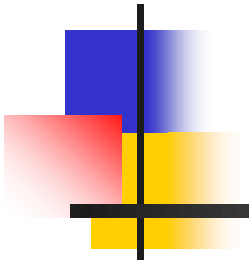
第9章 句法分析

(1/2)

北京市海淀区中关村东路95号
邮编：100190

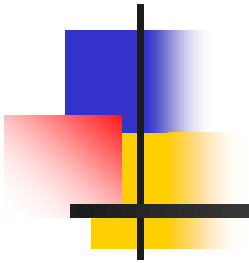
电话： +86-10-8254 4688
邮件：cqzong@nlpr.ia.ac.cn



9.1 概 述

9.1 概述

- **任务**：句法分析(syntactic parsing)的任务就是识别句子的句法结构(syntactic structure)。
- **类型**：
  - 短语结构分析 (Phrase parsing)
    - 完全句法分析 (Full parsing)
    - 局部句法分析 (Partial parsing)
  - 依存句法分析 (Dependency parsing)



9.2 短语结构分析

9.2 短语结构分析

◆ 句法分析的例子（参见前面第4章）

他还提出一系列具体措施的政策要点。

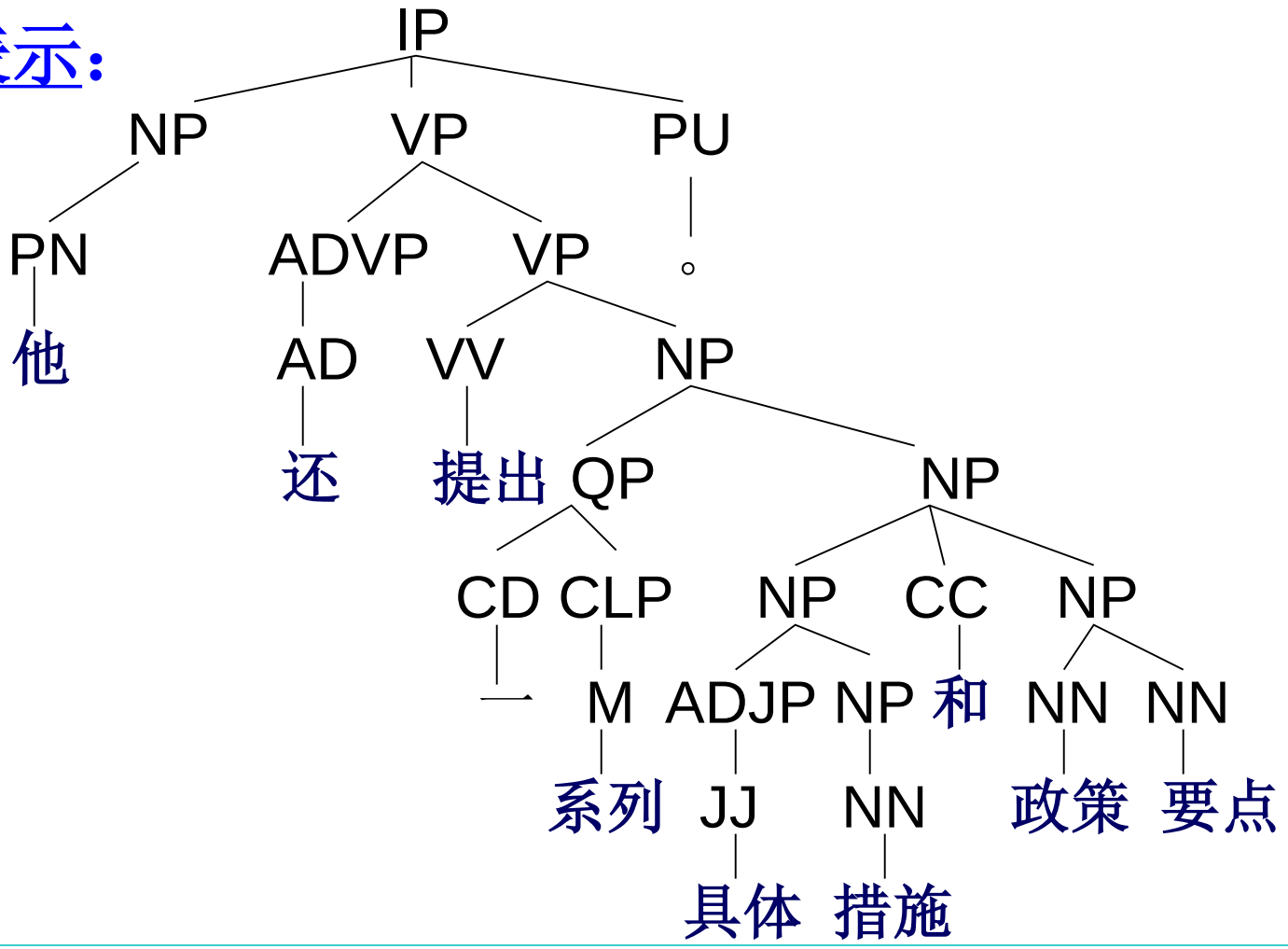
他/PN 还/AD 提出/VV 一/CD 系列/M 具体/JJ 措施/NN 和/CC 政策/NN 要点/NN 。/PU

9.2 短语结构分析

(IP (NP-SBJ (PN 他))
(VP (ADVP (AD 还))
(VP (VV 提出))
(NP-OBJ(QP (CD 一)
(CLP (M 系列)))
(NP (NP(ADJP (JJ 具体)
(NP (NN 措施)))
(CC 和)
(NP (NN 政策)
(NN 要点))))))
(PU 。))

9.2 短语结构分析

树状表示：



9.2 短语结构分析

- **目标**：实现高正确率、高鲁棒性(robustness)、高速度的自动句法分析过程。
- **困难**：自然语言中存在大量的复杂的结构歧义 (structural ambiguity)。

9.2 短语结构分析

◆结构歧义

例如：(1) I saw a boy in the park.

[I saw a boy] in the park.

I saw a [boy in the park].

(2) I saw a boy in the park with a telescope.

(3) I saw a boy swimming on the bridge.

(4) 关于鲁迅的文章。

(5) 把重要的书籍和手稿带走了。

9.2 短语结构分析

英语中的结构歧义随介词短语组合个数的增加而不断加深的，这个组合个数我们称之为开塔兰数(Catalan number，记作C_N)。

如果句子中存在这样 n (n为自然数)个介词短语，C_N 可由下式获得 [Samuelsson, 2000]：

$$C_N = \binom{2n}{n} \frac{1}{n+1} = \frac{(2n)!}{(n!)^2 (n+1)}$$

9.2 短语结构分析

- 基本方法和开源的句法分析器：
  - 基于CFG规则的分析方法：
    - 线图分析法 (chart parsing)
    - CYK 算法
    - Earley (厄尔利)算法
    - LR 算法 / Tomita 算法 … …
      - Top-down: Depth-first/ Breadth-first
      - Bottom-up

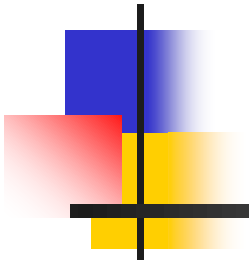
9.2 短语结构分析

- 基于 PCFG 的分析方法

  PCFG: Probabilistic Context-Free Grammar

  (有时也写作 Stochastic CFG, SCFG)

- 其他统计模型
- 部分开源的句法分析器



9.3 线图分析法

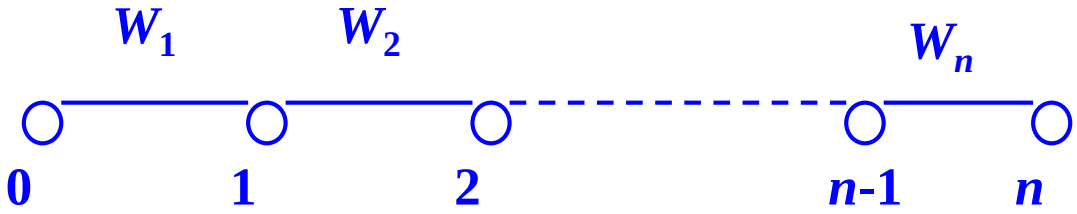
9.3 线图分析法

- 三种策略
  - 自底向上 (Bottom-up)
  - 从上到下 (Top-down)
  - 从上到下和从下到上结合

9.3 线图分析法

◆ 自底向上的 Chart 分析算法

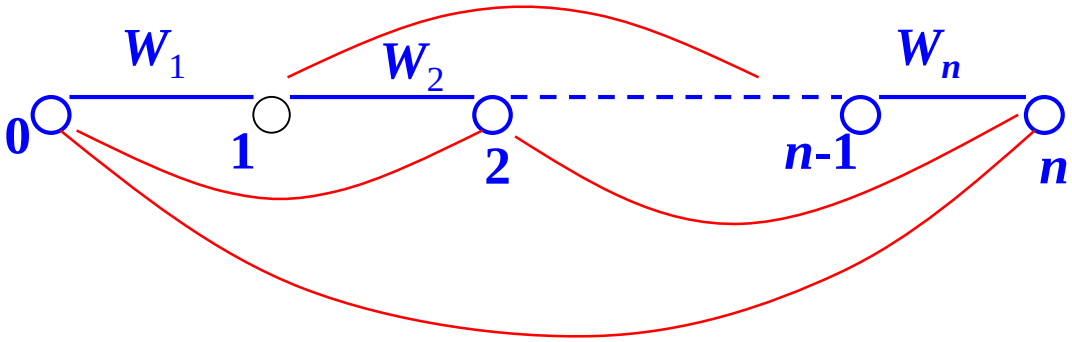
- 给定一组 **CFG** 规则：$XP \rightarrow \alpha_1 \ldots \alpha_n \quad (n \geq 1)$
- 给定一个句子的词性序列：$S = W_1 W_2 \cdots W_n$
- 构造一个线图：一组结点和边的集合；



- 建立一个二维表：记录每一条边的起始位置和终止位置。

9.3 线图分析法

执行操作：查看任意相邻几条边上的词性串是否与某条重写规则的右部相同，如果相同，则增加一条新的边跨越原来相应的边，新增加边上的标记为这条重写规则的头(左部)。重复这个过程，直到没有新的边产生。



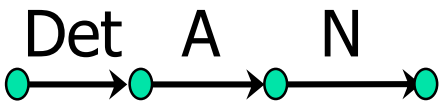
9.3 线图分析法

点规则：用于表示规则右部被归约(reduce)的程度。

设有规则：NP → Det A N

NP → Det N　　NP → A N

句子：The good book



NP → Det ∘ A N

NP → Det ∘ N

NP → Det A ∘ N

NP → Det A N ∘

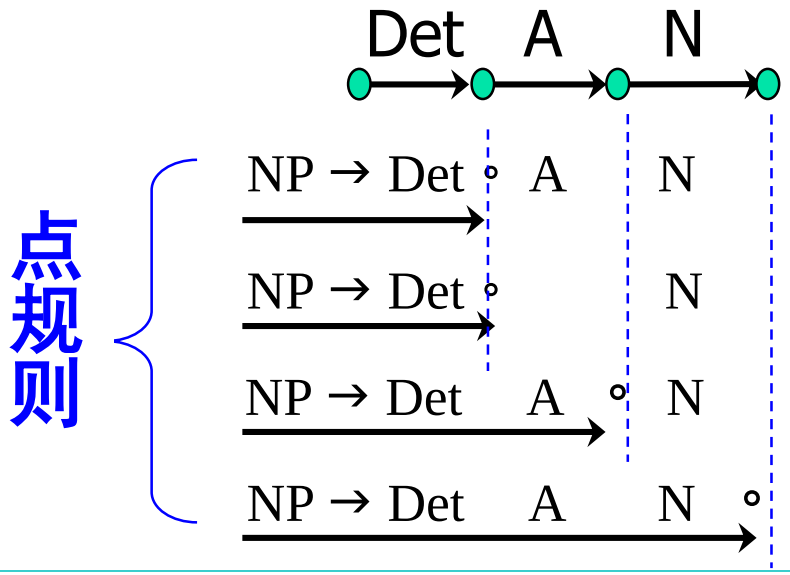
9.3 线图分析法

点规则：用于表示规则右部被归约(reduce)的程度。

设有规则：NP → Det A N

NP → Det N　　NP → A N

句子：The good book



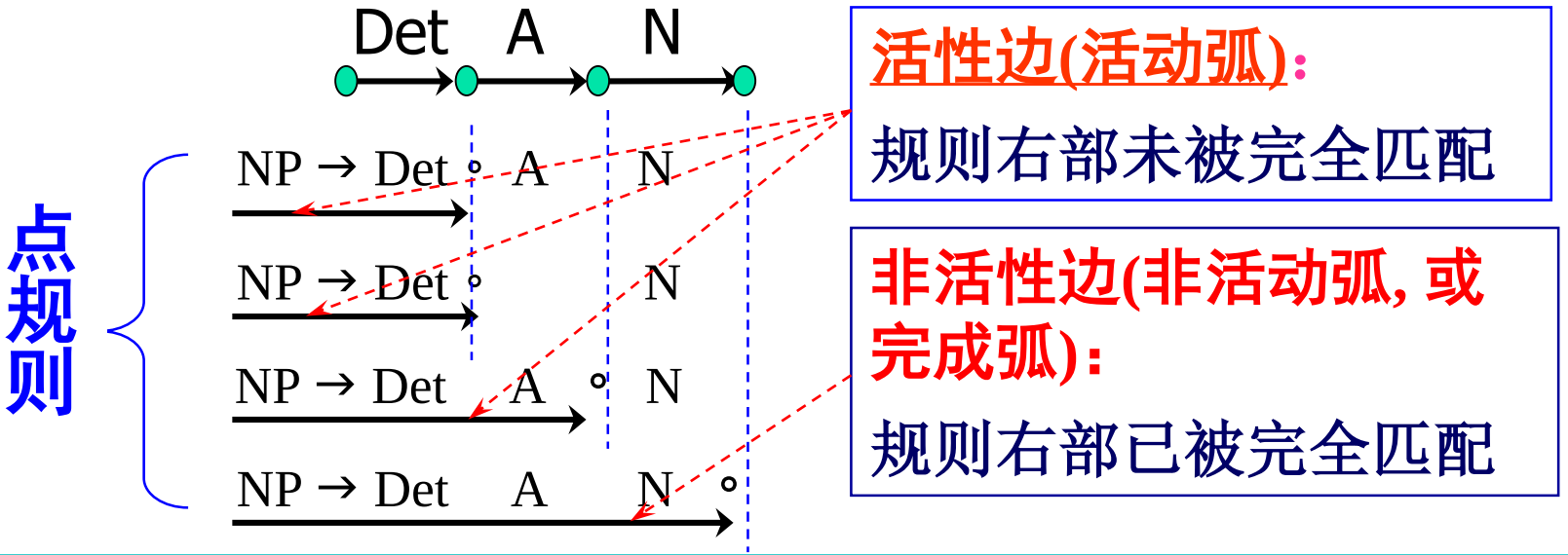
9.3 线图分析法

点规则：用于表示规则右部被归约(reduce)的程度。

设有规则：NP → Det A N

NP → Det N　　NP → A N

句子：The good book



9.3 线图分析法

例：G (S): S → NP VP, NP → Det N

VP → V NP, VP → VP PP

PP → Prep NP

输入句子： the boy hits the dog with a rod

9.3 线图分析法

例：G (S): S → NP VP, NP → Det N

VP → V NP, VP → VP PP

PP → Prep NP

输入句子： the boy hits the dog with a rod

①形态分析：the boy hit the dog with a rod

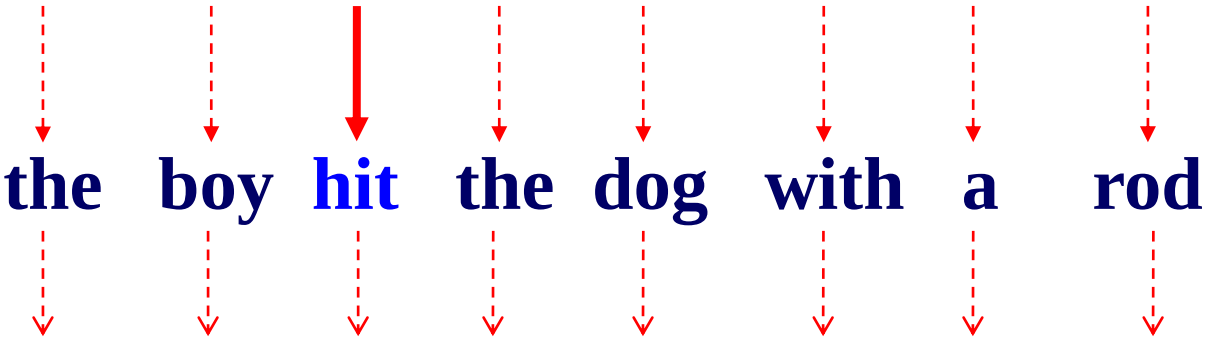
9.3 线图分析法

例：G (S): S → NP VP, NP → Det N

VP → V NP, VP → VP PP

PP → Prep NP

输入句子： the boy hits the dog with a rod



①形态分析： the boy hit the dog with a rod

②词性标注： Det N V Det N Prep Det N

9.3 线图分析法

③句法分析

Agenda　　ActiveArc　　Chart　　Acts

❶ Det (1, 2)　❷ NP → Det · N (1,2)　❸ Det (1, 2)　返回

(1) S → NP VP　　(2) NP → Det N　　(3) VP → VP PP
(4) VP → V NP　　(5) PP → Prep NP

9.3 线图分析法

③句法分析

Agenda　　ActiveArc　　Chart　　Acts

❶ Det (1, 2)　❷ NP → Det · N (1,2)　❸ Det (1, 2)　返回

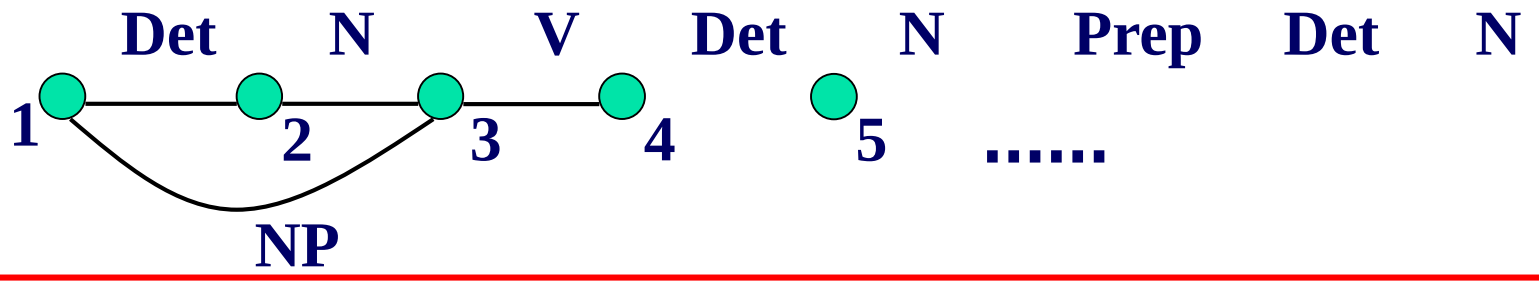
❹ N (2, 3)　无新的活动边加入　❺ N (2, 3)　扩展

(1) S → NP VP　(2) NP → Det N　(3) VP → VP PP
(4) VP → V NP　(5) PP → Prep NP

9.3 线图分析法

③句法分析

Agenda	ActiveArc	Chart	Acts
❶ Det (1, 2)	❷ NP → Det · N (1,2)	❸ Det (1, 2)	返回
❹ N (2, 3)	❻ NP → Det N · (1,3)	❺ N (2, 3)	扩展
❼ NP (1, 3)	❽ S → NP · VP (1, 3)	❾ NP (1, 3)	返回
❿ V (3, 4)	⓫ VP → V · NP (3, 4)	⓬ V (3, 4)	返回

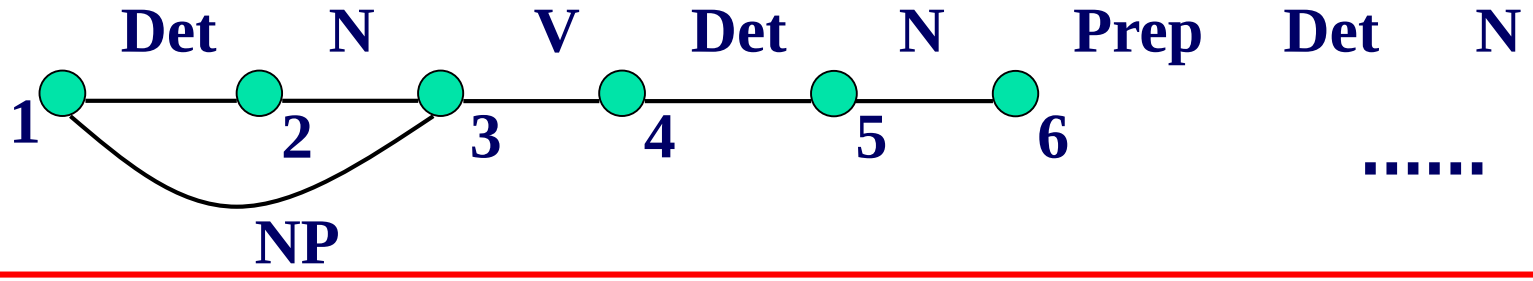


(1) S → NP VP　　(2) NP → Det N　　(3) VP → VP PP
(4) VP → V NP　　(5) PP → Prep NP

9.3 线图分析法

③句法分析

Agenda	ActiveArc	Chart	Acts
13 Det (4,5)	14 NP → Det · N (4,5)	15 Det (4,5)	返回
16 N (5,6)	无新的活动边加入	17 N (5,6)	扩展



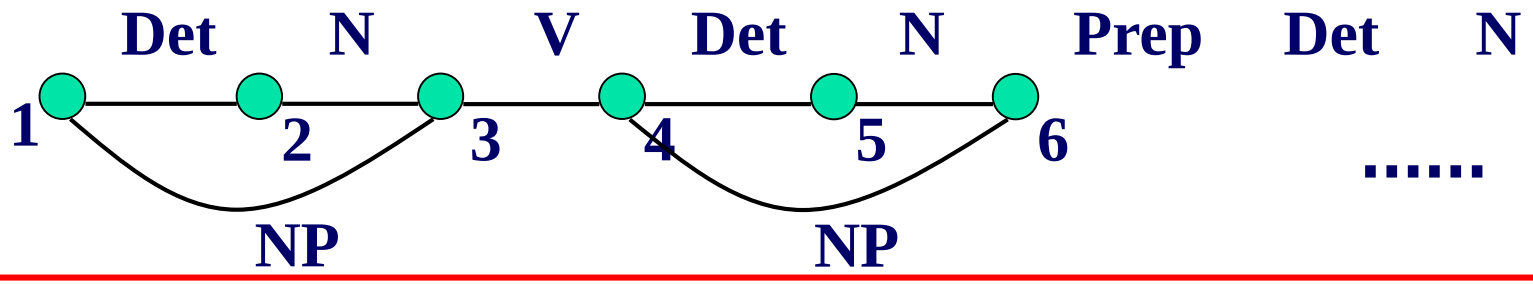
(1) S → NP VP　　(2) NP → Det N　　(3) VP → VP PP
(4) VP → V NP　　(5) PP → Prep NP

9.3 线图分析法

③句法分析

Agenda	ActiveArc	Chart	Acts
13 Det (4,5)	14 NP → Det · N (4,5)	15 Det (4,5)	返回
16 N (5,6)	18 NP → Det N · (4,6)	17 N (5,6)	扩展
19 NP(4,6)	20 S → NP · VP (4, 6)	21 NP(4,6)	扩展

将第11步的点规则 VP → V · NP (3, 4) 扩展



(1) S → NP VP　(2) NP → Det N　(3) VP → VP PP
(4) VP → V NP　(5) PP → Prep NP

9.3 线图分析法

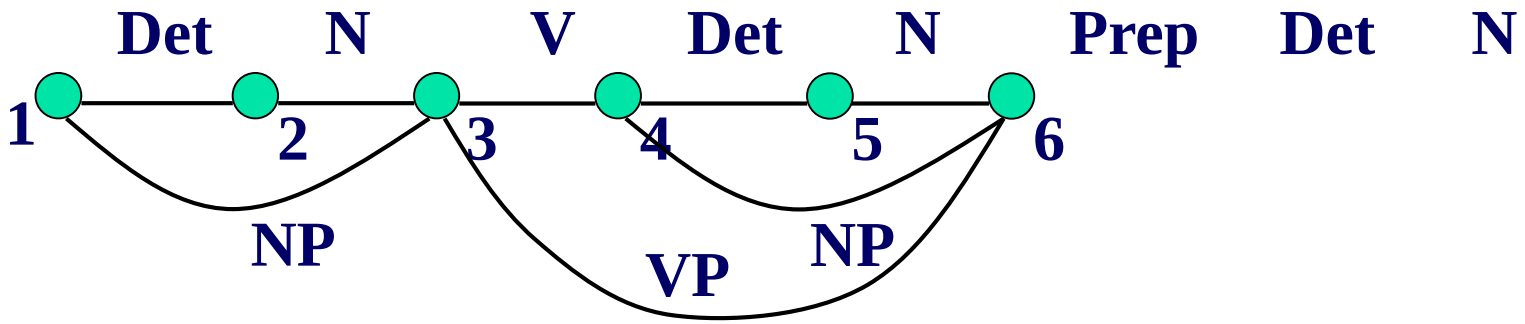
③句法分析

Agenda　　ActiveArc　　Chart　　Acts

22 VP → V NP ∘ (3, 6)

23 VP (3,6)　　24 VP → VP ∘ PP (3,6)　　25 VP (3,6)　　扩展

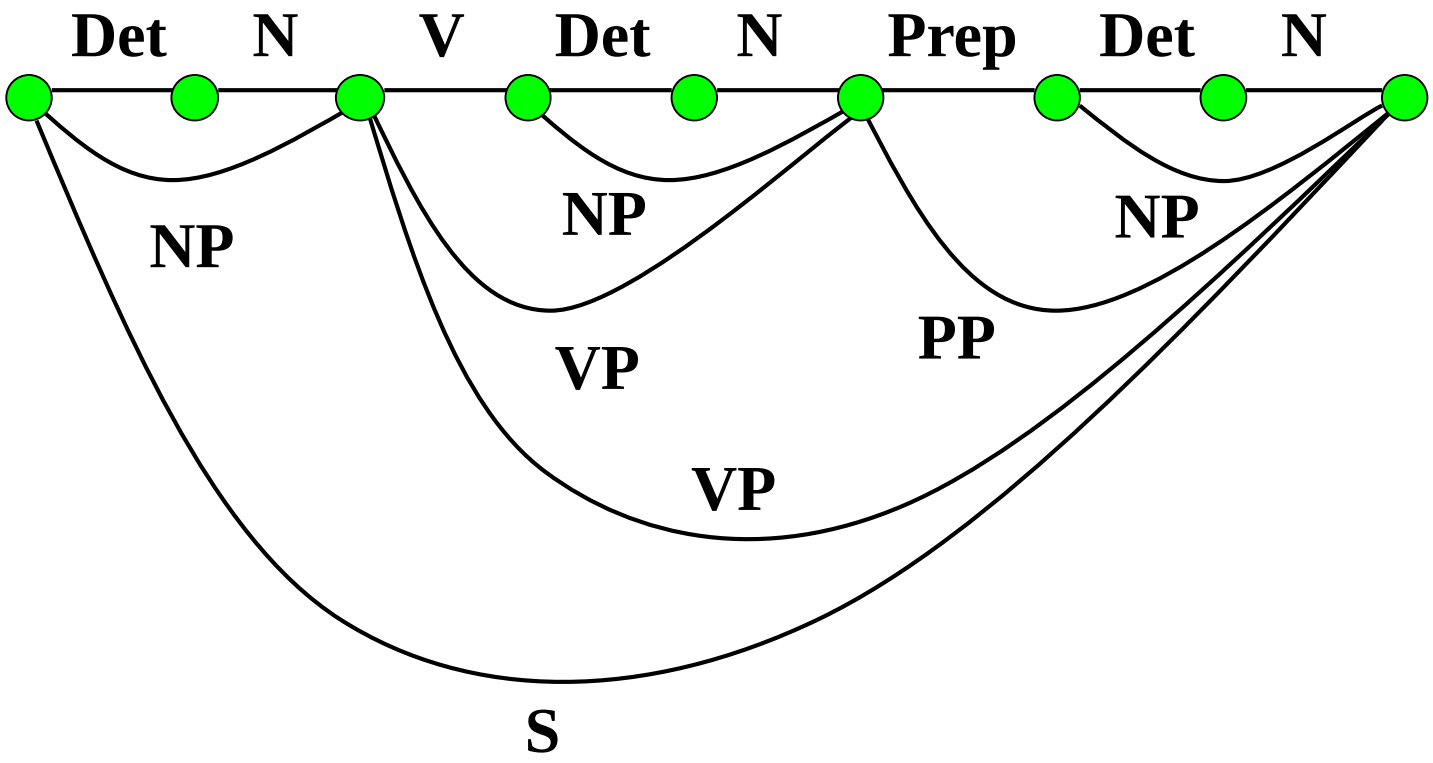
…… ……



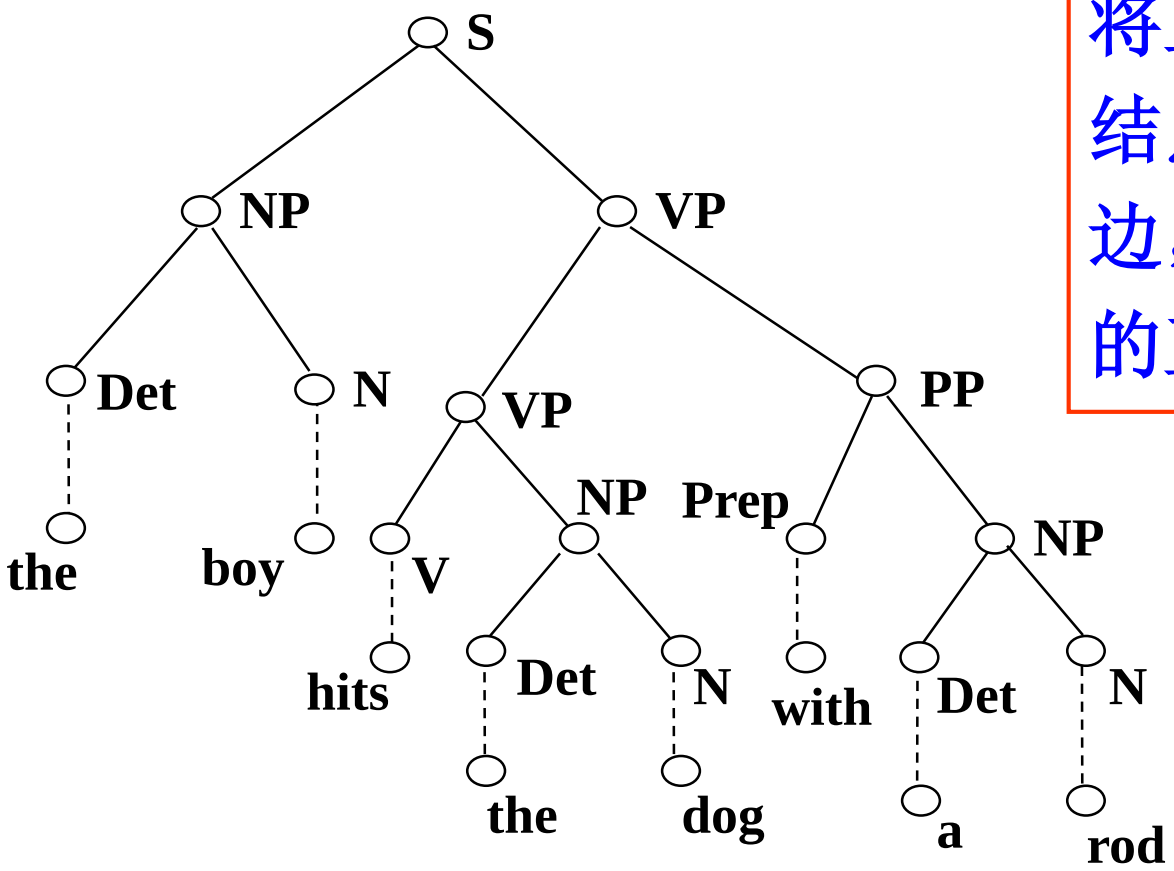
(1) S → NP VP　　(2) NP → Det N　　(3) VP → VP PP
(4) VP → V NP　　(5) PP → Prep NP

9.3 线图分析法

最后分析结果：



9.3 线图分析法



将上图中的边改为结点，将结点改为边，得到分析结果的直观图。

9.3 线图分析法

◆ 数据结构

- **线图(Chart)**：保存分析过程中已经建立的成分(包括终结符和非终结符)、位置(包括起点和终点)。通常以 $n \times n$ 的数组表示(n 为句子包含的词数)。
- **代理表(待处理表)(Agenda)**：记录刚刚得到的一些重写规则所代表的成分，这些重写规则的右端符号串与输入词性串(或短语标志串)中的一段完全匹配，通常以栈或线性队列表示。
- **活动边集(ActiveArc)**：记录那些右端符号串与输入串的某一段相匹配，但还未完全匹配的重写规则，通常以数组或列表存储。

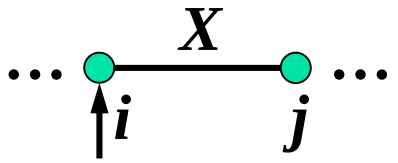
9.3 线图分析法

◆算法描述：

从输入串的起始位置到最后位置，循环执行如下步骤：

(1) 如果待处理表(Agenda)为空，则找到下一个位置上的词，将该词对应的(所有)词类X 附以 (i, j) 作为元素放到待处理表中，即$X\ (i, j)$。其中，i, j 分别是该词的起始位置和终止位置，$j > i$，$j - i$ 为该词的长度。

(2) 从 Agenda 中取出一个元素$X(i, j)$。



(3) 对于每条规则 $A \rightarrow X\gamma$，将 $A \rightarrow X \circ \gamma$ (i, j) 加入活动边集ActiveArc 中，然后调用 扩展弧子程序。

9.3 线图分析法

✧ **扩展弧子程序：**

(*a*) 将 ***X*** 插入图表(Chart)的 **(*i*, *j*)** 位置中。

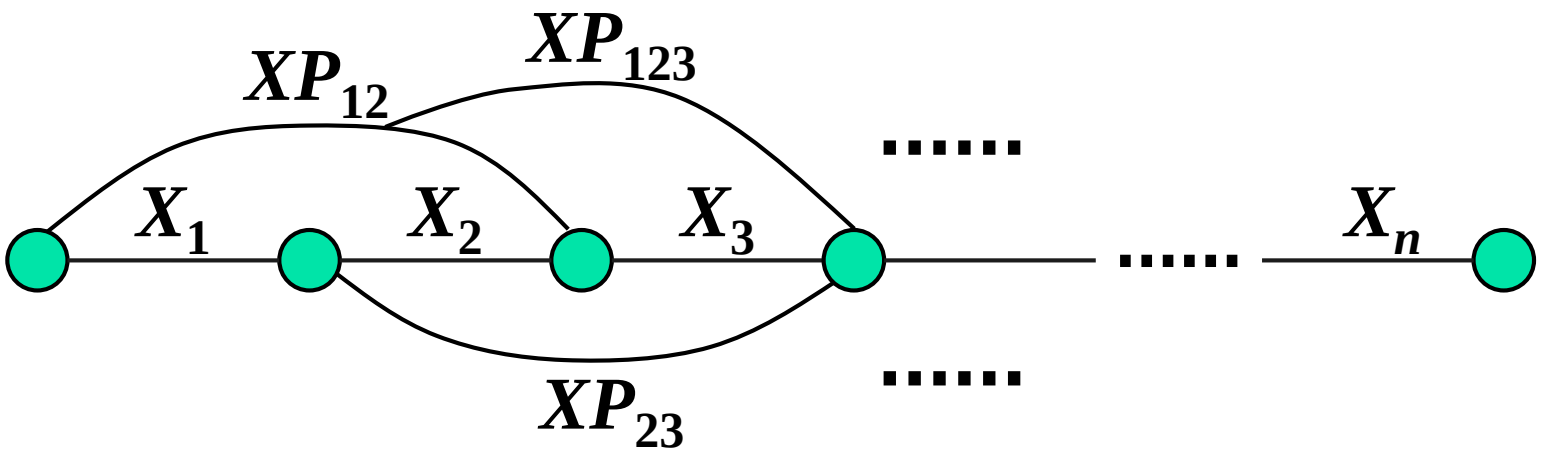
(*b*) 对于活动边集(ActiveArc)中每个位置为**(*k*, *i*) (1≤*k*<*i*)** 的点规则，如果该规则具有如下形式：$A \to \alpha \circ X$, 如果***A*=*S*,** 则把 ***S*(1, *n*+1)** 加入到 **Chart** 中，并给出一个完整的分析结果；否则，则将 ***A*(*k*, *j*)** 加入到**Agenda**表中。

(*c*) 对于每个位置为**(*k*, *i*)** 的点规则：$A \to \alpha \circ X\beta$，则将 $A \to \alpha X \circ \beta\,(k, j\,)$ 加入到活动边集中。

9.3 线图分析法

◆算法的时间复杂度分析

设 n为输入句子的长度，C 为上下文无关文法中的非终结符的数目，S 为点规则的状态数目(大于 CFG 规则的数目)，显然 $S > C$。因为 Agenda 表中的元素形式为 $X(i, j)$，因此, **Agenda** 表中最大的元素个数为：Cn^2。



9.3 线图分析法

由于**ActiveArc** 中的元素形式为：$A \rightarrow \alpha \circ X(i, j)$，所以**ActiveArc** 表中最大的元素数目为：Sn^2.

{Chart 表中的边的形式为：$A(i, j)$，因此，**Chart** 表中最大的元素数目为：Cn^2。**}**

我们来考察算法中每一步执行的最大次数：

9.3 线图分析法

◆算法描述：

从输入串的起始位置到最后位置，循环执行如下步骤：

(1) 如果待处理表(Agenda)为空，则找到下一个位置上的词，将该词对应的(所有)词类X 附以 (i, j) 作为元素放到待处理表Agenda中，其中i和j分别为该词的起始位置和终止位置，$j>i$，$j-i$为该词的长度。

最多执行的次数为：C

(2) 从 Agenda 中

最多执行的次数为：1

(3) 对于每条规则 $A \rightarrow X\gamma$，将 $A \rightarrow X \circ \gamma$ (i, j) 加入活动边集ActiveArc

最多执行的次数为：Sn^2

9.3 线图分析法

✧ **扩展弧子程序：**

(*a*) 将 *X* 插入图表

最多执行的次数为：1

(*b*) 对于活动边集(ActiveArc)中每个位置为(*k*, *i*) (*i* >*k*≥ 1)的点规则，如果该规则具有如下形式：$A \to \alpha \circ X$，如果 *A*=*S*, 则把 *S*(1, *n*+1) 加入到 Chart 中，并给出一个完整的分析结果；

最多执行的次数为：Sn^2

(*c*) 对于每个位置为(*k*, *i*) 的点规则：$A \to \alpha \circ X\beta$，则将 $A \to \alpha X \circ \beta\,(k, j)$

最多执行的次数为：Sn^2

9.3 线图分析法

每处理一个单词需要最多执行的最多操作次数为：

$$C+1+Sn^2+1+Sn^2+Sn^2=2+C+3Sn^2$$

由于算法对于长度为 n 的输入句子要执行 n 次循环，因此，Chart 算法最大执行的操作次数为：

$$n\times(2+C+3Sn^2)$$

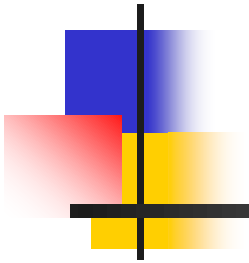
所以，Chart算法的时间复杂度为：

$O(Kn^3)$　　(K 为一常数)

9.3 线图分析法

◆ Chart parsing 算法评价

- ➢ 优点：
  - 算法简单，容易实现，开发周期短。
- ➢ 弱点：
  - 算法效率低，时间复杂度为 Kn^3；
  - 需要高质量的规则，分析结果与规则质量密切相关;
  - 难以区分歧义结构。



9.4 CYK分析算法

9.4 CYK分析算法

◆ Coke-Younger-Kasami (CYK) 算法

- 对 Chomsky 文法进行范式化：

  $A \rightarrow w$ 或 $A \rightarrow BC$

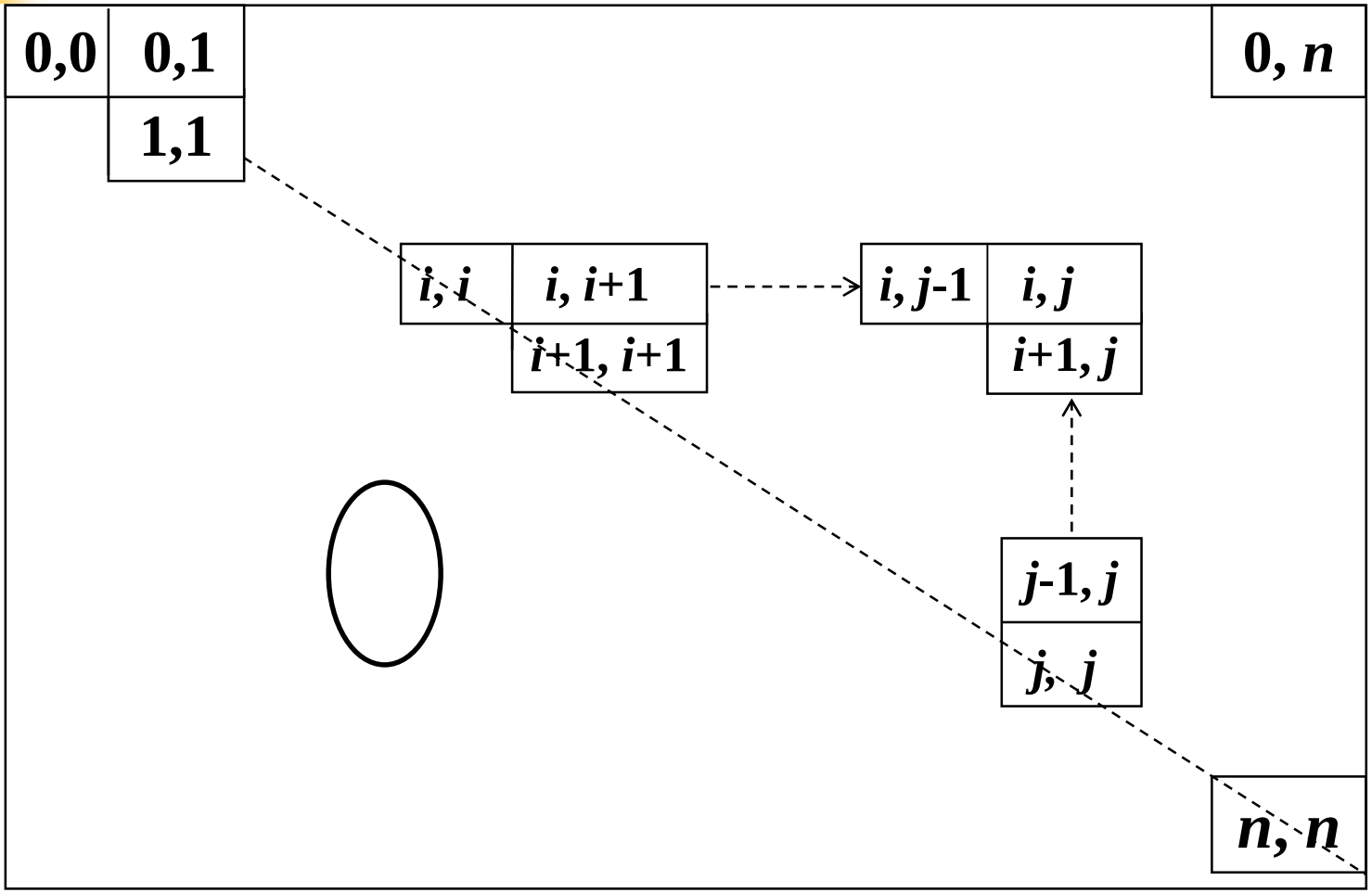
  $A, B, C \in V_N$， $w \in V_T$， $G=(V_N, V_T, P, S)$

- 自下而上的分析方法
- 构造 $(n+1)\times(n+1)$ 识别矩阵，n为输入句子长度。假设输入句子 $x=w_1w_2\ldots w_n$，w_i 为构成句子的单词，$n=|x|$。

9.4 CYK分析算法

- 识别矩阵的构成
  - 方阵对角线以下全部为0
  - 主对角线以上的元素由文法G的非终结符构成
  - 主对角线上的元素由输入句子的终结符号(单词)构成

9.4 CYK分析算法



9.4 CYK分析算法

◆ 识别矩阵构造步骤

(1) 首先构造主对角线，令 $t_{0,0}=0$，然后，从 $t_{1,1}$ 到$t_{n,n}$ 在主对角线的位置上依次放入输入句子x 的单词w_i。

(2) 构造主对角线以上紧靠主对角线的元素$t_{i,\,i+1}$, 其中，$i = 0, 1, 2 , \ldots, n\text{-}1$。对于输入句子 $x = w_1 w_2 \ldots w_n$，从 w_1 开始分析。

9.4 CYK分析算法

如果在文法G 的产生式集中有一条规则：

$$A \rightarrow w_1$$

则 $t_{0,1}=A$。

依此类推，如果有 $A \rightarrow w_{i+1}$，则 $t_{i,\,i+1}=A$。

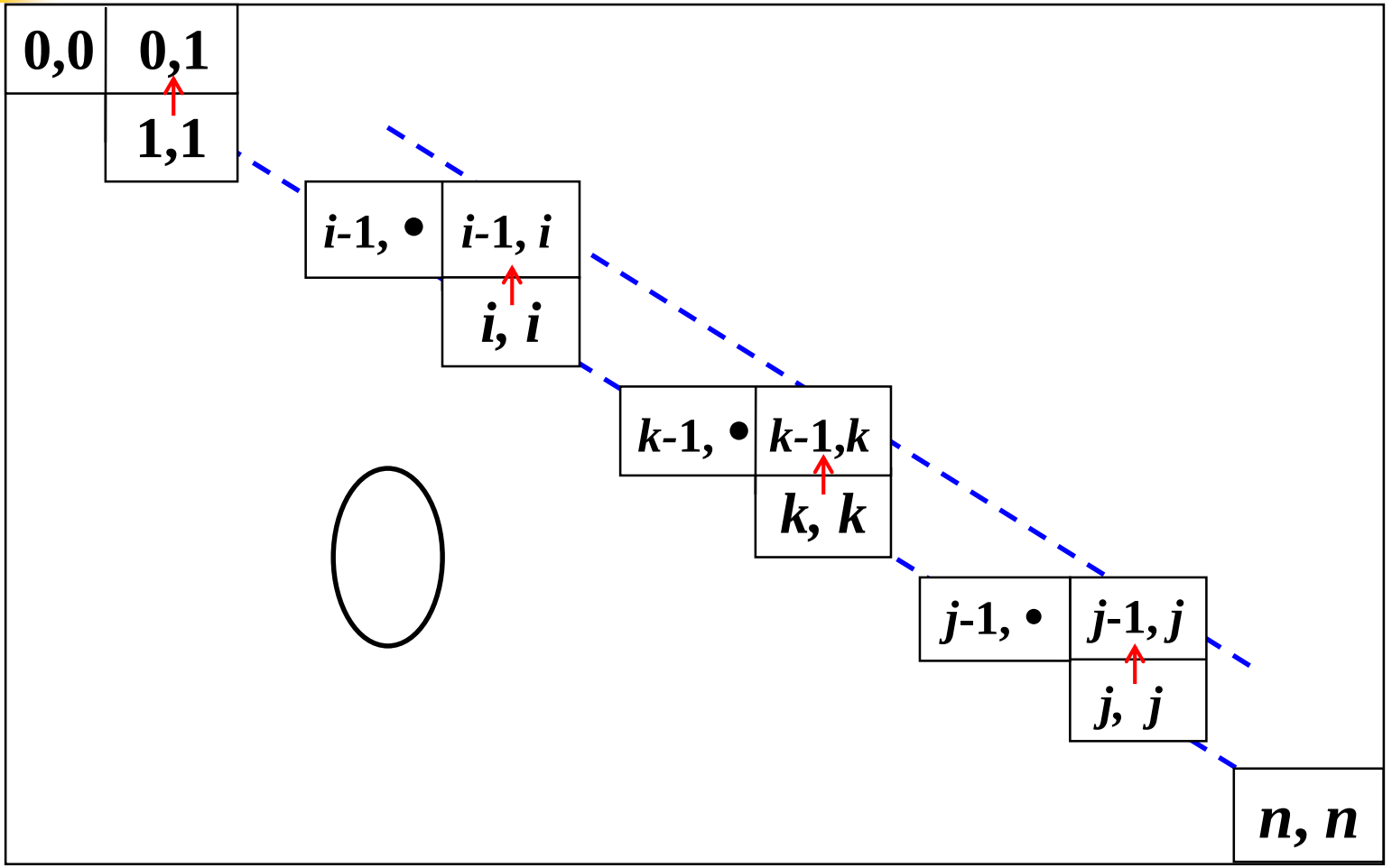
即，对于主对角线上的每一个终结符 w_i，所有可能推导出它的非终结符写在它的右边主对角线上方的位置上。

9.4 CYK分析算法

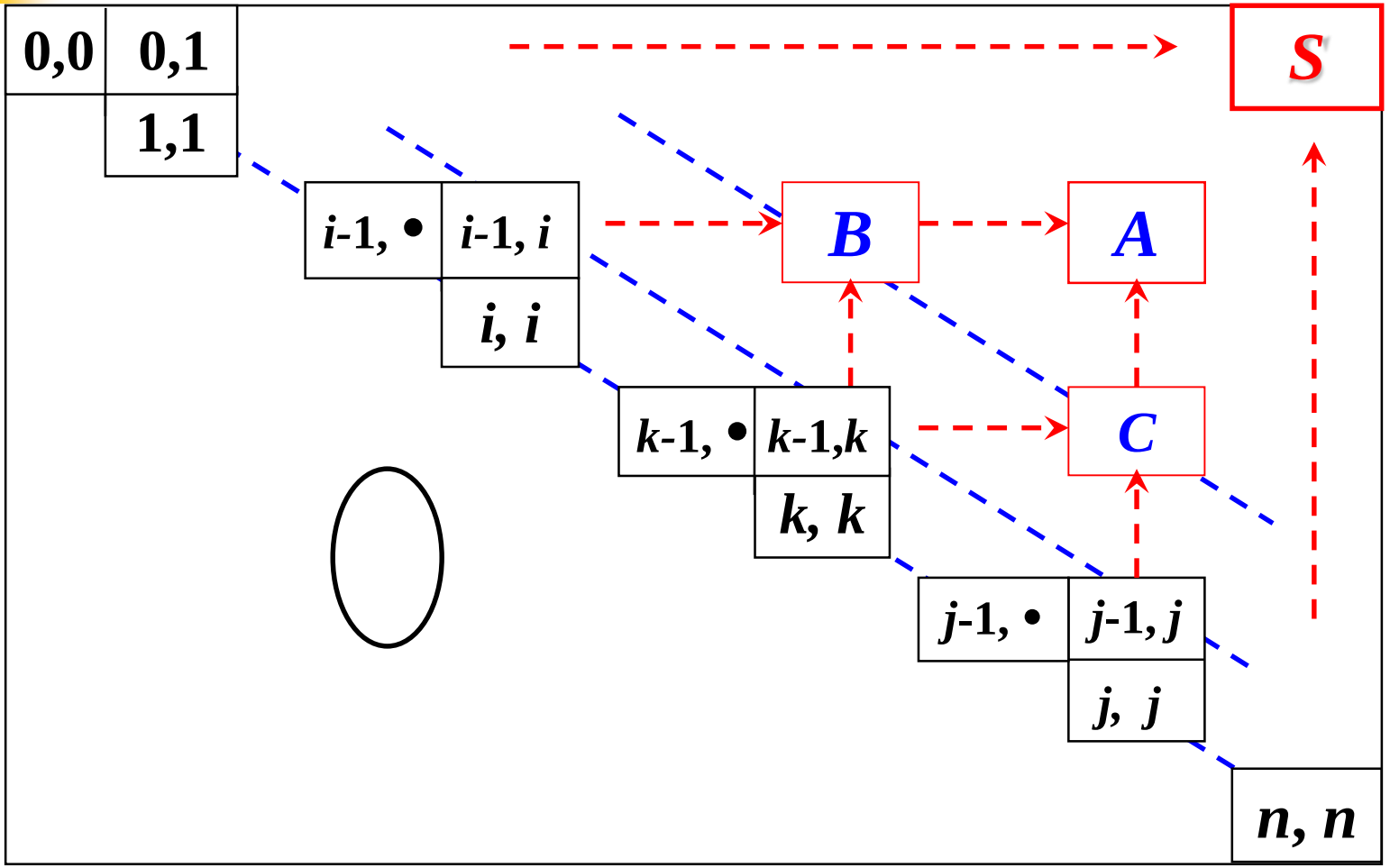
(3) 按平行于主对角线的方向，一层一层地向上填写矩阵的各个元素 $t_{i,j}$，其中，$i = 0,1, \ldots, n\text{-}d$，$j = d+i$，$d=2, 3, \ldots, n$。如果存在一个正整数 k，$i+1 \le k \le j-1$，在文法G的规则集中有产生式 $A \rightarrow BC$，并且，$B \in t_{i,k}$，$C \in t_{k,j}$，那么，将A写到矩阵 $t_{i,j}$ 位置上。

判断句子 x 由文法 G 所产生的充要条件是：$t_{0,n}=S$。

9.4 CYK分析算法



9.4 CYK分析算法



9.4 CYK分析算法

◆ 例子

给定文法 G(S)：

(1) S → P VP　　(2) VP → V V

(3) VP → VP N　　(4) P → 他

(5) V → 喜欢　　(6) V → 读

(7) N → 书

请用 CYK 算法分析句子：他喜欢读书

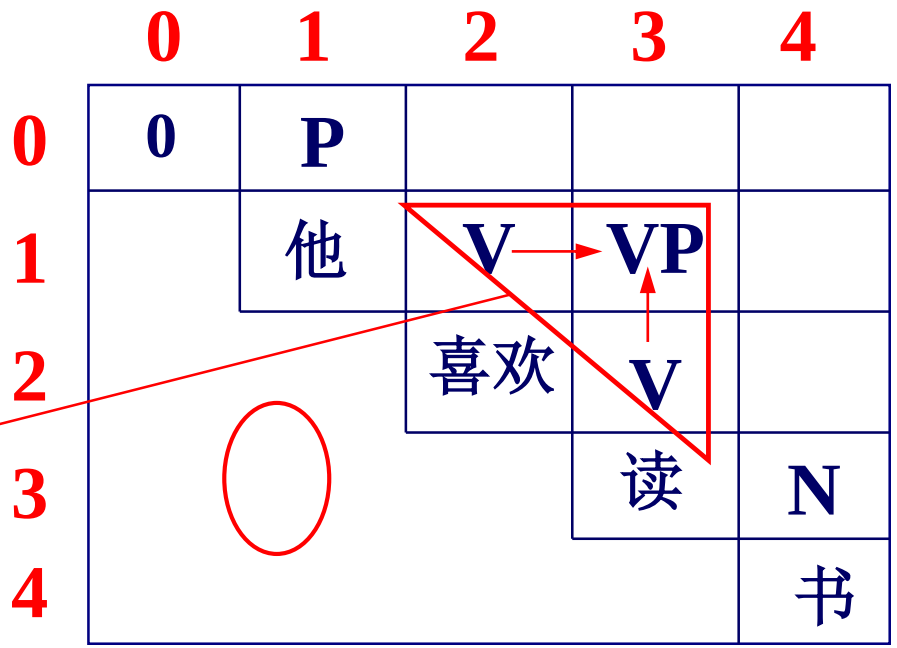
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。



(1) S → P VP
(2) VP → V V
(3) VP → VP N

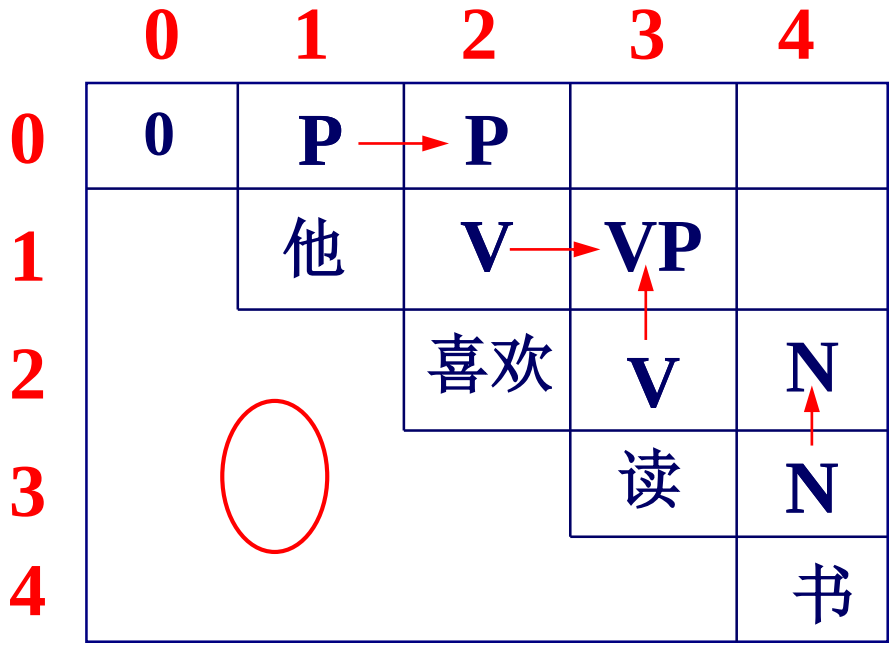
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。



(1) S → P VP

(2) VP → V V

(3) VP → VP N

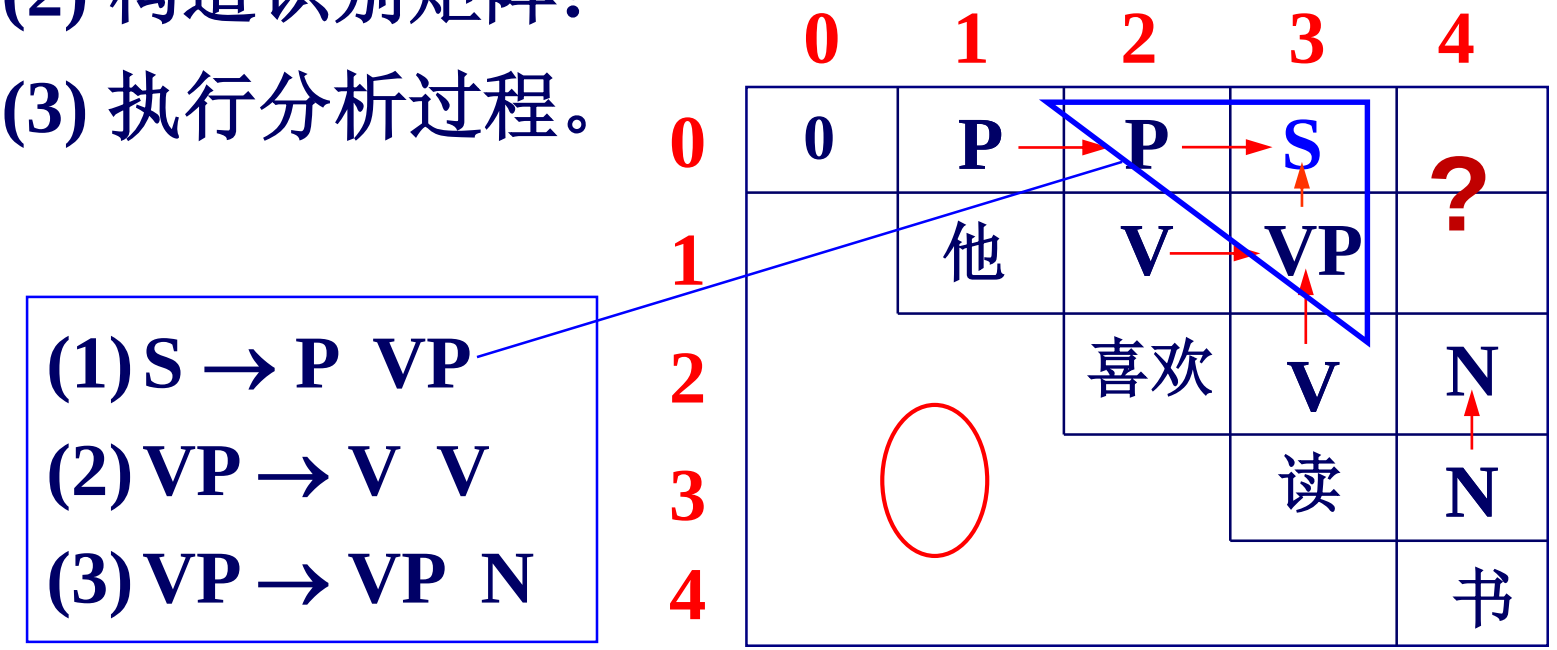
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。



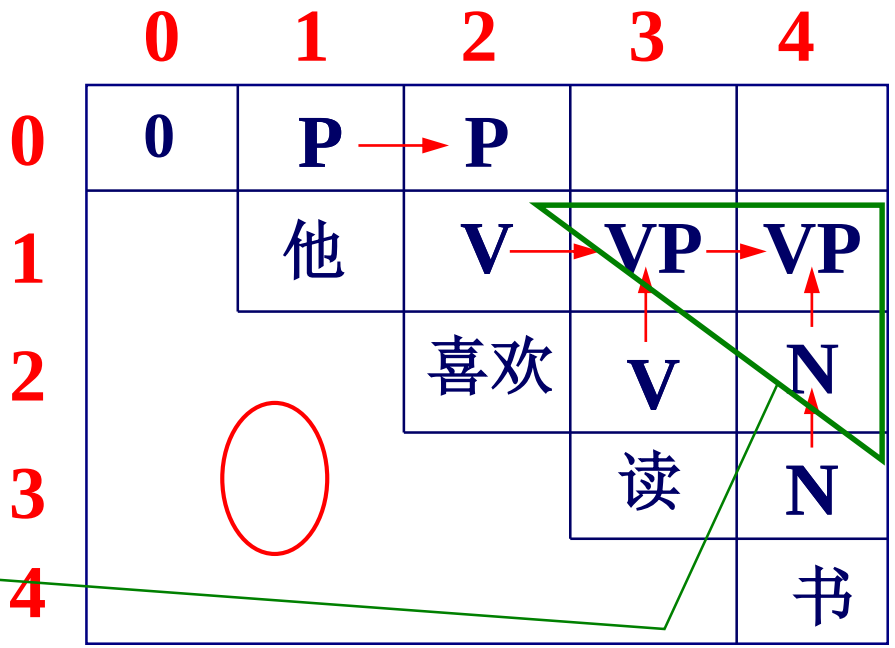
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。



(1) S → P VP

(2) VP → V V

(3) VP → VP N

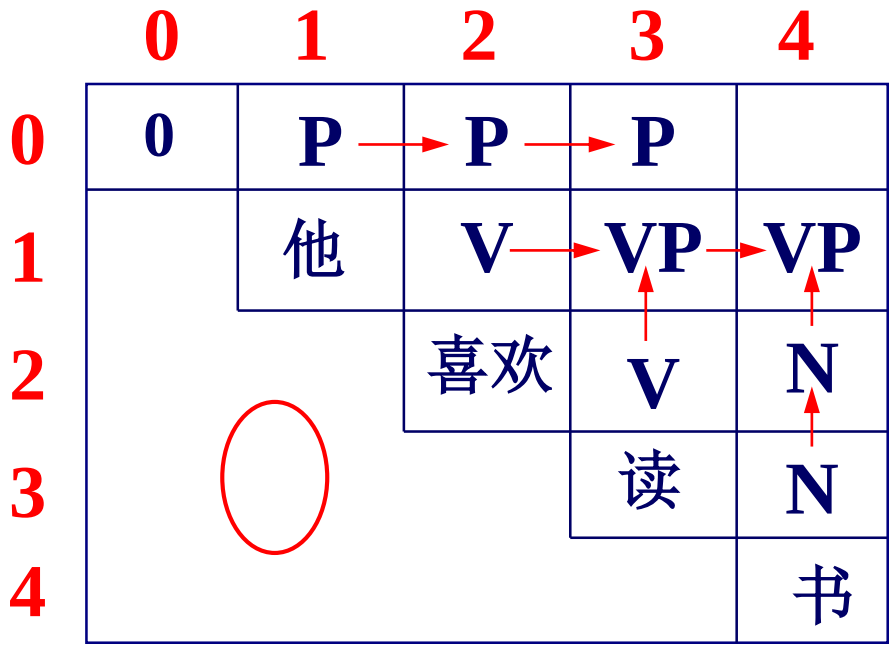
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。



(1) S → P VP

(2) VP → V V

(3) VP → VP N

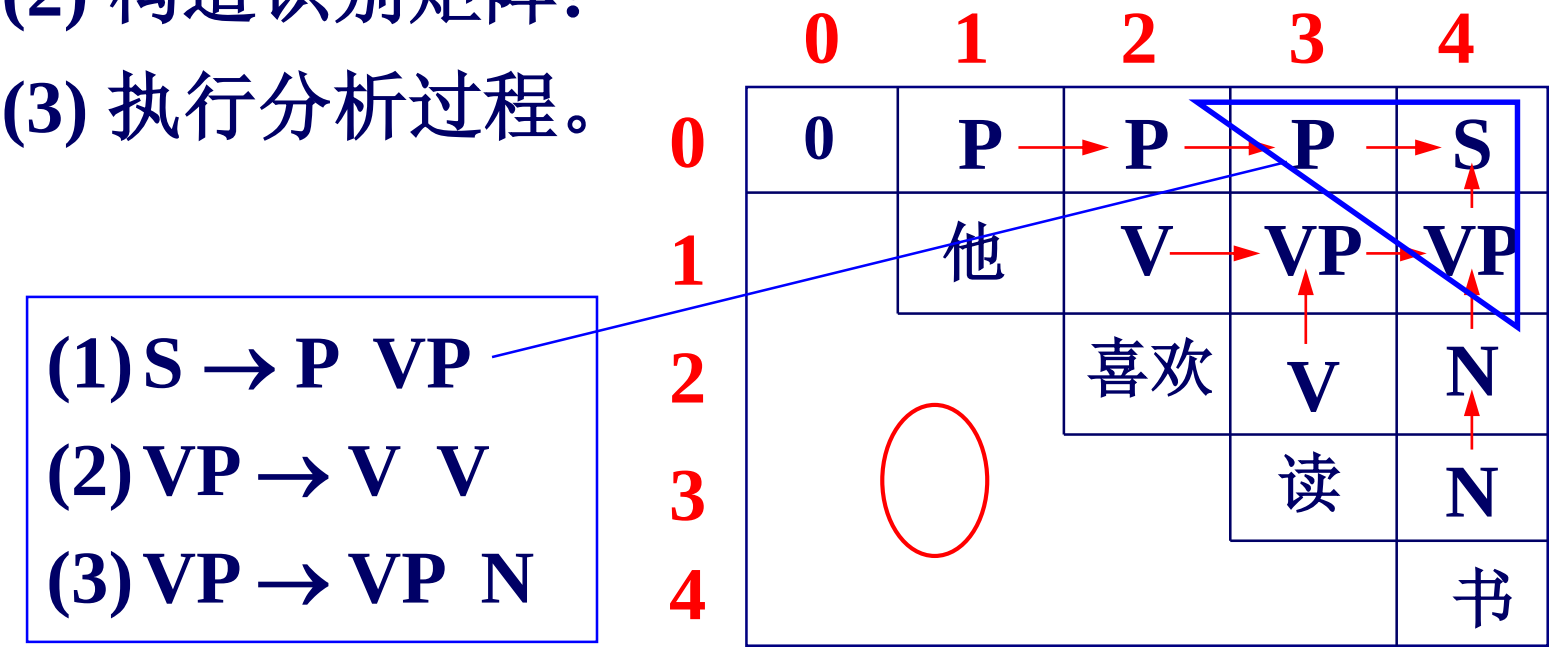
9.4 CYK分析算法

(1) 汉语分词和词性标注以后：

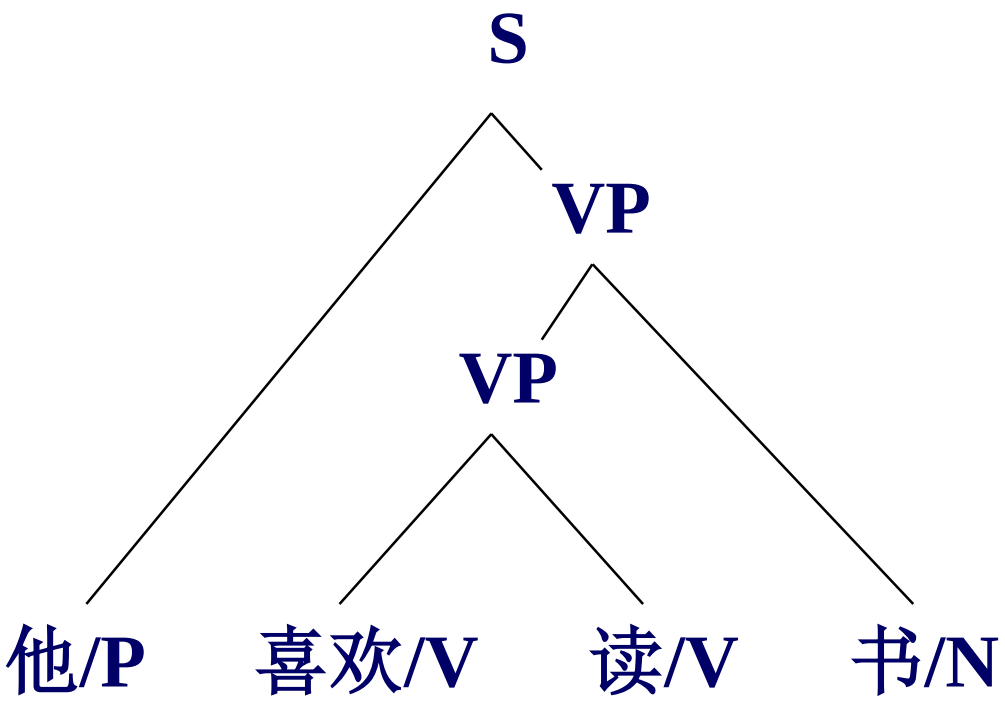
他/P 喜欢/V 读/V 书/N　　n=4

(2) 构造识别矩阵：

(3) 执行分析过程。

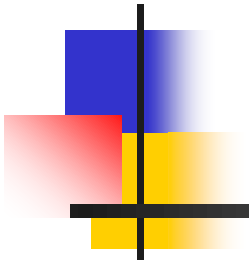


9.4 CYK分析算法



9.4 CYK分析算法

- CYK 算法的评价
  - 优点
    - 简单易行，执行效率高
  - 弱点
    - 必须对文法进行范式化处理
    - 无法区分歧义



9.5 概率上下文无关文法

9.5 概率上下文无关文法

◆ PCFG 规则

形式：$A \to \alpha, \quad p$

约束：$\sum_{\alpha} p(A \to \alpha) = 1$

例如：**NP → NN NN, 0.60**
NP → NN CC NN, 0.40 } $\sum p=1$

CD → QP, 0.99
CD → LST, 0.01 } $\sum p=1$

9.5 概率上下文无关文法

◆例-1：S → NP VP, 1.00　　NP → NP PP, 0.40

NP → astronomers, 0.10

NP → ears, 0.18　　NP → saw, 0.04

NP → stars, 0.18　　NP → telescopes, 0.1

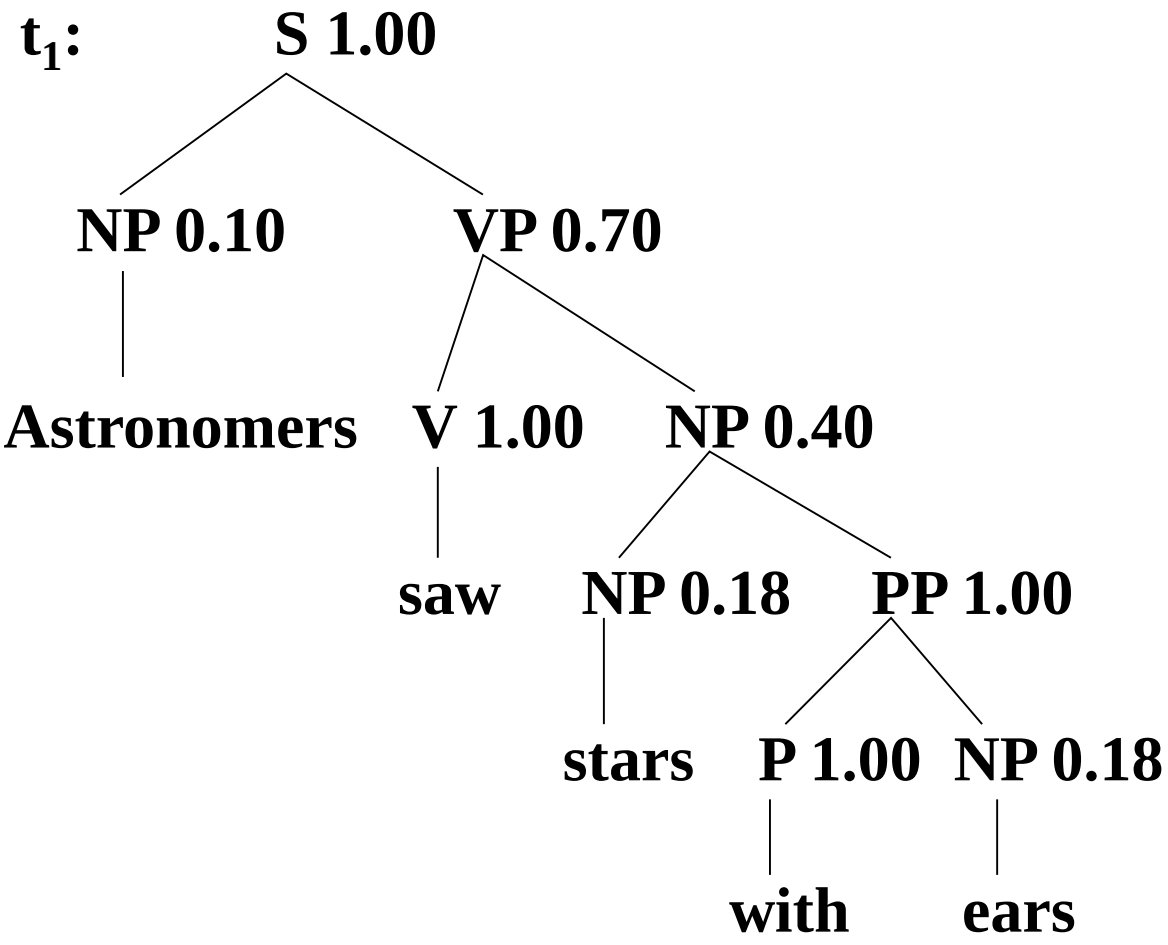
PP → P NP, 1.00　　P → with, 1.00

VP → V NP, 0.70　　VP → VP PP, 0.30

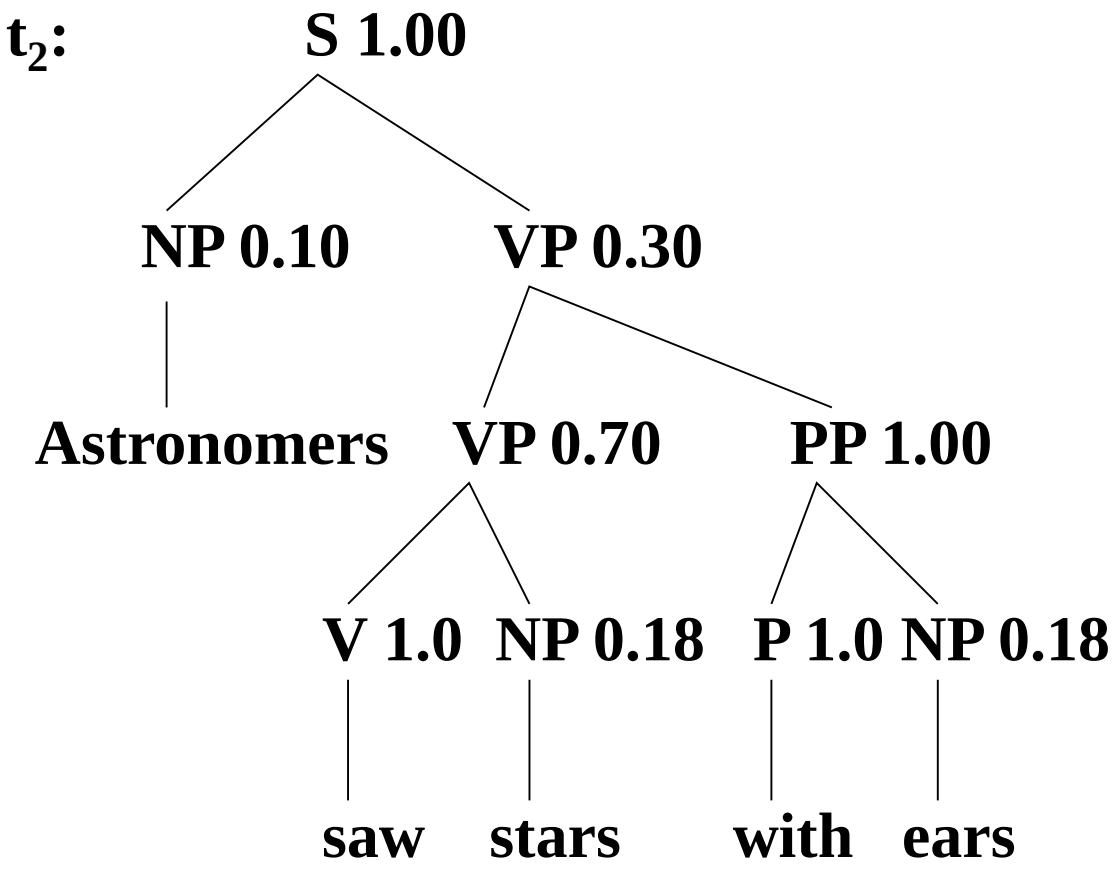
V → saw, 1.00

给定句子 S：*Astronomers saw stars with ears.*

9.5 概率上下文无关文法



9.5 概率上下文无关文法

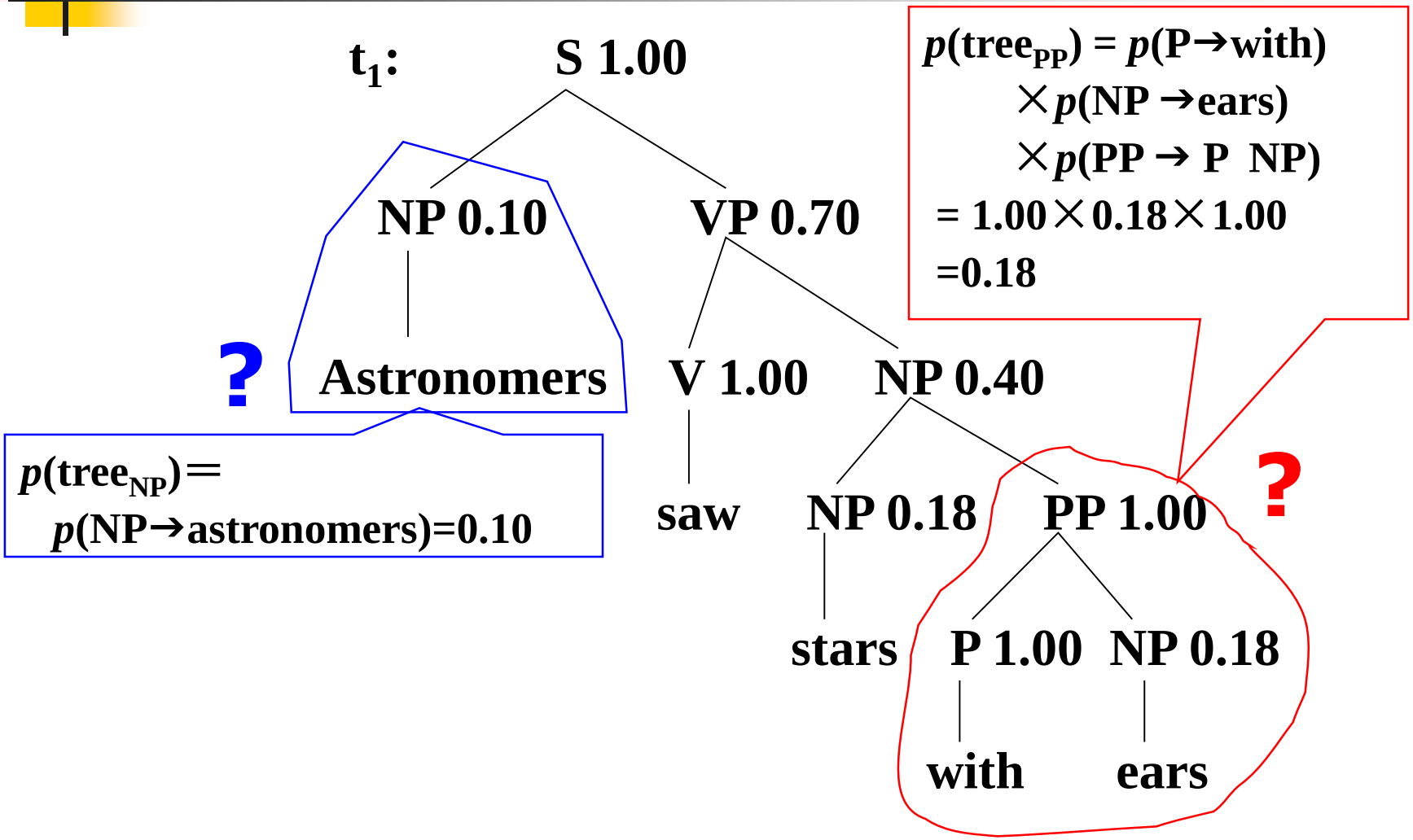


9.5 概率上下文无关文法

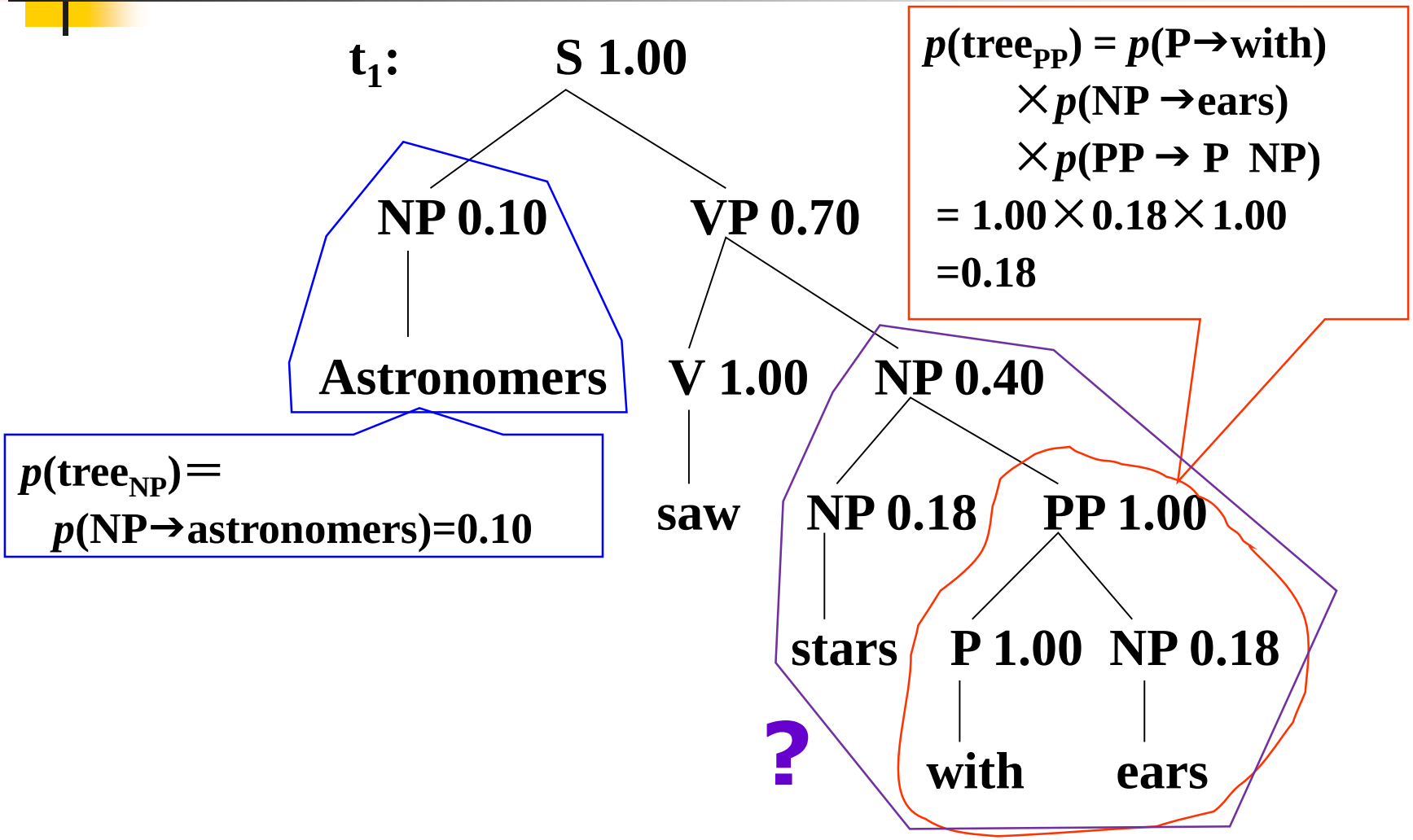
◆计算分析树概率的基本假设

- **位置不变性：**子树的概率与其管辖的词在整个句子中所处的位置无关，即对于任意的 k，$p(A_{k(k+C)} \rightarrow w)$ 一样。
- **上下文无关性：**子树的概率与子树管辖范围以外的词无关，即 $p(A_{kl} \rightarrow w|$任何超出$k\sim l$ 范围的上下文$) = p(A_{kl} \rightarrow w)$。
- **祖先无关性：**子树的概率与推导出该子树的祖先结点无关，即 $p(A_{kl} \rightarrow w \mid$ 任何除 A 以外的祖先结点$) = p(A_{kl} \rightarrow w)$。

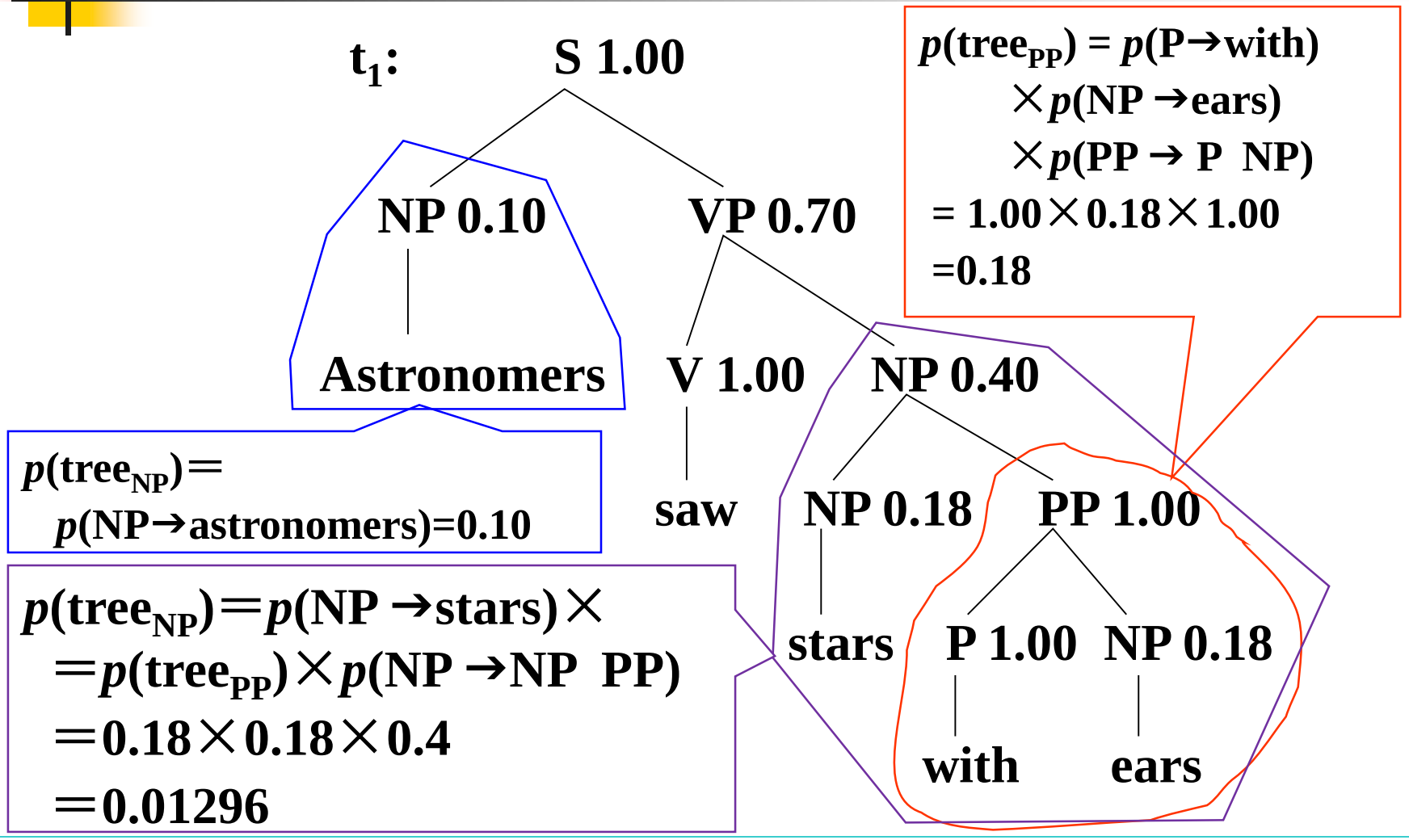
9.5 概率上下文无关文法



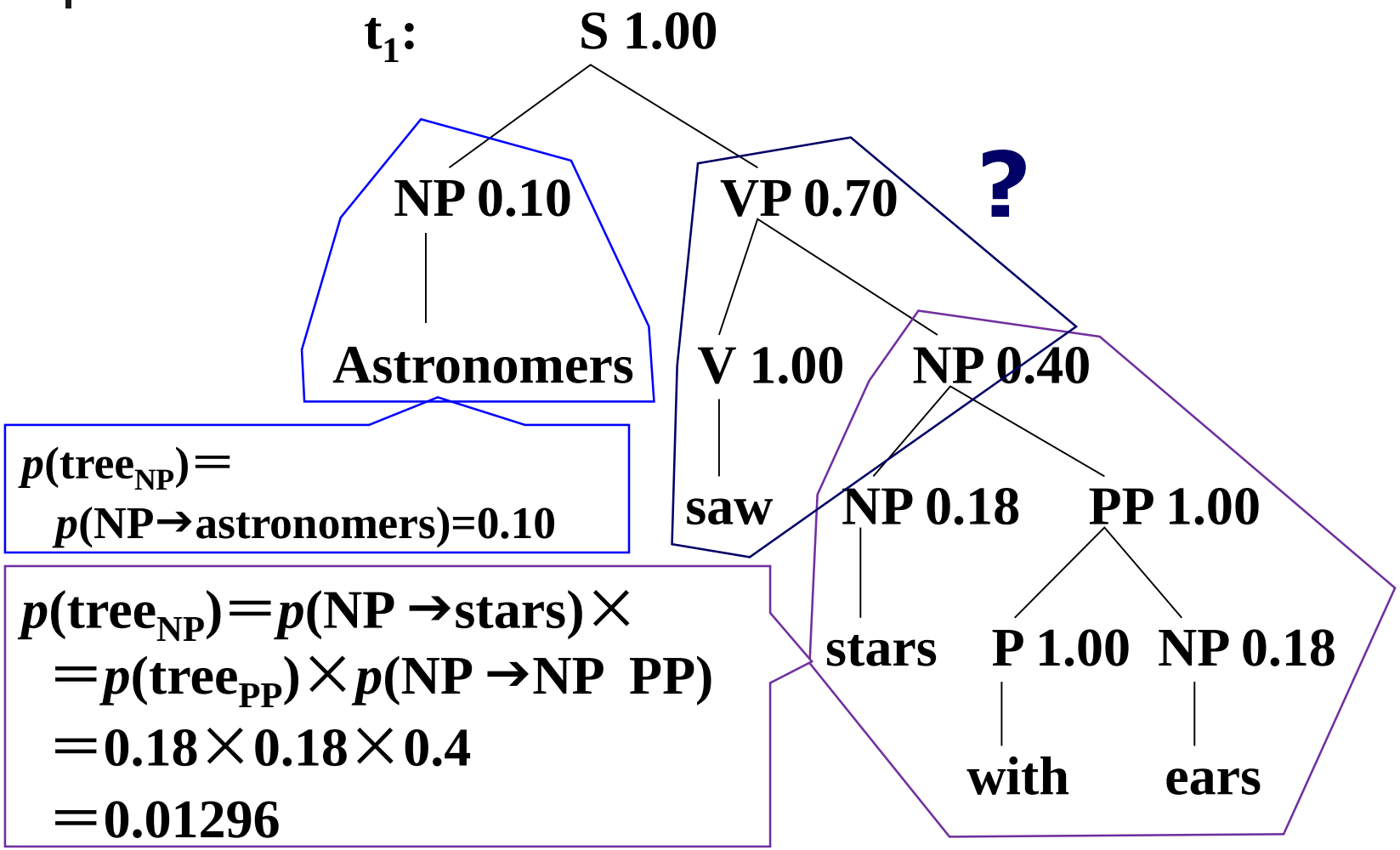
9.5 概率上下文无关文法



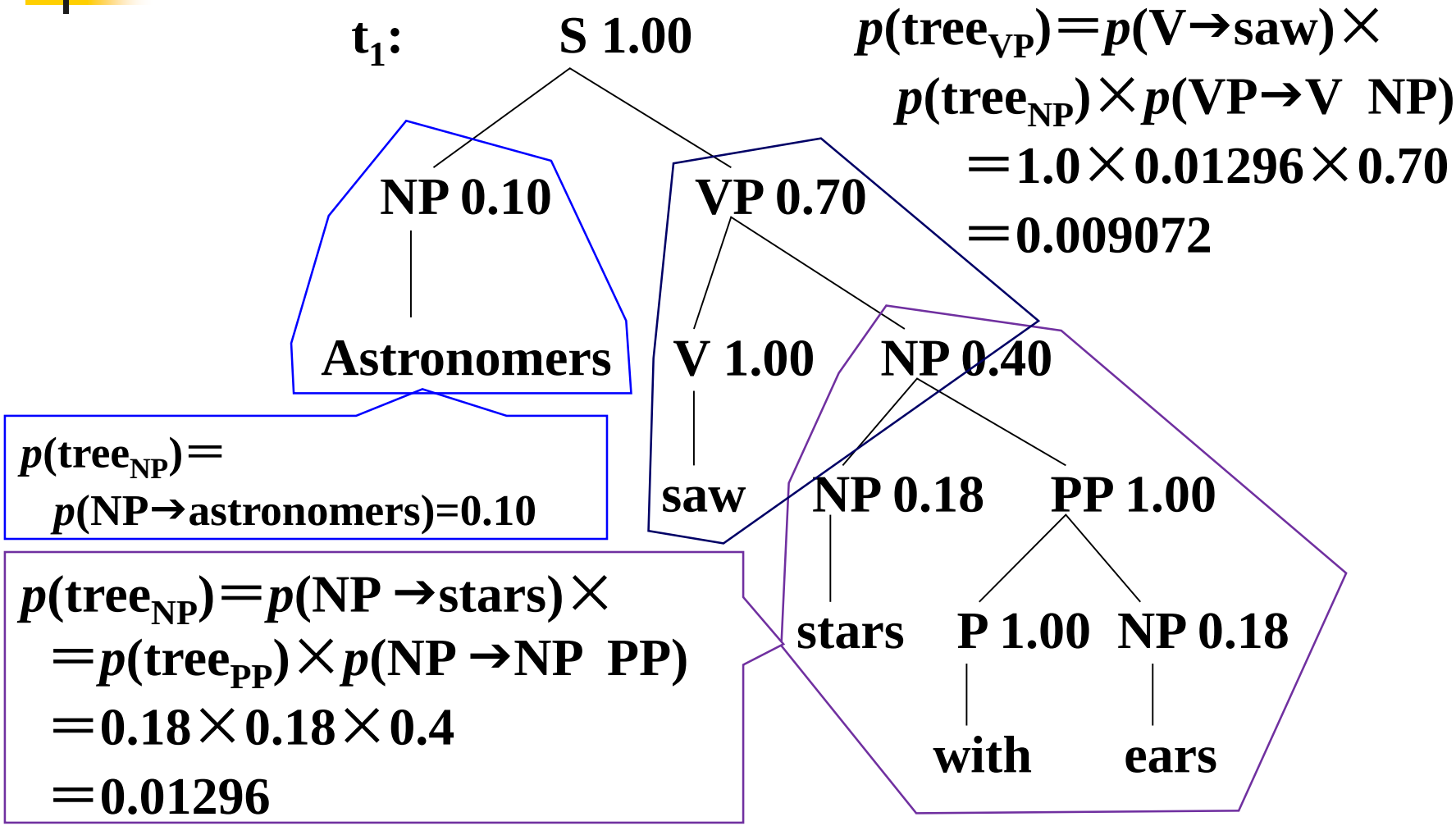
9.5 概率上下文无关文法



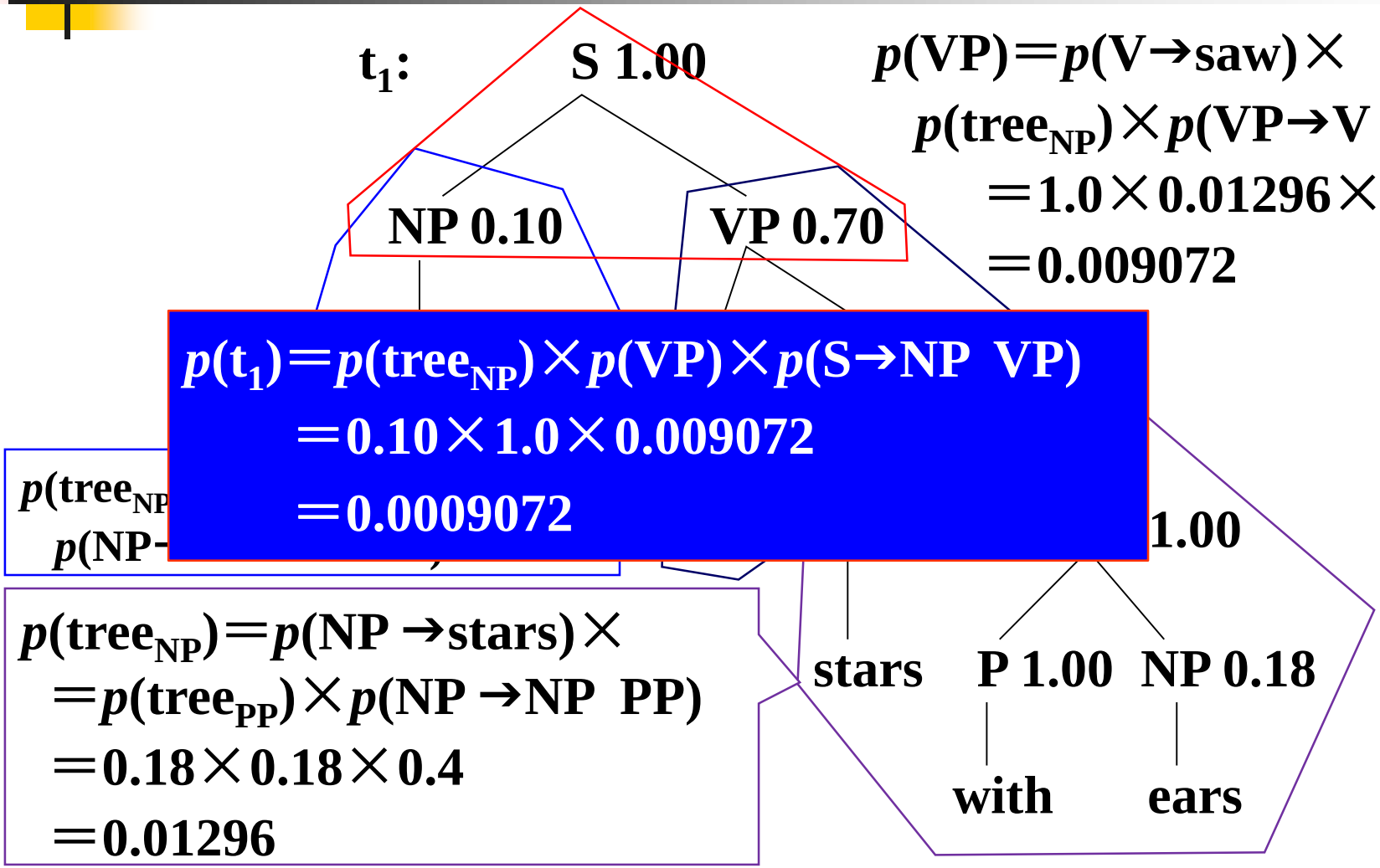
9.5 概率上下文无关文法



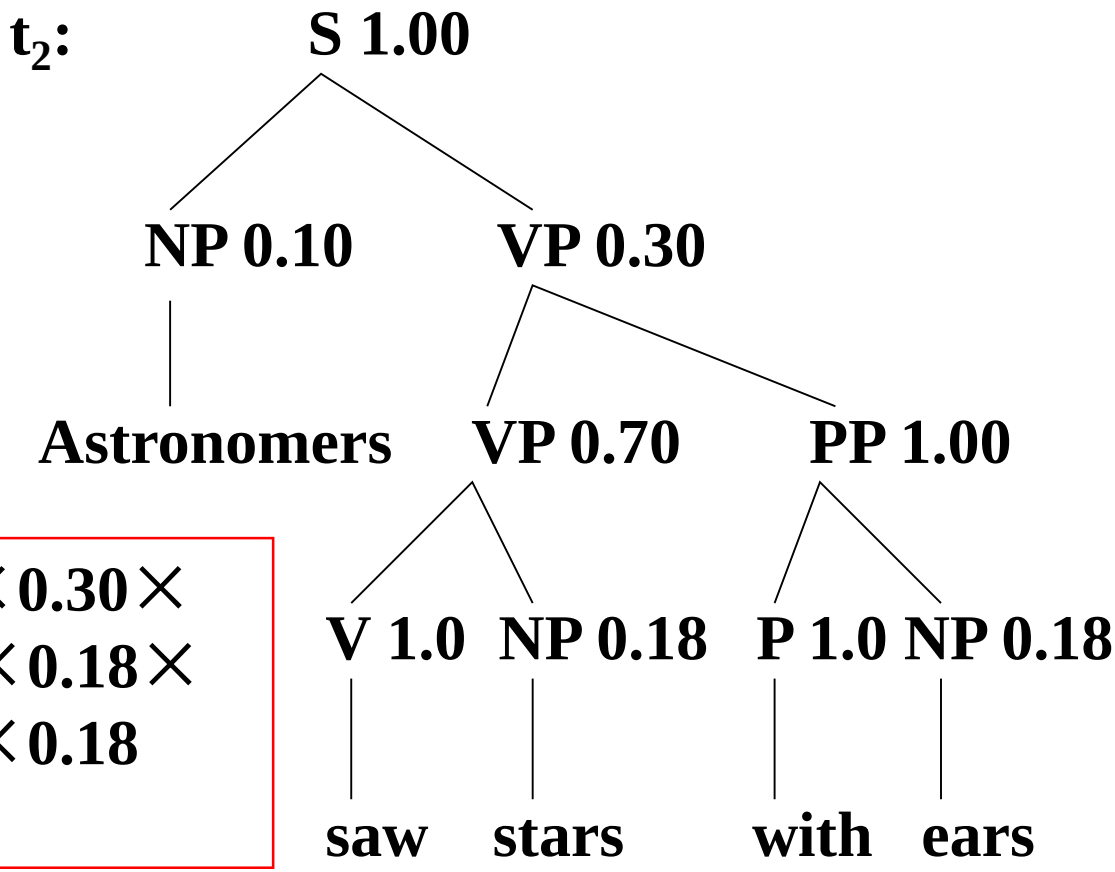
9.5 概率上下文无关文法



9.5 概率上下文无关文法



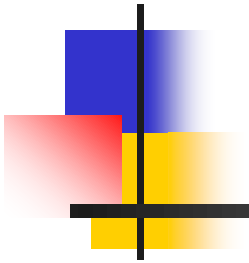
9.5 概率上下文无关文法



$$p(t_2) = 1.00 \times 0.10 \times 0.30 \times 0.70 \times 1.00 \times 0.18 \times 1.00 \times 1.00 \times 0.18 = 0.0006804$$

9.5 概率上下文无关文法

对于给定的句子 S，两棵句法分析树的概率不等，$P(t_1) > P(t_2)$，因此，可以得出结论：分析结果t_1正确的可能性大于t_2。



9.6 PCFG的三个问题

9.6 PCFG的三个问题

1、给定句子 $W=w_1w_2\ldots w_n$ 和 PCFG G，如何快速计算 $p(W|G)$？

2、给定句子 $W=w_1w_2\ldots w_n$ 和 PCFG G，如何快速地选择最佳句法结构树？

3、给定句子 $W=w_1w_2\ldots w_n$ 和 PCFG G，如何调节 G 的参数，使得 $p(W|G)$ 最大？

9.6 PCFG的三个问题

假设文法 G(*S*) 的规则只有两种形式：

$$A \rightarrow w, \quad w \in V_T$$

$$A \rightarrow BC, \quad B,C \in V_N$$

可以通过范式化处理，使CFG 规则满足上述形式。这种假设的文法形式称为乔姆斯基范式(Chomsky normal form, CNF)。

9.6 PCFG的三个问题

1、内向算法或外向算法解决第一个问题－快速地计算句子的句法树概率

◆内向算法

基本思想：利用动态规划算法计算由非终结符 ***A*** 推导出的某个字串片段 $w_i w_{i+1} \cdots w_j$ 的概率 $\alpha_{ij}(A)$。语句 $W = w_1 w_2 \cdots w_n$ 的概率即为文法 **G(*S*)**中***S*** 推导出的字串的概率 $\alpha_{1n}(S)$。

9.6 PCFG的三个问题

- **定义：** 内向变量 $\alpha_{ij}(A)$ 是由非终结符A 推导出的语句 W 中子字串 $w_i w_{i+1} \cdots w_j$ 的概率：

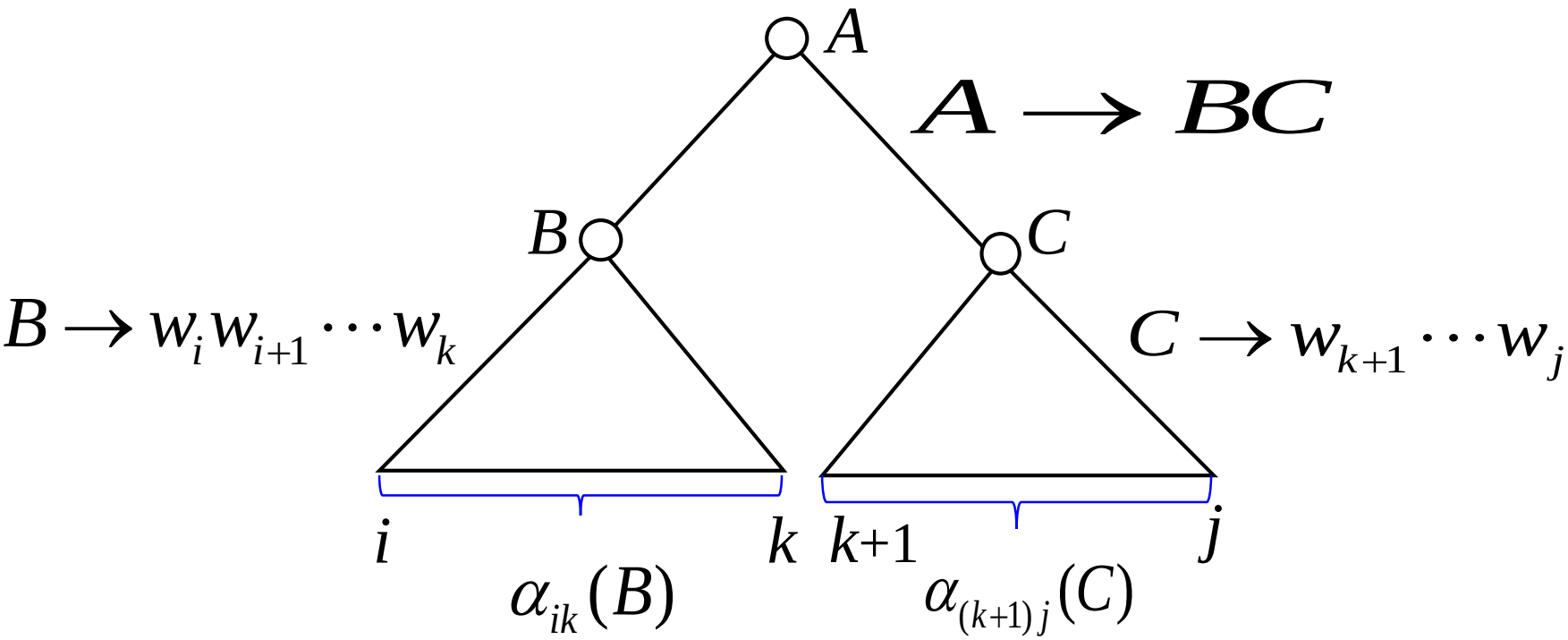
$$\alpha_{ij}(A) = p(A \overset{*}{\Rightarrow} w_i w_{i+1} \ \ldots \ w_j)$$

计算 $\alpha_{ij}(A)$ 的递推公式：

$$\alpha_{ii}(A) = p(A \rightarrow w_i)$$

$$\alpha_{ij}(A) = \sum_{B,C \in V_N} \sum_{i \le k \le j} p(A \rightarrow BC)\alpha_{ik}(B)\alpha_{(k+1)j}(C)$$

9.6 PCFG的三个问题



9.6 PCFG的三个问题

➢**解释：**

当 $i=j$ 时，字符串 $w_i\, w_{i+1} \ldots w_j$ 只是一个字 w_{ii}，可简单记作 w_i，由 A 推导出 w_i 的概率就是产生式 $A \rightarrow w_i$ 的概率 $p(A \rightarrow w_i)$；当 $i \neq j$ 时，也就是说，字符串 $w_i\, w_{i+1} \ldots w_j$ 至少有两个词，根据约定，A要推导出该词串，必须首先运用产生式 $A \rightarrow BC$，那么，可用B 推导出前半部 $w_i \ldots w_k$，用C 推导出后半部$w_{k+1} \ldots w_j$。由这一推导过程产生 $w_i\, w_{i+1} \ldots w_j$ 的概率为：$p(A \rightarrow BC)\alpha_{ik}(B)\alpha_{(k+1)j}(C)$。考虑到$B$、$C$ 和k 取值的任意性，应计算各种情况下概率的总和。

9.6 PCFG的三个问题

➢ **内向算法描述：**

输入：文法 **G(*S*)**，语句 $W = w_1 w_2 \cdots w_n$

输出：$p(S \overset{*}{\Rightarrow} w_1 w_2 \ \ldots \ w_n)$

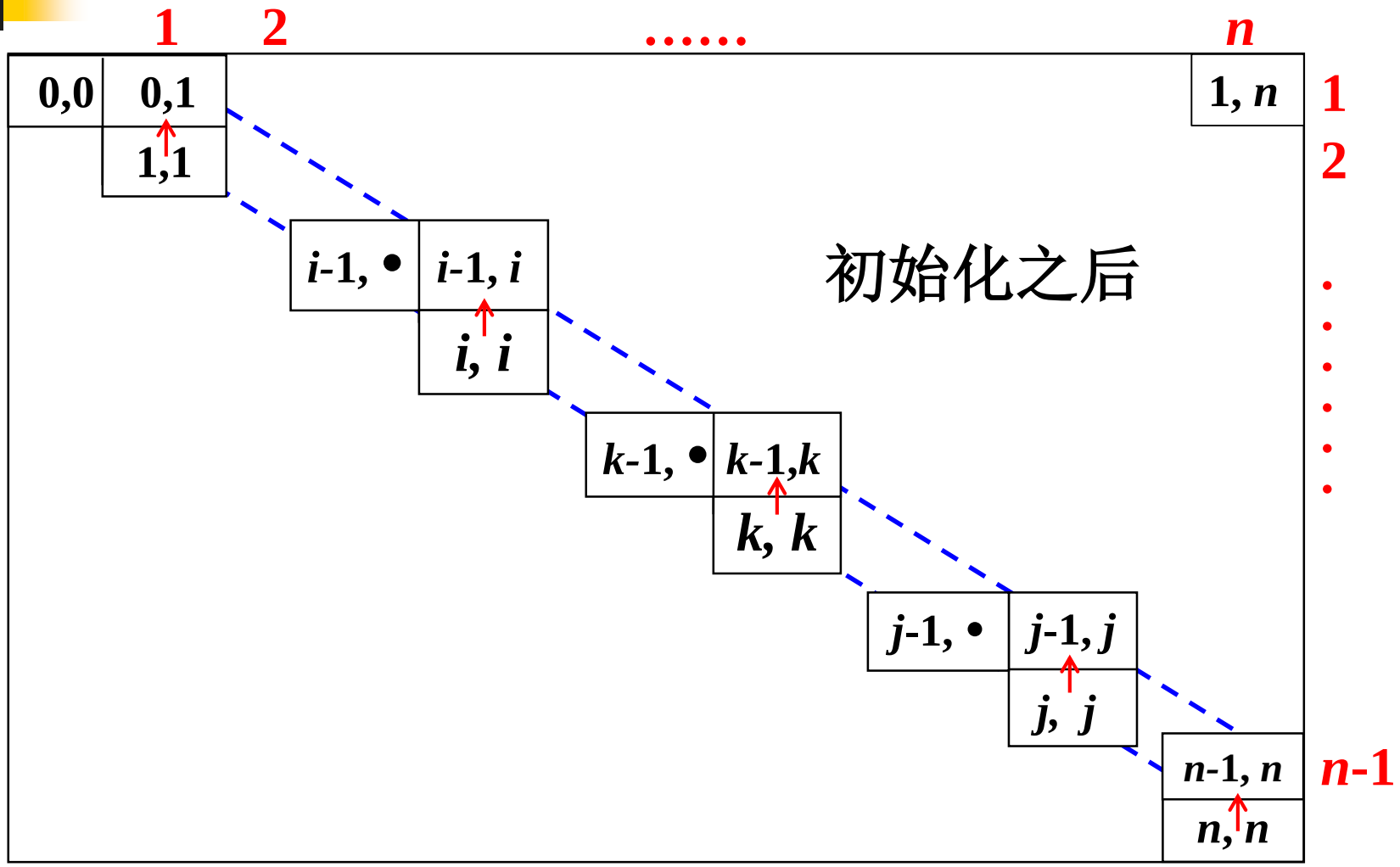
(1) 初始化：$\alpha_{ii}(A) = p(A \to w_i) \quad A \in V_N, \ 1 \le i \le j \le n$

(2) 归纳计算：j=1.. n, i=1..n- j, 重复下列计算：

$$\alpha_{i(i+j)}(A) = \sum_{B,C \in V_N} \sum_{i \le k \le i+j} p(A \to BC)\alpha_{ik}(B)\alpha_{(k+1)(i+j)}(C)$$

(3) 终结：$p(S \overset{*}{\Rightarrow} w_1 w_2 \cdots w_n) = \alpha_{1n}(S)$

9.6 PCFG的三个问题



9.6 PCFG的三个问题

循环执行：
(假设：n=20)

$$\alpha_{i(i+j)}(A)=\sum_{B,C\in V_N}\sum_{i\le k\le i+j}p(A\to BC)\alpha_{ik}(B)\alpha_{(k+1)(i+j)}(C)$$

j =1.. n, i =1..n- j (j 表示列，i 表示行)：

j =1: i =1..19: $\alpha_{12}(A)$, $\alpha_{23}(A)$, …, $\alpha_{(19)(20)}(A)$

j =2: i =1..18: $\alpha_{13}(A)$, $\alpha_{24}(A)$, …, $\alpha_{(18)(20)}(A)$

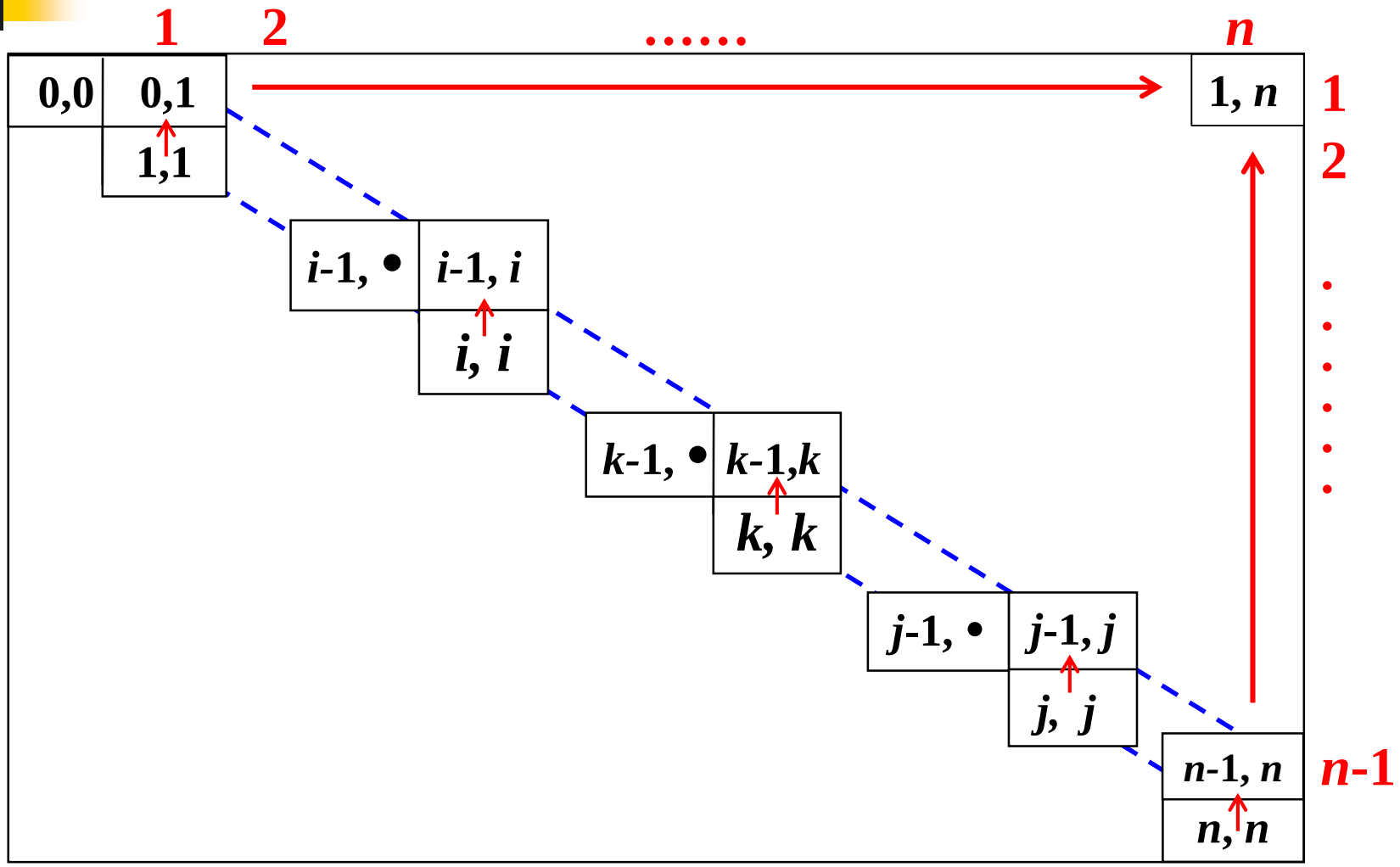
j =3: i =1..17: $\alpha_{14}(A)$, $\alpha_{25}(A)$, …, $\alpha_{(17)(20)}(A)$

……

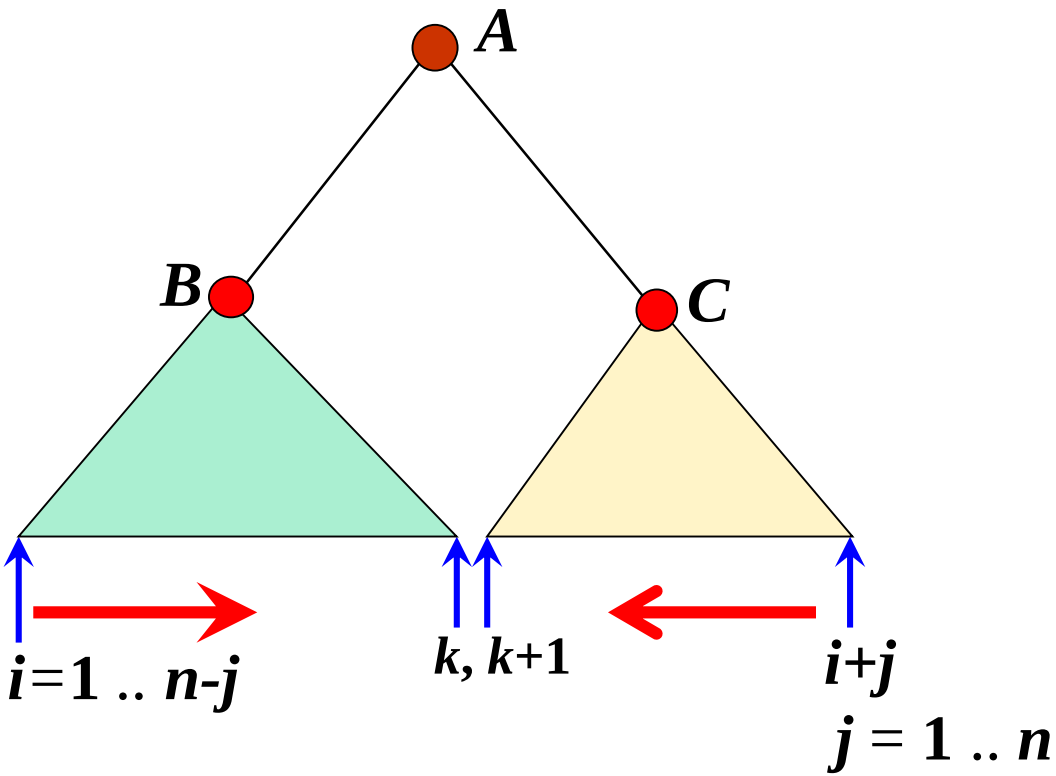
j =19: i =1: $\alpha_{1(20)}(A)$

j =20: 不执行

9.6 PCFG的三个问题



9.6 PCFG的三个问题



内向归纳，自底向上。

9.6 PCFG的三个问题

◆外向算法

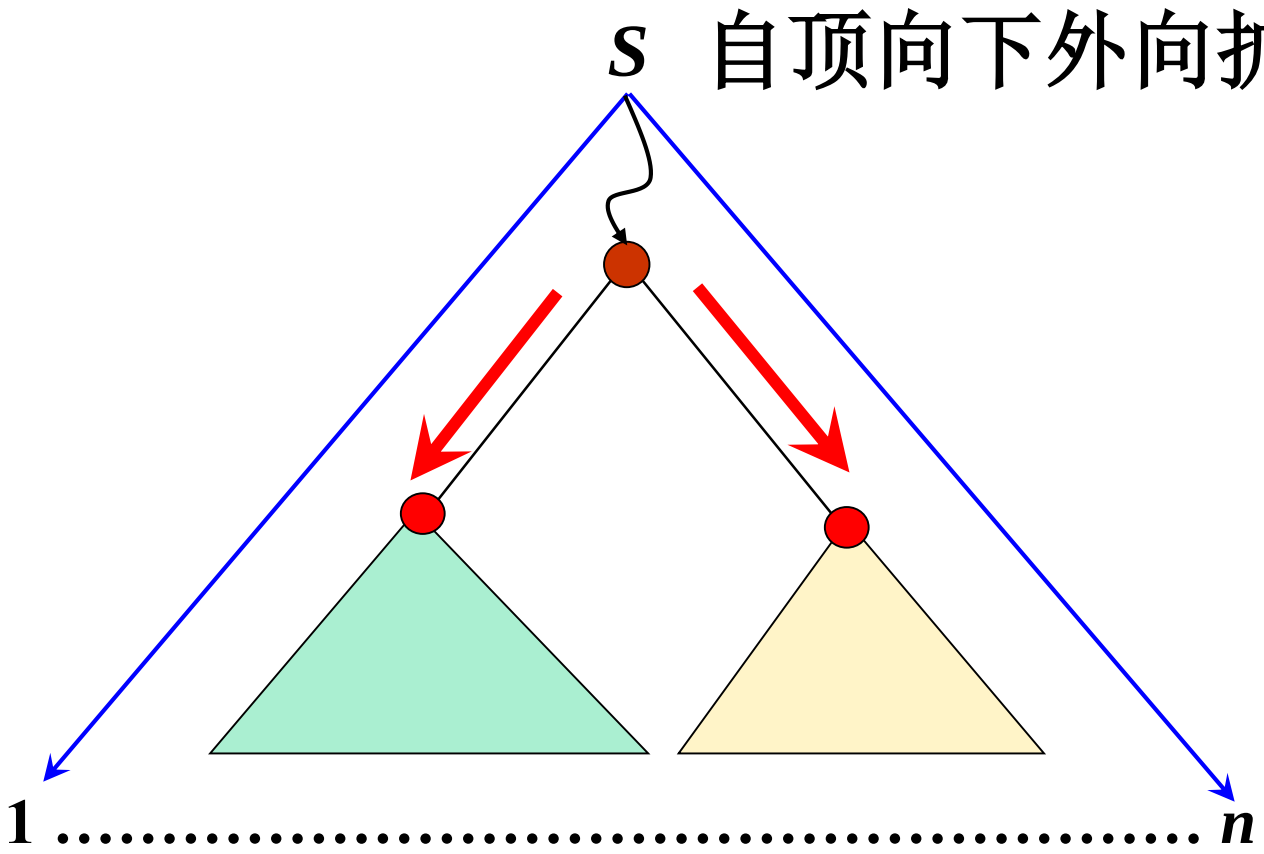
➢ **定义：**

外向变量 $\beta_{ij}(A)$ 是由文法初始符号 S 推导出语句 $W = w_1 w_2 \cdots w_n$ 的过程中，到达扩展符号串 $w_1 \cdots w_{i-1} A w_{j+1} \cdots w_n$ 的概率($A \Rightarrow w_i \ldots w_j$)：

$$\beta_{ij}(A) = p(S \overset{*}{\Rightarrow} w_1 \cdots w_{i-1} A w_{j+1} \cdots w_n)$$

$\beta_{ij}(A)$表示除了以A为根节点的子树以外的概率。

9.6 PCFG的三个问题

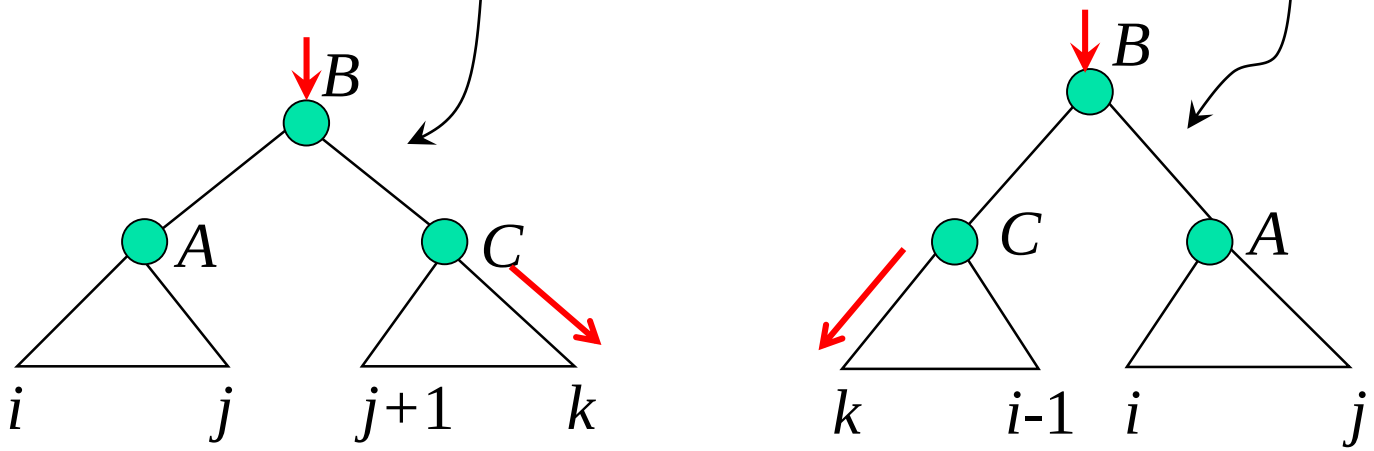


9.6 PCFG的三个问题

$\beta_{ij}(A)$ 可由动态规划算法求得, 其递推公式:

$$\beta_{1n}(A) = \delta(A, S) \qquad \text{（初始化）}$$

$$\beta_{ij}(A) = \sum_{B,C}\sum_{k>j} \beta_{ik}(B)\, p(B \to AC)\, \alpha_{(j+1)k}(C) + \sum_{B,C}\sum_{k<i} \beta_{kj}(B)\, p(B \to CA)\, \alpha_{k(i-1)}(C)$$



9.6 PCFG的三个问题

➢**解释：**

(1) 当 $i=1, j=n$ 时，即 $w_i w_{i+1} \cdots w_j$ 是整个语句W 时，根据乔姆斯基语法范式的约定，不可能有规则 $S \rightarrow A$，因此，由S 推导出 W 的过程中，如果 $A \neq S$ 的话，A 推导出 W 的概率为 0(只有S 才能推导出 W)，即$\beta_{1n}(A)=0$ 。

如果 $A=S$，$\beta_{1n}(A)$ 为由初始符S 推导出 W 的概率，因此，$\beta_{1n}(A)=1$。

9.6 PCFG的三个问题

(2) 当 $i \neq 1$ 或者 $j \neq n$ 时，如果在 S 推导出 W 的过程中出现了字符串 $w_1 \cdots w_k A w_{j+1} \cdots w_n$，则该推导过程必定使用了规则 $B \rightarrow AC$ 或 $B \rightarrow CA$。假定运用了规则 $B \rightarrow A\,C$ 推导出 $w_i \cdots w_j w_{j+1} \cdots w_k$，则该推导可以分解为以下三步：

(a) 由 S 推导出 $w_1 \cdots w_{i-1} B w_{k+1} \cdots w_n$，其概率为 $\beta_{ik}(B)$；

(b) 运用产生式 $B \rightarrow AC$ 扩展非终结符 B，其概率为 $p(B \rightarrow AC)$；

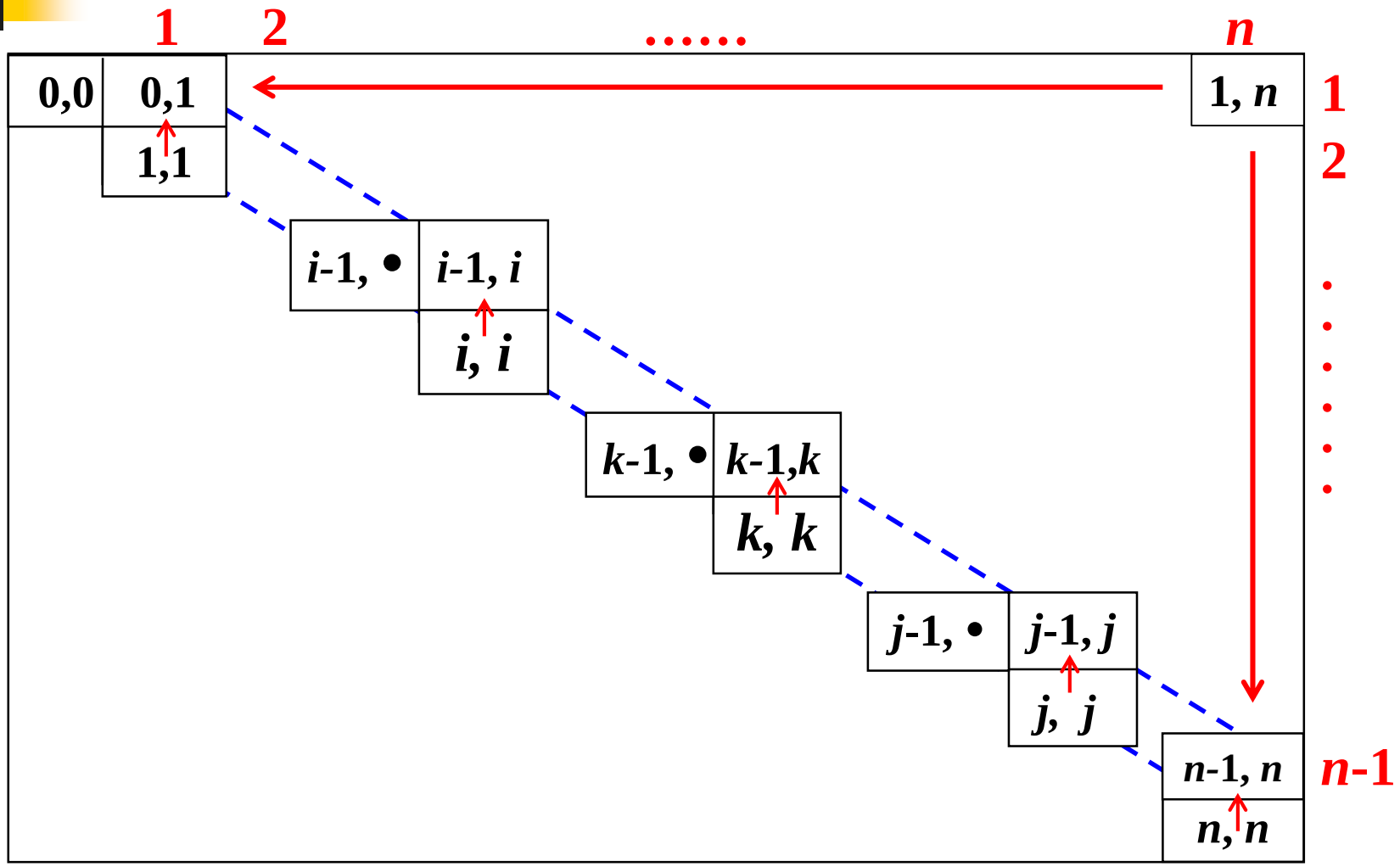
(c) 由非终结符 C 推导出 $w_{j+1} \cdots w_k$，其概率为 $\alpha_{(j+1)k}(C)$。

9.6 PCFG的三个问题

考虑到B, C 和k 的任意性，在计算$\beta_{ik}(B)$ 时，必须考虑所有可能的B, C 和k，因此，计算概率时必须考虑所有情况下的概率之和。

同样方法，可以计算出运用产生式 $B \rightarrow CA$ 推导出 $w_k \cdots w_{i-1} w_i \cdots w_j$ 的概率。

9.6 PCFG的三个问题



9.6 PCFG的三个问题

➢外向算法描述：

输入：PCFG $G=(S, N, T, P)$，语句 $W = w_1 w_2 \cdots w_n$ ；

输出：$\beta_{ij}(A)$, $A \in N$, $1 \le i \le j \le n$。

(1) 初始化：$\beta_{1n}(A) = \delta(A, S)$，$A \in N$；

(2) 归纳：j 从n-1 到0，i 从1 到 n-j，重复计算：

$$\beta_{i(i+j)}(A) = \sum_{B,C} \sum_{i+j<k\le n} p(B \to AC)\alpha_{(i+j+1)k}(C)\beta_{ik}(B)$$

$$+\sum_{B,C} \sum_{1\le k<i} p(B \to CA)\alpha_{k(i-1)}(C)\beta_{k(i+j)}(B)$$

(3) 终结：$p(S \overset{*}{\Rightarrow} w_1 w_2 \cdots w_n) = \beta_{1n}(S)$

9.6 PCFG的三个问题

2、Viterbi 算法解决第二个问题—最佳分析结果搜索

注意：γ区别于内向算法中的α。

- **定义：**

Viterbi 变量 $\gamma_{ij}(A)$ 是由非终结符 ***A*** 推导出语句 ***W*** 中子字串 $w_i w_{i+1} \cdots w_j$ 的最大概率。

变量 $\psi_{i,j}$ 用于记忆字串 $W = w_1 w_2 \cdots w_n$ 的 **Viterbi** 语法分析结果。

9.6 PCFG的三个问题

- **Viterbi 算法描述：**

输入：文法 G(*S*)，语句 $W = w_1 w_2 \cdots w_n$

输出：$\gamma_{1n}(S)$

(1) 初始化： $\gamma_{ii}(A) = p(A \to w_i)$ $\quad A \in V_N$，$1 \le i \le j \le n$

(2) 归纳计算： j=1..n, i=1..n- j, 重复下列计算：

$$\gamma_{i(i+j)}(A) = \max_{B,C \in V_N; i \le k \le i+j} p(A \to BC)\gamma_{ik}(B)\gamma_{(k+1)(i+j)}(C)$$

$$\psi_{i(i+j)}(A) = \max_{B,C \in V_N; i \le k \le i+j} p(A \to BC)\gamma_{ik}(B)\gamma_{(k+1)(i+j)}(C)$$

(3) 终结： $p(S \overset{*}{\Rightarrow} w_1 w_2 \cdots w_n) = \gamma_{1n}(S)$

9.6 PCFG的三个问题

3、内外向算法解决第三个问题－参数估计

- **基本思路**

如果有大量已标注语法结构的训练语料，则可直接通过计算每个语法规则的使用次数，用最大似然估计方法计算 **PCFG** 规则的概率参数，即：

$$\hat{p}(N^j \rightarrow \zeta) = \frac{C(N^j \rightarrow \zeta)}{\sum_{\gamma} C(N^j \rightarrow \gamma)}$$

9.6 PCFG的三个问题

多数情况下，没有可利用的标注语料，只好借助**EM** (Expectation Maximization) 迭代算法估计**PCFG**的概率参数。

初始时随机地给参数赋值，得到语法G_0，依据 G_0 和训练语料，得到语法规则使用次数的期望值，以期望次数运用于最大似然估计，得到语法参数新的估计值，由此得到新的语法 G_1，由 G_1 再次得到语法规则的使用次数的期望值，然后又可以重新估计语法参数。循环这个过程，语法参数将收敛于最大似然估计值。

9.6 PCFG的三个问题

◆内外向算法

给定 CFG G 和训练数据 $W = w_1 w_2 \cdots w_n$， 语法规则 $A \to BC$ 使用次数的期望值为：

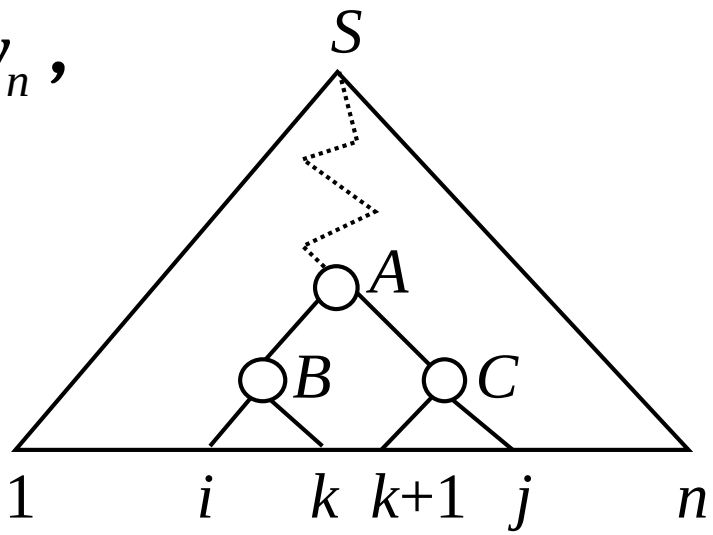
$$Count(A \to BC) = \sum_{1 \le i \le k \le j \le n} p(A_{ij}, B_{ik}, C_{(k+1)j} \mid w_1 \cdots w_n, G)$$

$$= \frac{1}{p(w_1 \cdots w_n \mid G)} \sum_{1 \le i \le k \le j \le n} p(A_{ij}, B_{ik}, C_{(k+1)j}, w_1 \cdots w_n \mid G)$$

$$= \frac{1}{p(w_1 \cdots w_n \mid G)} \sum_{1 \le i \le k \le j \le n} \beta_{ij}(A)\, p(A \to BC)\, \alpha_{ik}(B)\, \alpha_{(k+1)j}(C) \quad \ldots (NW1)$$

9.6 PCFG的三个问题

➢解释：给定语句 $W = w_1 \cdots w_n$，PCFG G 中产生式 $A \rightarrow BC$ 在产生 W 的过程中被使用次数的期望值为：在所有可能的情况下，即在条件：$1 \le i \le k \le j \le n$ 下，W 的语法分析结构中 $w_i \cdots w_k$ 由 B 导出，$w_{k+1} \cdots w_j$ 由 C 导出，$w_i \cdots w_j$ 由 A 导出的概率总和。



9.6 PCFG的三个问题

类似地，语法规则 $A \rightarrow a$ 的使用次数的期望值为：

$$Count(A \rightarrow a) = \sum_{1 \le i \le n} p(A_{ii} \mid w_1 \cdots w_n, G)$$

$$= \frac{1}{p(w_1 \cdots w_n \mid G)} \sum_{1 \le i \le n} p(A_{ii}, w_1 \cdots w_n \mid G)$$

$$= \frac{1}{p(w_1 \cdots w_n \mid G)} \sum_{1 \le i \le n} \beta_{ii}(A)\, p(A \rightarrow a)\, \delta(a, w_i)$$

… (NW2)

9.6 PCFG的三个问题

G 的参数可由下面的公式重新估计：

$$\hat{p}(A \to \mu) = \frac{Count(A \to \mu)}{\sum_{\mu} Count(A \to \mu)} \quad \ldots \text{(NW3)}$$

其中，μ 要么为终结符号，要么为两个非终结符号串，即 $A \to \mu$ 为乔姆斯基语法范式要求的两种形式。

9.6 PCFG的三个问题

➢**内外向算法：**

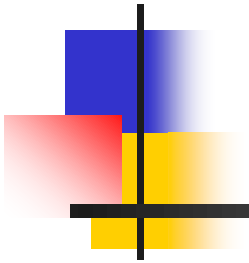
(1) 初始化：随机地给 $p(A \to \mu)$ 赋值，使得 $\sum_{\mu} p(A \to \mu) = 1$，由此得到语法 G_0。令 $i=0$；

(2) EM迭代：

E-步：由 G_i 根据公式(NW1)和(NW2)，计算期望值 $Count(A \to BC)$ 和 $Count(A \to a)$；

M-步：用 E-步所得的期望值，根据公式(NW3)重新估计 $p(A \to \mu)$，得到 G_{i+1}。

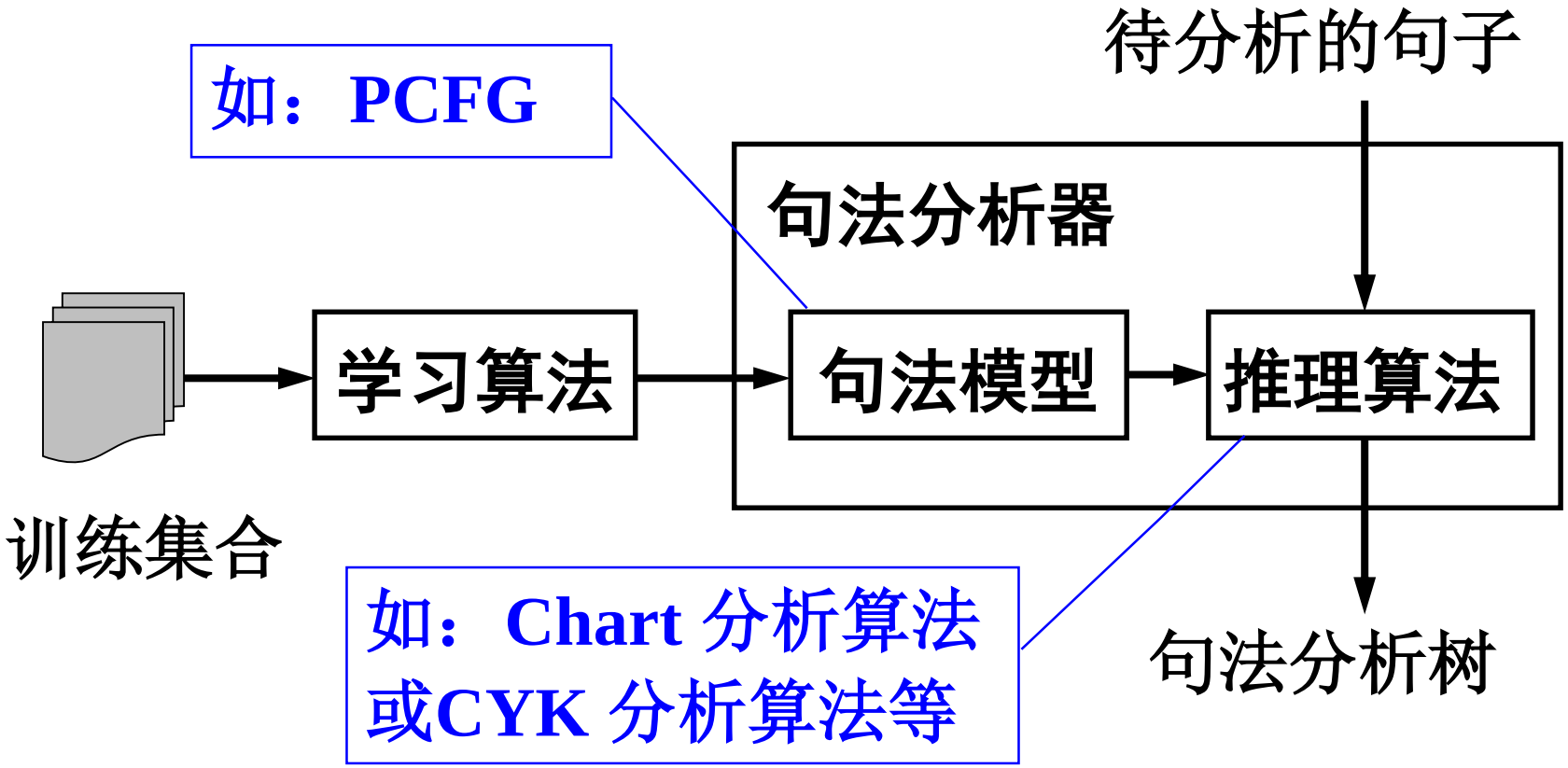
(3) 循环：$i=i+1$，重复EM步骤，直至 $p(A \to \mu)$ 值收敛。



9.7 基于PCFG的句法分析实例

9.7 基于PCFG的句法分析实例

统计句法分析器实现的一般方法：



9.7 基于PCFG的句法分析实例

给定如下 PCFG G(S)：

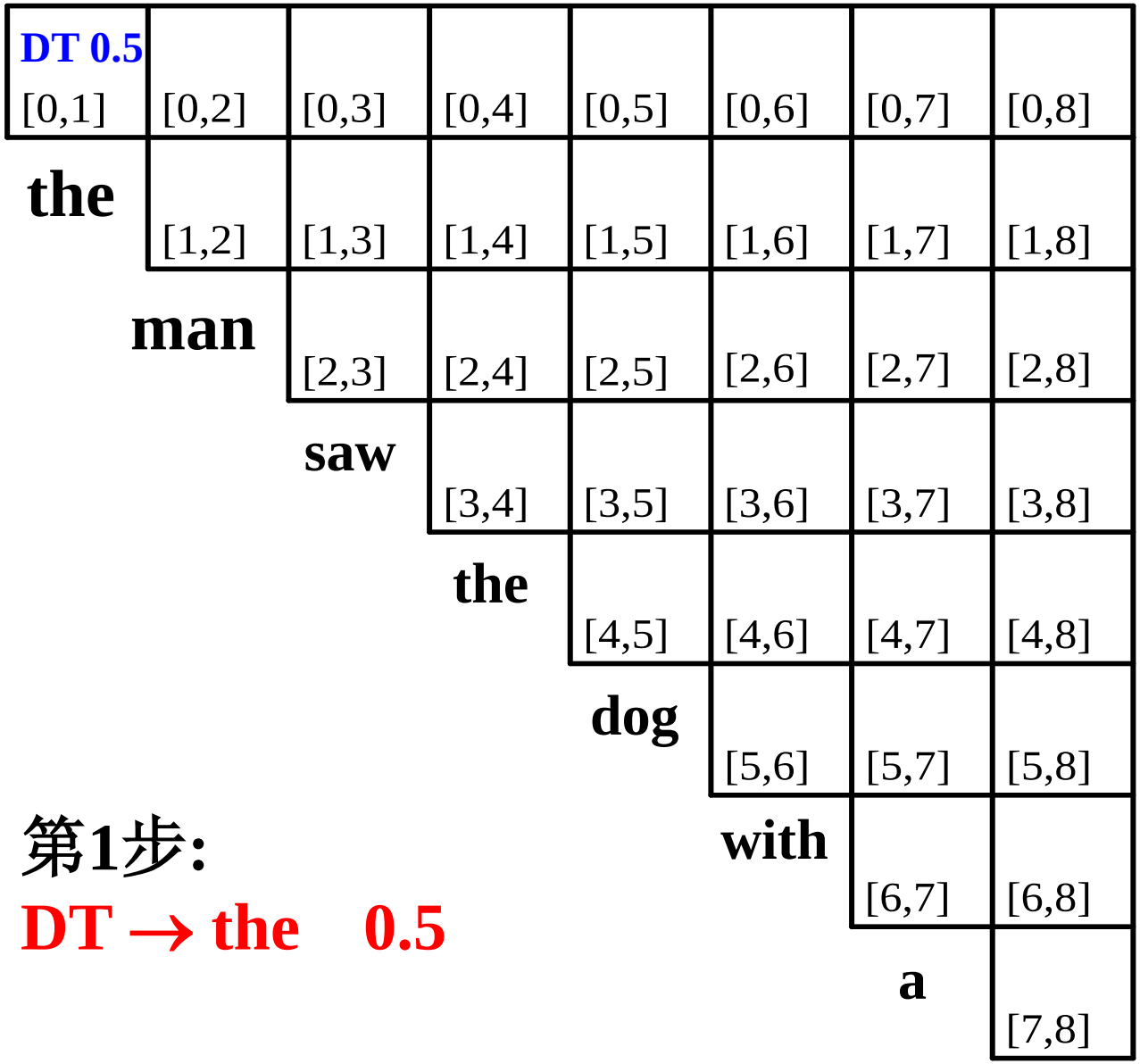
非终结符集合：N = {S, NP, VP, PP, DT, Vi, Vt, NN, IN}

终结符集合：= {sleeps, saw, man, woman, dog, telescope, the, with, in}

规则集：

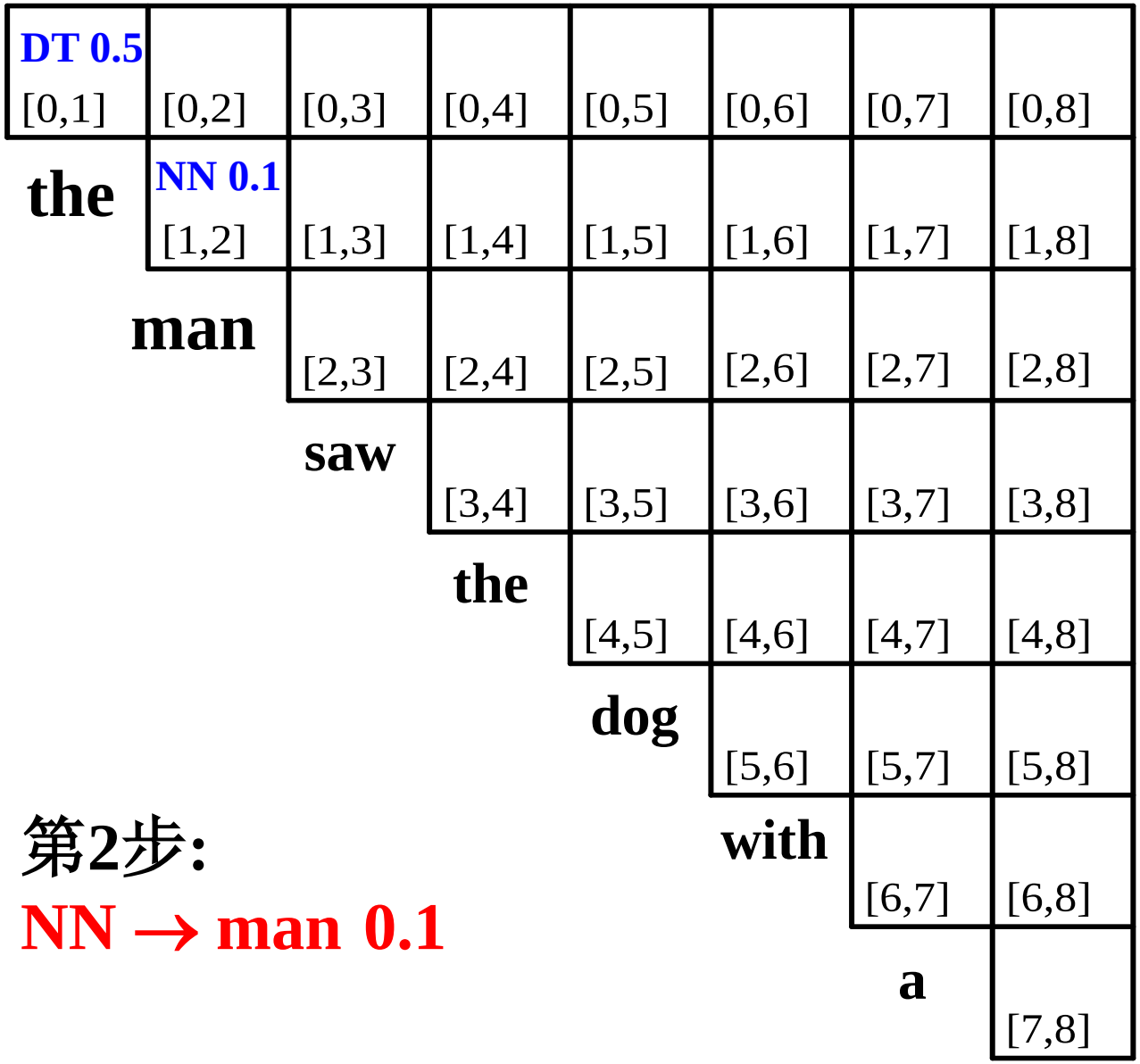
(1) S → NP VP	1.0	(9) Vt → saw	1.0
(2) VP → Vi	0.3	(10) NN → man	0.1
(3) VP → Vt NP	0.4	(11) NN → woman	0.1
(4) VP →VP PP	0.3	(12) NN → telescope	0.3
(5) NP → DT NN	0.8	(13) NN → dog	0.5
(6) NP → NP PP	0.2	(14) DT → the	0.5
(7) PP → IN NP	1.0	(15) DT → a	0.5
(8) Vi → sleeps	1.0	(16) IN → with	0.6
		(17) IN → in	0.4

输入句子：the man saw the dog with a telescope


DT 0.5
[0,1]
[0,2]
[0,3]
[0,4]
[0,5]
[0,6]
[0,7]
[0,8]
the
[1,2]
[1,3]
[1,4]
[1,5]
[1,6]
[1,7]
[1,8]
man
[2,3]
[2,4]
[2,5]
[2,6]
[2,7]
[2,8]
saw
[3,4]
[3,5]
[3,6]
[3,7]
[3,8]
the
[4,5]
[4,6]
[4,7]
[4,8]
dog
[5,6]
[5,7]
[5,8]
with
[6,7]
[6,8]
a
[7,8]
telescope

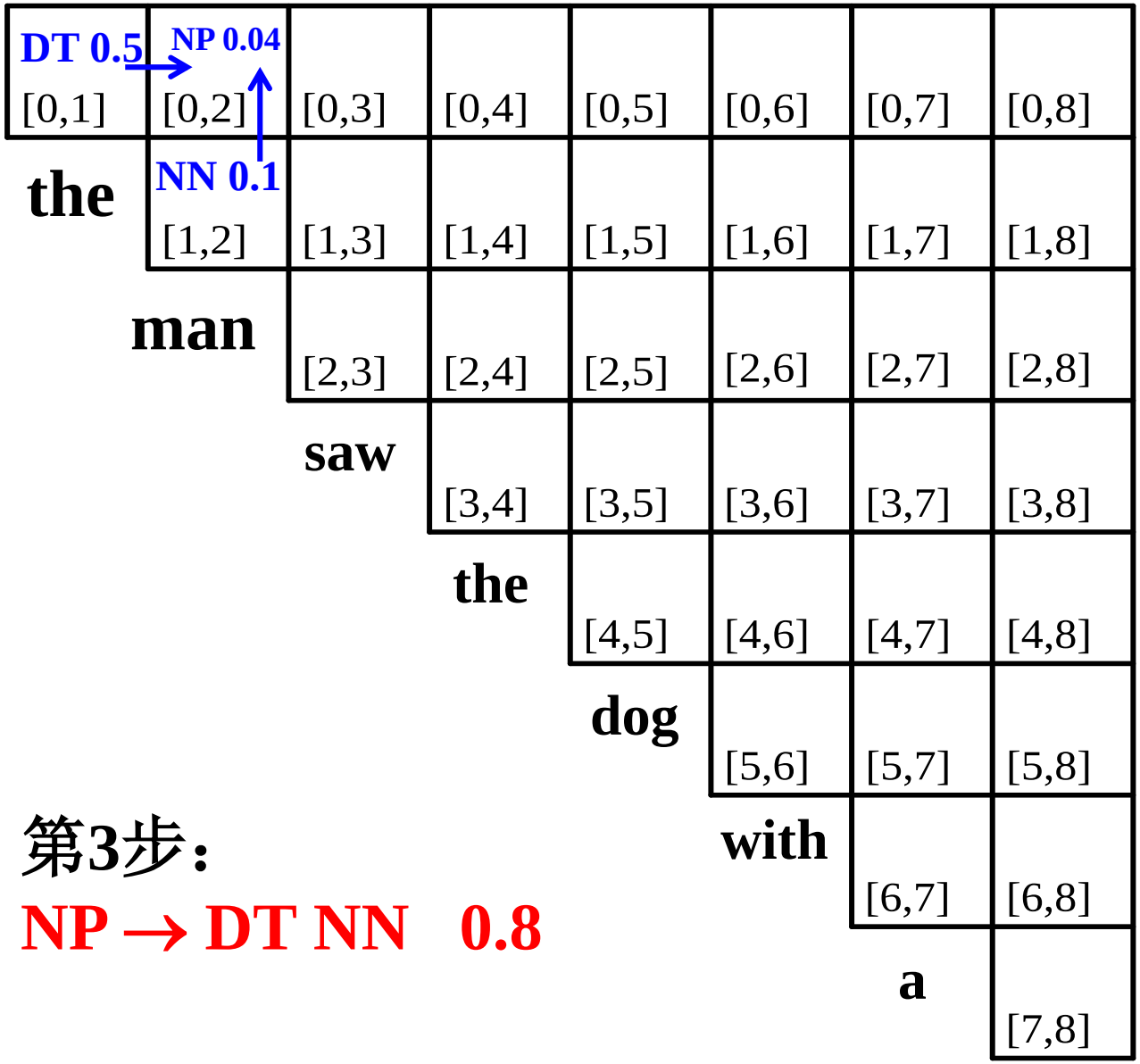
第1步:

DT → the 0.5


DT 0.5
[0,1]
[0,2]
[0,3]
[0,4]
[0,5]
[0,6]
[0,7]
[0,8]
the
NN 0.1
[1,2]
[1,3]
[1,4]
[1,5]
[1,6]
[1,7]
[1,8]
man
[2,3]
[2,4]
[2,5]
[2,6]
[2,7]
[2,8]
saw
[3,4]
[3,5]
[3,6]
[3,7]
[3,8]
the
[4,5]
[4,6]
[4,7]
[4,8]
dog
[5,6]
[5,7]
[5,8]
with
[6,7]
[6,8]
a
[7,8]
telescope

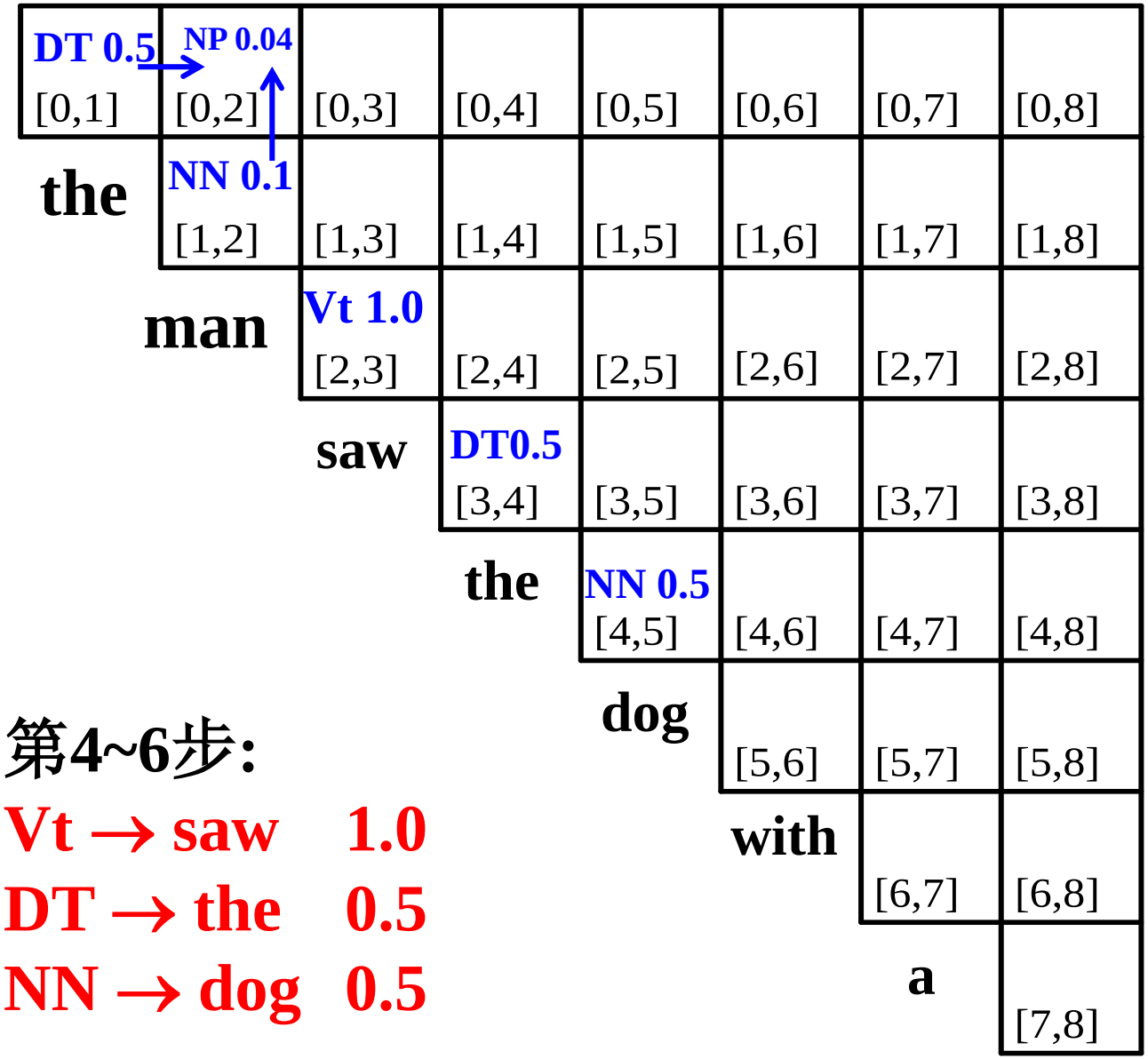
第2步:

NN → man 0.1



第3步：

NP → DT NN　0.8

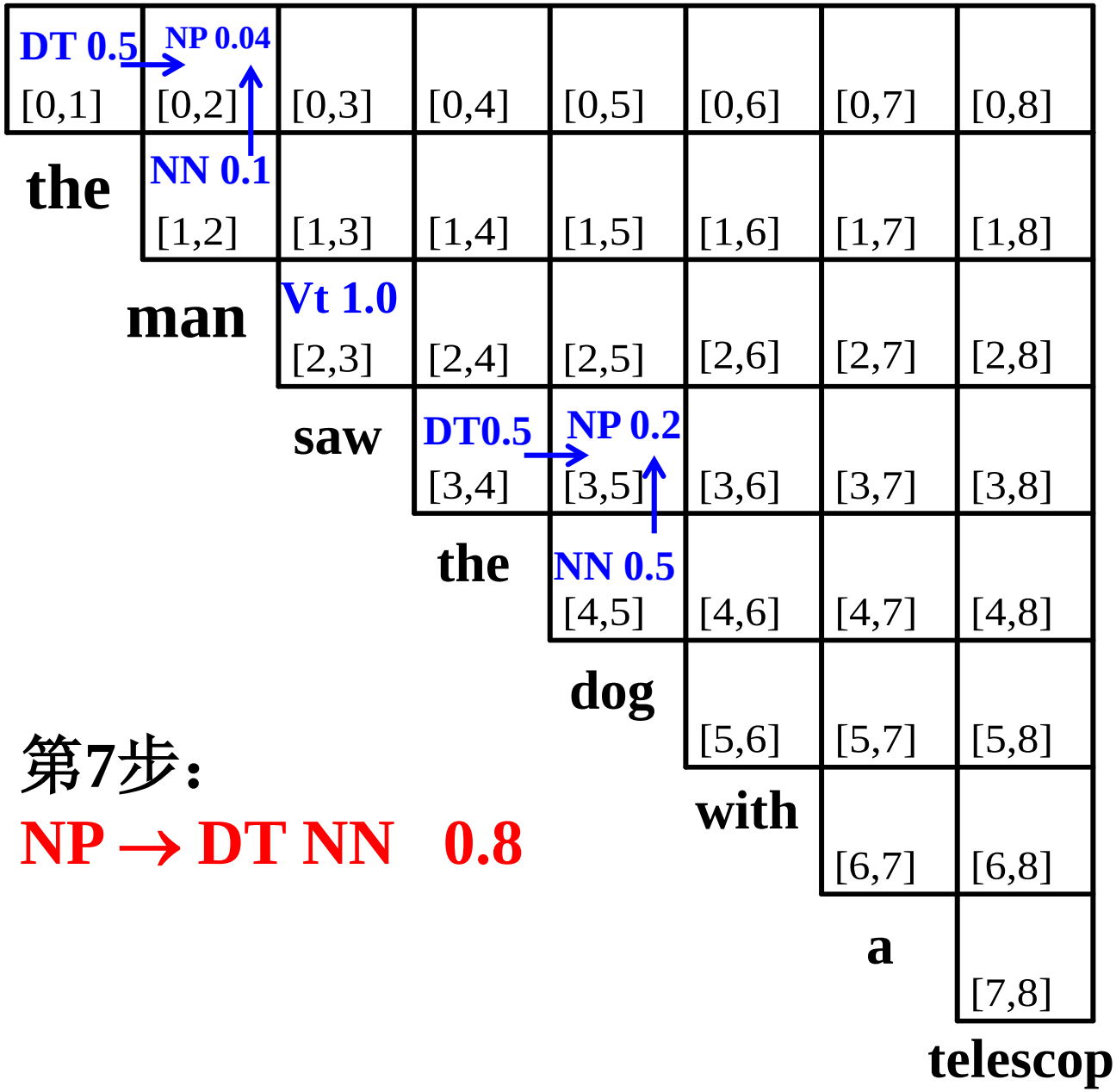


第4~6步：

Vt → saw 1.0

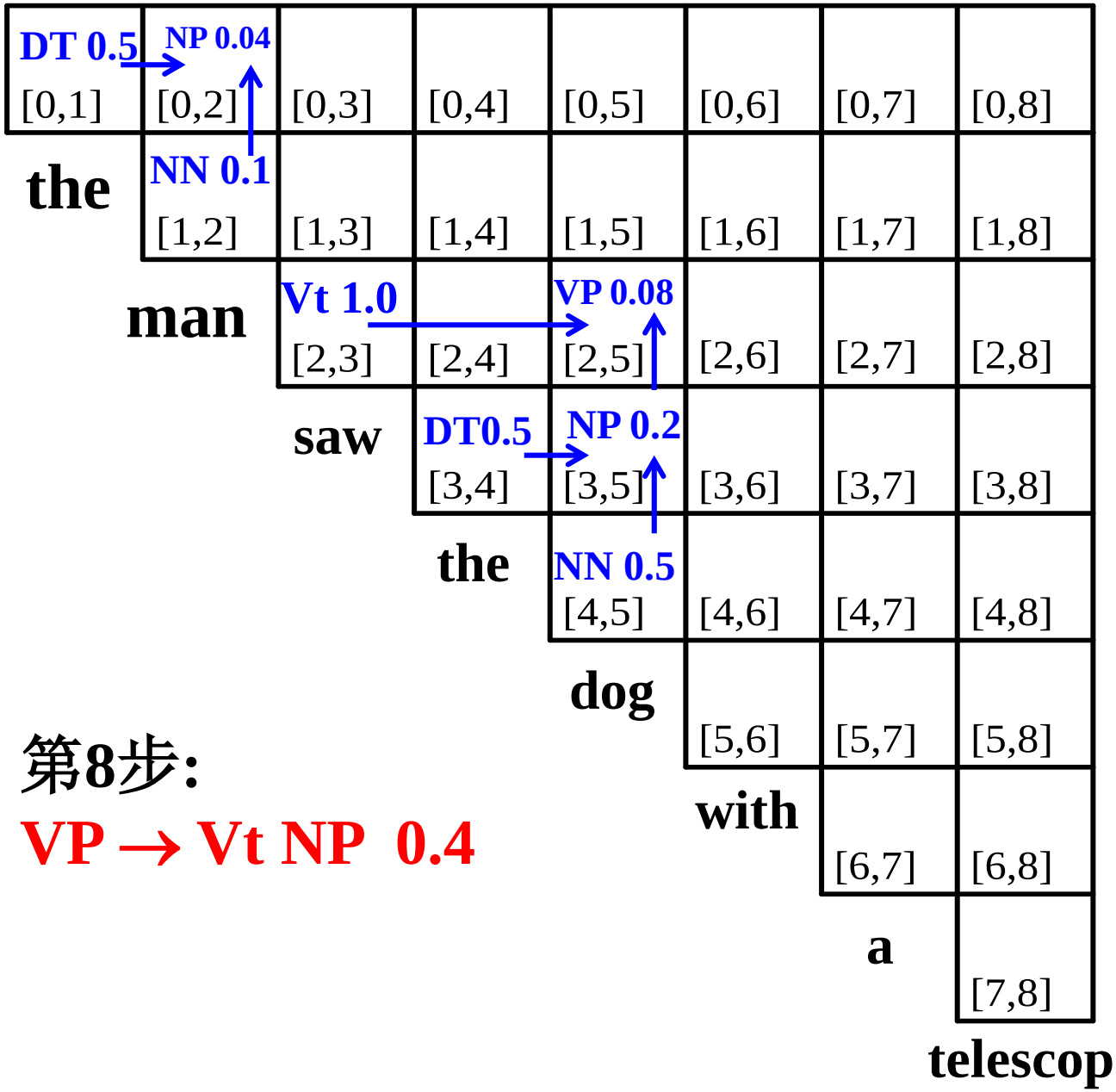
DT → the 0.5

NN → dog 0.5



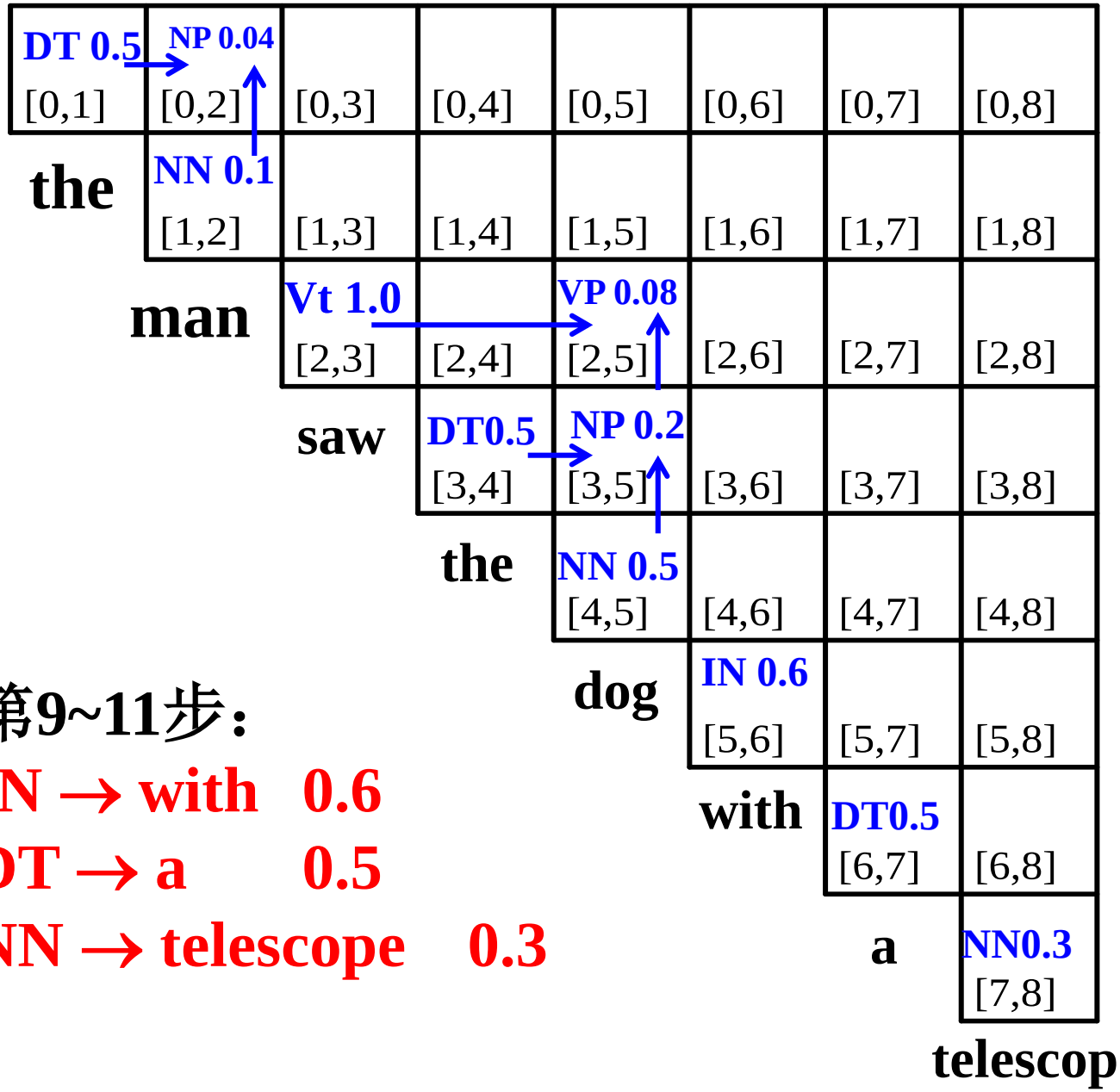
第7步：

NP → DT NN　0.8



第8步：

VP → Vt NP 0.4

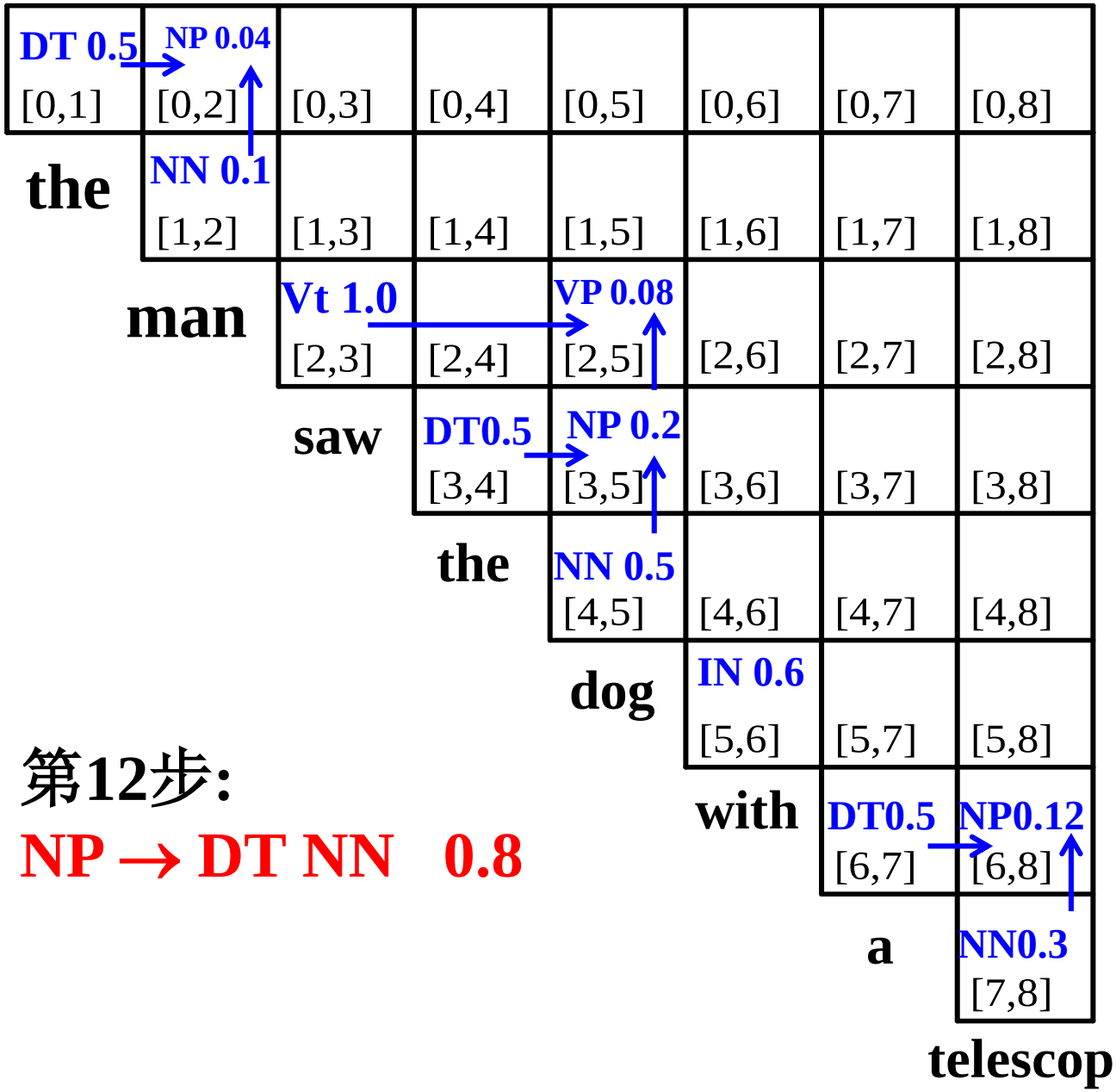


第9~11步：

IN → with 0.6

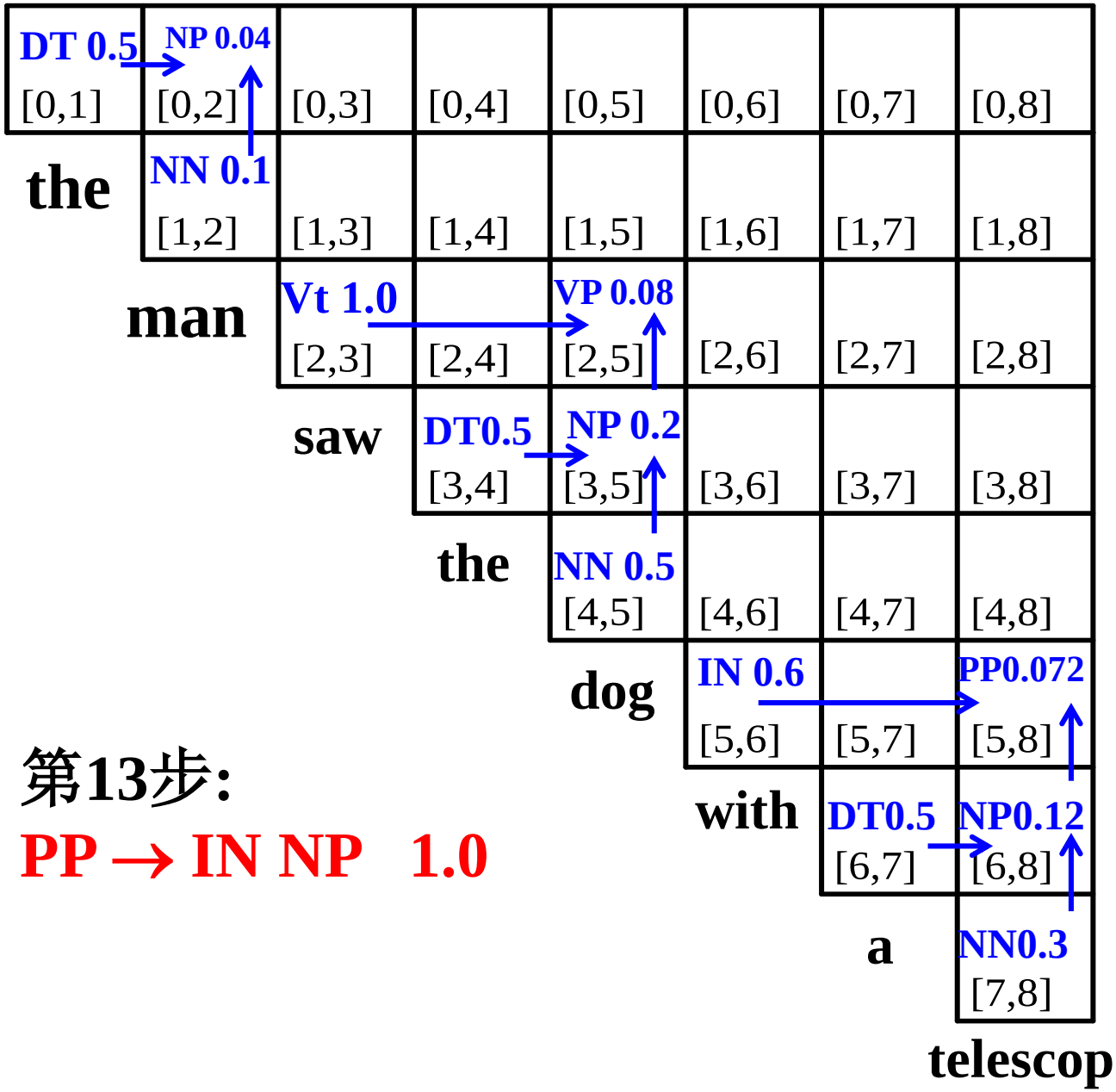
DT → a 0.5

NN → telescope 0.3



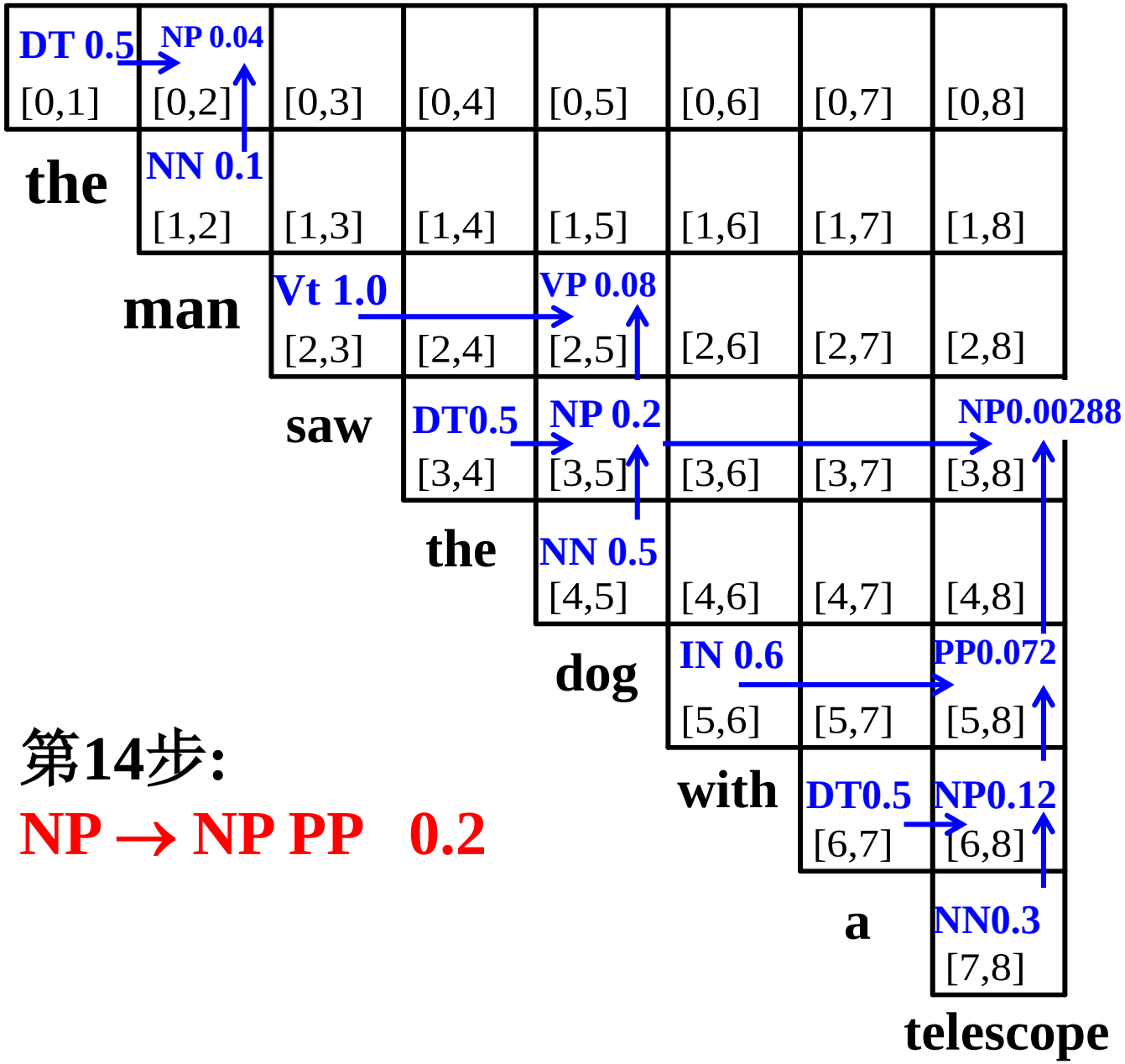
第12步:

NP → DT NN 0.8



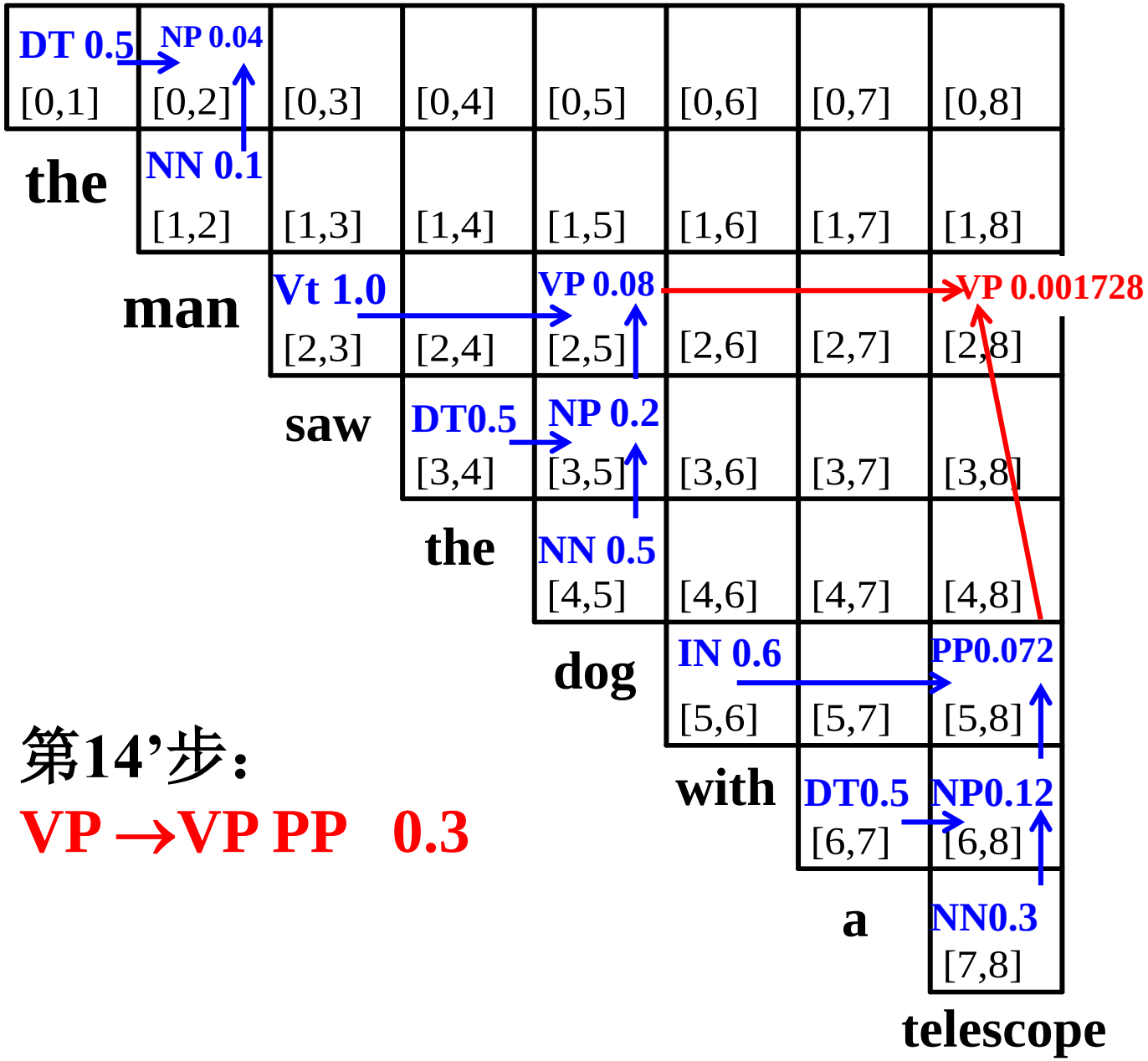
第13步:

PP → IN NP 1.0



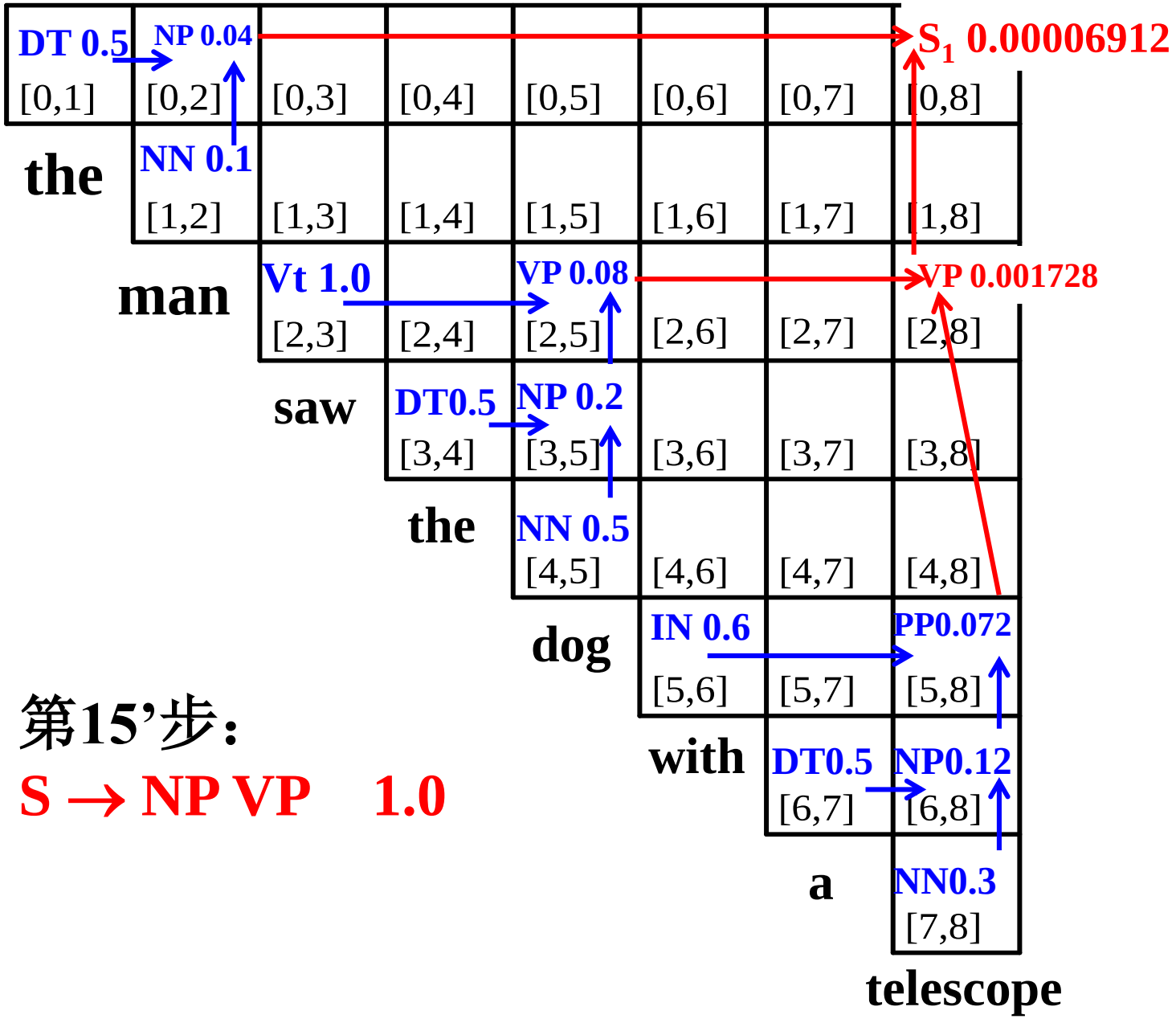
第14步:

NP → NP PP 0.2



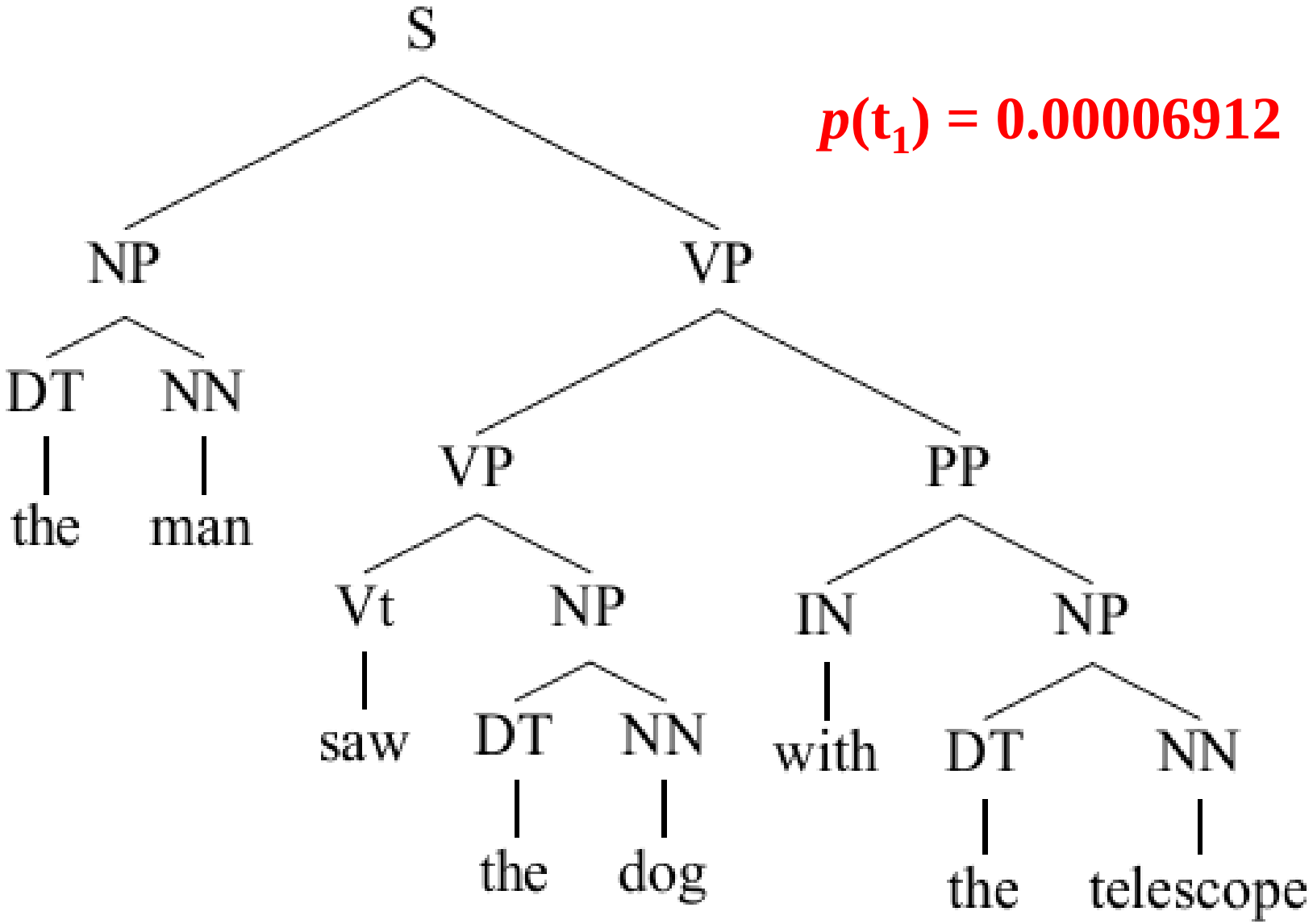
第14’步：

VP →VP PP 0.3



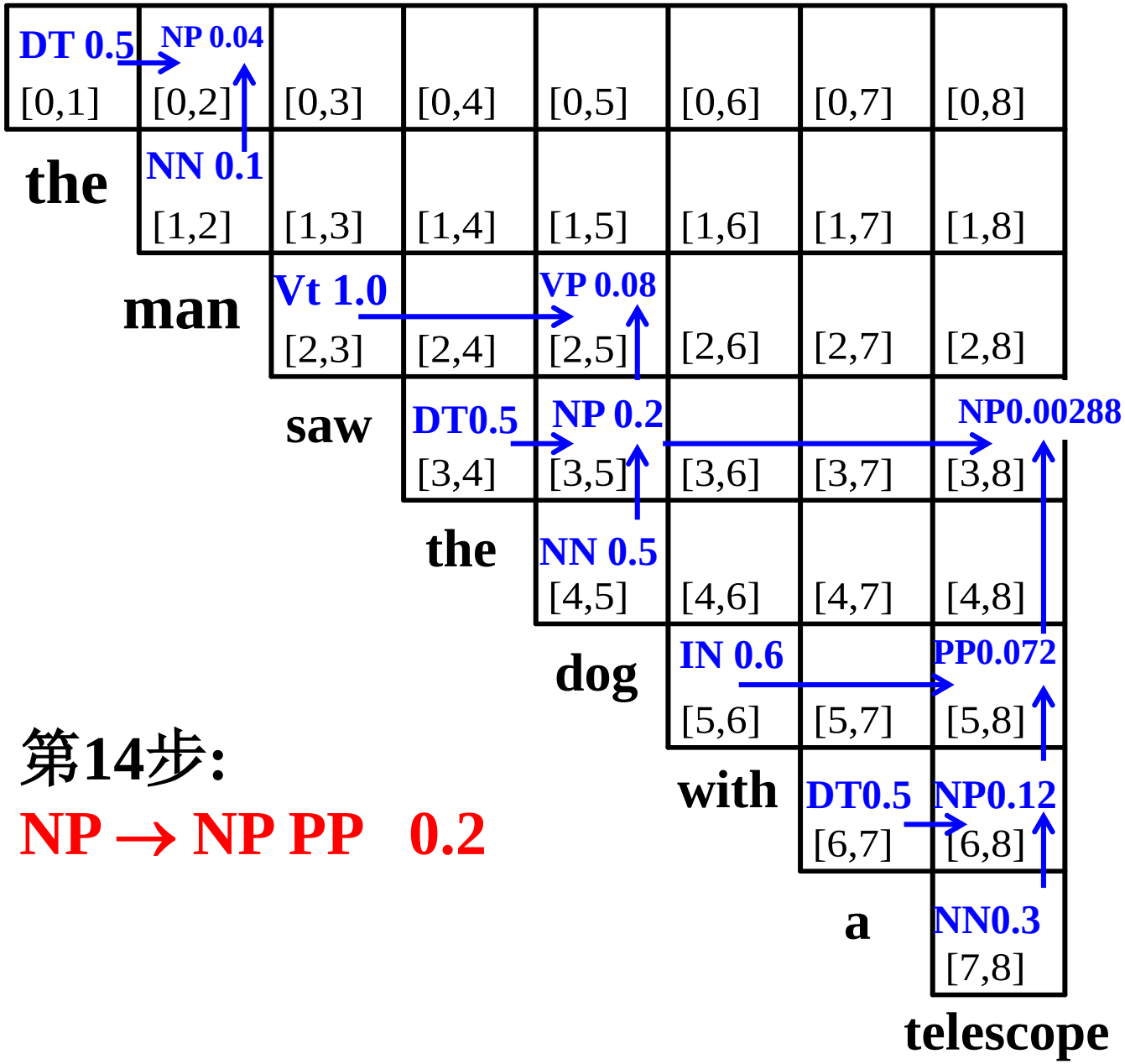
第15’步：

S → NP VP 1.0



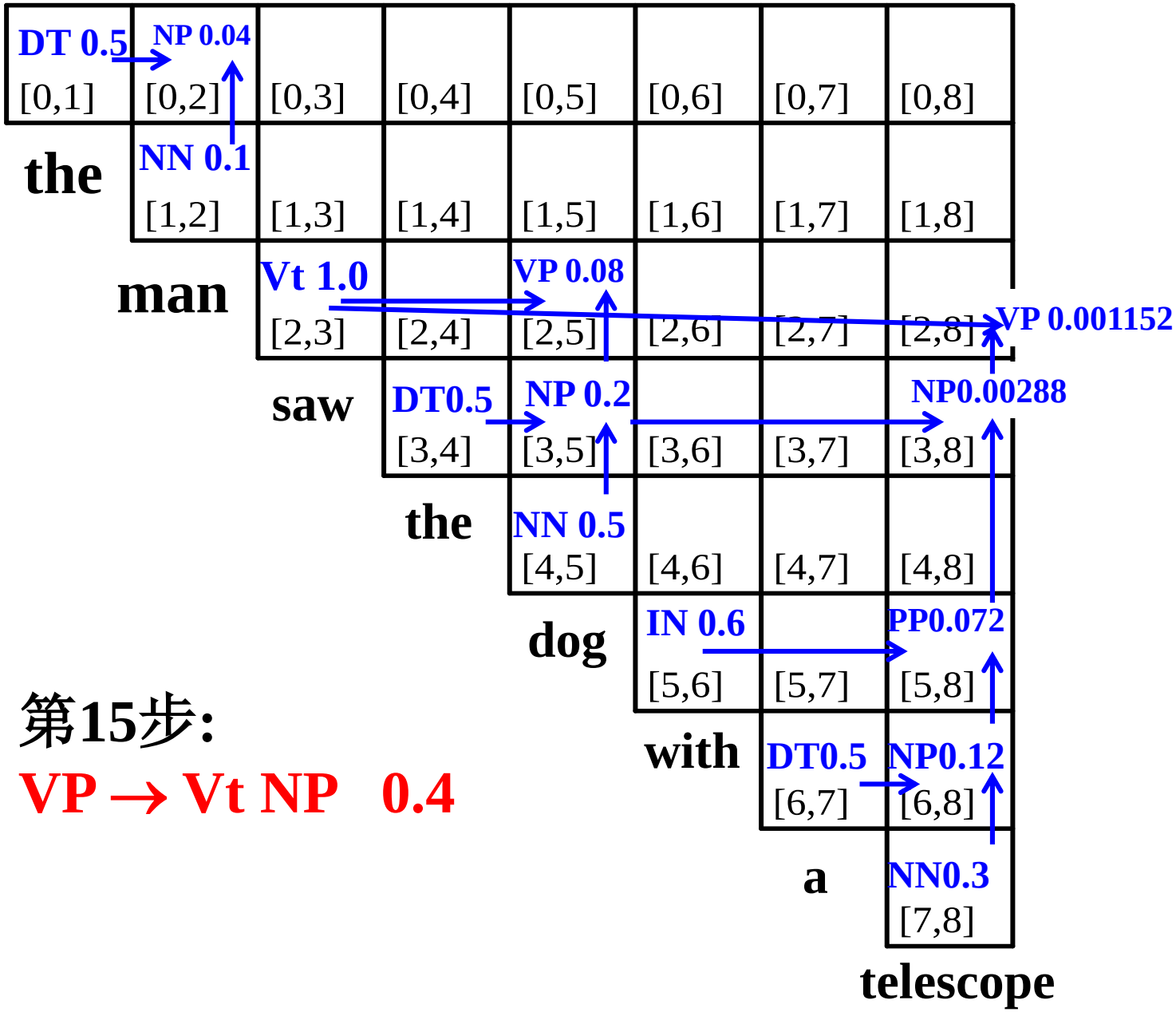
$p(t_1) = 0.00006912$

句法树t_1



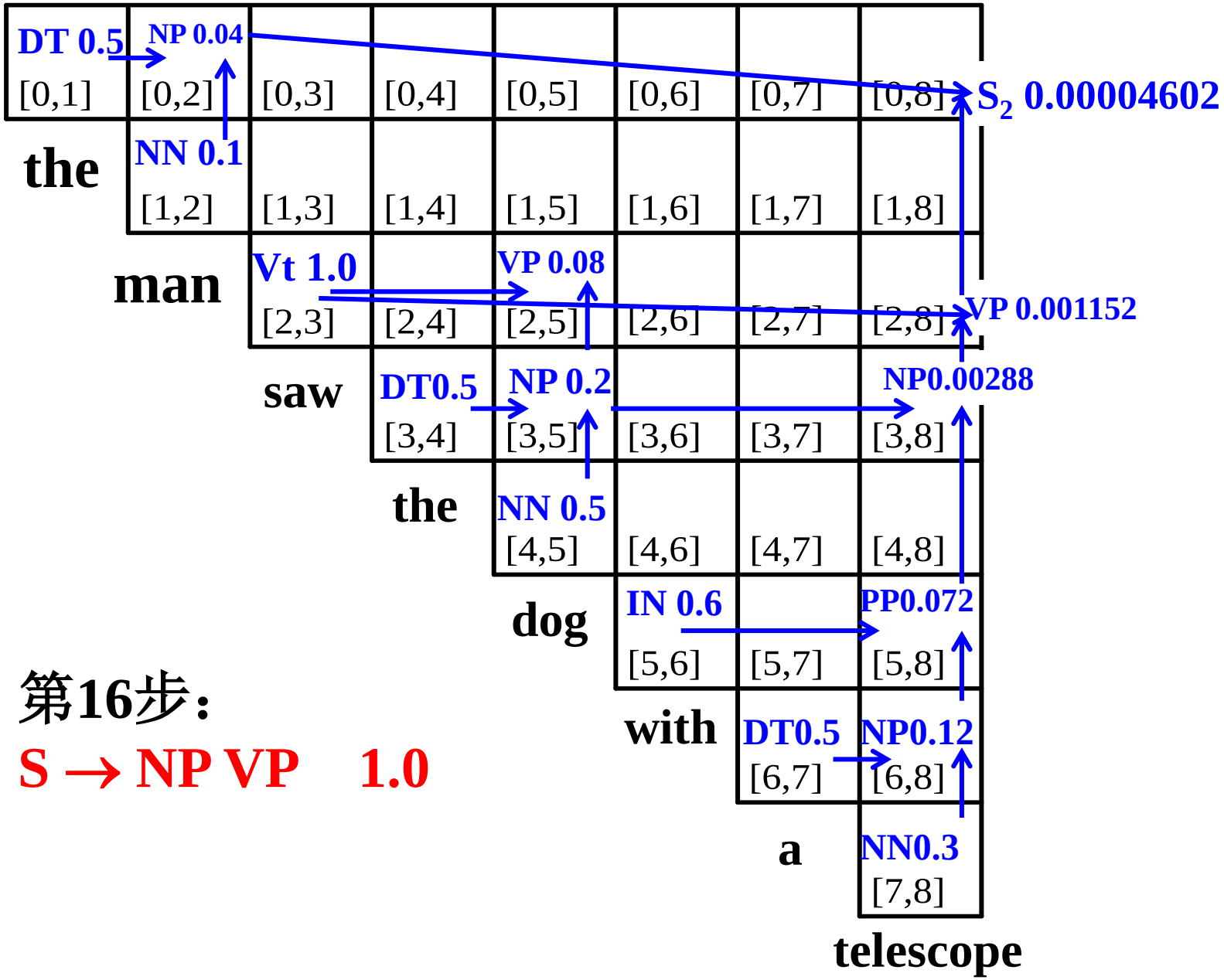
第14步:

NP → NP PP 0.2



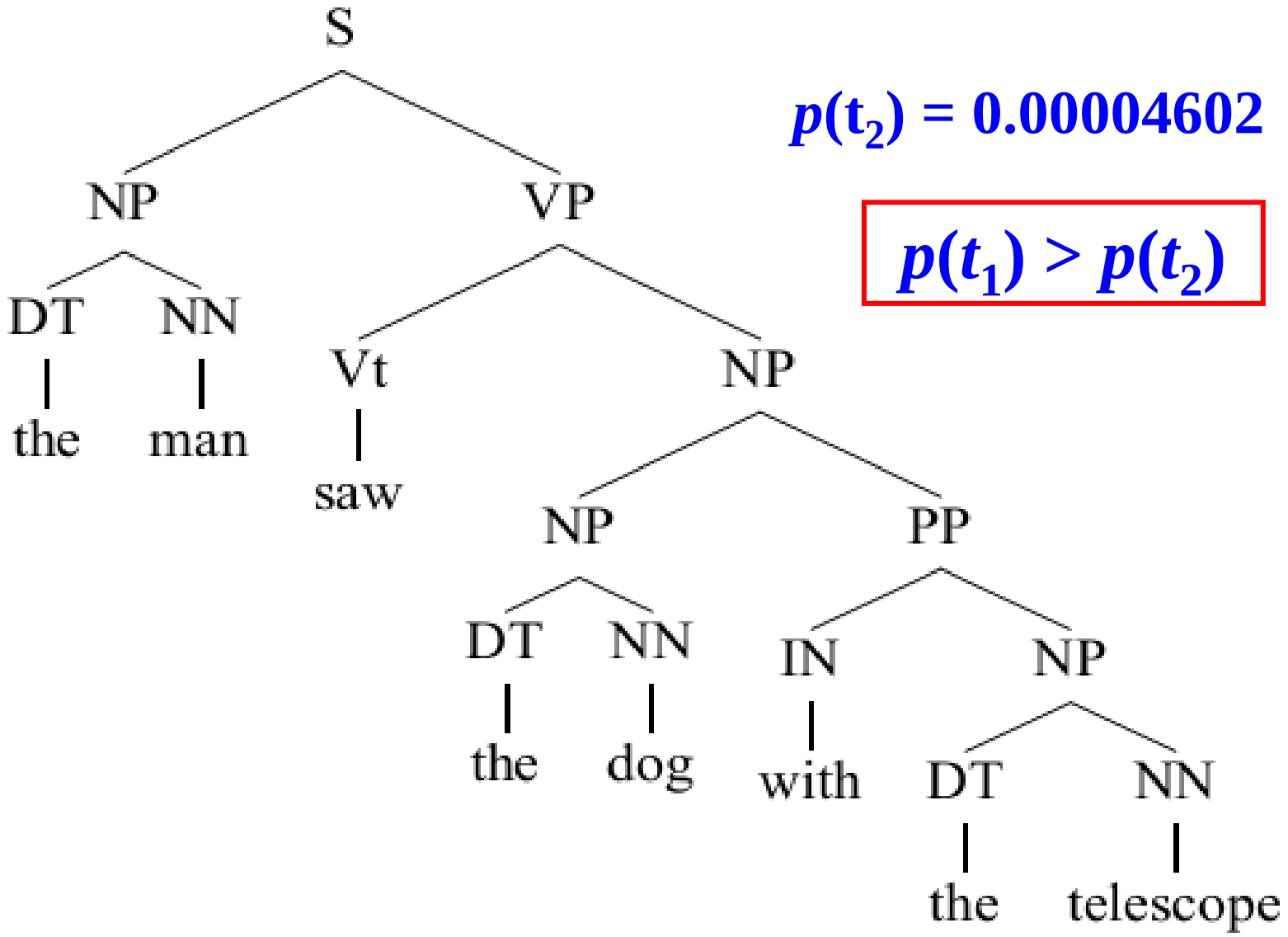
第15步:

VP → Vt NP 0.4



第16步：

S → NP VP 1.0



$p(t_2) = 0.00004602$

$p(t_1) > p(t_2)$

句法树t_2

9.7 基于PCFG的句法分析实例

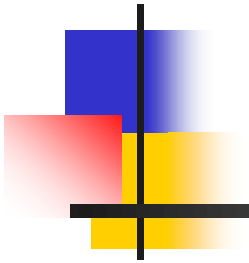
◆ PCFG 的评价

➢ 优点：

- 可利用概率减少分析过程的搜索空间；
- 可利用概率对概率较小的子树剪枝，加快分析效率；
- 可以定量地比较两个语法的性能。

➢ 弱点：

- 分析树的概率计算条件非常苛刻，甚至不够合理。



9.8 短语结构分析方法评估

9.8 短语结构分析方法评估

◆内部评测 (intrinsic evaluation)

一对评测方法本身的评测，用于指导句法分析系统及其语法的开发过程。主要指标有：

- 语法的覆盖性 (grammatical coverage)
- 平均分析基数 (average parse base)
- 结构一致性 (structural consistency)
- 排序的一致性 (best-first/ranked consistency)等

9.8 短语结构分析方法评估

◆对比评测 (comparative evaluation)

用于对比不同系统之间的性能差别，主要评测指标和方法包括：

- 树相似性 (tree similarity)
- 模型的熵 (或困惑度) (entropy/perplexity)
- 语法评估兴趣小组 (grammar evaluation interest group, GIEG)的评测方案等

9.8 短语结构分析方法评估

◆句法分析器性能评测

目前使用比较广泛的句法分析器评价指标是PARSEVAL测度，三个基本的评测指标：

- 精度(precision)：句法分析结果中正确的短语个数所占的比例，即分析结果中与标准分析树（答案）中的短语相匹配的个数占分析结果中所有短语个数的比例，即：

9.8 短语结构分析方法评估

$$P=\frac{\text{分析得到的正确的短语个数}}{\text{分析得到的所有的短语个数}}\times 100\%$$

- 召回率(recall)：句法分析结果中正确的短语个数占标准分析树中全部短语个数的比例，即：

$$R=\frac{\text{分析得到的正确的短语个数}}{\text{标准树库中(答案)的短语个数}}\times 100\%$$

- F-measure：$F=\frac{(\beta^2+1)\times P\times R}{\beta^2\times P+R}\times 100\%$

一般地，$\beta=1$，称作 $F1$ 测度。

9.8 短语结构分析方法评估

- 交叉括号数 (crossing brackets)：一棵分析树中与其他分析树中边界相交叉的成分个数的平均值。

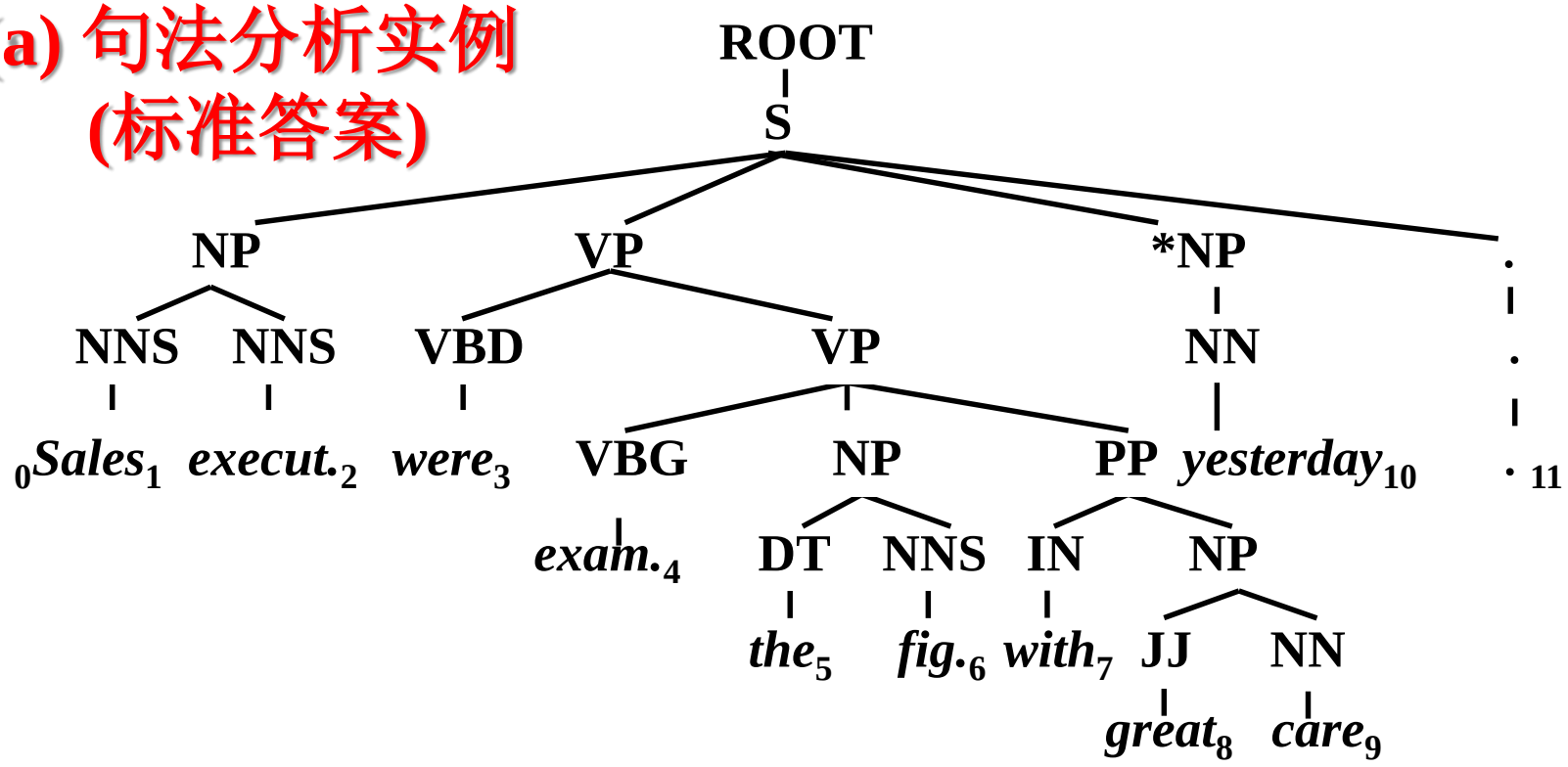
分析树中除了词性标注符号以外的其他非终结符节点通常采用如下标记格式：XP-(起始位置：终止位置)。其中，XP为短语名称；(起始位置：终止位置)为该节点的跨越范围，起始位置指该节点所包含的子节点的起始位置，终止位置为该节点所包含的子节点的终止位置。在计算PARSEVAL指标时，通常需要计算分析结果与标准分析树之间括号匹配的数目或括号交叉的数目。

9.8 短语结构分析方法评估

- 例如：下面的图(a)为句子“*Sales executives were examining the figures with great care yesterday.*”的正确分析树(答案标准)。

9.8 短语结构分析方法评估

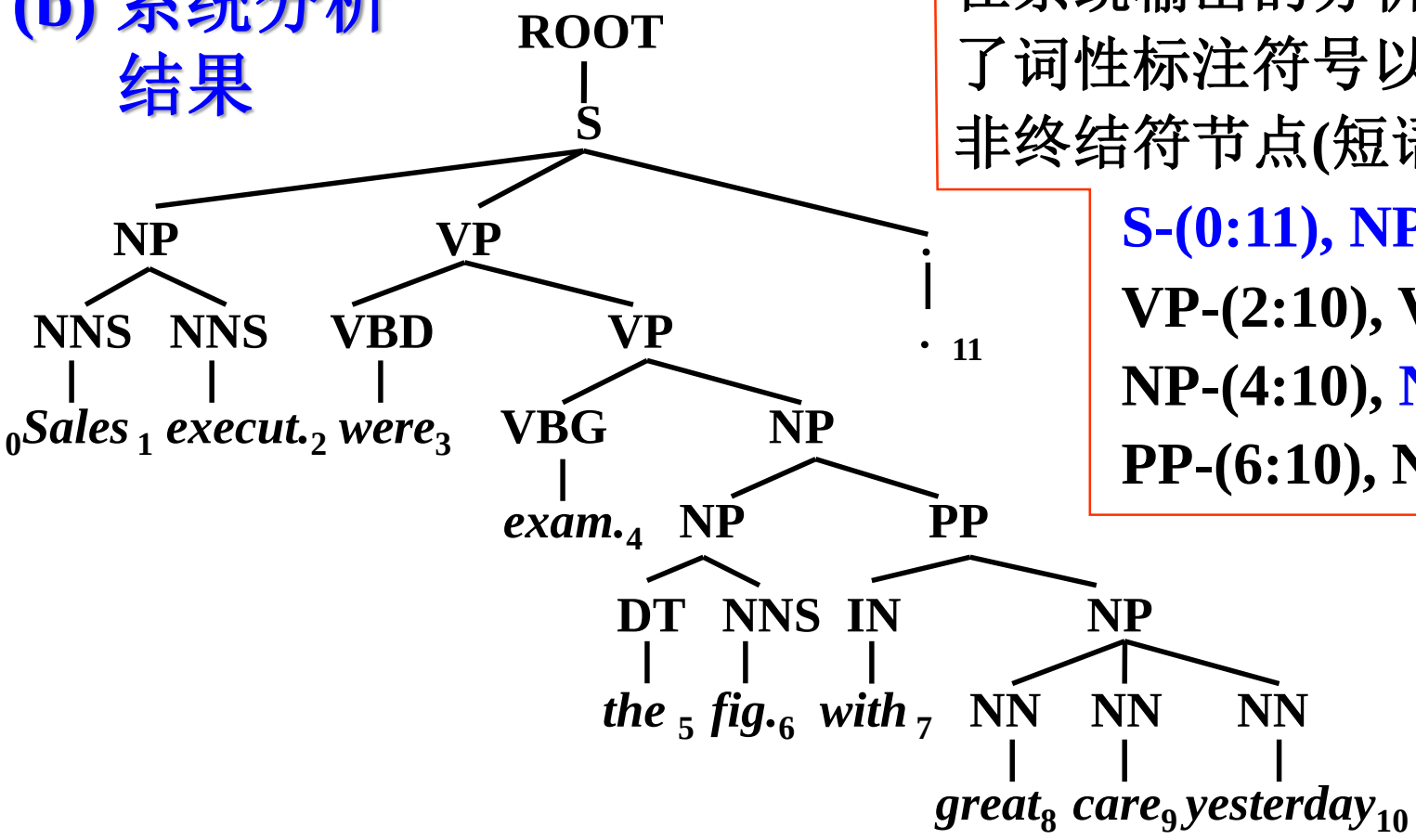
(a) 句法分析实例 (标准答案)



在标准答案树中，除了词性标注符号以外(即除了叶子节点和其直接父节点以外)的其他非终结符节点(短语)有：**S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), NP-(4:6), PP-(6:9), NP-(7:9), *NP-(9:10)** 。

9.8 短语结构分析方法评估

(b) 系统分析结果



在系统输出的分析树中，除了词性标注符号以外的其他非终结符节点(短语)有：

S-(0:11), NP-(0:2), VP-(2:10), VP-(3:10), NP-(4:10), NP-(4:6), PP-(6:10), NP-(7:10)。

9.8 短语结构分析方法评估

标准答案：S-(0:11), NP-(0:2), VP-(2:9), VP-(3:9), NP-(4:6), PP-(6:9), NP-(7:9), *NP-(9:10)

系统结果：S-(0:11), NP-(0:2), VP-(2:10), VP-(3:10), NP-(4:10), NP-(4:6), PP-(6:10), NP-(7:10)

只有这3个短语与标准答案完全一样，因此，

$$\text{Precision} = \frac{3}{8} \times 100\% = 37.5\%$$

$$\text{Recall} = \frac{3}{8} \times 100\% = 37.5\%$$

注：图(a)中加*号的一元的节点 (*NP) 在计算时应该被排除在外，但在这里也被包括进来了。

9.8 短语结构分析方法评估

- 另外两个指标：

① 词性标注的准确率(tagging accuracy)。在该句子的分析树中，11个词中除了*great* 被错误地标注以外，其他的词性标注均为正确的，因此，词性标注准确率为10/11＝90.9%

②交叉括号数(crossing brackets)和交叉准确率(crossing accuracy)：在本例中不存在交叉括号，因此，交叉括号数为0，交叉准确率为100%。

9.8 短语结构分析方法评估

PARSEVAL评测方法的区分能力不是很强，而且在某些特殊情况下计算出的正确率和召回率存在较大的偏差。因此，有关专家提出了一些其他的评测方法，如基于依存结构的评测方法[D. Lin, 1995]、基于语法关系的评测方法[J. Carroll, 1998]等等 。

Lin, Dekang. 1995. A dependency-based method for evaluating broad-coverage parsers. In *Proceedings of the IJCAI-95*, Montreal, Canada, pp 1420-1425

Carroll, John, Ted Briscoe, and Antonio Sanfilippo. 1998. Parser Evaluation: a Survey and a New Proposal. In *Proceedings of the First Conference on Linguistic Resources*, pp 447-455

部分公开的短语结构分析器

- ✧ Berkeley Parser： http://nlp.cs.berkeley.edu/Main.html#Parsing
- ✧ Stanford Parser： http://nlp.stanford.edu/downloads/lex-parser.shtml
- ✧ Collins Parser： http://people.csail.mit.edu/mcollins/code.html
- ✧ Charniak Parser： http://www.cs.brown.edu/people/ec/#software
- ✧ Bikel Parser
  http://www.cis.upenn.edu/~dbikel/software.html#stat-parser
- ☼ Oboe Parser (可执行程序)
  http://www.openpr.org.cn/index.php/NLP-Toolkit-for-Natural-Language-Processing/

Thanks

进一步了解

基于PCFG分析方法的改进

基于PCFG分析方法的改进

1. 中心词驱动(Head-driven)模型

- 基本思想：每个非终结符标注上对应的中心词(及其词性)，例如：

$$VP \rightarrow VBD\ \ NP\ \ PP$$

$$\downarrow$$

$$VP(dumped, VBD) \rightarrow VBD(dumped, VBD)\ \ NP(sacks, NNS)\ \ PP(into, P)$$

将每条规则看作一个马尔科夫过程

(1) 由父节点生成中心子节点；

(2) 自右向左依次生成中心子节点左边的节点；

(3) 自左向右依次生成中心子节点右边的节点。

基于PCFG分析方法的改进

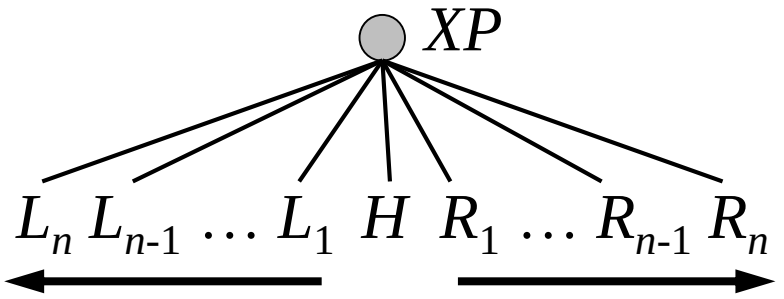
- 概率计算
  - 规则： $XP \rightarrow L_n L_{n-1} \ldots L_1 H R_1 \ldots R_{n-1} R_n$

    其中，XP 为规则左端的非终结符，H 为中心子节点，L_i 和 R_i为非中心子节点。
  - 产生中心子节点 $H(hw, ht)$的概率为: $P_H(H(hw,ht) \mid XP, hw, ht)$, 其中，$hw$ 为中心词，ht 为中心词的标记。
  - 产生中心子节点左边的非终结符，概率为:

$$\prod_{i=1}^{n+1} P_L(L_i(lw_i, lt_i) \mid XP, H, hw, ht)$$

  - 产生右边的非终结符，概率为:

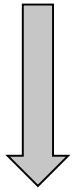
$$\prod_{i=1}^{n+1} P_R(R_i(rw_i, rt_i) \mid XP, H, hw, ht)$$



基于PCFG分析方法的改进

✧ 整个句法树的概率是所有规则概率的乘积：

VP(dumped, VBD) → VBD(dumped, VBD) NP(sacks, NNS) PP(into, P)



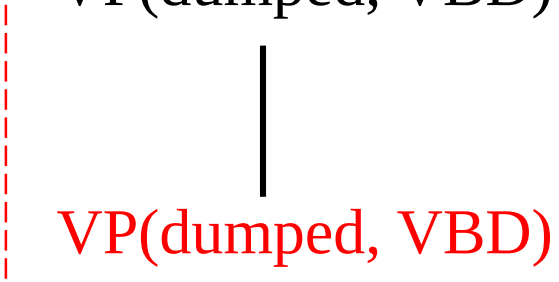
在规则右端的左右两边分别添加**STOP**非终结符，
用以标记非终结符生成过程的结束位置。

VP(dumped, VBD) → STOP VBD(dumped, VBD) NP(sacks, NNS) PP(into, P) STOP

➢ **步骤1：** 利用概率

P(VBD(dumped, VBD)|VP(dumped, VBD))

生成中心子节点 VBD(dumped, VBD)

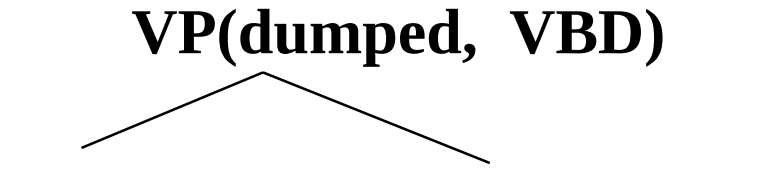


基于PCFG分析方法的改进

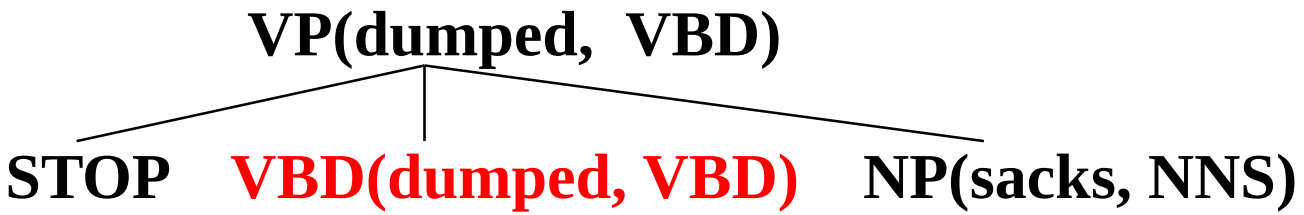
- **步骤2：**产生中心子节点左边的非终结符(因为该规则中不存在，所以只产生 STOP)，概率为：

  P_L(STOP|VP(dumped, VBD), VBD(dumped, VBD))

- **步骤3：**产生中心子节点右边的第一个非终结符 NP(sacks, NNS)，概率为：



P_R(NP(sacks, NNS)|VP(dumped, VBD), VBD(dumped, VBD))



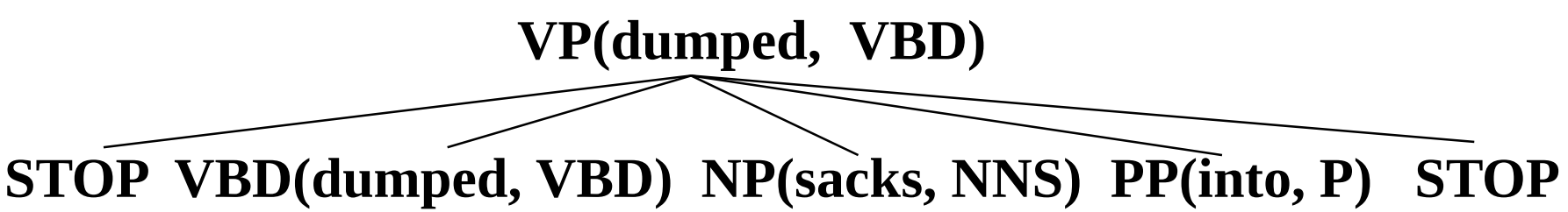
基于PCFG分析方法的改进

- **步骤4：**产生非终结符 PP(into, P)，概率为：

P_R(PP(into, P)|VP(dumped, VBD), VBD(dumped, VBD))

- **最后**，生成最右端的 STOP，概率为：

P_R(STOP|VP(dumped, VBD), VBD(dumped, VBD))



基于PCFG分析方法的改进

- 词汇化规则的概率可以表示为如下形式：

$$P(\text{VP(dumped, VBD)} \rightarrow \text{VBD(dumped, VBD)}\ \ \text{NP(sacks, NNS)}\ \ \text{PP(into, P)})$$

$$P(\text{VP(dumped, VBD)} \rightarrow \text{STOP}\ \ \text{VBD(dumped, VBD)}\ \ \text{NP(sacks, NNS)}\ \text{PP(into, P)}\ \text{STOP})$$

$$= P_H(\text{VBD}|\text{VP, dumped}) \times P_L(\text{STOP}|\text{VP, VBD, dumped}) \times P_R(\text{NP(sacks, NNS)}|\text{VP, VBD, dumped}) \times P_R(\text{PP(into, P)}|\text{VP, VBD, dumped}) \times P_R(\text{STOP}|\text{VP, VBD, dumped})$$

词汇化的句法分析器(lexicalized parser)

基于PCFG分析方法的改进

- 进一步的改进：
  - Collins(1999)模型1 将“距离(distance)”信息整合到模型中，使得模型能够更好地生成依恋(close attachment)结构和右分支(right-branching)结构；
  - Collins(1999)模型2 对模型1进行了扩展，使模型能够更好地区分修饰语(complement)和补足语(adjunct)，同时还直接对中心词子类别框架(sub-categorization frames)的概率分布进行了参数化；

基于PCFG分析方法的改进

- **Collins(1999)模型3** 根据生成短语结构语法(generalized phrase structure grammar, GPSC)，针对疑问词移位(wh-movement)现象进行了建模;
- **Charniak(2000)** 利用最大熵模型对规则中的每个概率进行计算;
- **Bikel(2004)** 是对Collins(1999)的重新实现，并将其扩展到多语言句法分析。

基于PCFG分析方法的改进

◆代表性论文

✧ Collins' Parser —

- Michael Collins. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics, 29*(4): *589-637*

✧ Charniark Parser —

- Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL, pp.132-139*
- Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of ACL, pp.173-180*

✧ Bikel Parser —

- Daniel M. Bikel. 2004. Intricacies of Collins' parsing model. *Computational Linguistics,* 30(4): *479-511*

基于PCFG分析方法的改进

- 开源的短语句法分析器
  - Collins Parser
    http://people.csail.mit.edu/mcollins/code.html
  - Bikel Parser
    http://www.cis.upenn.edu/~dbikel/software.html#stat-parser
  - Charniak Parser
    http://www.cs.brown.edu/people/ec/#software
  - Oboe Parser (可执行程序)
    http://www.openpr.org.cn/index.php/NLP-Toolkit-for-Natural-Language-Processing/

基于PCFG分析方法的改进

- **性能表现**

在宾州树库上的实验结果：

	≤40 words			≤100 words		
	LP	LR	F-Score	LP	LR	F-score
Collins (1999) M1	88.2	87.9	88.0	87.7	87.5	87.6
Collins (1999) M2	88.7	88.5	88.6	88.3	88.1	88.2
Collins (1999) M3	88.7	88.6	88.6	88.3	88.0	88.1
Charniak (2000)	90.1	90.1	90.1	89.6	89.5	89.5

基于PCFG分析方法的改进

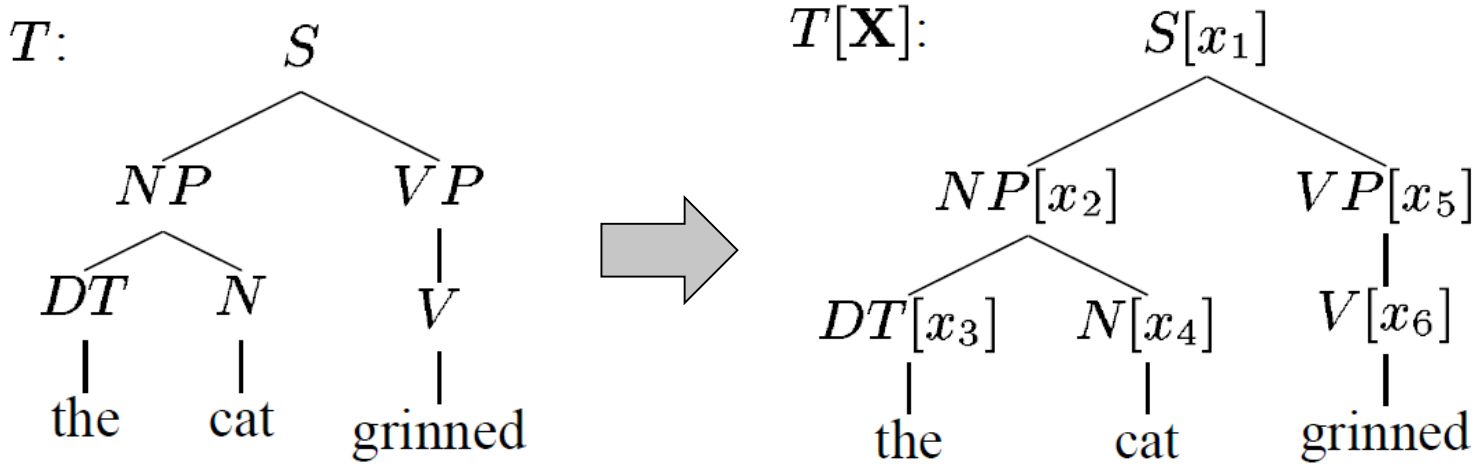
2. 带隐含变量的概率上下文无关文法 (PCFG with latent annotations, PCFG-LA)

- 目前最为成功的非词汇化模型
- 基本观点：观察到的句法树是不完全数据，完全的数据还应包含一些隐含变量。
- 解决方法：为每个非终结符标注一个隐含变量
  - 例如非终结符A，标注隐含变量后变为$A[x]$。其中，x的取值范围认为设定，一般取1到16之间的整数。

基于PCFG分析方法的改进

- **PCFG-LA举例**

下面图(1)为观察到的句法树，图(2)为标注隐含变量之后的句法树:



图(1)　　图(2)

基于PCFG分析方法的改进

- 带有隐含变量的句法树的概率为：

$$
\begin{aligned}
& P(T[\mathbf{X}]) \\
= & \pi(S[x_1]) \times \beta(S[x_1] \rightarrow NP[x_2]VP[x_5]) \\
& \times \beta(NP[x_2] \rightarrow DT[x_3]N[x_4]) \\
& \times \beta(DT[x_3] \rightarrow the) \times \beta(N[x_4] \rightarrow cat) \\
& \times \beta(VP[x_5] \rightarrow V[x_6]) \times \beta(V[x_6] \rightarrow grinned)
\end{aligned}
$$

- 相应的(原)句法树的概率为：

$$
P(T) = \sum_{x_1 \in H} \sum_{x_2 \in H} \cdots \sum_{x_6 \in H} P(T[\mathbf{X}])
$$

H=[1..16]

基于PCFG分析方法的改进

- **PCFG-LA的参数学习**
  - 该模型类似于HMM，原始非终结符对应 HMM中的观察状态，隐含标记对应 HMM中的隐含状态，因此，PCFG-LA可以借鉴 HMM 的训练和解码算法，通过EM算法进行参数的训练。
  - 考虑非终结符 *A* 跨度为(*r*, *t*)，其孩子结点为 *B* 和 *C*，相应的跨度分别为(*r*, *s*)和(*s*, *t*)
    - *A*的内向概率定义为：$\mathrm{P_{IN}}(r,t,A_x) \stackrel{\text{def}}{=} P(w_{r:t}|A_x)$
    - *A*的外向概率定义为：$\mathrm{P_{OUT}}(r,t,A_x) \stackrel{\text{def}}{=} P(w_{1:r}A_x w_{t:n})$

基于PCFG分析方法的改进

- 内向概率和外向概率可以通过如下的方程递归进行计算：

$$\mathrm{P_{IN}}(r,t,A_x) = \sum_{y,z} \begin{array}{l}\beta(A_x \rightarrow B_y C_z) \\ \times \mathrm{P_{IN}}(r,s,B_y)\mathrm{P_{IN}}(s,t,C_z)\end{array}$$

$$\mathrm{P_{OUT}}(r,s,B_y) = \sum_{x,z} \begin{array}{l}\beta(A_x \rightarrow B_y C_z) \\ \times \mathrm{P_{OUT}}(r,t,A_x)\mathrm{P_{IN}}(s,t,C_z)\end{array}$$

$$\mathrm{P_{OUT}}(s,t,C_z) = \sum_{x,y} \begin{array}{l}\beta(A_x \rightarrow B_y C_z) \\ \times \mathrm{P_{OUT}}(r,t,A_x)\mathrm{P_{IN}}(r,s,B_y)\end{array}$$

基于PCFG分析方法的改进

- 参数学习使用EM算法
  - 计算期望阶段(Expectation step)，规则在句法树中的后验概率为：

$$P((r,s,t,A_x \rightarrow B_y C_z)|w,T) \propto \mathrm{P}_{\mathrm{OUT}}(r,t,A_x) \times \beta(A_x \rightarrow B_y C_z)\mathrm{P}_{\mathrm{IN}}(r,s,B_y)\mathrm{P}_{\mathrm{IN}}(s,t,C_z)$$

  - 最大化阶段(Maximization step)，规则的概率为：

$$\beta(A_x \rightarrow B_y C_z) := \frac{\#\{A_x \rightarrow B_y C_z\}}{\sum_{y',z'} \#\{A_x \rightarrow B_{y'} C_{z'}\}}$$

注意：这里句法树的结构是已知的，未知的仅仅是每个非终结符上的隐含标记，因此，内向－外向算法与句子长度呈线性关系，而不是立方关系。

基于PCFG分析方法的改进

PCFG-LA的解码

- 解码的目标是获得概率最大的观察树(observable tree):

$$T_{best} = \underset{T \in G(w)}{\operatorname{argmax}} P(T|w) = \underset{T \in G(w)}{\operatorname{argmax}} P(T)$$

- 但每个观察树对应着指数多个完全(complete)树:

$$P(T \mid w) = \frac{1}{P(T)} \sum_{x_1 \in H} \sum_{x_2 \in H} \cdots \sum_{x_n \in H} P(T[x_1, x_2, ..., x_n])$$

→ $|H|^n$ complete trees

n 为树中节点数

所以，不能使用动态规划算法求解(NP-hard)，只能寻求近似解码算法。

基于PCFG分析方法的改进

- PCFG-LA的近似解码算法：
  - 方法1：重排序PCFG的N-best句法树
    - 利用原始的PCFG生成N-best列表
    - 使用PCFG-LA对该N-best列表进行重排序
    - 选取最好的结果作为最终句法树
  - 方法2：利用Viterbi完全树作为最终结果
    - 利用PCFG-LA分析出得分最高的句法树
    - 将隐含标记删除后得到的句法树作为最终结果
  - 方法3：利用近似分布进行Viterbi搜索
    - 比较复杂，详见原文。

非词汇化的句法分析器 (unlexicalized)

基于PCFG分析方法的改进

- 代表性论文
  - Stanford Parser —
    - Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of ACL,* pp. 423-430
  - Berkeley Parser —
    - Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. *Proc. NAACL-HLT*, pp. 404–411
    - Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. *Proc. the 21st COLING and 44th ACL*, pp.433 - 440

基于PCFG分析方法的改进

- 开源的短语句法分析器
  - Berkeley Parser

    http://nlp.cs.berkeley.edu/Main.html#Parsing
  - Stanford Parser

    http://nlp.stanford.edu/downloads/lex-parser.shtml

基于PCFG分析方法的改进

◆ 性能表现

宾州树库语料，参阅论文：
Slav Petrov and Dan Klein. 2007. Improved Inference for Unlexicalized Parsing. *Proc. of NAACL-HLT*, pp. 404–411

Parser	≤ 40 words LP	≤ 40 words LR	all LP	all LR
ENGLISH				
Charniak et al. (2005)	90.1	90.1	89.5	89.6
Petrov et al. (2006)	90.3	90.0	89.8	89.6
Unlexicalized	**90.7**	**90.5**	**90.2**	**89.9**
ENGLISH (reranked)				
Charniak et al. (2005)	**92.4**	**91.6**	**91.8**	**91.0**
GERMAN				
Dubey (2005)	F_1 76.3		-	
Unlexicalized	**80.8**	**80.7**	**80.1**	**80.1**
CHINESE				
Chiang et al. (2002)	81.1	78.8	78.0	75.2
Unlexicalized	**86.9**	**85.7**	**84.8**	**81.9**

基于PCFG分析方法的改进

- 我们利用宾州树库语料对 Berkeley Parser 的测试情况：

语种	训练集	开发集	测试集
汉语(句子数)	18089	350	348
英文(句子数)	39832	1700	2416

- 性能表现：

语种	句子类型	正确率(%)	召回率(%)	F1
英语	长度小于等于40个词	90.7	90.5	90.6
	所有的句子	90.2	8	90.5
汉语	长度小于等于40个字	86.9	85.7	86.3
	所有的句子	84.8	81.9	83.32