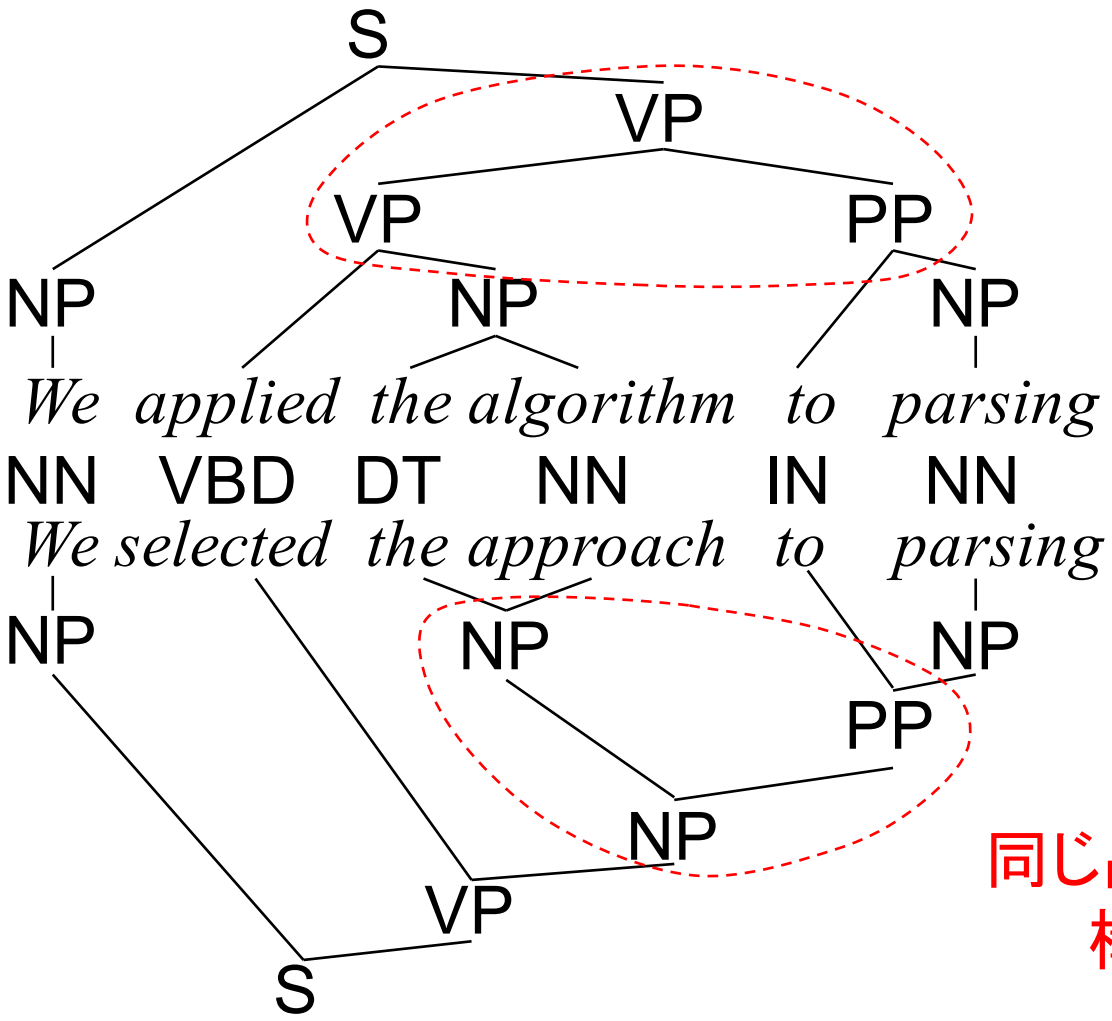
CFG+機械学習

- 同様に、CFGと機械学習を組み合わせることもできる
  - 期待値計算 → ログ線形モデル
  - argmax → パーセプトロン
  - マージン → SVM



自然言語の構文解析

- PCFGによる構文解析は精度が低い（70%～80%）



VP → VP PP

or

NP → NP PP

同じ品詞列でも単語によって
構文木の形が変わる

まとめ

- 構文解析
- 文脈自由文法 (CFG)
  - 導出、構文木
  - CKY法
- 確率文脈自由文法 (PCFG)
  - ビタビアルゴリズム
  - 教師付き学習
  - 教師なし学習