第四章 句法结构分析

王峰
华东师大计算机系

主要内容

- 自然语言的形式化表示
  - 自动机
  - 形式文法
- 句法结构分析
  - 自顶向下
  - 自底向上
  - Earley算法
  - 概率上下文无关文法
  - 增加特征结构与合一约束的句法结构分析
  - 依存句法分析
  - 汉英句法结构特点对比

1 自然语言的形式化表示

关于自然语言的知识

对于自然语言，人具有以下三个层面的能力：

1. **人们一般可以判断一个表达形式是否属于一种语言。(句法知识)**

例1

A. 张三爱好下围棋

A’. 下围棋是张三的爱好

B. 张三喜欢下围棋

B’. 下围棋是张三的喜欢

人们能够判断出句子B’不属于汉语（即不被说汉语者接受）。

关于自然语言的知识

2. 对于一种语言中的两个表达形式，人们一般可以判断二者之间是否具有某种关系，比如同义关系，两个表达式所对应的命题之间的逻辑蕴含关系，等等。(语义知识)

例2

C. 这件事容易办 → C’. 办这件事容易

D. 这件事好办 → D’. 办这件事好

人们能够判断句子C跟C’是同义关系，但D跟D’不是同义关系。

关于自然语言的知识

3. 对于一种语言中两个同义的表达式，人们一般可以判断在特定场合下使用哪一个表达式更好。**(篇章知识)**

例3

E. 马文才害死了梁山伯

F. 梁山伯被马文才害死了

G. E / F ，欺骗了祝英台。

例4

H. 桌子上有两本书

I. 有两本书在桌子上

J. H / I ，一盏台灯，还有一个笔筒。

例5

M.张三送了一套茶具给李四， N3/ N2/ N1

N1 李四很喜欢一套茶具。

N2 李四很喜欢茶具。

N3 李四很喜欢。

语言知识的表示

- 用自然语言来描述关于自然语言的知识
  - 不够精确
  - 不被计算机所接受
- 用形式语言来描述关于自然语言的知识
  - 无歧义，结构清晰简单
  - 可计算，结构化的数据
- 只有用形式化的方式描述语言知识，才能将知识用于计算，模拟人的语言能力。

形式语言(Formal Language)的一些例子

- 2＋5 ＝ 7
- $2H_2 + O_2 = 2H_2O$
- P & Q （P: 董永是放牛郎；Q:董永喜欢七仙女）
- IS_COWBOY (*x*) & IS_Vega(*y*) & LOVE (*x, y*)

```
# include "stdio.h"
int main ( )
{
        printf("Hello, world!\n");
        return 0;
}
```

语言的形式定义

- 语言：是字母表上的字符串的任意集合。
- *字母表*：有限个任意符号组成的非空集合Σ。
  - 例1：所有汉字组成的集合构成一个字母表。
  - 例2：汉语中所有的词也构成一个字母表。
  - 例3：字母a, b, c也组成一个字母表。
- *字符串*：由字母表Σ上的字符组成的有限长度的序列。

例1. 若字母表$\Sigma = \{a, b\}$，则定义在Σ上的语言可以是

L1={ab, ba}

L2={ab, abab, ababab, …}

例2. 字母表Σ = {0,1,2,3,4,5,6,7,8,9,+,-,.}

语言L3：十进制有限小数

如何描述（严格定义）一个语言

- 枚举
  - 给出语言中的所有句子
  - 对于含无限多个句子的语言不合适

列举性定义

- 自动机
  - 给出识别该语言中句子的机械方法

过程性定义

- 文法
  - 给出生成语言中所有句子的方法
  - 当且仅当能够用该方法产生的句子才属于该语言

描述性定义

有限状态自动机

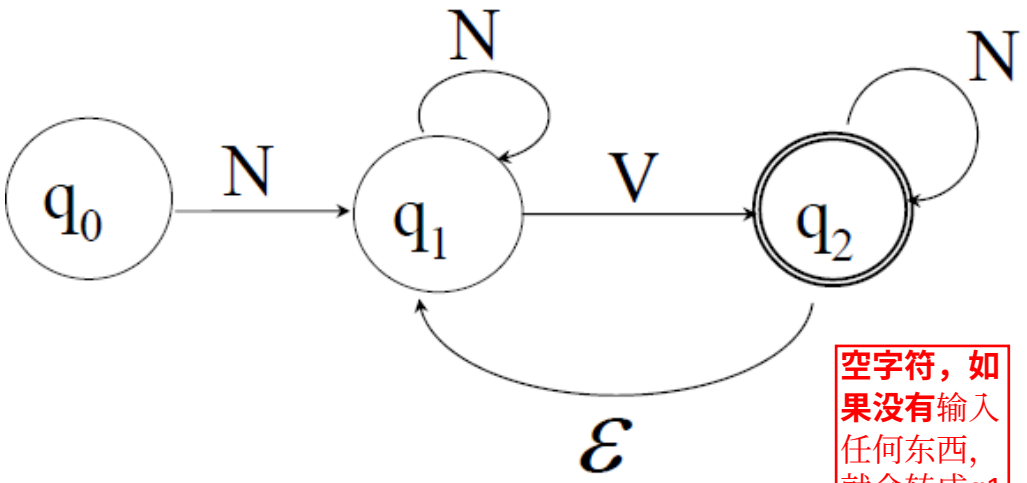
有限状态自动机 (Finite State Automata)

一个有限状态自动机M是一个五元组：
$(Q, \Sigma, q_0, F, \delta)$

- 有限个状态组成的状态集: Q
- 有限字母组成的字母表: Σ
- 开始状态 $q_0 \in Q$
- 终止状态的集合 $F \subseteq Q$
- 状态转移函数$\delta(q, i)$: $Q \times \Sigma \to Q$

有限状态自动机

开始状态**q0**，
唯一



终止状态**q2**，
可有多个

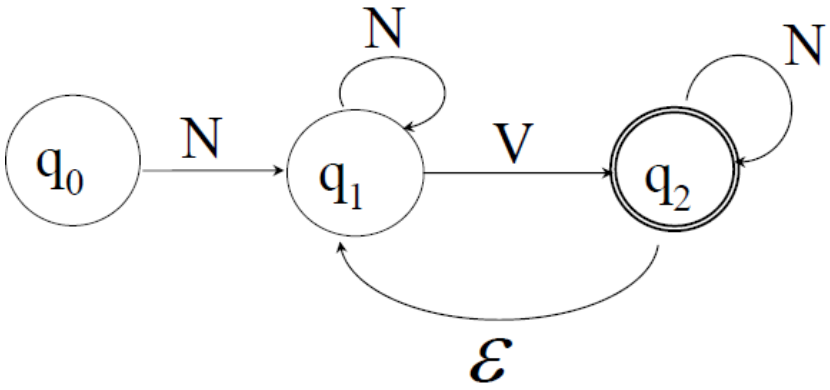
空字符，如果没有输入任何东西，就会转成q1

状态转移表
(state transition table)

弧(输入) / 状态 转移	N	V	ε
q_0	q_1		
q_1	q_1	q_2	
q_2	q_2		q_1

状态转移过程示例

给定字符串x，依次读入x中的字符。在读完该字符串后，如果该自动机停在一个属于F的**终止**状态，那么它就**接受**该字符串，反之则**拒绝**该字符串。

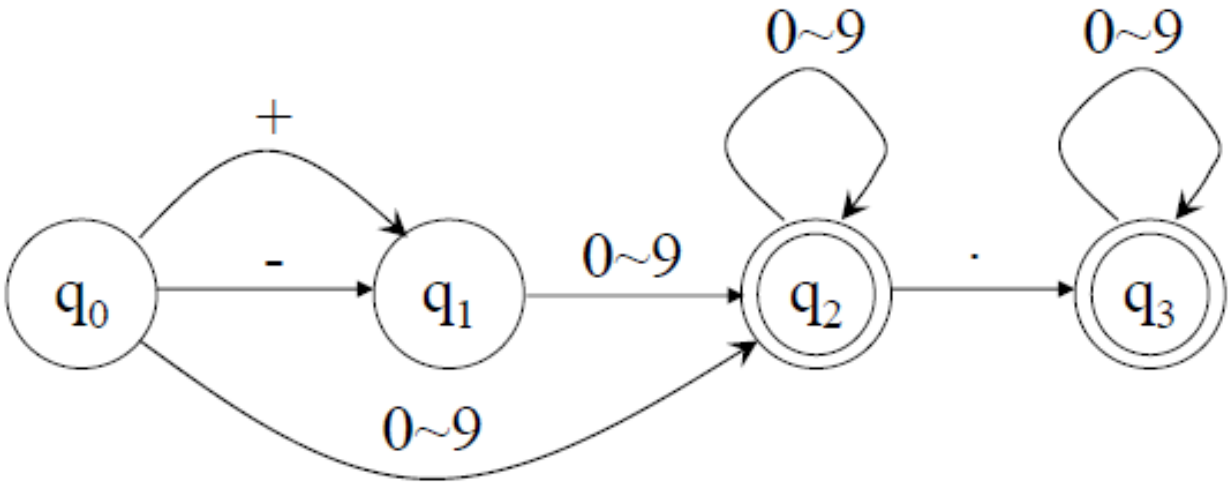


可以转化到上下文无关文法

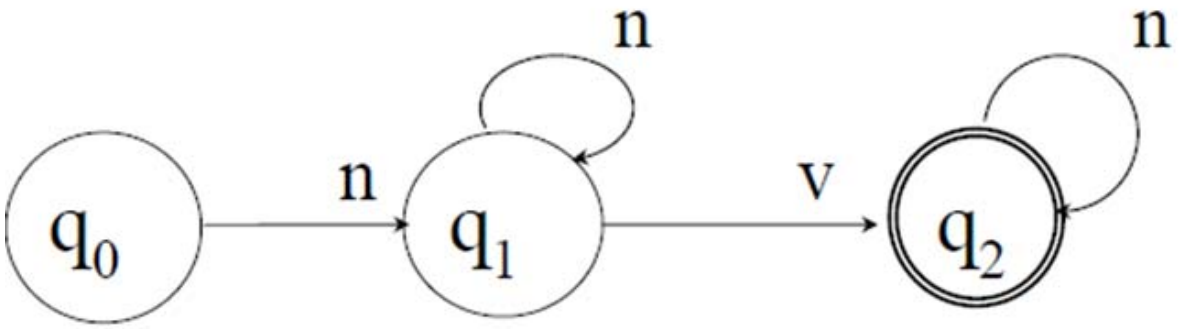
字符串		
N V	√	q0 -> q1 -> q2
N V N	√	q0 -> q1 -> q2 -> q2
N N V N N N	√	q0 -> q1 -> q1 -> q2 -> q2 -> q2 -> q2
N V N V N	√	q0 -> q1 -> q2 -> q2 -> q1 -> q2 -> q2
N N N	×	q0 -> q1 -> q1
V N	×	q0 -> ?

识别一个十进制实数的自动机

- 字母表：{0,1,2,3,4,5,6,7,8,9,+,-,.}
- 语言：十进制实数



一个识别简单汉语句子的自动机



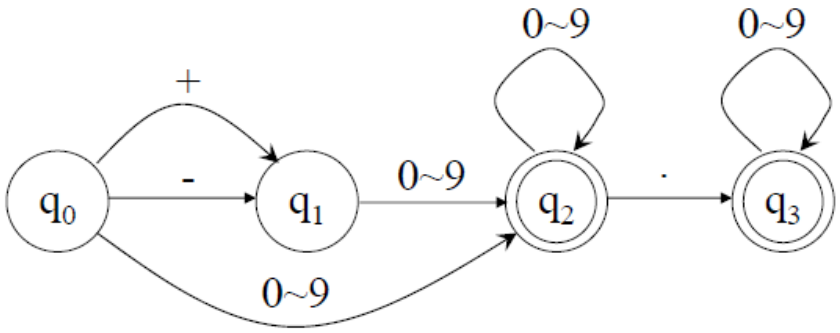
- 董永/n 喜欢/v 七仙女/n
- 牛郎/n 董永/n 喜欢/v 七仙女/n
- 董永/n 七仙女/n 喜欢/v

- 喜欢/v 董永/n 七仙女/n

接受的不一定合法，合法的也不一定被接受。

正则表达式（Regular Expression）

- 用来描述或者匹配一系列符合某个句法规则的字符串的单个字符串



(\+|-)?[0-9]+(\.[0-9]*)?

正则表达式与有限状态自动机可以互相对应！

元字符	描述
|	将两个匹配条件进行“或”运算
?	匹配0或1个它之前的那个字符。
[c1-c2]	匹配括号中的任何一个字符。
*	匹配0或多个在它之前的那个字符。
+	匹配1或多个在它之前的那个字符。
.	匹配任何单个字符。
\(\)	将 \(和 \) 之间的表达式定义为“组”，并且将匹配这个表达式的字符保存到一个临时区域，可以用 \1 到\9 的符号来引用。

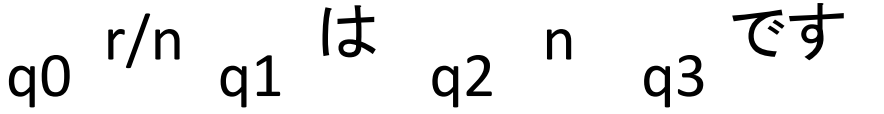
前面的可以没有，也可以只有一个

句法结构

例：これはいぬです

分词：　これ/r　は/u　いぬ/n　です/u

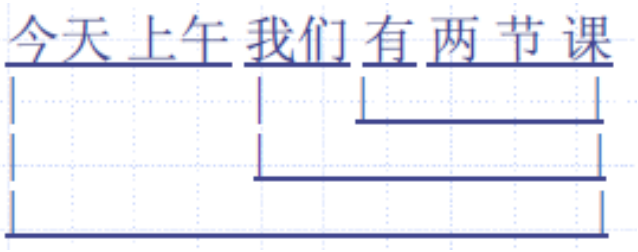
句型：[代词/名词] は [名词] です
[代词/名词] は [名词] でわ　あ り ま せ ん
……



句法结构

句法：词是如何组成句子的？

- 句子是词的线性序列，但词和词之间结合的松紧程度并不一样。
- 句子在构造上具有层次性，较小的成分还可以进一步组成较大的成分。

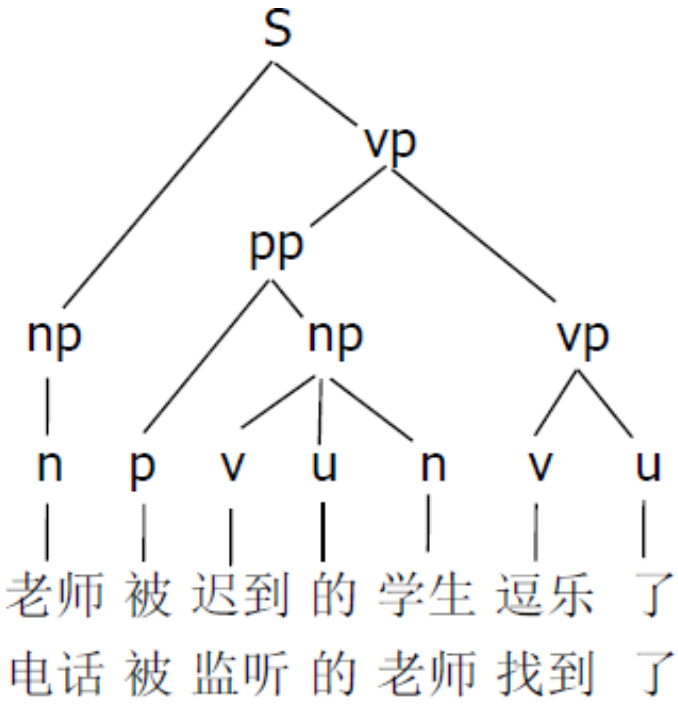


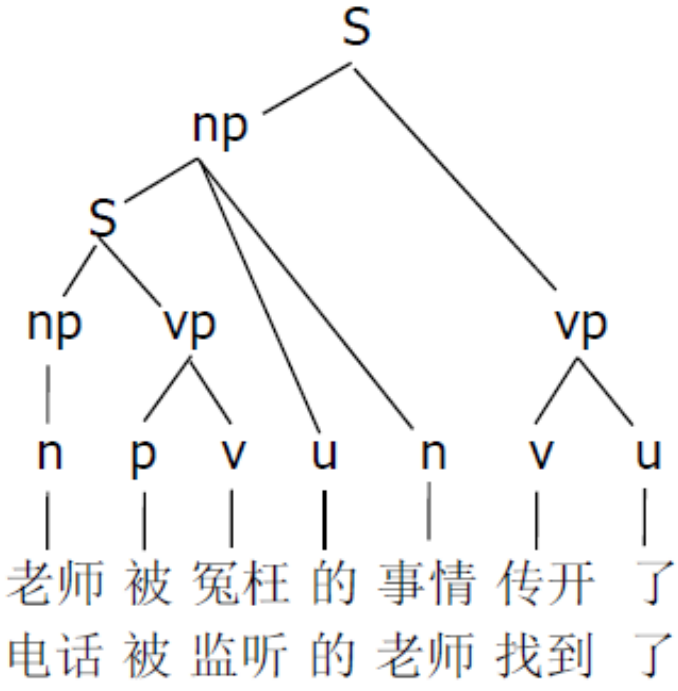
- 短语
  - 名词性短语(np, noun phrase)　两节课、机器零件
  - 动词性短语(vp)　吃包子、洗干净
  - 形容词性短语(ap)　很干燥、多么美妙
  - 处所词性短语(sp)　地板上、盒子里
  - 数量短语(mp)　一封(公开信)、多名(乘客)
  - 介词短语(pp)　向雷锋(学习)、被老师(警告)
  - ……

句子内部结构的树图表示

例：下面这些句子有何不同？

1. 老师 被 迟到 的 学生 逗乐了
2. 老师 被 冤枉 的 事情 传开了
3. 电话 被 监听 的 老师 找到了





从FSA到形式文法

同一个线性字符串，根据所处上下文环境的不同而解释为不同的树结构。

1. 听说服装设计很吃香
   听说那套服装设计得很有品位
2. 听说孩子丢了
   听说孩子丢了一只鞋
3. 听说北京队大败
   听说北京队大败上海队

- FSA：无法描述自然语言的层次结构特性

上下文无关文法

形式文法

生成式的表示方法

一个形式文法G由四个部分组成，可记作$G = \{V_N, V_T, S, P\}$，其中：

- V_N：称为文法G的非终端符号字母表，V_N不出现在G所表示的语言集合的句子中；
- V_T：称为文法G的终端符号字母表，G所表示的语言的句子由V_T中的元素组成，$V_N \cap V_T = \emptyset$；
- S：代表开始符号，$S \in V_N$；
- P：代表一组变换式组成的集合，P 中的式子具有如下形式：

$$\alpha \to \beta$$

主要区别

形式文法

$\alpha \to \beta$

- 称为产生式规则（production rule）或 重写规则（rewriting rule）

- 产生式规则需要满足下面的条件：

1. α可以是V_N和V_T上的任意字符串，不能是空字符；
2. β可以是V_N 和V_T上的任意字符串，可以是空字符；
3. P中至少有一个产生式中的α得由S来充当。

形式文法的Chomsky Hierarchy

分级	名称	产生式规则的限制
0	PSG	$A \to \beta$, with $A \in (V_N \cup V_T)^+$ and $\beta \in (V_N \cup V_T)^*$
1	CSG	$\alpha_1 A \alpha_2 \to \alpha_1 \beta \alpha_2$, with $A \in V_N$ and $\alpha_1, \alpha_2 \in (V_N \cup V_T)^*$ and $\beta \in (V_N \cup V_T)^+$
2	CFG	$\alpha \to \beta$, with $\alpha \in V_N$ and $\beta \in (V_N \cup V_T)^*$
3	RG	$A \to \beta B$ or $A \to \beta$, with $A, B \in V_N$ and $\beta \in {V_T}^*$

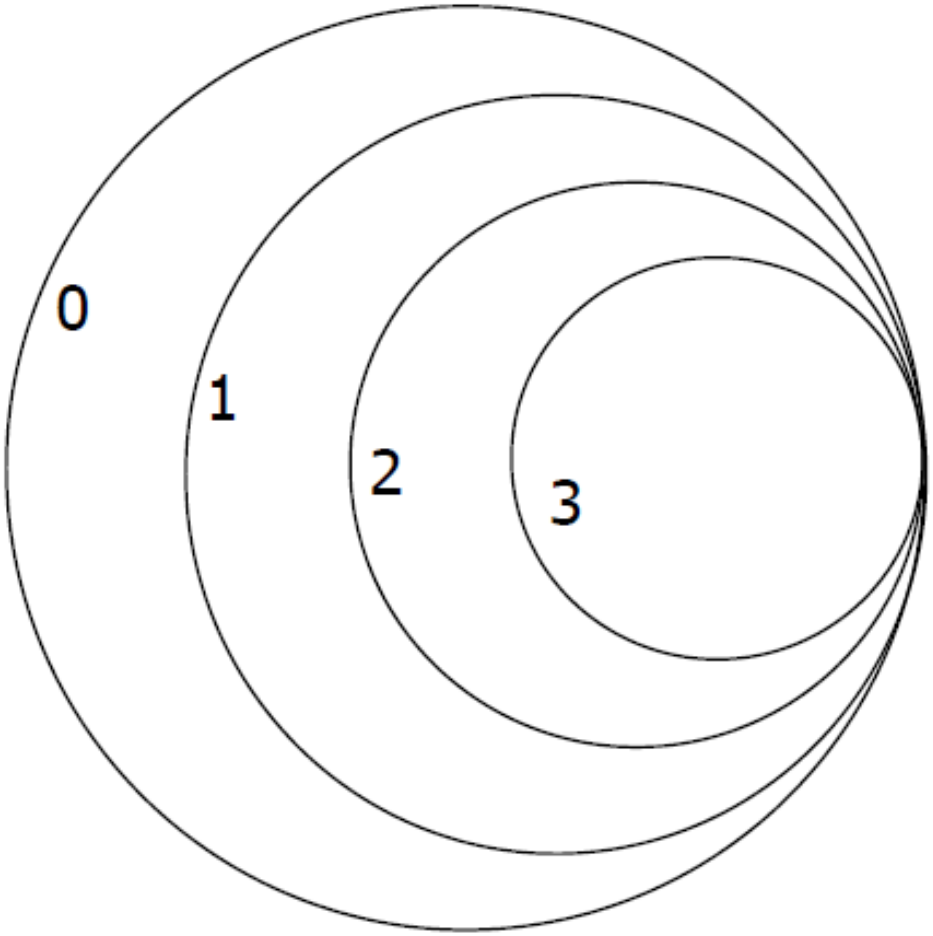
最松的

上下文无关文法
正则文法

Noam Chomsky, 1956, Three Models for the Description of Language, *IRE Trans. on Information Theory*, 2(1956), pp.113-124.

Noam Chomsky, 1959, On Certain Formal Properties of Grammars, *Information and Control*, 2(1959), pp. 137-167.

形式文法的Chomsky Hierarchy



G_0：无限制重写文法 PSG

G_1：上下文相关文法 CSG

G_2：上下文无关文法 CFG

G_3：正则文法 RG

L_0：递归可枚举语言

L_1：上下文相关语言

L_2：上下文无关语言

L_3：正则语言

自然语言处于1**与**2**之**间，分析时最常用的是上下文无关文法

上下文无关文法（Context-Free Grammar）

对产生式规则$\alpha \to \beta$ 做如下约定：

$$|\alpha| = 1, \qquad \alpha \in V_N, \qquad \beta \in (V_N \cup V_T)^*$$

这样的形式文法就是“上下文无关文法”。

- 上下文无关文法(CFG)是最常用的句法知识形式化工具
- 为了便于计算机处理自然语言，计算语言学研究人员提出了许多形式语法系统(grammar formalism)，例如：功能合一语法(FUG)、词汇功能语法(LFG)、中心词驱动的短语结构语法(HPSG)等。在这些语法形式化系统中，上下文无关文法是一个核心组成部分。
- 许多句法分析算法都建立在上下文无关文法的基础上。

用CFG描写汉语句法规则

- 名词性短语

np -> n|r	桌子、他们
np -> mp np	一本书、五斤牛肉
np -> ap np	英俊的小伙子
np -> np np	英语教师
np -> vp u	卖菜的

- 动词性短语

vp -> v	死
vp -> vp np	吃苹果
vp -> dp vp	认真准备
vp -> vp ap	洗干净

一个上下文无关文法的例子

设文法$G_0 = \{V_N, V_T, S, P\}$，其中

- $V_N = \{S, NP, VP, N, V\}$,
- $V_T = \{$喜欢，知道，董永，七仙女$\}$,
- P中产生式如下：

1. S -> NP VP
2. VP -> VP NP
3. VP -> VP S
4. VP -> V
5. NP -> N
6. N -> 董永
7. N -> 七仙女
8. V -> 喜欢
9. V -> 知道

直接推导、推导、句型、句子、语言

- 直接推导：S => NP VP
- 推导：S=> NP VP => NP V => N V
  - 上式可以简写为：S ⇒* N V
- 句型：由S推导出的$V_N \cup V_T$上的字符串
  - NP VP , NP V, N V, … 是G_0的句型
- 句子：仅含终结符号的句型
- 语言：给定一个文法G_0，该文法所产生的所有句子组成的集合，称为该文法所定义的语言。

G_0 所描述的语言L_0

S1: 董永喜欢七仙女

S2: 董永知道董永喜欢七仙女

S3: 七仙女知道董永

S4: 七仙女喜欢董永知道董永

S5: 七仙女喜欢董永董永董永七仙女

……

1. S -> NP VP
2. VP -> VP NP
3. VP -> VP S
4. VP -> V
5. NP -> N
6. N -> 董永
7. N -> 七仙女
8. V -> 喜欢
9. V -> 知道

不属于L_0 的字符串

S1′: 知道喜欢知道七仙女

S2′: 董永董永七仙女知道喜欢

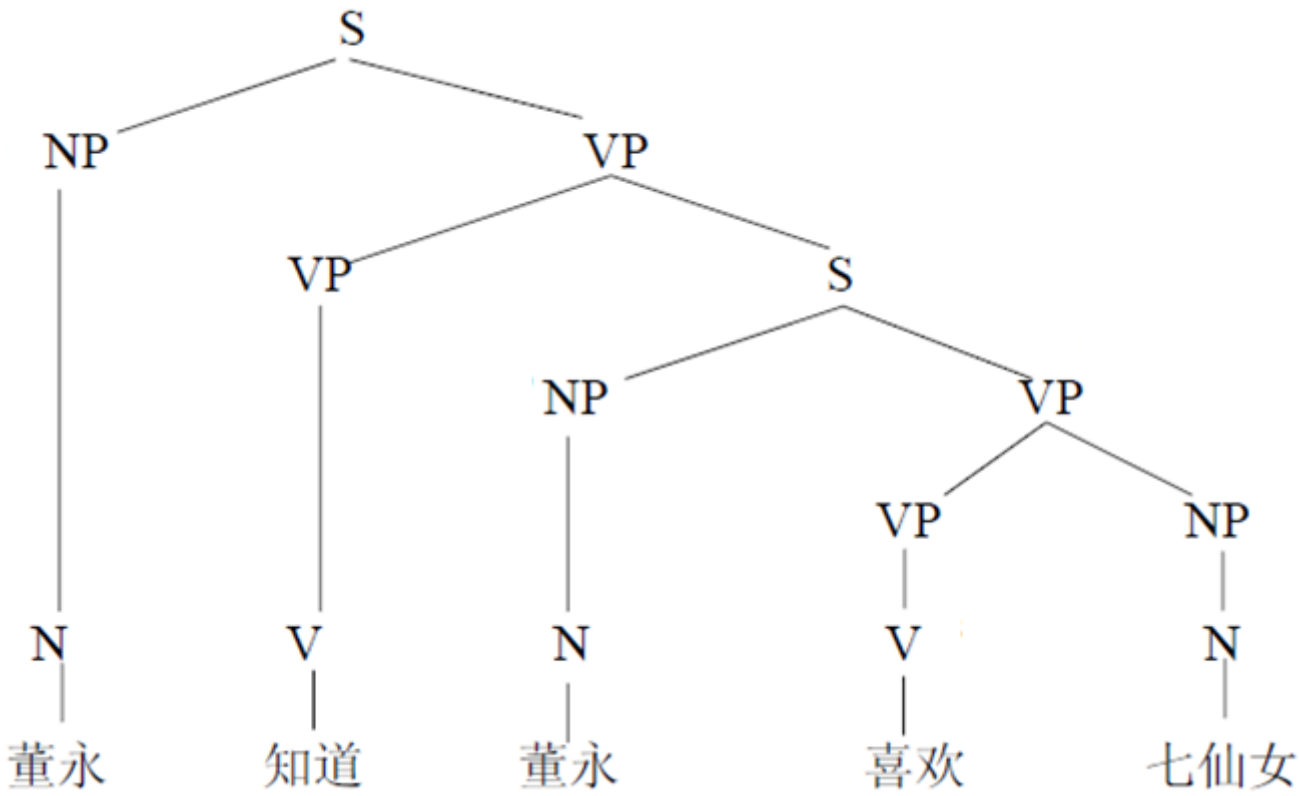
S3′: 七仙女董永喜欢

……

1. S -> NP VP
2. VP -> VP NP
3. VP -> VP S
4. VP -> V
5. NP -> N
6. N -> 董永
7. N -> 七仙女
8. V -> 喜欢
9. V -> 知道

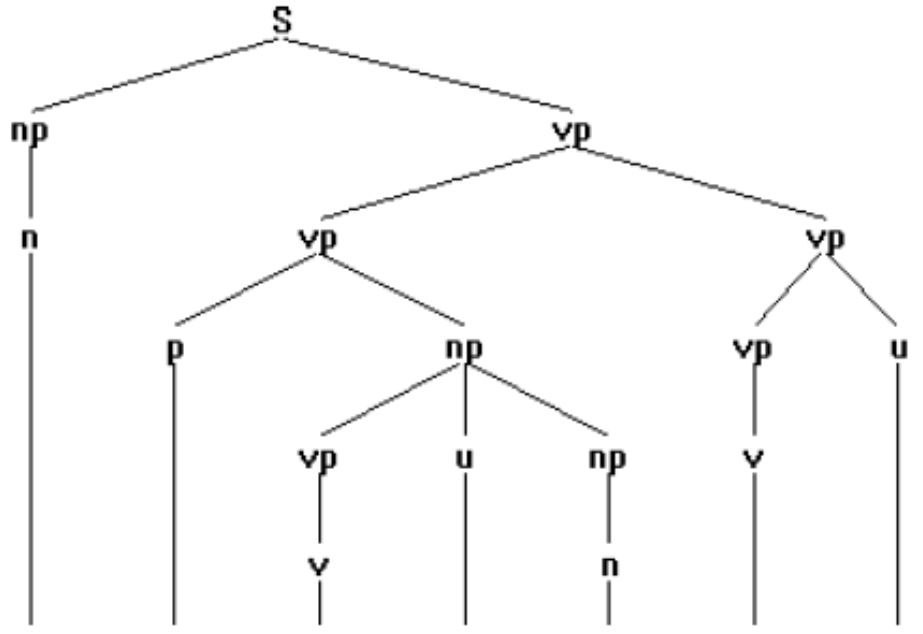
句子结构的树形描述

句法结构分析树



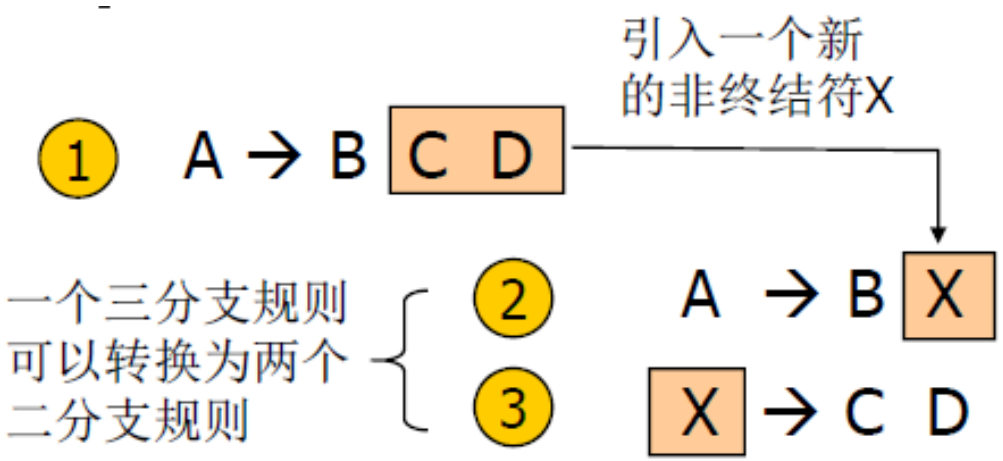
用CFG描述自然语言

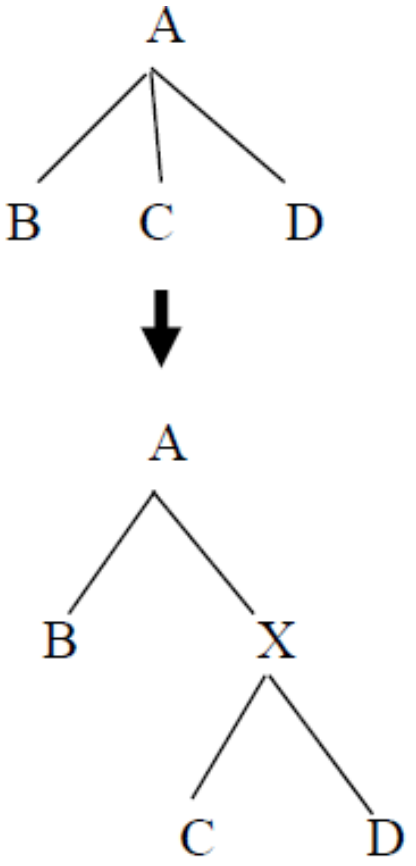
1. S → np vp
2. np → vp u np
3. vp → pp vp
4. vp → vp u
5. pp → p np
6. np → n
7. vp → v
8. n → 老师 | 学生 ...
9. v → 迟到 | 逗乐 ...
10. p → 被 ...
11. u → 的| 了 ...



老师 被 迟到 的 学生 逗乐 了
学生 被 迟到 的 老师 逗乐 了
迟到的学生 被 逗乐 的 老师 逗乐 了
....

乔姆斯基范式（ Chomsky Normal Form, CNF）





CNF 的 规则形式

- A → B C
- A → a

文法的三个作用

- **生成**：产生语言L中所有的句子；
- **判定**：一个字符串是否属于语言L；
- **分析**：得到L中句子的结构树。
  - 分析出句子的结构是进行自然语言信息处理的基础。

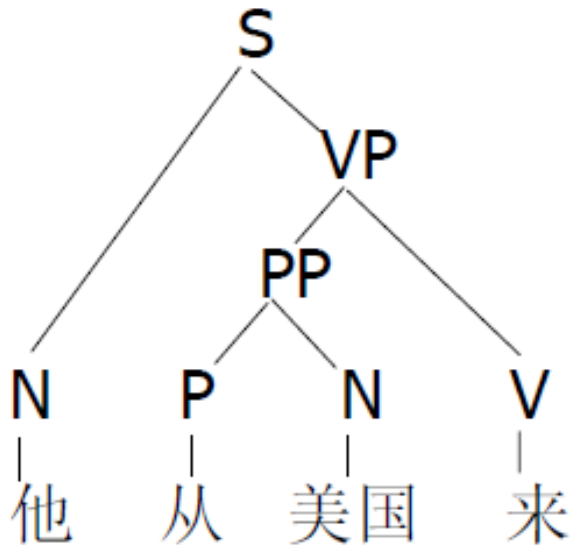
2 句法结构分析

句法分析

- 句法分析
  - 给定一个字符串S，判定S是否属于L；
  - 给定一个字符串S，如果S属于L，给出S对应的树结构。

他从美国来

*他从来美国



如何进行句法结构分析

- **句法结构分析：** 从“线性串”到“树结构”的映射。
- 需要做两件事：

1. 语言模型 - 形式语法理论的任务
   - ◆对自然语言定义一个文法
   - – 语言成分有多少类（范畴）？
   - – 成分间组合模式有多少种？
   - – 成分组合的约束条件是什么？

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V

2. 搜索算法 – 计算技术的任务
   - – 如何快速找到正确的结构树？

张三 是 县长 派 来 的
N V N V V 的

句法结构分析算法

- 自顶向下 (Top-down)句法分析
- 自底向上 (Bottom-up) 句法分析
- Earley算法
- 概率方法PCFG
- ……

自顶向下分析法

自顶向下的方法又称为基于预测的方法：

- 先产生对后面将要出现的成分的预期，然后通过逐步吃进待分析的字符串来验证预期；
  - 如果预期得到了证明，就说明待分析的字符串可以被分析为所预期的句法结构；
  - 如果某一个环节上预期出了差错，那就要用另外的预期来替换 (即回溯) ；
  - 如果所有环节上所有可能的预期都被吃进的待分析字符串所“反驳”，那就说明待分析的字符串不可能是一个合法的句子，分析失败。

分析用例

张三是县长派来的

苍蝇是瞎子打死的

主意是董永想出来的

……

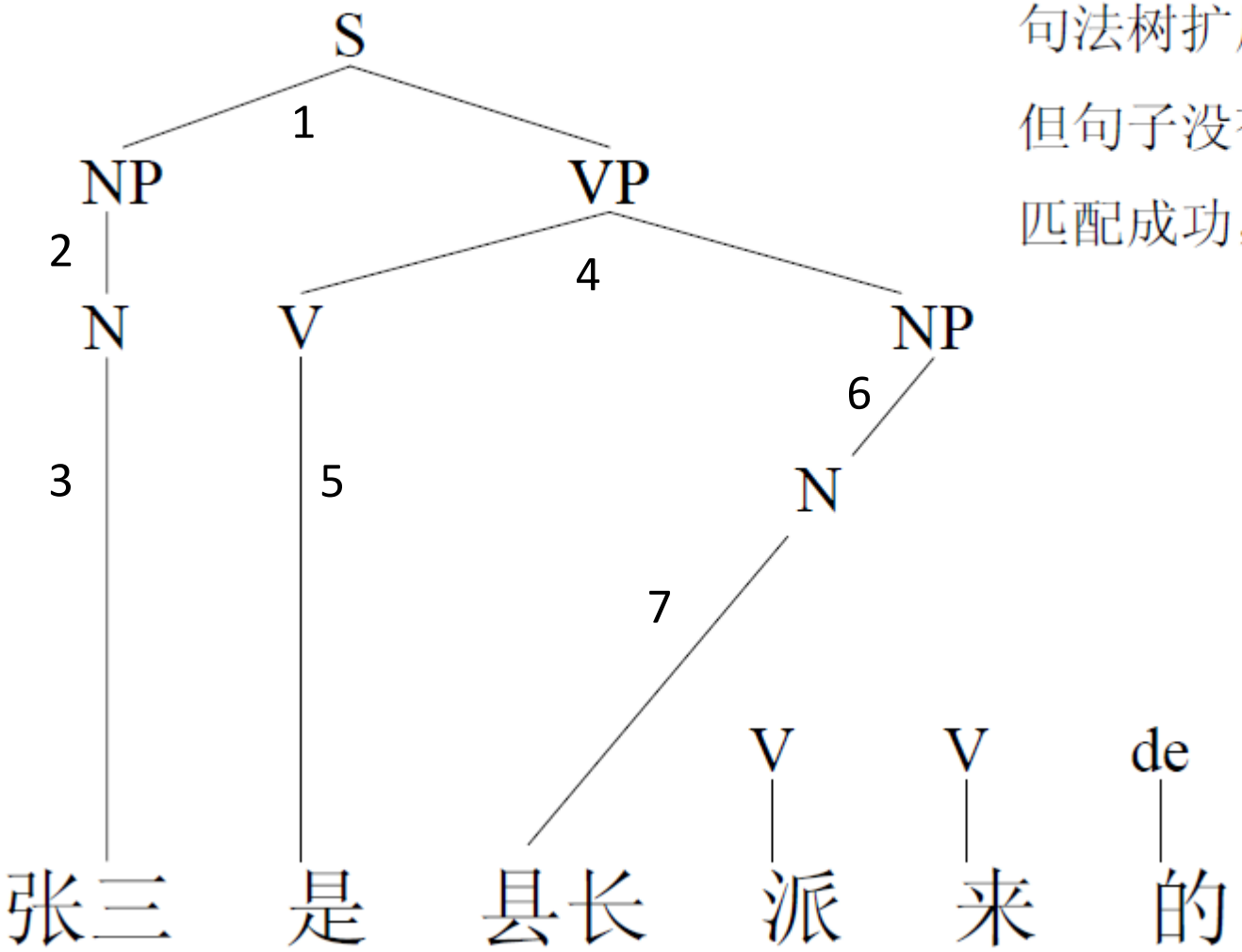
N V N V V 的

词典：
张三：N
县长：N
是：V
派：V
来：V
的：de

规则：
(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V

自顶向下分析示例

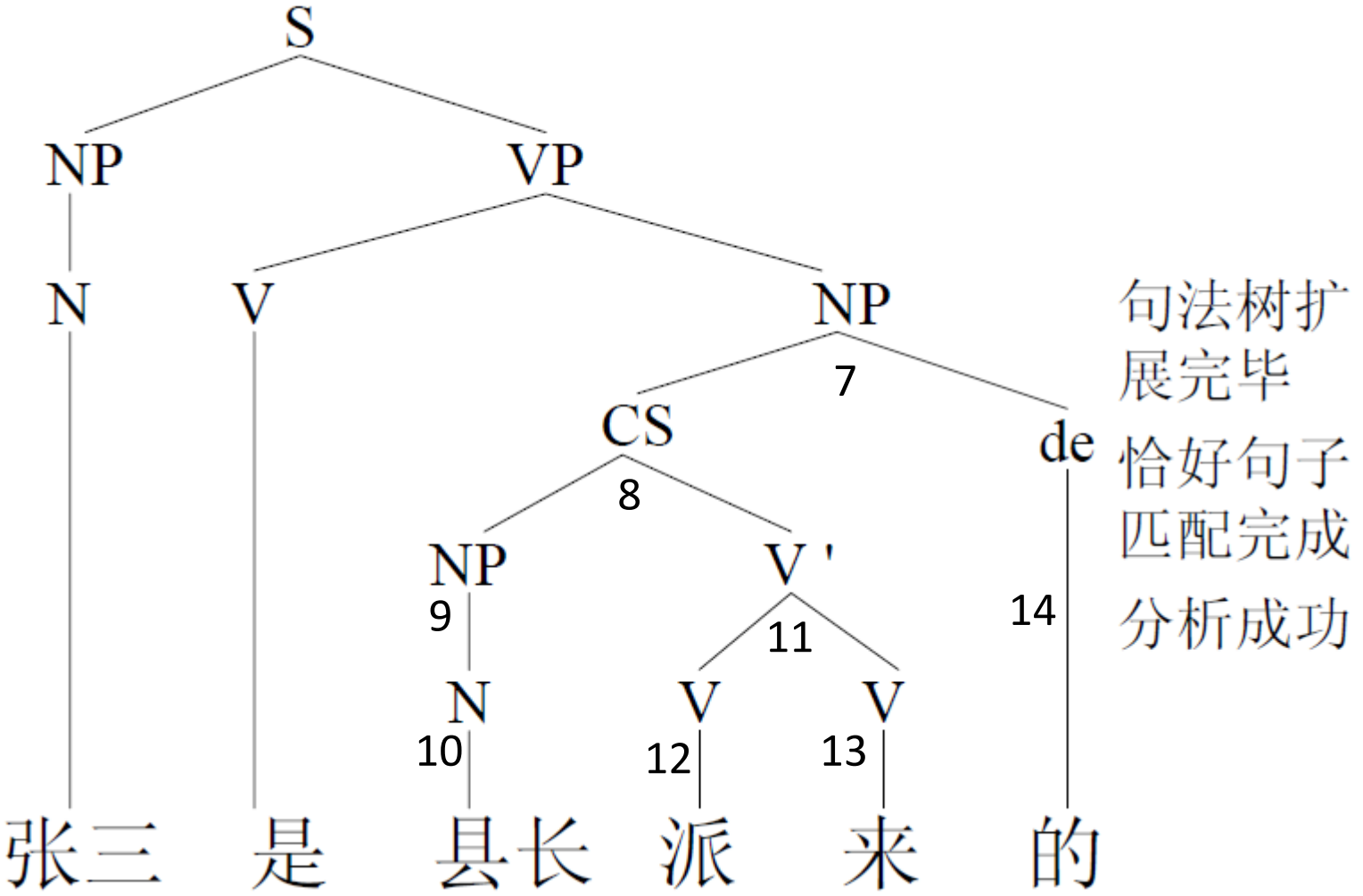
(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V



句法树扩展完毕，
但句子没有完全
匹配成功，回溯

自顶向下分析示例

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V



句法树扩展完毕
恰好句子匹配完成
分析成功

自底向上分析法

自底向上的方法也叫基于归约的方法：

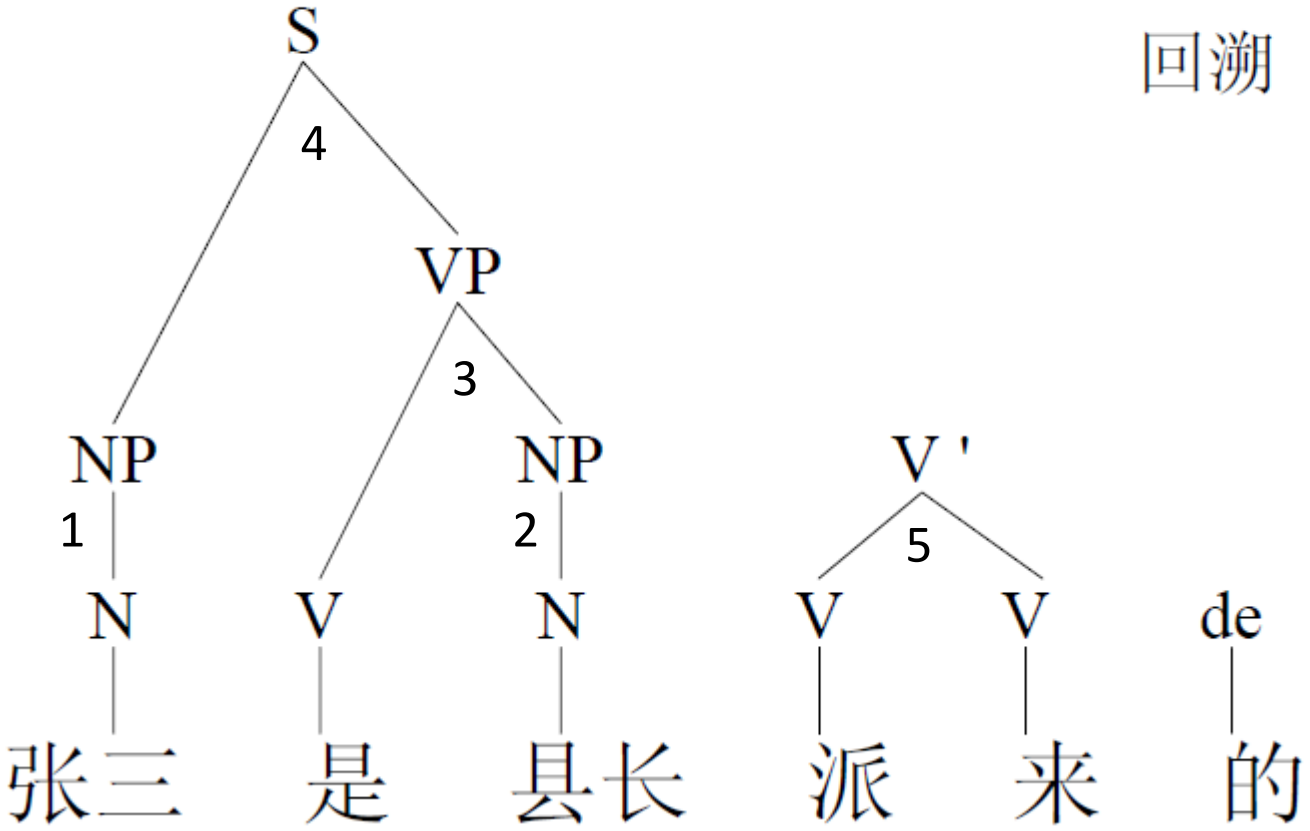
- 先逐步吃进待分析字符串，把它们从局部到整体层层归约为可能的成分；
  - 如果整个待分析字符串被归约为开始符号S，那么分析成功；
  - 如果在某个局部证明不可能有任何从这里把整个待分析字符串归约为句子的方案，那么就需要回溯；
  - 如果经过回溯始终无法将待分析字符串归约为S，那么分析失败。

自底向上分析示例

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V

无规则可用

回溯

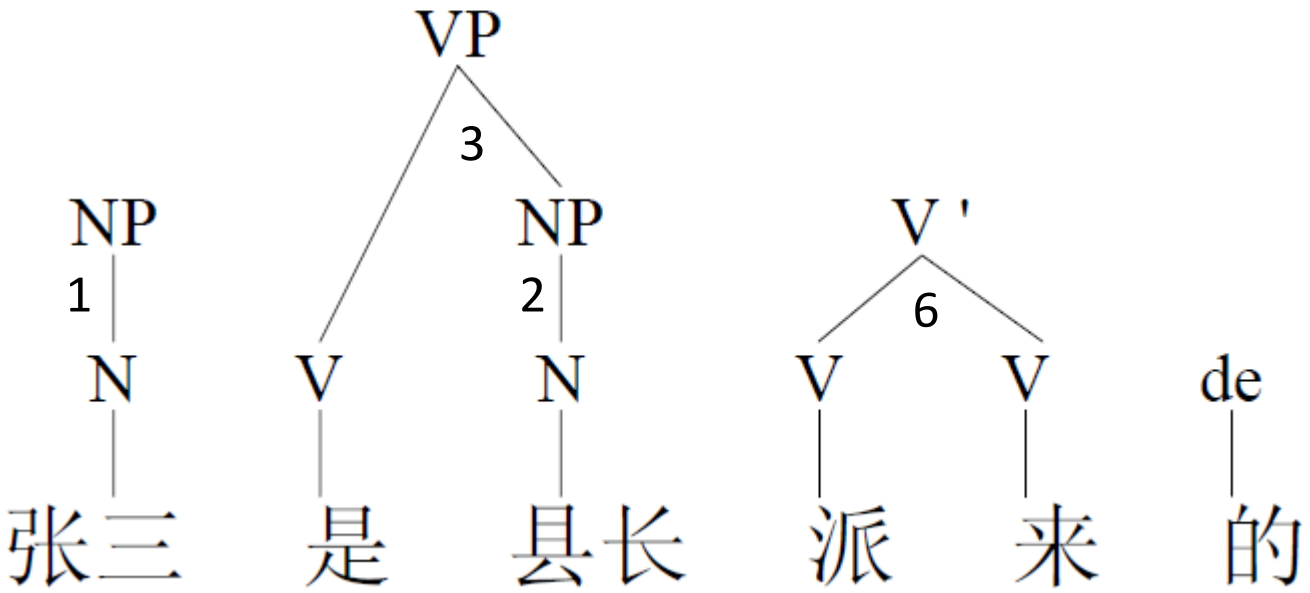


自底向上分析示例

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V

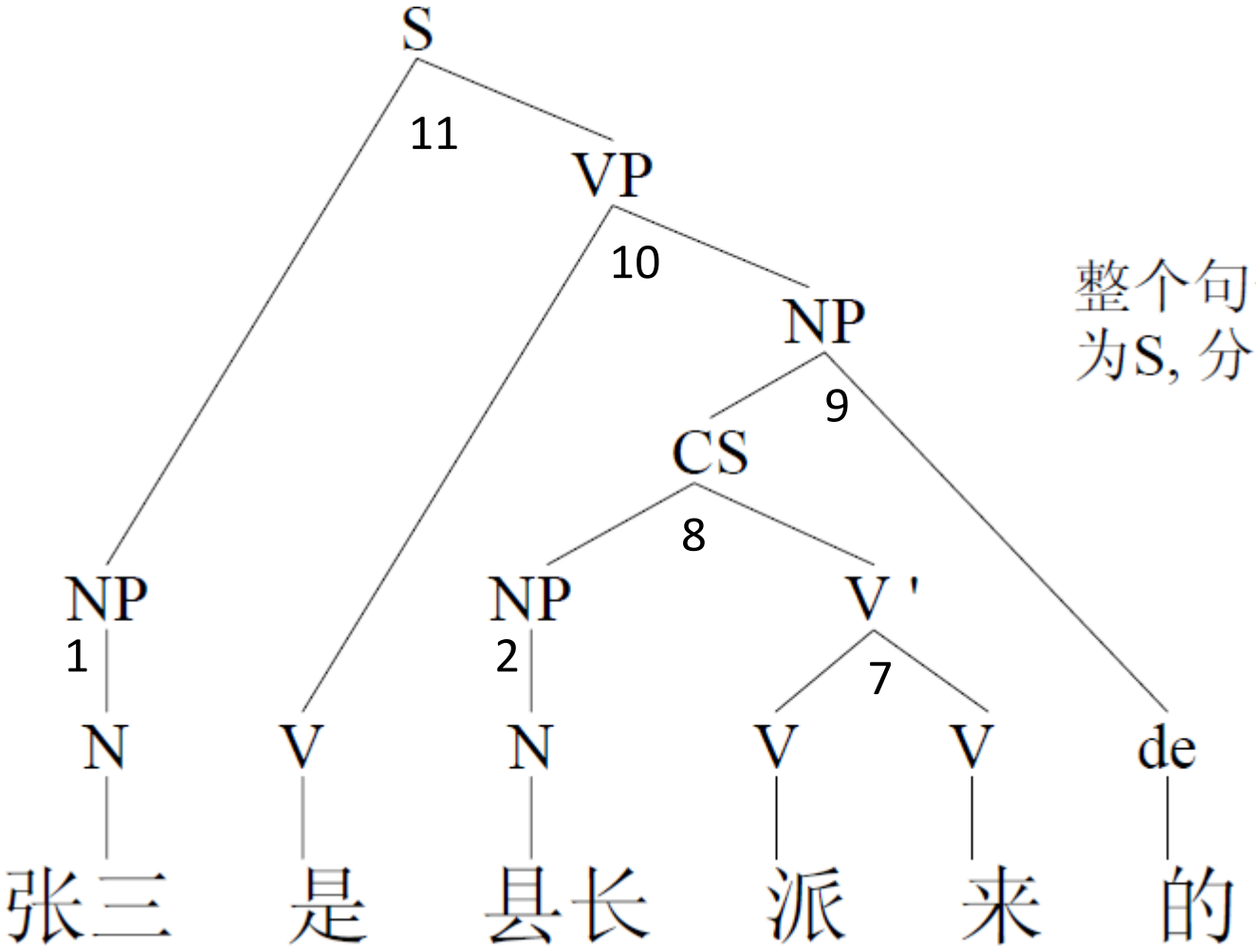
无规则可用

回溯



自底向上分析示例

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V



整个句子归结为S, 分析成功

句法分析方法的问题

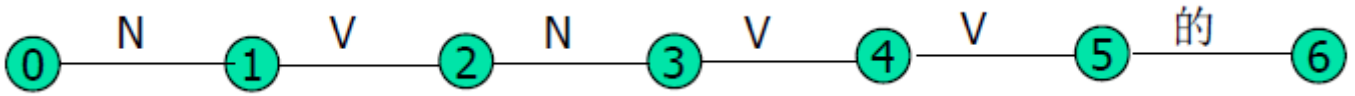
- 对于自顶向下的分析方法而言，当有多个重写规则可用时，如何避免选择导致分析失败的规则？
- 对于自底向上的分析方法而言，如果有多种可能的归约，如何避免导致分析失败的归约?
- 回溯导致效率低下
- 回溯也会导致重复计算

- 目前已经提出了多种句法分析算法，这些算法都可以相对有效地完成句法分析。
  - Earley分析算法
  - 广义LR分析算法
  - 基于chart的分析算法
  - CYK分析算法
  - ……

Earley算法

- 1970年，由Earley提出，所以称为Earley算法。
- 属于一种自顶向下的分析算法，时间复杂度是$O(N^3)$，其中N是待分析句子中的词数。
- 避免反复分析某些子树

张三是县长派来的　　N V N V V 的



基本概念：状态(state)

- 一个状态由3部分组成：
  - 上下文无关文法规则 --- *一个子树*
  - 圆点· --- --- --- --- --- --- *该子树的完成状态*
    - 圆点左边的部分是已分析的，右边是待分析的
  - 状态的起止位置: --- --- --- *子树与输入句子的对应关系*
    - 整数 i ：状态起点（已分析子串的起点）
    - 整数 j ：状态终点（已分析子串的终点）, $i<=j$

- 比如：$S \rightarrow NP \cdot VP[0,4]$

  点在最右端，为[完成状态]

  否则，为[未完成状态]

基本操作/算子（operator）

- **预测（Predicator）：**如果圆点右方是一个非终结符，那么以该非终结符为左部的规则都有匹配的希望，也就是说分析器可以预测这些规则都可以建立相应的项目。
- **扫描（Scanner）：**如果圆点右方是一个终结符，就将圆点向右方扫描一个字符间隔，把匹配完的字符“让”到左方。
- **归约（Completer）：**如果圆点右方没有符号（即圆点已经在状态的结束位置），那么表示当前状态所做的预测已经实现，因而可以将当前状态S_i与已有的包含当前状态的状态S_j进行归约（合并），从而扩大S_j覆盖的子串范围。

算子的形式定义

- **Predicator:** 对于状态 $Z \to \alpha \cdot X\beta[j,k]$其中$X$是非终结符对于语法中每条形如 $X \to \gamma$的规则，都可以形成一个新状态: $X \to \cdot\, \gamma[k,k]$。
- **Scanner:**对于状态$Z \to \alpha \cdot X\beta[j,k]$其中$X$是终结符，如果$X$与输入字符串中第k个字符匹配，就形成一个新状态：$Z \to \alpha X \cdot \beta[j,k+1]$。
- **Completer:**对于一个已经“完成”的状态 $Z \to \gamma \cdot [j,k]$如果已有状态集合中有形如 $X \to \alpha \cdot Z\beta[i,j]$这样的状态，就形成一个新状态：$X \to \alpha Z \cdot \beta[i,k]$。

说明：以上α,β,γ是终结符或非终结符串，
其中α,β均可为空字符， $i \le j \le k$。

Earley算法：算法描述

设输入字符串长度为n, 字符间隔可记做0,1,2,…,n。

1. 将文法规则中形如$S \to \alpha$的规则形成为状态：$< S \to \cdot \alpha[0,0] >$加入到状态集合中（种子状态/seed state）；
2. 对当前分析句子的每个词，依次进行循环：

   对状态集中的每个状态，依次进行循环：

   a) 如果当前状态是[未完成状态]，且点后不是终结符，则执行Predicator；

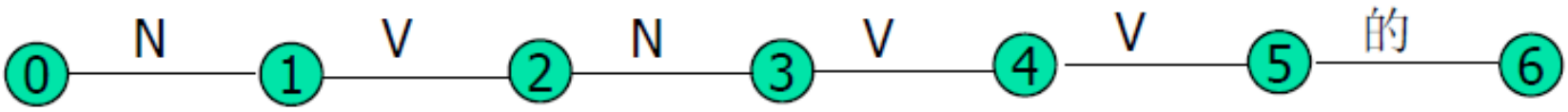
   b) 如果当前状态是[未完成状态]，且点后是终结符，则执行Scanner；

   c) 如果当前状态是[完成状态]，则执行Completer；
3. 如果最后得到形如$< S \to \alpha \cdot [0, n] >$这样的状态，那么输入字符串被接受为合法的句子，否则分析失败。

Earley算法过程示例

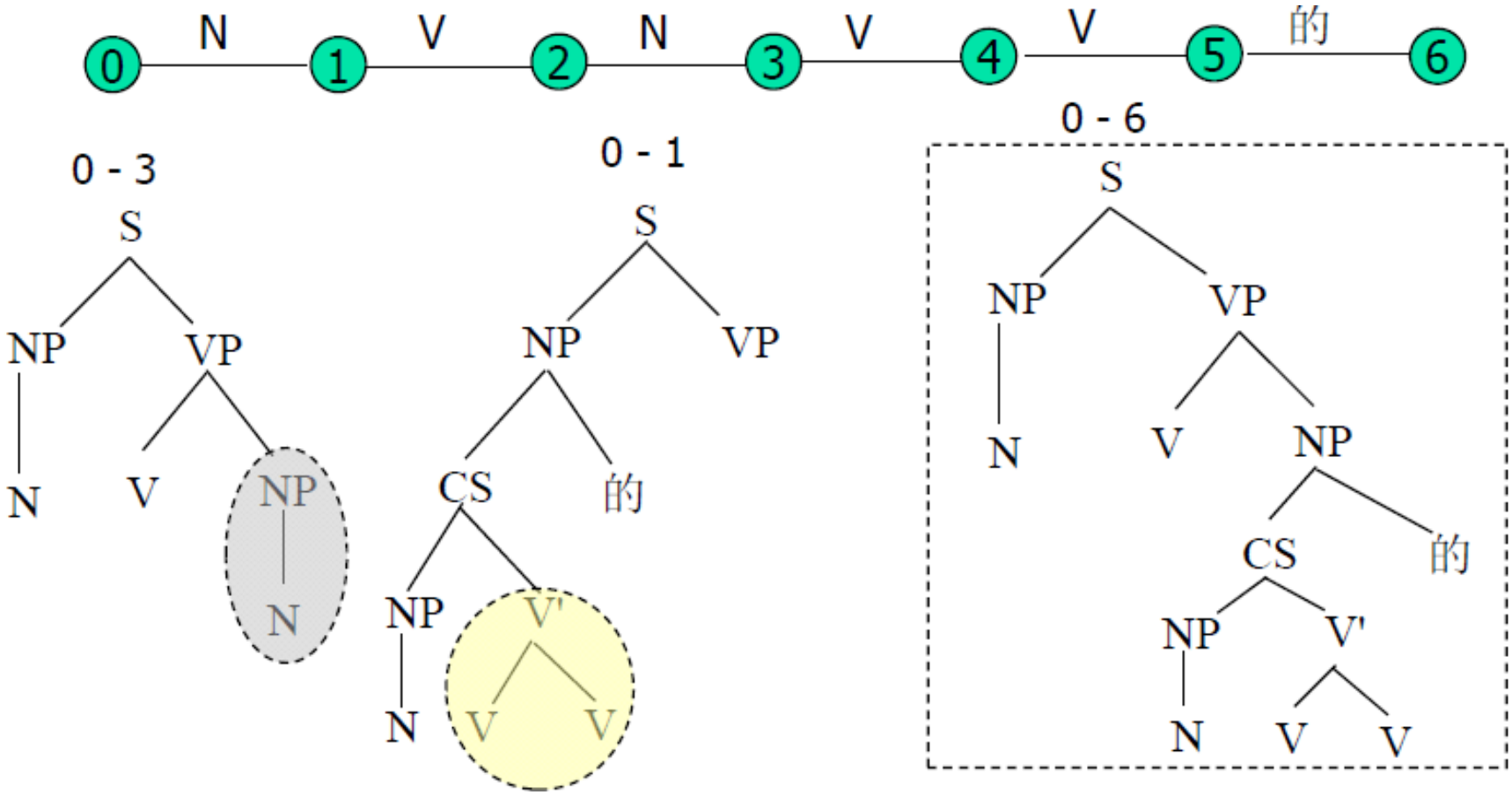
张三是县长派来的
老虎是瞎子打死的
主意是董永想出来的
……
N V N V V 的

(1) S → NP VP
(2) NP → N
(3) NP → CS 的
(4) CS → NP V'
(5) VP → V NP
(6) V' → V V



0	S →· NP VP [0, 0] NP →· N [0, 0] NP →· CS 的 [0, 0] CS →· NP V' [0, 0] ①						种子 预测 预测 预测
1	NP → N · [0, 1] CS → NP · V' [0, 1] ② S → NP · VP [0, 1]	VP →· V NP [1,1] V' →· V V [1,1] ③					扫描 归约 预测
2		VP → V · NP[1,2] V' → V · V [1,2] ④	NP →· N[2,2] NP →· CS 的[2,2] ⑤ CS → ·NP V'[2,2]				扫描 预测 预测
3	S → NP VP· [0,3] ⑧	VP→ V NP· [1,3] ⑦	NP → N · [2,3] CS → NP · V'[2,3] ⑥	V' →·V V[3,3] ⑨			扫描 归约 预测
4				V' → V · V[3,4]	⑩		扫描
5			CS → NP V' ·[2,5] NP → CS· 的[2,5] ⑫	V' → V V· [3,5] ⑪			扫描 归约
6	**S → NP VP· [0, 6]** ⑮	VP → V NP· [1,6] ⑭	NP → CS 的· [2,6] CS → NP· V'[2,6] V' →·V V[6,6] ⑬				扫描 归约 预测
	0	1	2	3	4	5	6
	张三 /N	是/V	县长/N	派/V	来/V	的	

Earley算法构造分析树示意图

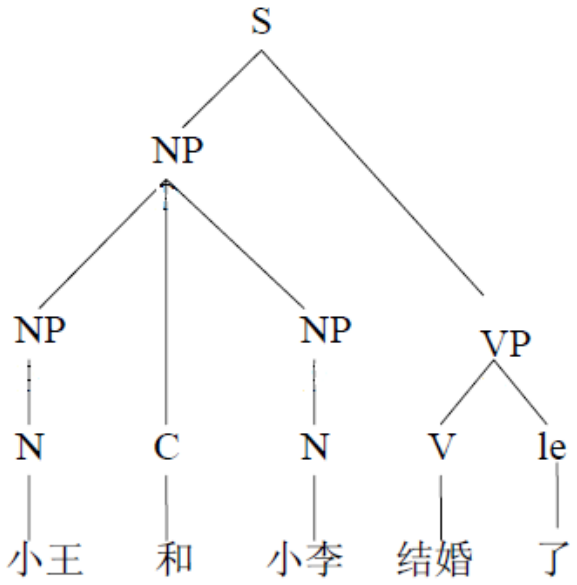


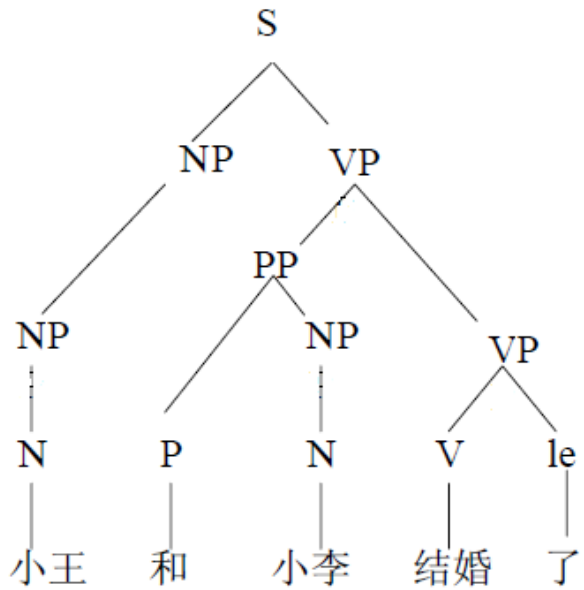
概率上下文无关文法

- 句法分析的歧义
  - 一个线性字符串可以分析为几种不同的树形结构

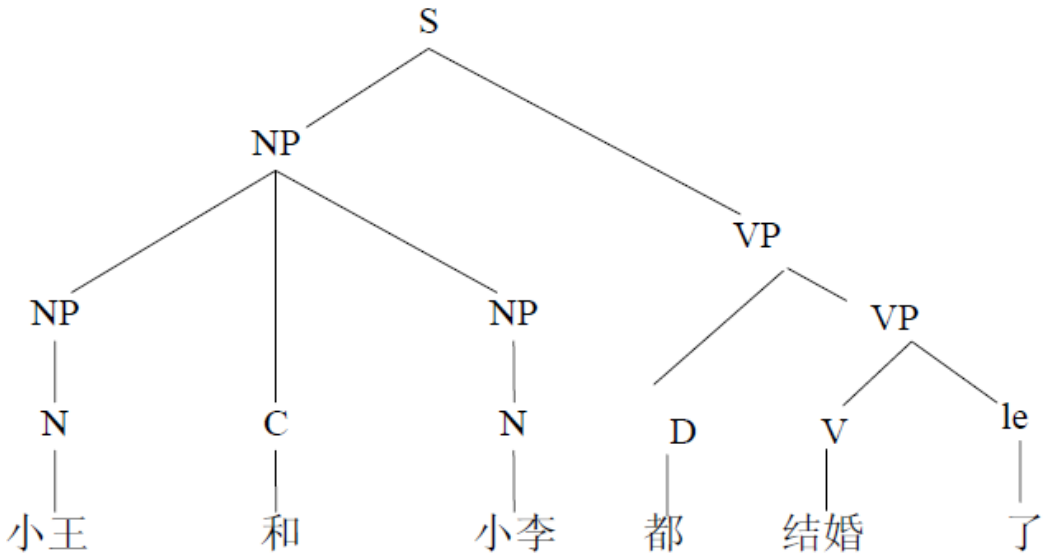
- Put the frog on the napkin in the box.
  - Put the frog on the napkin in the box.
  - Put the frog on the napkin in the box.

小王和小李结婚了





小王和小李都结婚了



概率上下文无关文法 Probablistic CFG

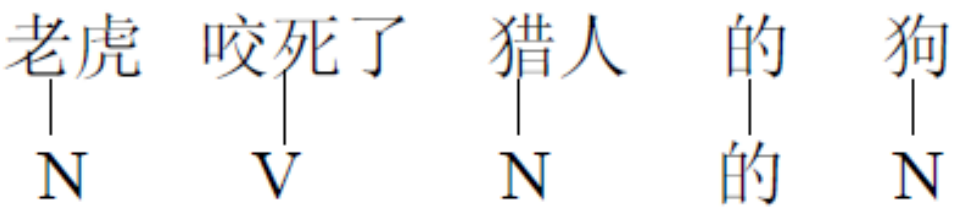
- PCFG: 为CFG中的每条规则增加一个概率值

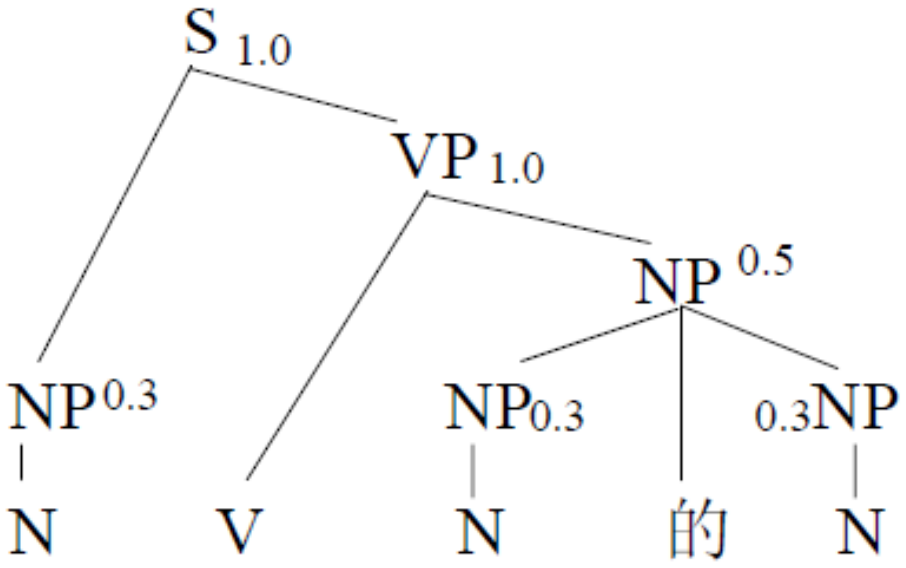
$$\sum_{\beta} P(\alpha \rightarrow \beta) = 1$$

CFG
S→NP VP
VP → V NP
NP → N
NP → NP 的 NP
NP → VP 的 NP

PCFG	
S→NP VP	1.0
VP → V NP	1.0
NP → N	0.3
NP → NP 的 NP	0.5
NP → VP 的 NP	0.2

分析树及其概率





$$P(S) = 1.0*0.3*1.0*0.5*0.3*0.3$$
$$= 0.0135$$

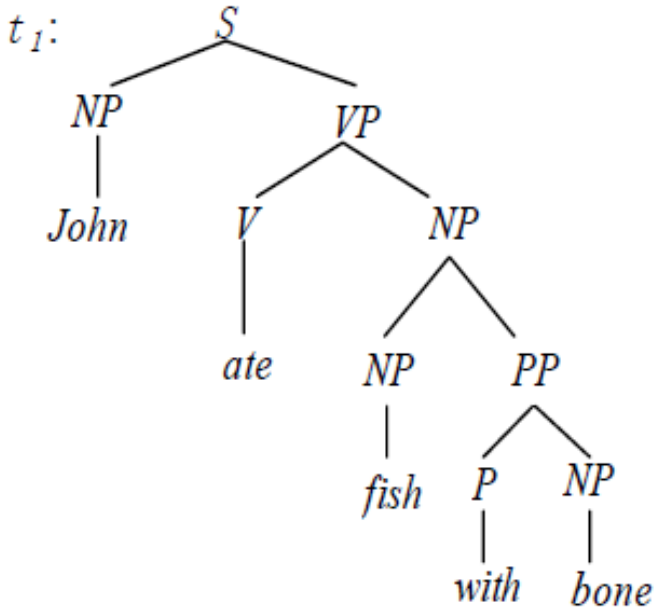
用概率来帮助判别歧义

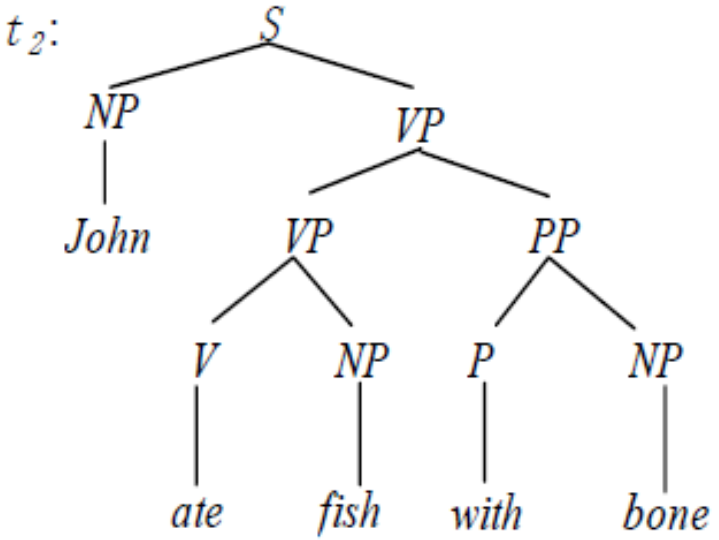
sentence = "John ate fish with bone"

$S \rightarrow NP\ VP$	1.0	$NP \rightarrow NP\ PP$	0.4
$PP \rightarrow P\ NP$	1.0	$NP \rightarrow John$	0.1
$VP \rightarrow V\ NP$	0.7	$NP \rightarrow bone$	0.18
$VP \rightarrow VP\ PP$	0.3	$NP \rightarrow star$	0.04
$P \rightarrow with$	1.0	$NP \rightarrow fish$	0.18
$V \rightarrow ate$	1.0	$NP \rightarrow telescope$	0.1

P(t1) = 0.0009072

P(t2) = 0.0006804





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
- 上下文无关性：子树的概率与子树管辖范围以外的词无关
- 祖先无关性：子树的概率与推导出该子树的祖先结点无关

PCFG的三个问题

给定一个符号串$W = w_1 w_2 \cdots w_n$和概率上下文无关文法 G,

1. 如何快速计算由G产生符号串W的概率 $P(W|G)$?
2. 如果 W 有多种可能的语法树，如何选择一棵最好的树? - Viterbi算法

$$\arg\max_{tree} P(tree|W, G)$$

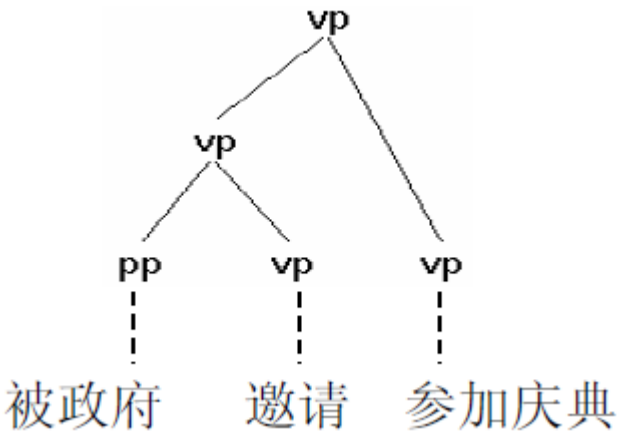
3. 如何调整 G 的规则概率参数，使得$P(W|G)$最大?

$$\arg\max_{G} P(W|G)$$

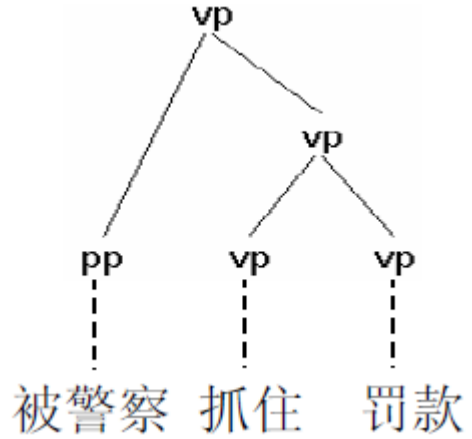
加入特征结构的CFG

CFG文法的不足

- CFG 文法要更精确地描述句法成分的组合，就要增加非终结符



vp → pp vp
vp → vp vp

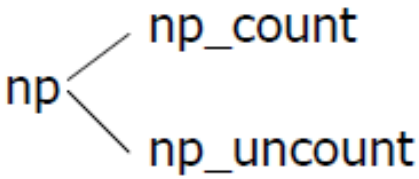


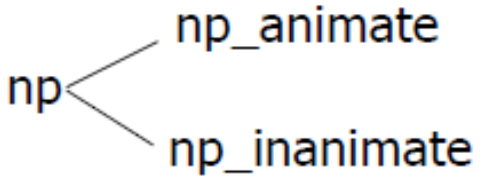
vp_1 → pp vp_2
Vp_3 → vp_1 vp_4

vp_5 → pp vp_6
Vp_6 → vp_7 vp_8

CFG文法的不足

- 范畴划分有不同的颗粒度（granularity）
  – np vp ap 等各类短语都可以区分更小的子类
- 范畴划分有不同的角度（perspective）





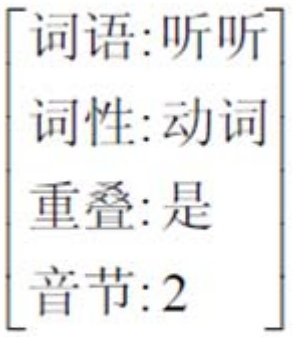
- 为了增强对句法组合的约束，CFG文法中的非终结符需要不断细化。
- 但增加非终结符的方式不可取，因此引入“特征结构”的表达形式。

特征结构（Feature Structure）

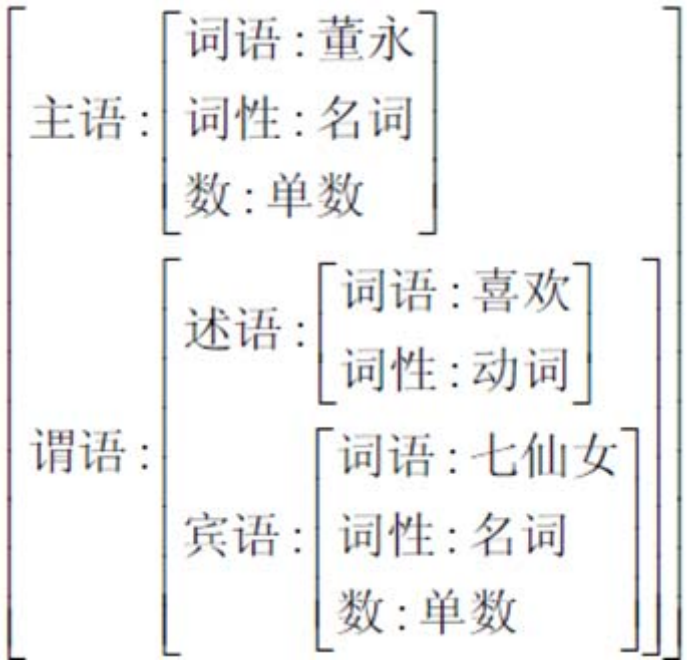
- 又名复杂特征集（Complex Feature Set）
- 特征结构定义为“特征”的集合
- 所谓“特征”，是一个由“属性”和“值”组成的二元组，“属性”也称为“特征名”，“值”也称为“特征值”
- 在特征结构中，要求所有的“特征”的“属性”互不相同
- 空特征结构：不含任何特征的特征结构，记作：[]

$$\begin{bmatrix} attribute_1 : value_1 \\ attribute_2 : value_2 \\ \cdots \\ \cdots \\ attribute_n : value_n \end{bmatrix}$$

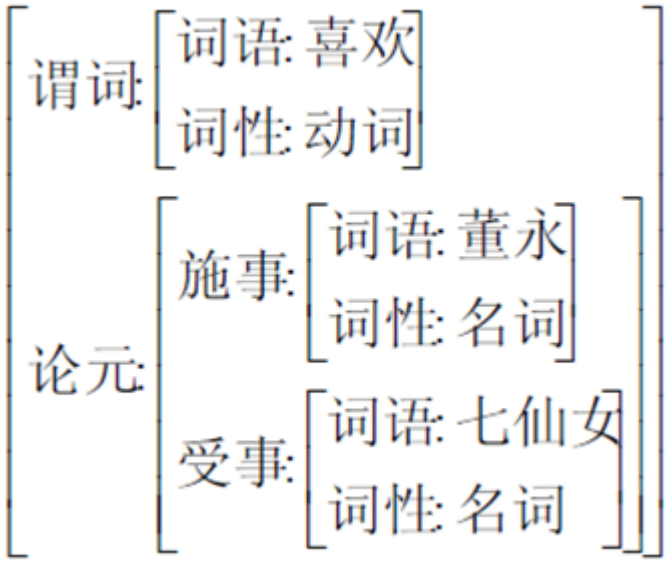
特征结构示例（框式表示法）



a. 简单特征结构



b. 复杂特征结构（嵌套）



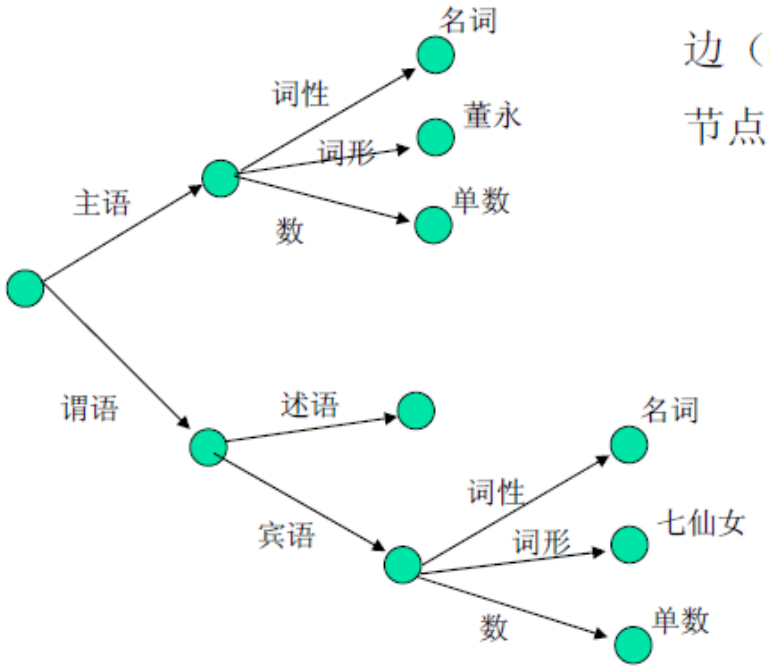
c. 复杂特征结构

特征结构的表示法

表(list) 表示法

```
((主语: (词语:董永)(词性:名词)(数:单数))
   (谓语: (述语:(词语:喜欢)(词性:动词))
      (宾语:(词语:七仙女)(词性:名词)(数:单数))))
```

有向无环图（**Directed Acyclic Graph**）



边（edge）表示特征

节点（node）表示特征值

特征结构间的包孕关系subsumption

- 特征结构$F1$包孕$F2$，记作$F1 \dot{\subseteq} F2$，当且仅当
  1. 若特征$f \in F1$，则$f \in F2$，并且
  2. 若f的值是特征结构，则$value_{F1}(f) \dot{\subseteq} value_{F2}(f)$；
  3. 若f的值是简单原子，则$value_{F1}(f) = value_{F2}(f)$。
- 空特征结构包孕任何特征结构

特征结构包孕关系举例

$$[Number{:}\ SG] \dot{\subseteq} \begin{bmatrix} Number{:}\ SG \\ PERSON{:}\ 3 \end{bmatrix}$$

$$[Agree{:}\,[Number{:}\ SG]] \dot{\subseteq} \begin{bmatrix} CAT{:} & NP \\ Agree{:} & \begin{bmatrix} Number{:} & SG \\ PERSON{:} & 3 \end{bmatrix} \end{bmatrix}$$

$$[\,] \dot{\subseteq} \begin{bmatrix} Number{:}\ SG \\ PERSON{:}\ 3 \end{bmatrix}$$

$$[Number{:}\ SG] \dot{\subseteq} [Number{:}\,PL]$$

特征结构的合一运算

- 合一运算（Unification）：将两个独立的特征结 构$F1$，$F2$组合为一个新的特征结构$F3$，满足条件： $F1 \dot{\subseteq} F3$ 并且 $F\ 2\ \dot{\subseteq} F3$。

- 合一的含义是：对两个特征结构进行类似于集合求并的一种运算，从而可以在“小”的特征结 构基础上形成“大”的特征结构
- 这种运算非常适于刻划“小”的语言单位发展为“大”的语言单 位的过程中的信息增加，即F3中包含了F1，F2所包含的信息。

合一运算实例（一）

$$A = \begin{bmatrix} 结构:述宾 \\ 功能:述语 \\ 词性:动词 \\ 及物:是 \end{bmatrix} \qquad B = \begin{bmatrix} 词语:咳嗽 \\ 词性:动词 \\ 及物:否 \end{bmatrix}$$

$$A \cup B = \phi$$ 合一失败

“合一”的作用：检查两个特征结构所包含的信息是否相容

合一运算实例（二）

令 $A=\begin{bmatrix}施事:C\\谓词:知道\end{bmatrix}$ $B=\begin{bmatrix}词语:董永\\语义类:人\end{bmatrix}$，其中 $C=$ [语义类:人]

则将 C 和 B 合一后，特征结构A变为：

$$\begin{bmatrix}施事:\begin{bmatrix}语义类:人\\词语:董永\end{bmatrix}\\谓词:知道\end{bmatrix}$$

合一成功

“合一”的作用：合一成功，特征结构的信息量增加

合一运算的性质

- 交换律：$A \,\bar{\cup}\, B = B \,\bar{\cup}\, A$
- 结合律： $A \,\bar{\cup}\, (B \,\bar{\cup}\, C) = (A \,\bar{\cup}\, B) \,\bar{\cup}\, C$

说明：

- 合一运算的结果与执行顺序无关；
- 合一运算使得特征结构真正成为一种“描述性”知识表示方法，而不是“过程性”的表示方法；
- 特征结构的“描述性”特点有利于在词典中给出词语的个性化描述。

为CFG文法增加特征结构合一描述

- 产生式规则： 描述一个语言的基本范畴及其组合模式。

- 基于特征结构的合一约束：

  ① 描述基本范畴之间发生组合关系的条件；
  ② 描述组合后整体的功能特征。

规则描述内容示例

规则描述内容 / 实例	内部构成	外部功能
一件衣服	qp + np， 定中结构， qp是定语， np是中心语；	√ 名词性短语（np）， √ 主谓结构的主语， √ 述宾结构的宾语， √ 定中结构的中心语； × 述补结构的补语， × 定中结构的定语， × 状中结构的状语和中心语；

“一件衣服”相关的规则示例

&& {R1} np -> qp !np

① $.内部结构=定中,$.定语=%qp,$.中心语=%np,$.zuodingyu=否,…,

② %np.数量名=是,…,

③ IF %qp.量词子类=个体 THEN %np.个体量词=%qp.原形 ENDIF,…

&& {R2} qp -> mp !q

$.内部结构=数量,$.定语=%mp,$.中心语=%q,$.zuodingyu=是,…,

&& {R3} np -> !n

$.内部结构=单词

① 说明np整体的特征，包括内部结构，定语，中心语，整体的功能分布特征等；
② 说明对中心语np的约束条件（独立条件）
③ 说明对定语mp与中心语np之间的相互约束条件（相关条件）

在词典中描述关于词的规则

词语	特征结构
……	……
一	[词性:m,数词子类:基数]
件	[词性:q,量词子类:个体,表数:数]
衣服	[词性:n, 数量名:是,个体量词:件|套,...]
心胸	[词性:n, 数量名:否,...]
……	……

规则的应用示例

1. 她买了一件衣服
2. 董永拿走了七仙女的一件衣服
3. * 一个心胸
4. * 一个衣服
5. * np[[一件衣服] [领子]]
6. * np[[一件] [衣服领子]]

规则的应用示例：状中式ap的内部差异

1. 很 好
2. 更 好
3. 不 好
4. 更 不 好

ap → dp ap :: $.内部结构=状中,...
IF %dp.yx=很|不 THEN $.comp=No ENDIF

ap → pp ap :: $.内部结构=状中, ...
%ap.comp=Yes

1. （张三）比李四 好
2. 很好 ×
3. 更好
4. 不好 ×
5. 更不好

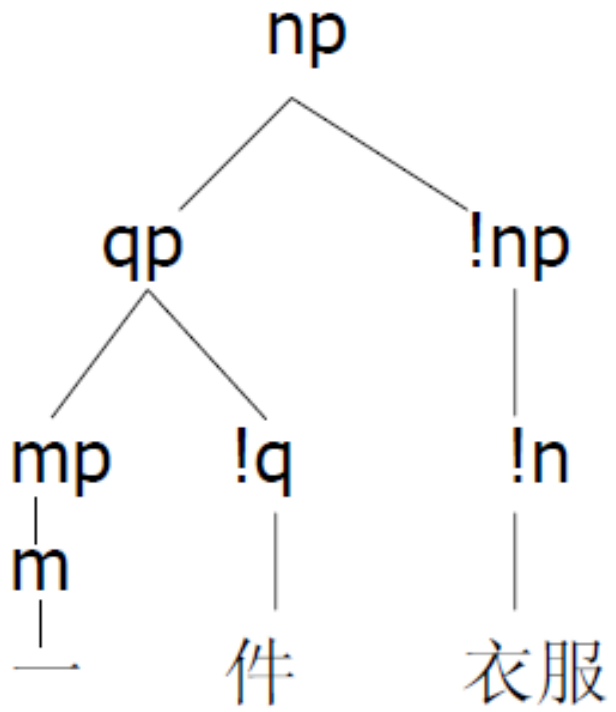
"很好、更好、不好、更不好"都是 dp + ap 形成的状中式ap组合模式，但是在"比 np __"环境中，"很好、不好"不能进入。这个限制可以通过特征结构的合一约束表达。

带合一约束的Earley算法

将合一运算融入到Earley算法中

1. 根据重写规后所附的合一约束为一个节点生成特征结构；
2. 对状态进行改进；
3. 对Predicator，Scanner，Completer等操作进行改进；
4. 生成新状态时对特征结构的包孕关系进行检查。

“一件衣服”的分析结果



$$\begin{bmatrix} \text{短语类型} = np \\ \text{内部结构} = \text{定中} \\ \text{定语} = \begin{bmatrix} \text{短语类型} = qp \\ \text{内部结构} = \text{定中} \\ \text{定语} = \begin{bmatrix} \text{短语类型} = m\text{p} \\ \text{内部结构} = \text{单词} \\ \text{词语} = \text{"一"} \end{bmatrix} \\ \text{中心语} = \begin{bmatrix} \text{词类} = q \\ \text{内部结构} = \text{单词} \\ \text{词语} = \text{"件"} \end{bmatrix} \end{bmatrix} \\ \text{中心语} = \begin{bmatrix} \text{短语类型} = np \\ \text{内部结构} = \text{单词} \\ \text{词语} = \text{"衣服"} \end{bmatrix} \end{bmatrix}$$

依存句法分析

依存句法分析

- 现代依存语法理论的创立者是法国语言学家Lucien Tesnière(1893-1954)。

- L. Tesnière的思想主要反映在他1959年出版的《结构句法基础》。

依存句法分析

- L. Tesnière 的理论认为：

一切结构句法现象可以概括为关联(connexion)、组合(jonction)和转位(tanslation)这三大核心。句法关联建立起词与词之间的从属关系，这种从属关系是由支配词和从属词联结而成；动词是句子的中心并支配别的成分，它本身不受其他任何成分支配。

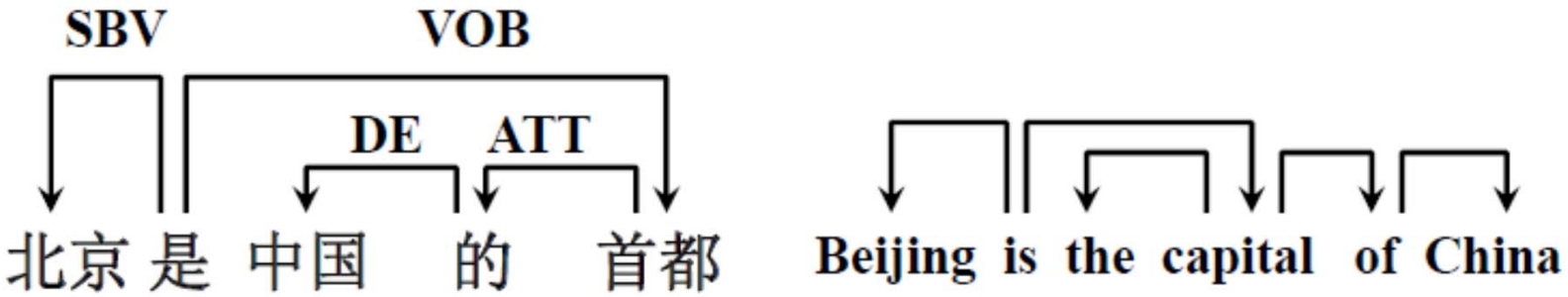
依存句法分析

- 欧洲传统的语言学突出一个句子中主语的地位，句中其它成分称为“谓语”。依存语法打破了这种主谓关系，认为“谓语”中的动词是一个句子的中心，其他成分与动词直接或间接地产生联系。

依存句法分析

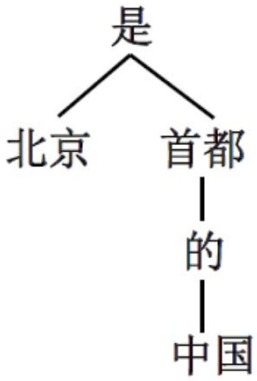
- 在依存语法理论中，“依存”就是指词与词之间支配与被支配的关系，这种关系不是对等的，而是有方向的。
- 处于支配地位的成分称为支配者**(governor, regent, head)**，而处于被支配地位的成分称为从属者**(modifier, subordinate, dependency)**。

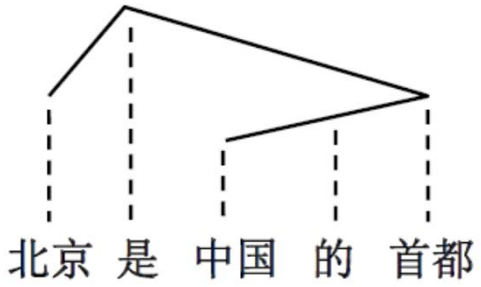
依存句法分析



两个有向图用带有方向的弧两个有向图用带有方向的弧(或称边，edge)来来表示两个成分之间的依存关系；支配者在有向弧的发出端，被支配者在箭头端，我们通常说被支配者依存于支配者配者依存于支配者。

依存句法分析





依存树：用树表示的依存结构，树中子节点依存于该节点的父节点。

依存投射树：带有投射线的树结构,实线表示依存联结关系位置低的成份依存于位置高的成份，虚线为投射线。

依存句法分析

1970年计算语言学家**J. Robinson**在论文《依存结构和转换规则》中提出了依存语法的四条公理：

(1) 一个句子只有一个独立的成分；

(2) 句子的其他成分都从属于某一成分；

(3) 任何一成分都不能依存于两个或多个成分；

(4) 如果成分**A**直接从属于成分**B**，而成分**C**在句子中位于**A**和**B**之间，那么，成分**C**或者从属于**A**，或者从属于**B**，或者从属于**A**和**B**之间的某一成分。

依存句法分析

这四条公理相当于对依存图和依存树的形式约束为：

- 单一父结点**(single headed)**
- 连通**(connective)**
- 无环**(acyclic)**
- 可投射**(projective)**

由此来保证句子的依存分析结果是一棵有“根**(root)**”的树结构。

依存句法分析

依存语法的优势

- 简单，直接按照词语之间的依存关系工作，是天然词汇化的；
- 不过多强调句子中的固定词序，对自由语序的语言分析更有优势；
- 受深层语义结构的驱动，词汇的依存本质是语义的；
- 形式化程度较短语结构语法浅，对句法结构的表述更为灵活。

依存句法分析

- 依存句法分(dependency parsing)的任务就是分析出句子中所有词汇之间的依存关系。
- 建立一个依存句法分析器一般需要完成以下三部分工作：

(1) 依存句法结构描述

(2) 分析算法设计与实现

(3) 文法规则或参数学习

依存句法分析

目前依存句法结构描述一般采用有向图方法或依存树方法，所采用的句法分析算法可大致归为以下四类：

- 生成式的分析方法**(Generative parsing)**
- 判别式的分析方法**(Discriminative parsing)**
- 决策式的分析方法**(Deterministic parsing)**
- 基于约束满足的分析方法**(Constraint satisfaction parsing)**

依存句法分析

(1) 生成式的分析方法

- 基本思想：采用联合概率模型$Score(x, y|\theta)$ (其中，$\boldsymbol{x}$为输入句子，$\boldsymbol{y}$ 为依存分析结构，θ为模型的参数)生成一系列依存句法树，并赋予其概率分值
- 然后采用相关算法找到概率打分最高的分析结果作为最后输出。
- 这是一种完全句法分析方法，它搜索整个概率空间，得到整个句子的依存分析结果。

依存句法分析

- 二元文法的词汇关系模型**(Bigram lexical affinities)**

$$\Pr(words,tags,links) \approx \prod_{1\le i\le n} \Pr(tag(i)\,|\,tag(i+1),tag(i+2)) \cdot \Pr(word(i)\,|\,tag(i)) \cdot \prod_{1\le i,j\le n} \Pr(L_{ij}\,|\,tword(i),tword(j))$$

其中，***tword*(*i*)** 表示符号***i*** 的标记 **(*tag*(*i*))** 和词本身**(*word*(*i*))**；L_{ij}是取值**0**或**1**的二值函数，L_{ij}=1表示***i*** 和***j*** 具有依存关系，L_{ij}=0表示***i*** 和***j*** 不具有依存关系；***n***是句子长度。

依存句法分析

- 一个标记序列**(tags)**由马尔柯夫**(Markov)**过程产生，某一个标记由该标记前面的两个标记决定，词由标记决定
- 观察每一对词**(words)**是否可以构成链接关系**(link)**的决策依赖于**[tags, words]**，即 **link** 对词汇是敏感的。
- 最终生成**words**，**tags**，**links** 的联合概率模型。

依存句法分析

方法评价

- 优点：
  - 此类方法的准确率较高
- 弱点：
  - 采用联合概率模型，在进行概率乘积分解时做了独立性假设
  - 因为采用全局搜索，算法的复杂度较高，一般为$O(n^3)$或$O(n^5)$

依存句法分析

(2) 判别式的分析方法

- 判别式句法分析方法采用条件概率模型，避开了联合概率模型所要求的独立性假设。
- 最大跨度树模型**(Maximum Spanning Trees, Mst)**
  - 定义整棵句法树的打分是树中各条边打分的加权和：

依存句法分析

$$s(\boldsymbol{x}, \boldsymbol{y}) = \sum_{(i,j)\in y} s(i,j) = \sum_{(i,j)\in y} \mathbf{w}\cdot\mathbf{f}(i,j)$$

- 其中，***s*** 表示打分值，***y*** 是句子***x*** 的一棵依存树, **(*i*, *j*)**是***y***中的结点对。**f(.)**是取值为 **1** 或 **0** 的高维二元特征函数向量，表示结点x_i和x_j之间的依存关系，如果一棵依存树中两个词“打”和“球”存在依存关系, 则：

$$f(i,j) = \begin{cases} 1 & \text{如果 } x_i = \text{'打'} \; and \; x_j = \text{'球'} \\ 0 & \text{其他} \end{cases}$$

- **w** 是特征**f(*i*,*j*)** 的权值向量，**w**在确定了特征后由样本训练得到。

依存句法分析

- 该方法基本思想就是，在点和边组成的跨度树**(spanning tree)**中找到加权和分值最高的边的组合。
- 跨度树中任意两个由词表示的节点之间都有边，根据特征和权值为每条边打分，求解最佳分析结果转化为搜索打分最高的最大跨度树问题。
- 该方法将寻求最佳依存分析转化为最优路径搜索问题，使得诸多机器学习方法和运筹学的方法得以应用，在可计算性上具有优势，
- 该方法的大部分精力放在如何降低算法复杂度上。
- 同样由于是全局搜索，算法复杂度较高。

依存句法分析

(3) 决策式的(确定性的)分析方法(deterministic parsing)

- 基本思想：
  - 模仿人的认知过程，按照特定方向每次读入一个词。
  - 每读入一个词，都要根据当前状态做出决策(比如判断是否与前一个词发生依存关系)。一旦决策做出，将不再改变。所做决策即“采取什么样的分析动作(action)”。
  - 分析过程可以看作是一步一步地作用于输入句子之上的分析动作(action)的序列。

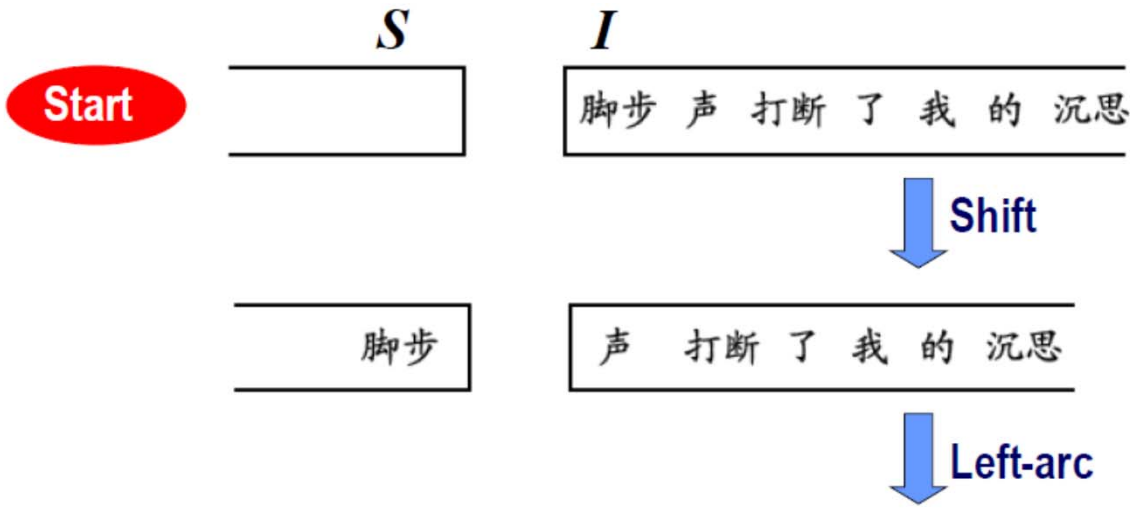
依存句法分析

- **J. Nivre**等(2003)提出的自左向右、自底向上的分析算法：
- 当前分析状态的格局(configuration)是一个三元组：**(*S, I, A*)**，***S, I, A***分别表示栈、未处理结点序列和依存弧集合。分析体系主要包含两种分析动作组合: Reduce和 Shift 。

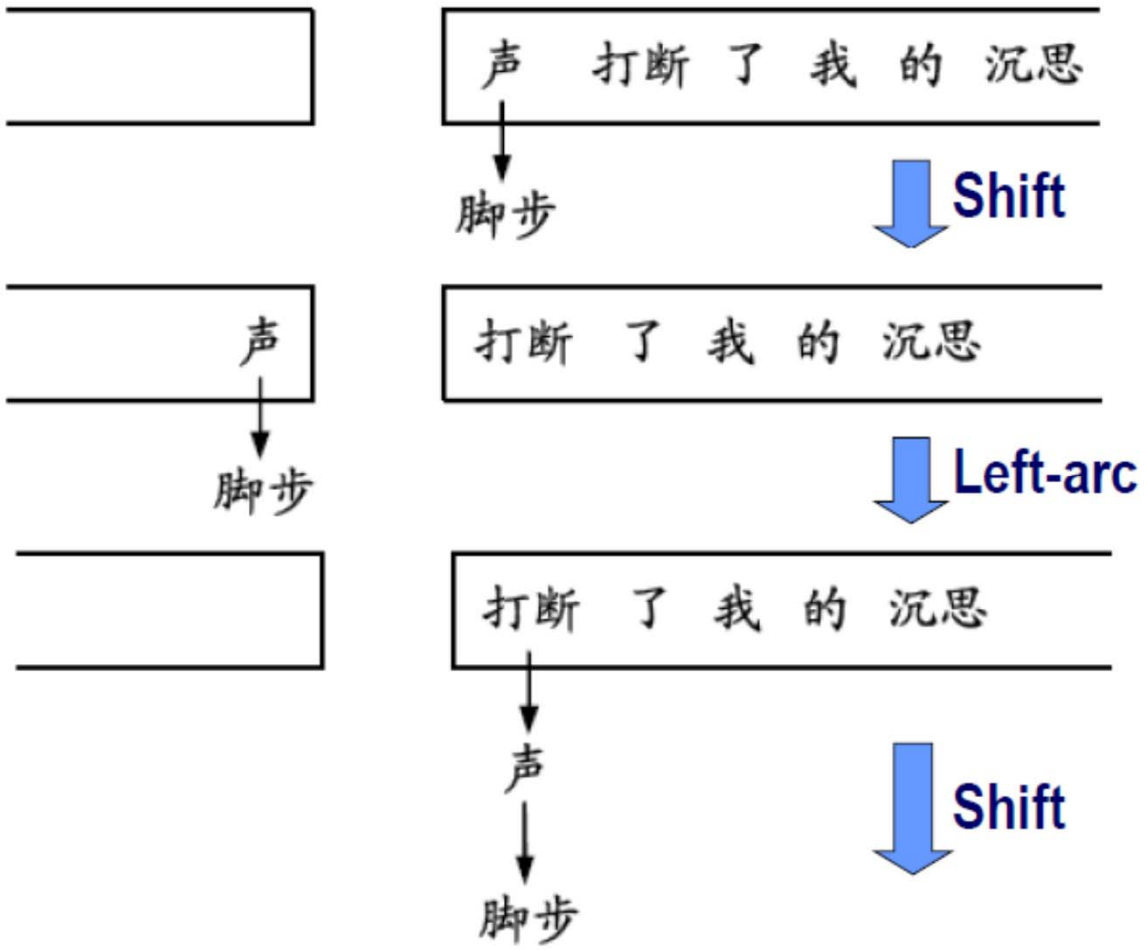
依存句法分析

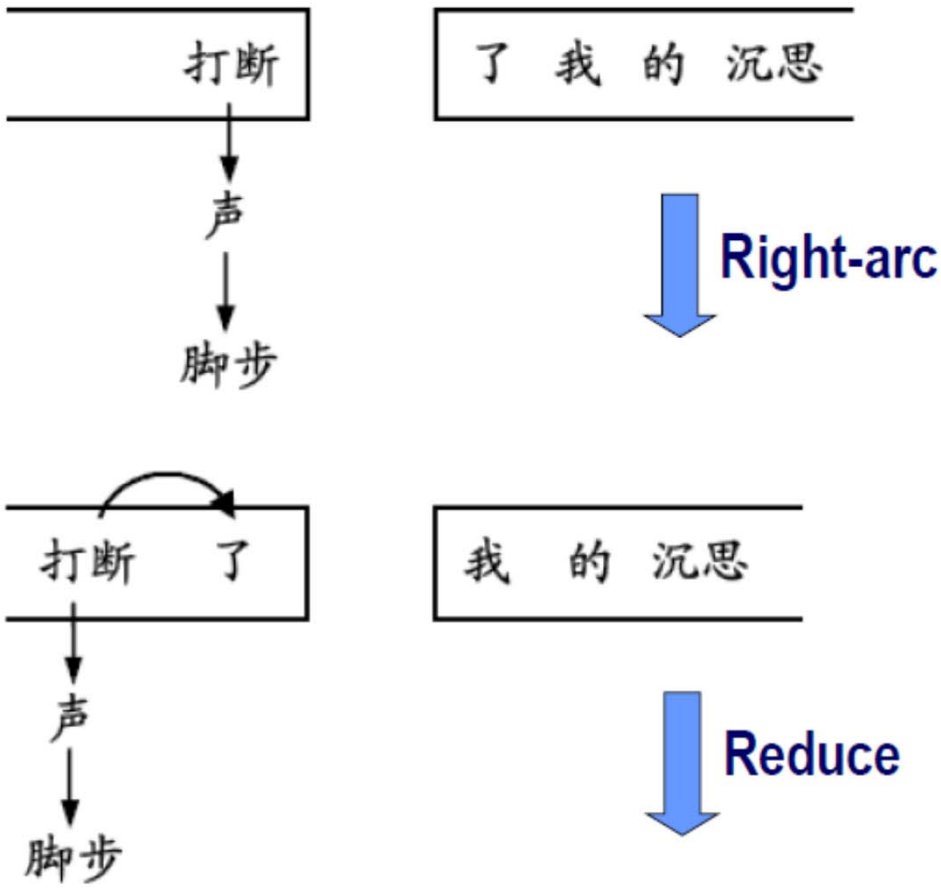
举例：分析如下句子：

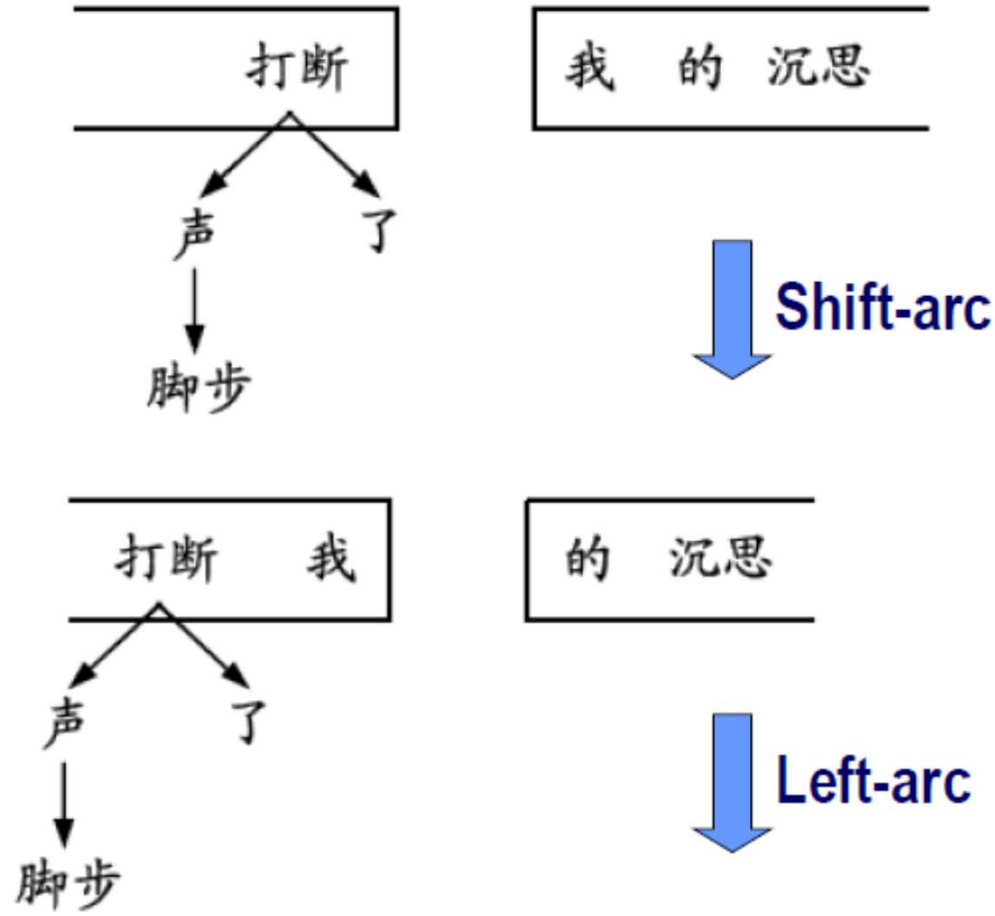
脚步　声　打断　了　我　的　沉思

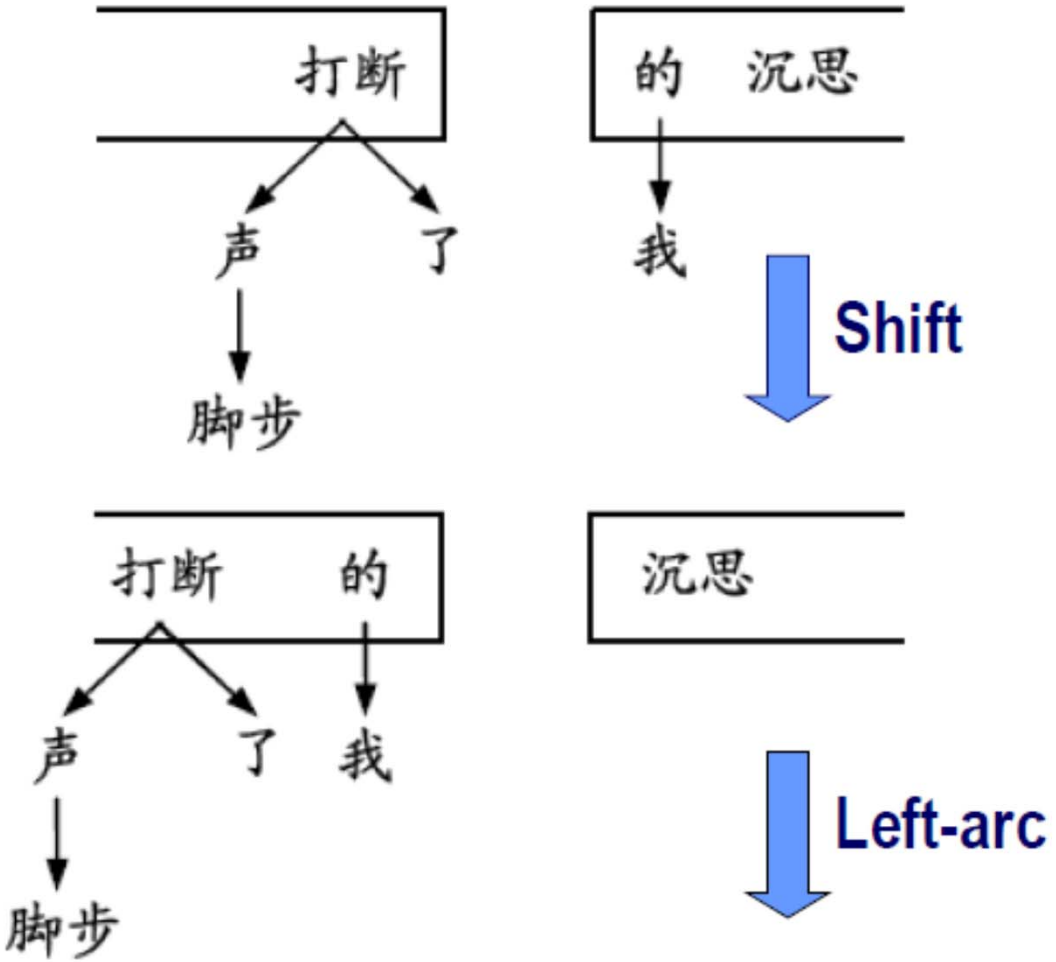


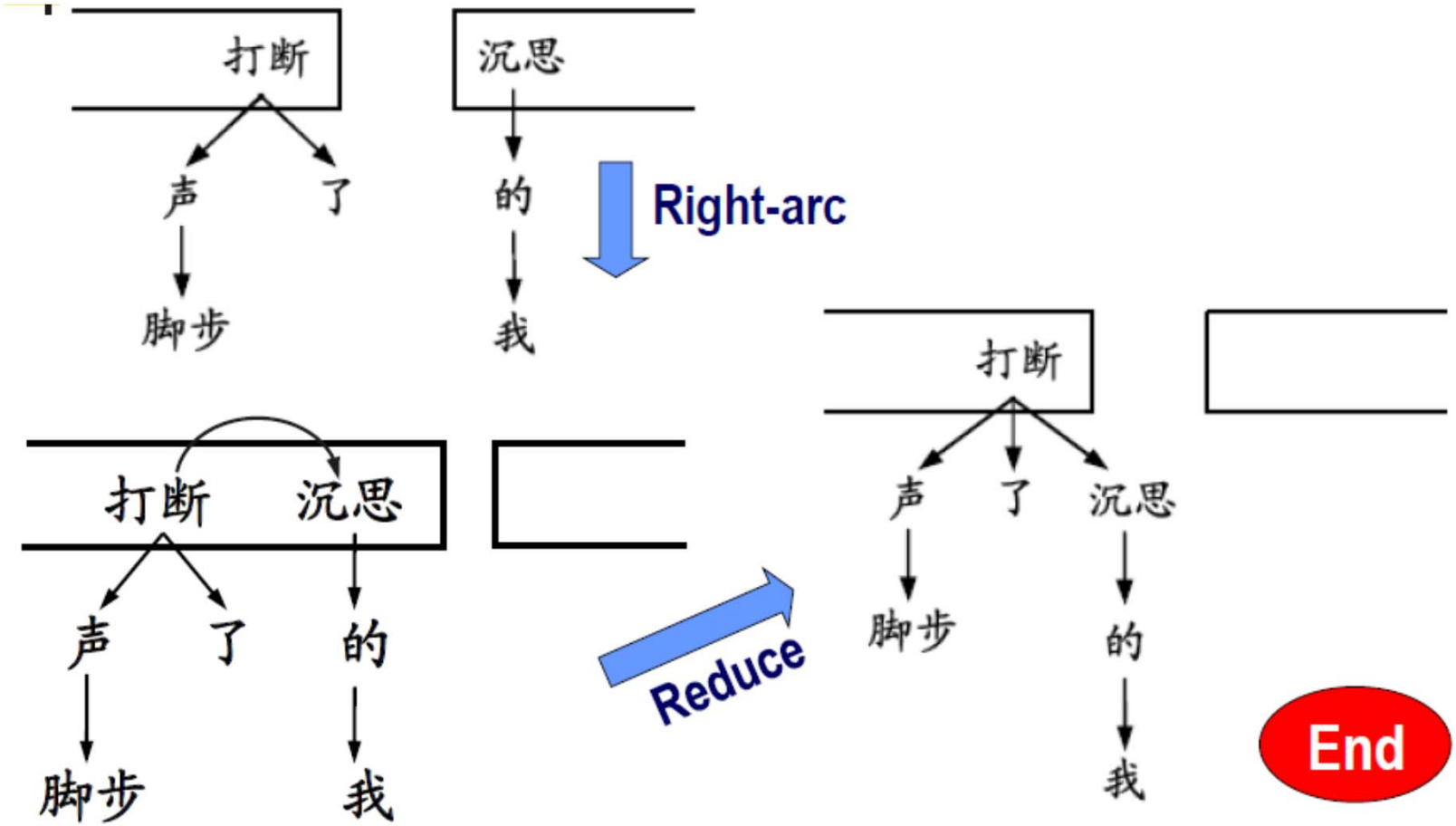
依存句法分析

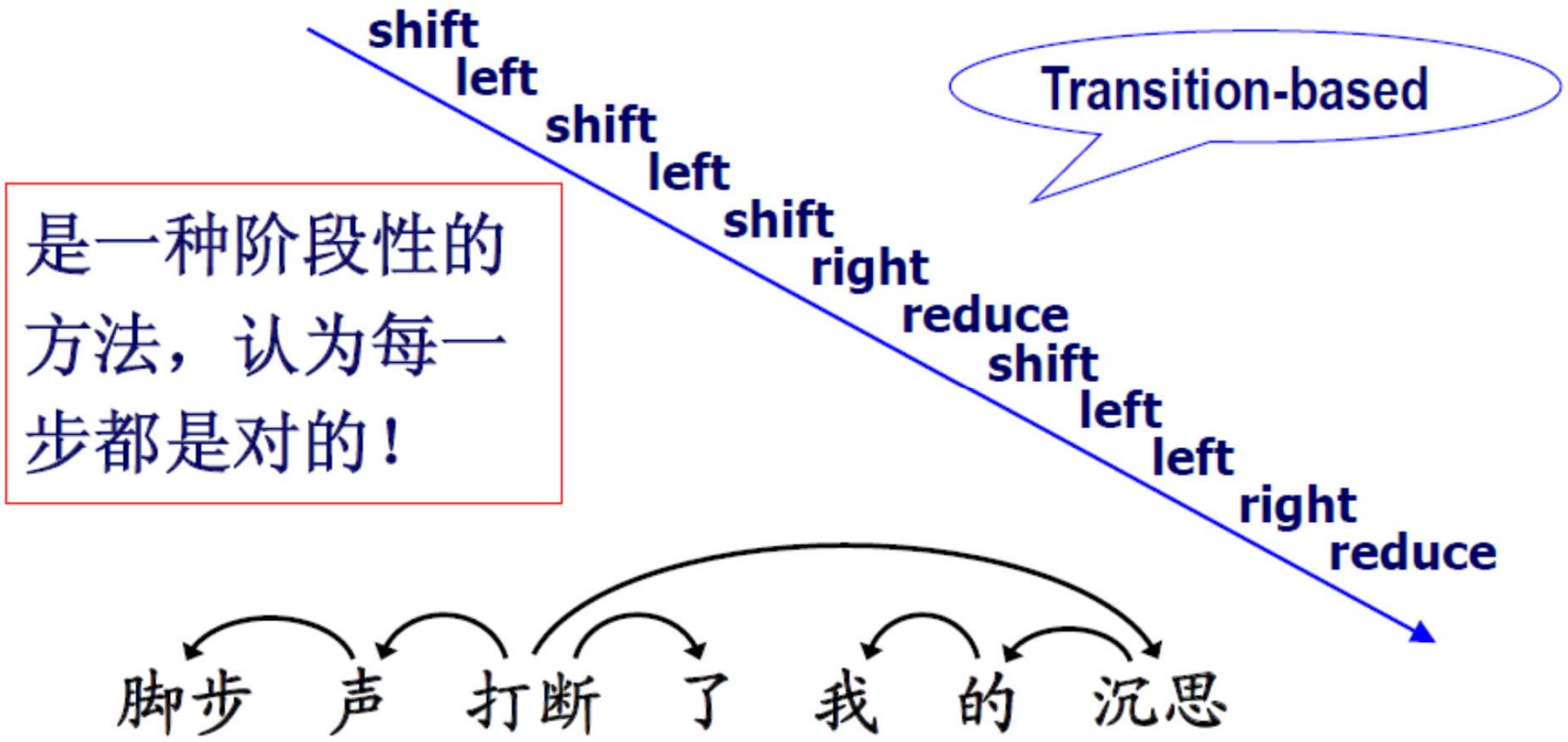



打断
了 我 的 沉思
声
脚步
Right-arc
打断 了
我 的 沉思
声
脚步
Reduce


打断
我 的 沉思
声
了
脚步
Shift-arc
打断 我
的 沉思
声
了
脚步
Left-arc


打断
的 沉思
声
了
我
脚步
Shift
打断 的
沉思
声
了
我
脚步
Left-arc


打断
沉思
声
了
的
脚步
我
Right-arc
打断
沉思
声
了
的
脚步
我
Reduce
打断
声
了
沉思
脚步
的
我
End


shift
left
shift
left
shift
right
reduce
shift
left
left
right
reduce
Transition-based
是一种阶段性的方法，认为每一步都是对的！
脚步 声 打断 了 我 的 沉思

依存句法分析

方法评价

- 优点：
  - 算法可以使用之前产生的所有句法结构作为特征；
  - 可以达到线性复杂度：$O(n)$。
- 弱点：
  - 以局部最优的加和代替全局最优，导致错误传递；
  - 准确率稍逊于全局最优算法。

部分开源的依存句法分析器

- **Stanford Parser http://nlp.stanford.edu/downloads/lex-parser.shtml**
- **MST Parser (Minimum-Spanning Tree Parser) http://www.seas.upenn.edu/~strctlrn/MSTParser/MSTParser.html**
- **MaltParser http://maltparser.org/index.html**
- **MINIPAR Parser http://webdocs.cs.ualberta.ca/~lindek/minipar.htm**
- **Layer-based Dependency Parser (LDPar) http://www.openpr.org.cn/**

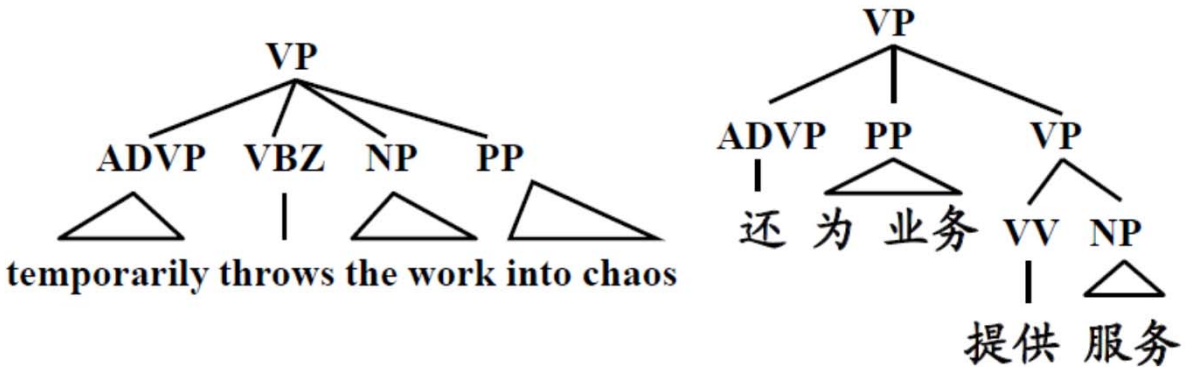
汉英句法结构特点对比

汉英句法结构特点对比

1. 汉语比英语更少地使用功能词**(function words)**，且没有形态变化：汉语中不使用限定词**(**“这、这个、那个”等**)**的名词普遍存在，复数标记**(**“们”等**)**有限并且很少出现。

汉英句法结构特点对比

2. 英语短语绝大多数以左部为中心，而汉语短语比较复杂，大多数短语类是以右部为短语中心，除了动词和介词的补语在它们的中心词之后。如：



汉英句法结构特点对比

- 介词短语 **into chaos** 在动词 **throw** 的右边，而在汉语例子中恰好相反，介词短语“为业务”在动词前面。
- 这种差异意味着在英语句子中附加在动词后面的补语引起的歧义是句法分析器需要解决的主要问题，而在汉语句子中很少有这种歧义存在。

汉英句法结构特点对比

- 在汉语句子中没有做主语的先行代词的情况普遍存在，但在英语中这种情况很少出现。
- 这样就使得汉语句法分析器很难判断一个输入到底是没有主语的子句**(IP)**结构还是仅仅是一个动词短语**VP**，如：

  He thinks it is true. / 他认为□是对的。

汉英句法结构特点对比

- 从本质上讲，英语是一种“结构型”语言，一个完整的句法结构即表示一个完整的句子。当多个单句连接起来构成复句的时候，单句与单句之间需要有显式的连接词或者短语。
- 汉语则不同，汉语“表意型”的语言特点，使得汉语句子通常受语义的牵引，一个句子是表达一个完整意义的语言单元，这种特点在长句中表现得特别明显。
- 因此，在汉语中存在一种独特的长句构成方式，就是一连串独立的简单句通过逗号或分号，连接成一个复杂的“句群”式的长句。

汉英句法结构特点对比

- 这些长句内部的各个简单句是为了表意的需要而连接在一起的，它们彼此的句法结构完全是独立的，表示彼此之间逻辑关系的连接词不是必需的。
- 因此，在很多情况下，它们之间的分隔标记仅仅是一个逗号或者分号。这类长句在汉语中称之为“流水复句”，例如：

“我现已步入中年，每天挤车，搞得我精疲力尽，这种状况，直接影响我的工作，家里的孩子也没人照顾。”

汉英句法结构特点对比

- 从中文资源联盟 **(Chinese LDC)** 发布的汉语树库**(TCT 973)**中随机地抽取出 **4431** 个长度超过**20**个词的长句，其中，流水复句有**1830**个，占全部长句的**41.3%[**李幸**, 2005]**。

汉英句法结构特点对比

汉语长句的层次化句法分析方法

1. 对包含“分割”标点的长句进行分割；
2. 对分割后的各个子句分别进行句法分析(即第一级分析)，分析得到的各个最大概率的子树根节点的词类或者短语类别标记作为第二级句法分析的输入；
3. 通过第二遍分析找到各子句或短语之间的结构关系，从而获得最终整句的最大概率分析树。

汉英句法结构特点对比

- 例句：我喜欢在春天去观赏桃花，在夏天去欣赏荷花，在秋天去观赏红叶，但更喜欢在冬天去欣赏雪景。