第6节 句法结构与依存分析

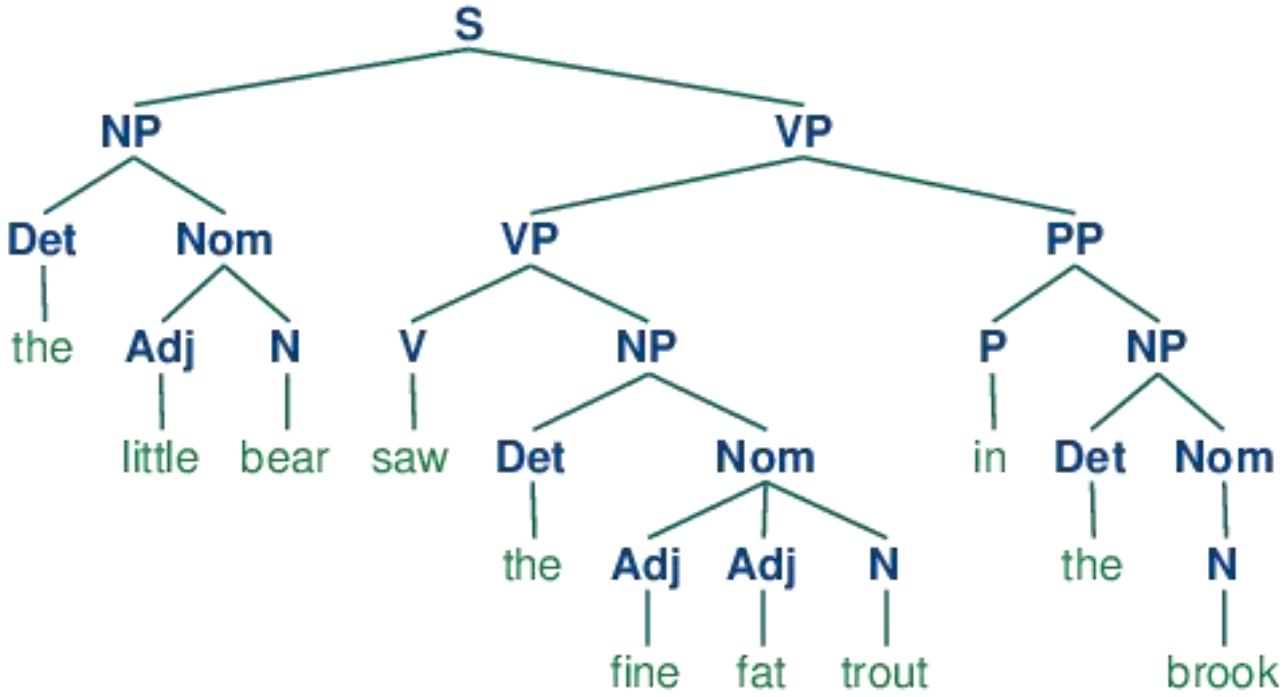
1. 句法结构

关于句法结构的两种观点：成分句法 和 依存句法

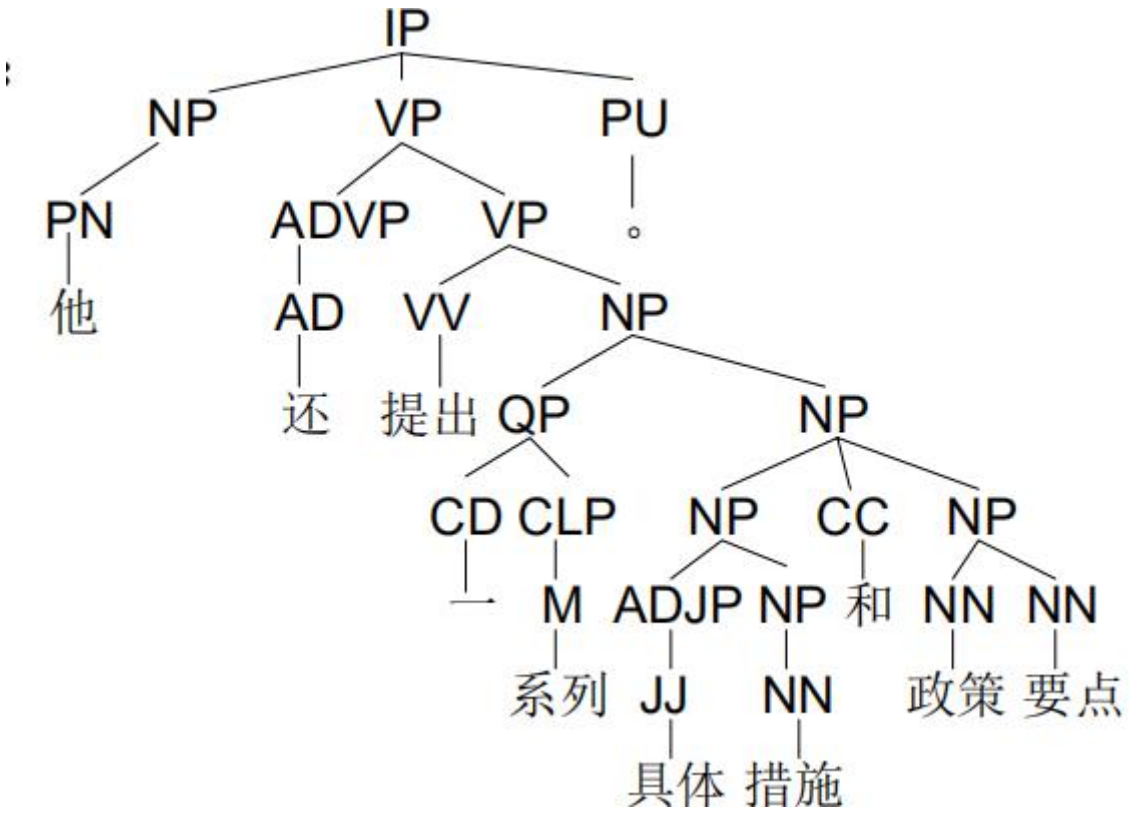
1.1 成分句法（Constituency）

- 也叫短语结构文法（phrase structure grammar）= 上下文无关语法 (CFGs)
- 这种短语语法用固定数量的rule分解句子为短语和单词、分解短语为更短的短语或单词

例如句子：The little bear saw the fine fat trout in the brook.

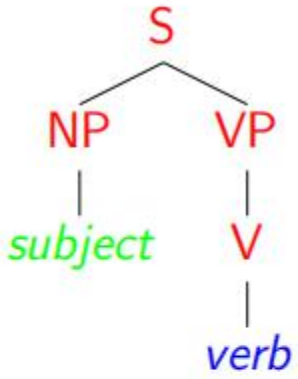


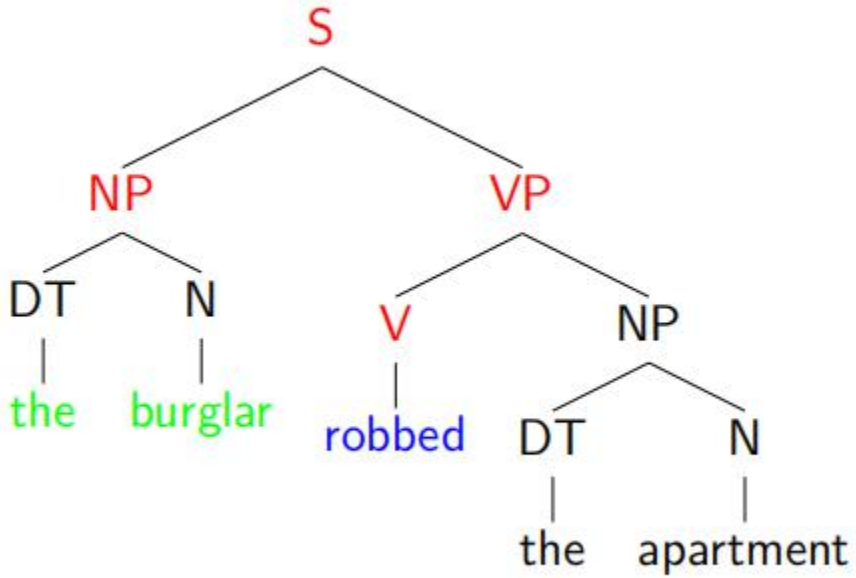
例如句子：他还提出一系列具体措施和政策要点



Symbol	Meaning	Example
S	sentence	the man walked
NP	noun phrase	a dog
VP	verb phrase	saw a park
PP	prepositional phrase（介词短语）	with a telescope
Det	determiner（限定词）	the
N	noun	dog
V	verb	walked
P	preposition	in

我们分析一个句子的结构





⇒ "the burglar" is the subject of "robbed"

英语句子的顺序是 subject – verb – object

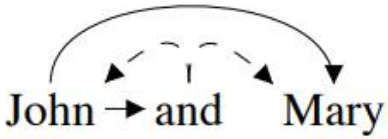
日语句子的顺序是 subject – object – verb

English:	IBM bought Lotus
Japanese:	*IBM Lotus bought*
English:	Sources said that IBM bought Lotus yesterday
Japanese:	*Sources yesterday IBM Lotus bought that said*

1.2 依存句法

■ 依赖关系结构（Dependency structure）揭示了句子中词的依赖关系。如果一个单词修饰另一个单词，则称该单词依赖于另一个单词。

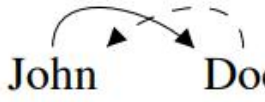
■ 所有依赖弧线形成一个有向树，根节点是一个root符号，除了根节点以外的每个单词w都有一条从根到w的有向路径。



(a) Coordination

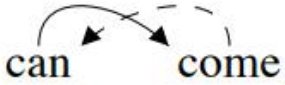
(b) Infinitive Verbs

(c) Noun Phrases



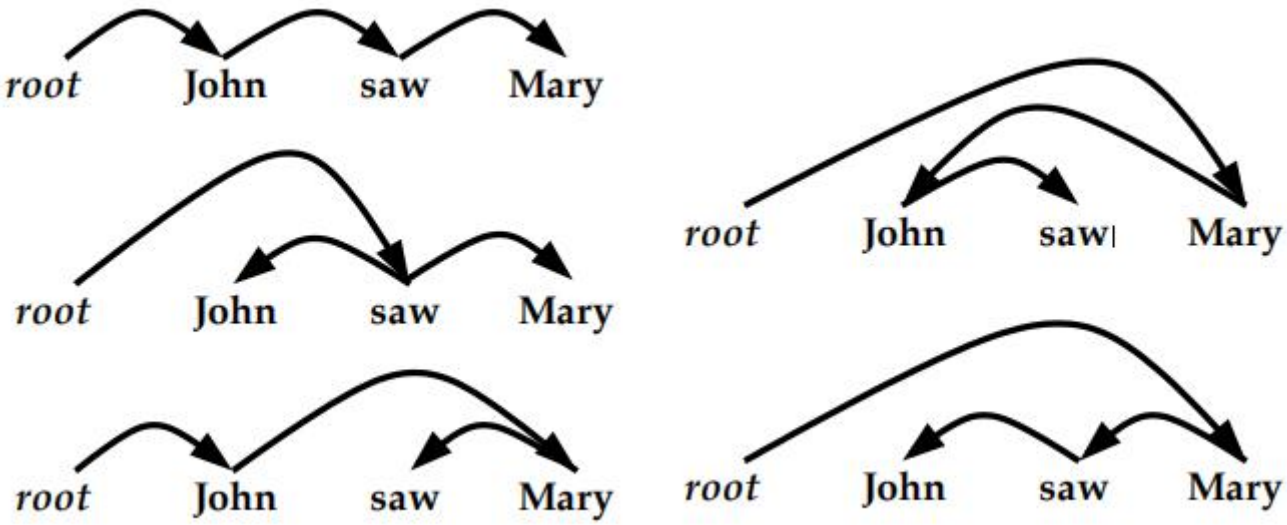
(d) Noun Sequence

(e) Prepositional Phrases



(f) Verb Groups

例如句子：分析句子John saw Mary，所有依赖项如下



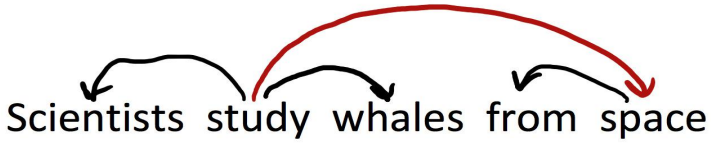
两种语法结构能够揭示句子中不同的信息，所以当你在其他任务中，需要用到句子中的短语结构就用constituent，而需要用到词与词之间的依赖关系就用dependency

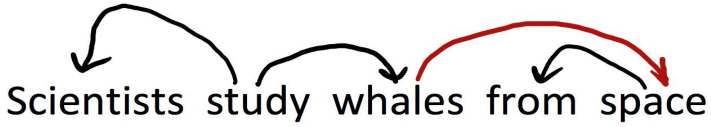
依存句法树能够根据成分句法树转换而来，但成分句法树不能通过依存树转化来。转换的规则是head-finding rules from Zhang and Clark 2008

1.2.1 歧义

通过句法树可以表达歧义，一个确定的句法树对应句子的一个确定解读。

比如对介词短语依附（attachment of prepositional phrases (PPs)）





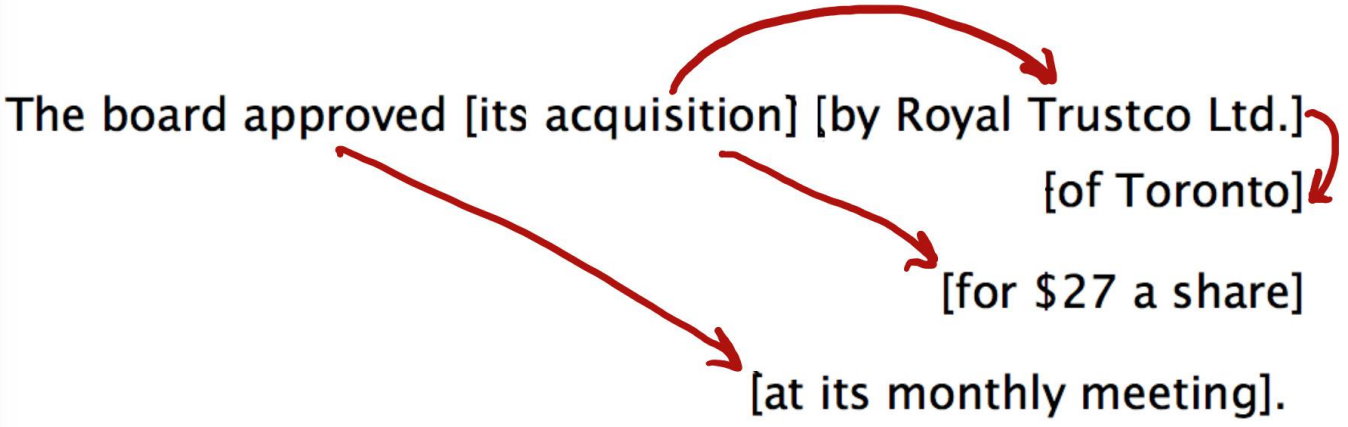
“from space”这个介词短语到底依附谁？不同的答案导致对句子不同的理解

结构歧义

- I saw a boy in the park.
- [I saw a boy] in the park.
- I saw a [boy in the park].
- 关于鲁迅的文章。
- 把重要的书籍和手稿带走了。

依附歧义

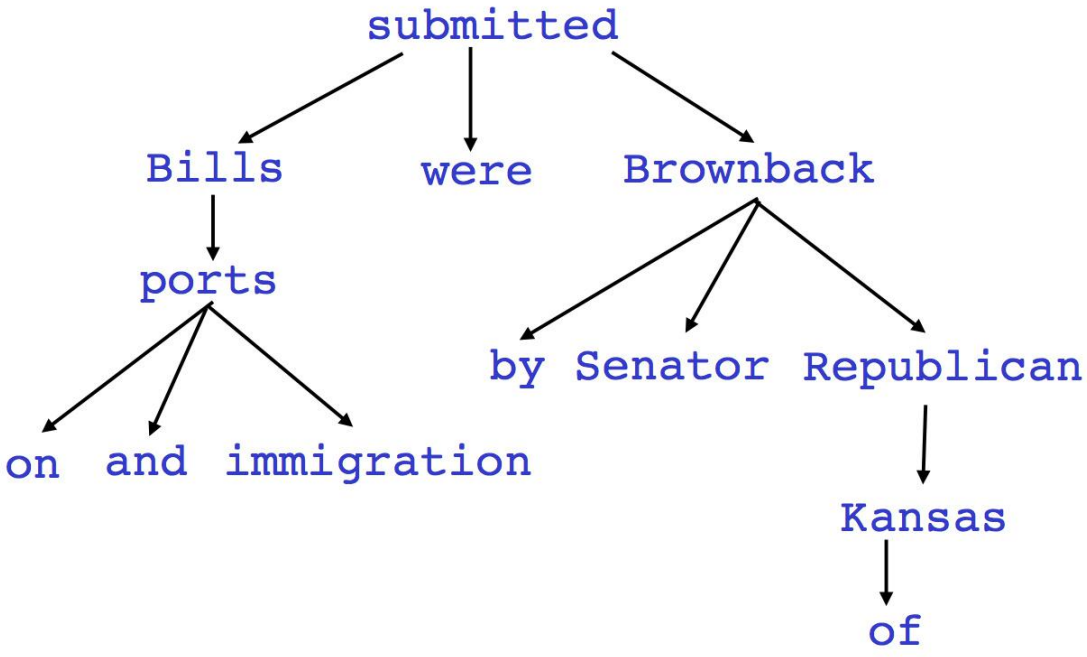
很难确定如何把一个短语（介词短语、状语短语、分词短语、不定式）依附到其他成分上去，比如下列句子



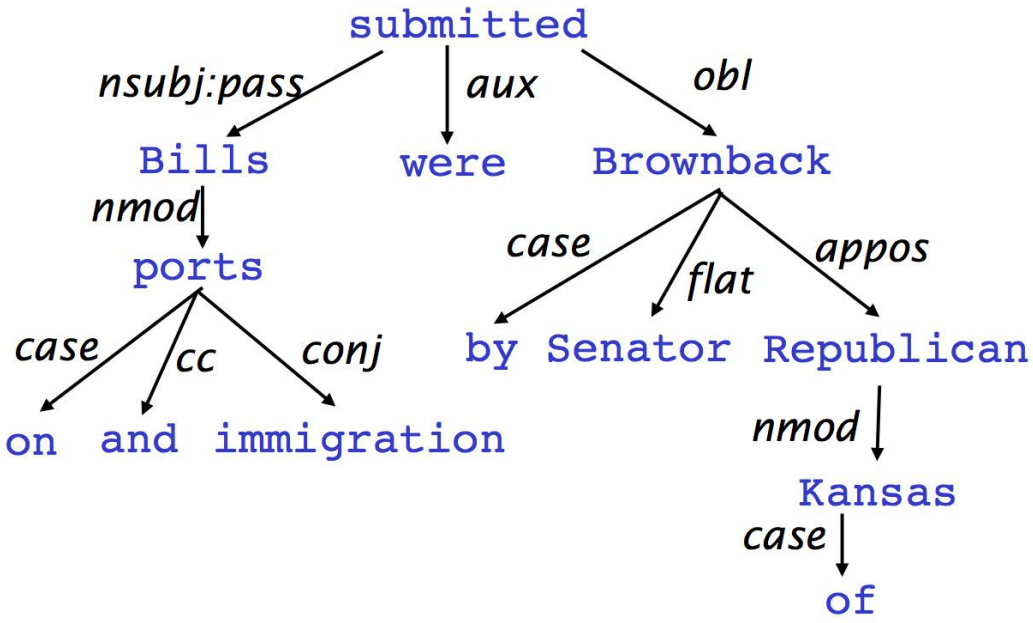
每个括号中都是一个短语，它们依附的对象各不相同。对于n个短语来讲，组成的树形结构有 $C_n = (2n)!/[(n+1)!n!]$ 这是Catalan数，指数级增长，常用于树形结构的计数问题

1.2.2. 依存文法与依存结构

- 依存句法树标注简单，parser准确率高，所以后来（特别是最近十年）基本上就是依存句法树的天下（至少80%）
- 不标注依存弧label的依存句法树就是短语结构树的一种：



- 依赖语法假设句法结构由词汇项间的关系组成，这种二元不对称关系(箭头)称为依赖性(dependencies)
- 箭头通常标记语法关系的名称（主语，介词宾语，同位语等）
- 箭头连接head（被修饰的主题）与指向dependent（修饰语）
- 通常，依赖关系形成一棵树（连接的，非循环的，single-head）



1.2.3. 依存句法分析方法

- **Dynamic programming**

  在两端生成heads解析条目，而不是在中间

- **Graph algorithms**

  为句子创建最小生成树

- **Constraint Satisfaction**

  在某个图上逐步删除不符合要求的边，直到成为一棵树

- **Transition-based parsing** or **deterministic dependency parsing**

  主流方法，基于贪心决策动作拼装句法树

基于转换的依存分析（Arc-standard transition），具体见参考资料\5LN455-F8.pdf

句法分析可用的特征

依赖分析的信息来源是什么?

① 双词汇亲和(Bilexical affinities)，比如discussion与issues

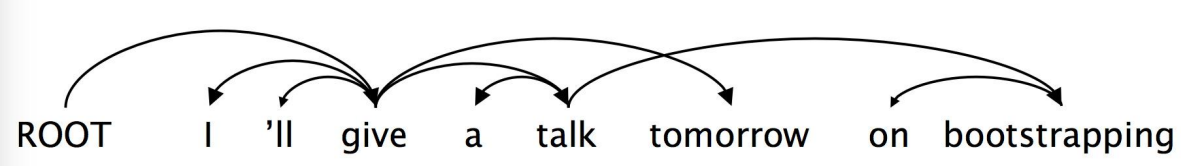
② 词语间距(Dependency distance)，因为一般相邻的词语才具有依存关系

③ 中间词语(Intervening material)，如果中间词语是动词或标点，则两边的词语不太可能有依存

④ 词语配价(Valencyof heads)，一个词语最多有几个依赖者

依存句法分析

有几个约束条件:



① root只能被一个词依赖

② 无环

③ 英语中大部分句子是projective（箭头不交叉）的，少数是non-projective（箭头交叉）

2. 句法分析的基本方法

- 基于CFG规则的分析方法
  - CYK
  - 线图分析法 (chart parsing)
  - Earley (厄尔利)算法
  - LR算法
    - Top-down: Depth-first/- Breadth-first
    - Bottom-up
- 基于概率上下文无关文法的分析方法
  - Probabilistic Context-Free Grammar (PCFG)

假设有以下文法：

$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$

① w = aaabb in L(G) ?

② w = aaabbb in L(G) ?

自底向上解析过程为：

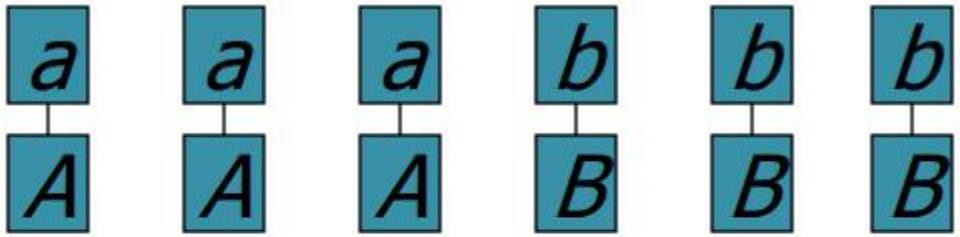
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



自底向上解析过程为：

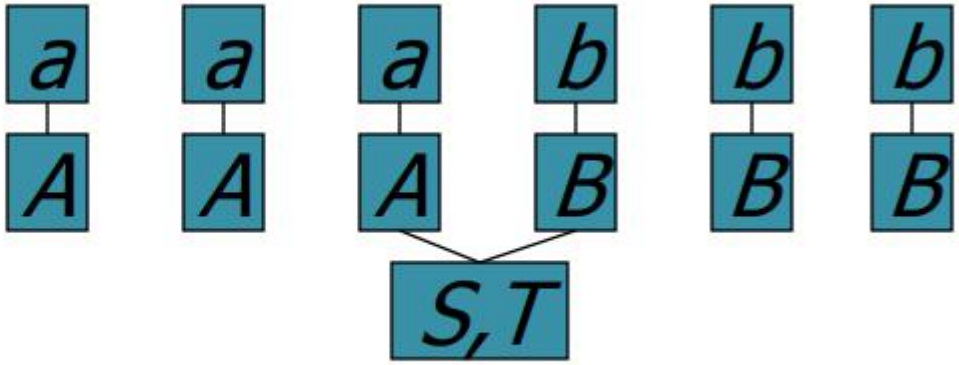
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



自底向上解析过程为：

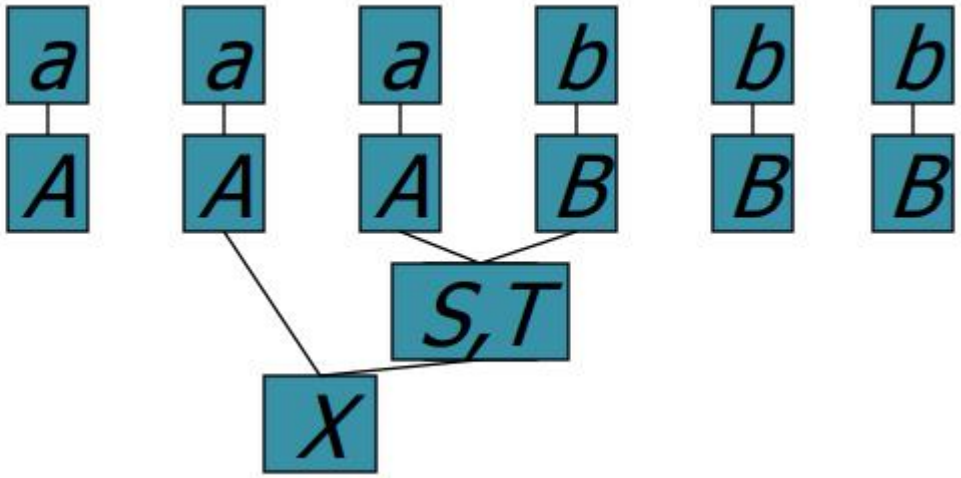
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



自底向上解析过程为：

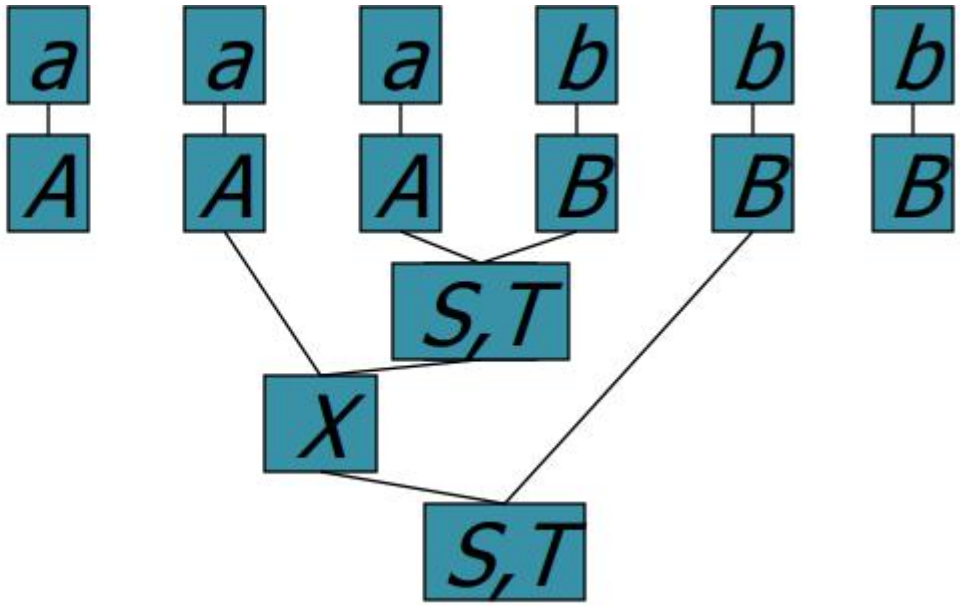
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



自底向上解析过程为：

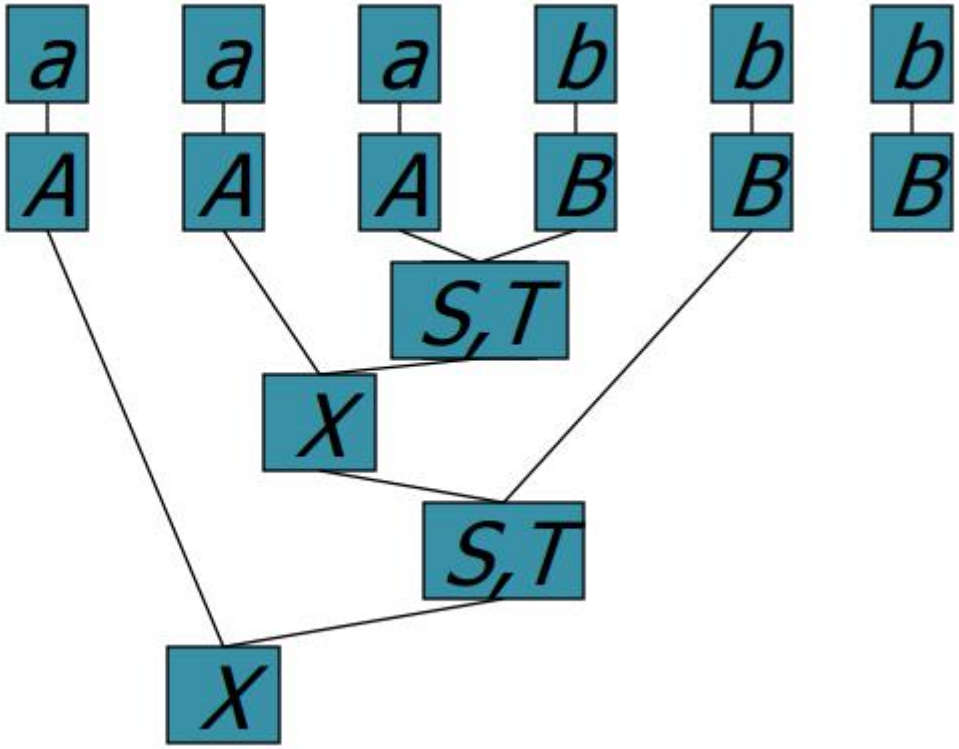
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



自底向上解析过程为：

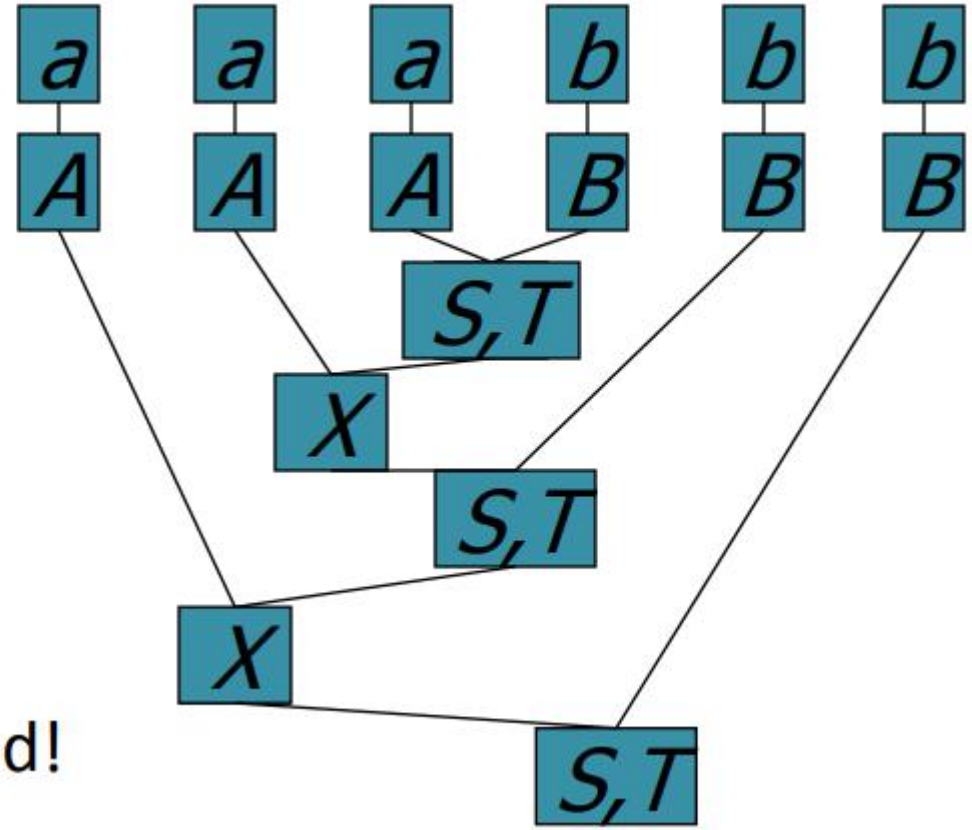
$S \rightarrow \varepsilon \mid AB \mid XB$

$T \rightarrow AB \mid XB$

$X \rightarrow AT$

$A \rightarrow a$

$B \rightarrow b$



S is included so

aaabbb accepted!

图解来自：Dor Altshuler -CKY (Cocke-Kasami-Younger) and Earley Parsing Algorithms

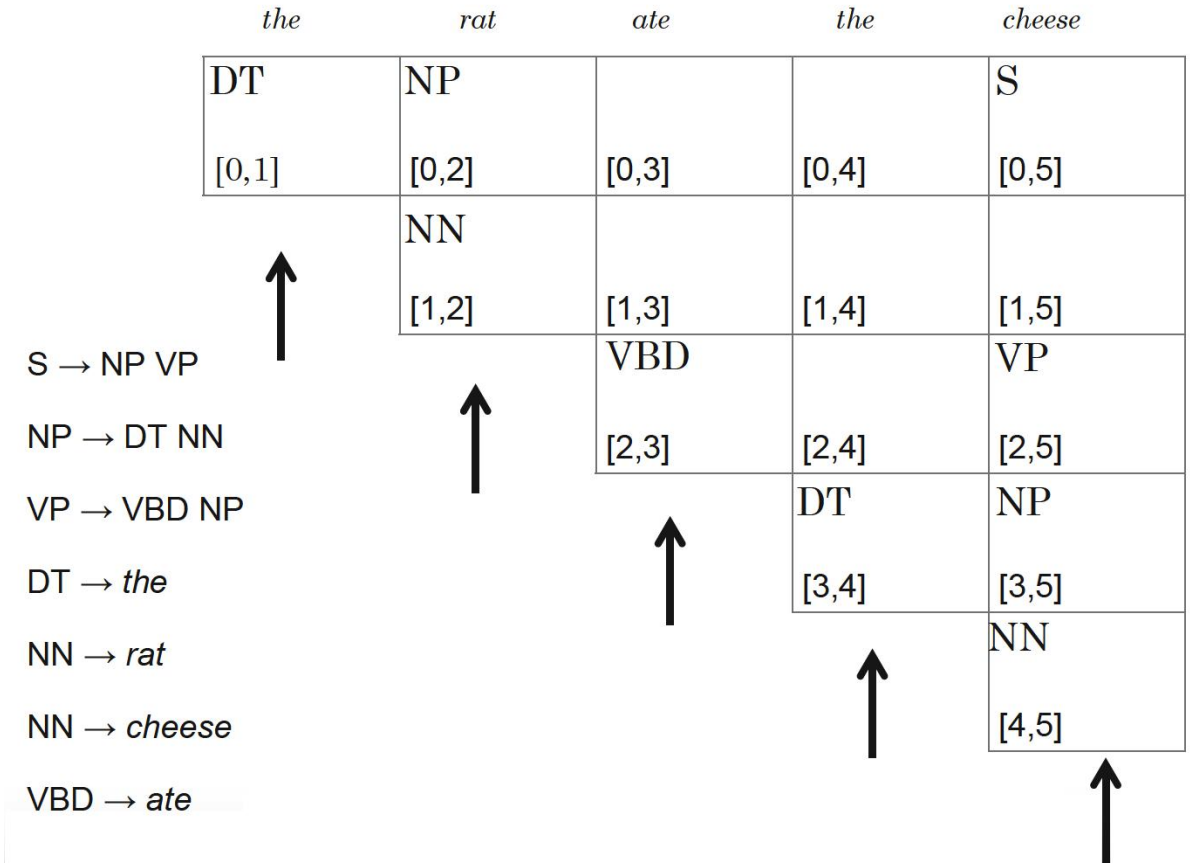
2.1 CYK算法

- 自下而上的分析方法
- 上下文无关文法（CFG）

$$A \to a \text{ 或 } A \to BC$$

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

- 构造 $(n+1)\times(n+1)$ 识别矩阵



具体演示：

https://en.wikipedia.org/wiki/CYK_algorithm#/media/File:CYK_algorithm_animation_showing_every_step_of_a_sentence_parsing.gif

CYK算法描述

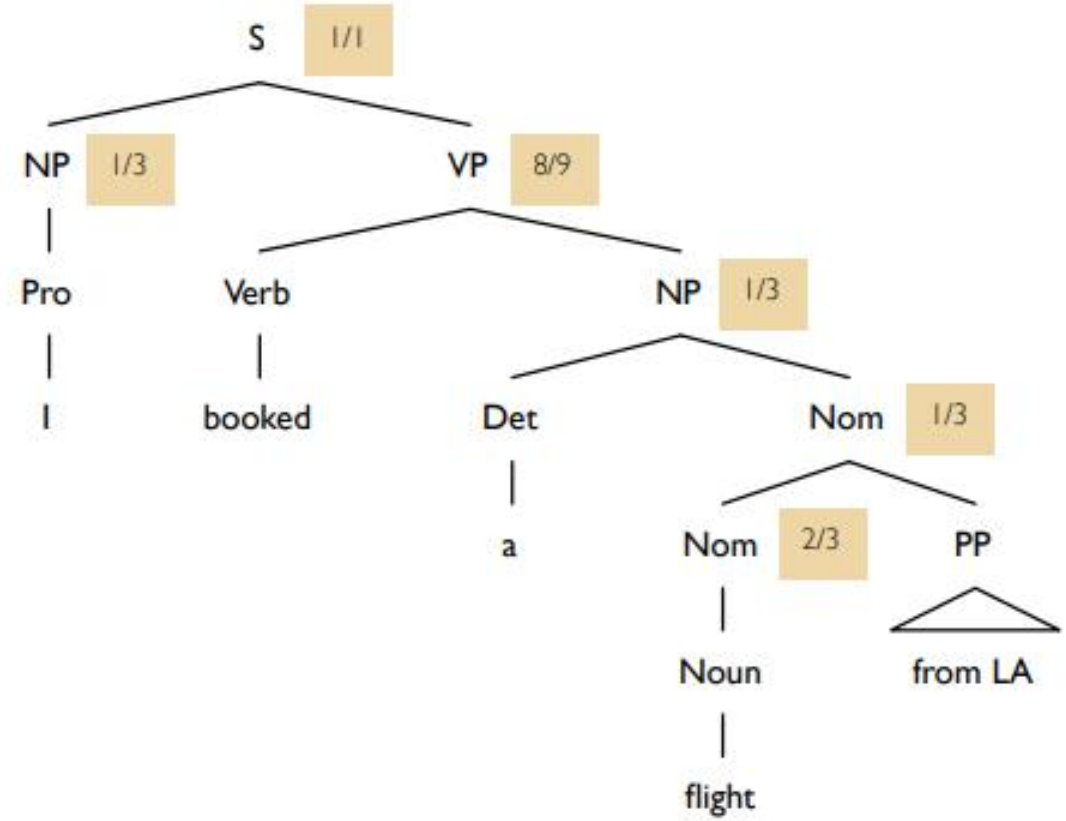
```
function CKY (word w, grammar P) returns table
for i <- from 1 to LENGTH(w) do
    table[i-1, i] <- {A | A      wi ∈ P }
for j <- from 2 to LENGTH(w) do
    for i <- from j-2 down to 0 do
        for k <- i + 1 to j – 1 do
            table[i,j] <- table[i,j] ∪ {A | A -> BC ∈ P,
                B ∈ table[i,k], C ∈ table[k,j] }

If the start symbol S ∈ table[0,n] then w ∈ L(G)
```

2.2 基于概率上下文无关文法的分析方法 (PCFG)

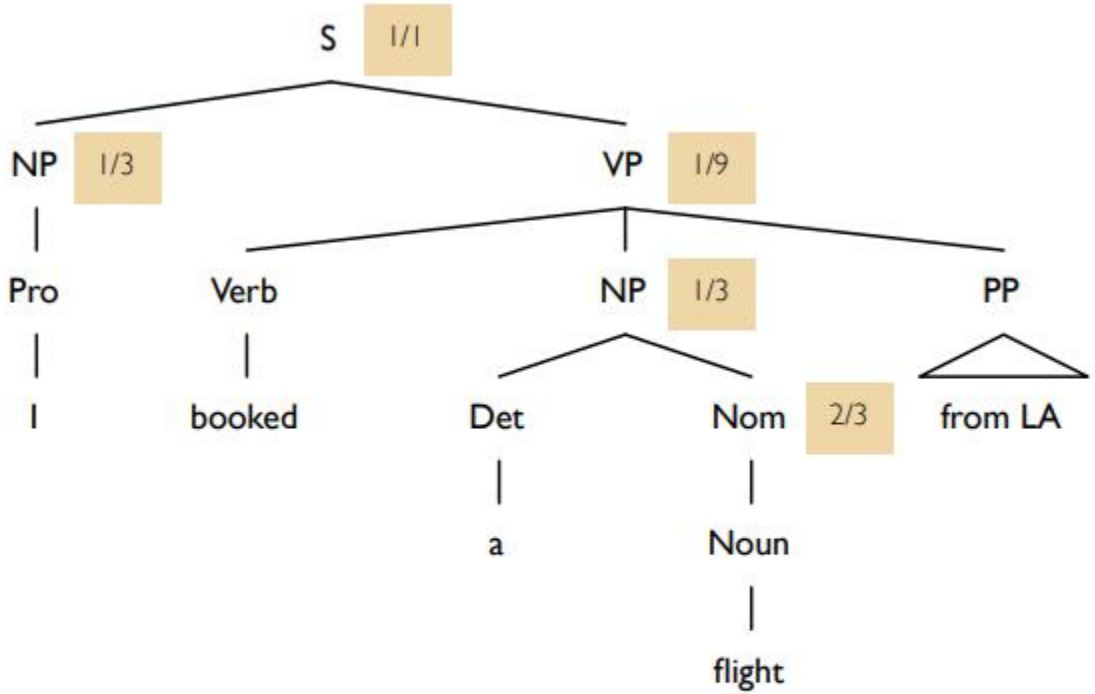
- 上下文无关文法
- 每条规则有对应的概率
- 规则概率和为1

假设以下例子：

Rule	Probability
S → NP VP	1
NP → Pronoun	1/3
NP → Proper-Noun	1/3
NP → Det Nominal	1/3
Nominal → Nominal PP	1/3
Nominal → Noun	2/3
VP → Verb NP	8/9
VP → Verb NP PP	1/9
PP → Preposition NP	1



Probability: 16/729



Probability: 6/729

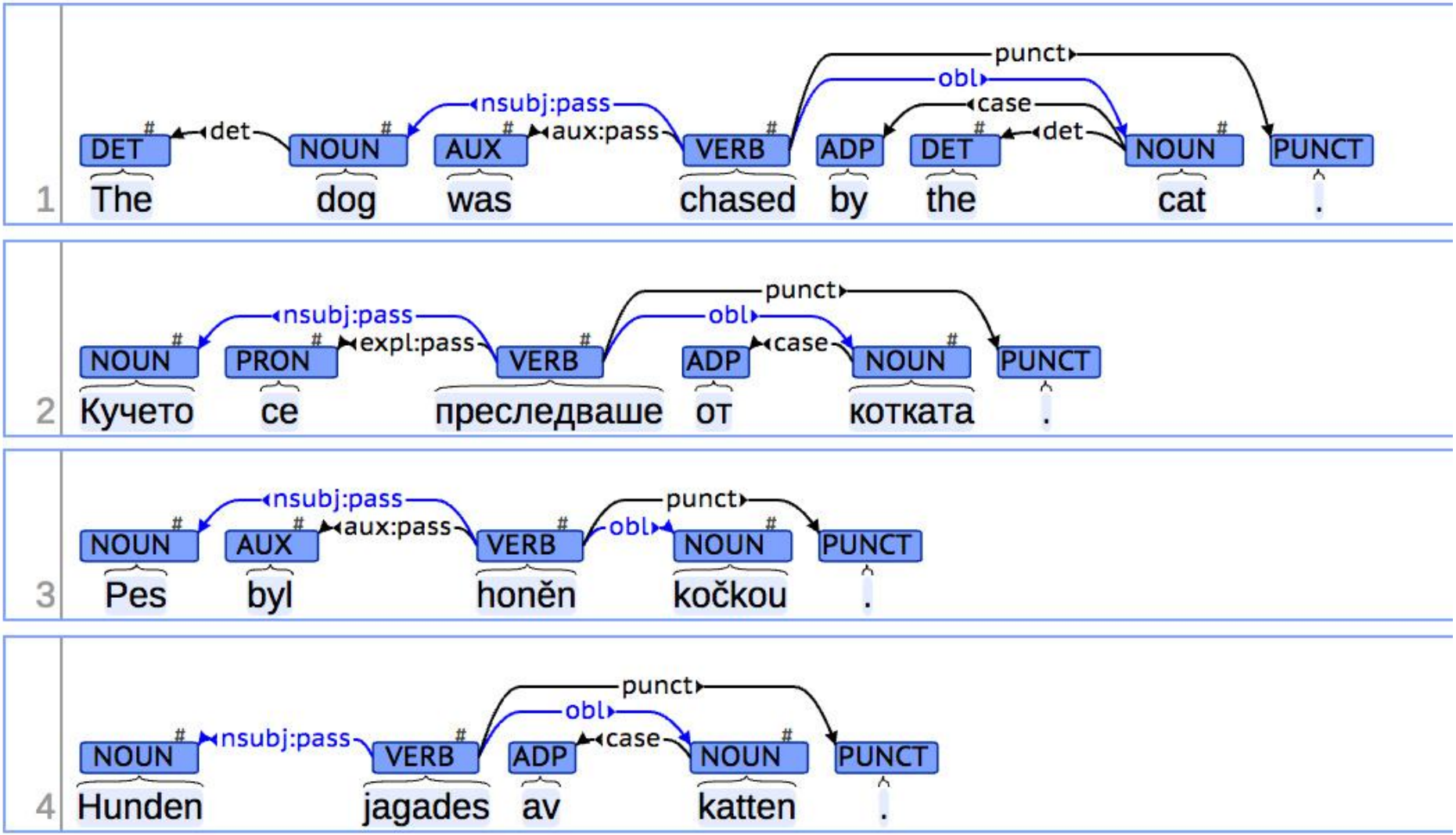
注释数据集的崛起：Universal Dependencies treebanks

虽然上下文无关文法中的语法集很容易写，无非是有限数量的规则而已，但人工费时费力标注的树库却茁壮成长了起来。在1993年首次面世的Universal Dependencies treebanks如今在Google的赞助下发布了2.0，其授权大多是署名-相同方式共享，覆盖了全世界绝大多数语言(不包括简体中文)。

其官网是：http://universaldependencies.org/

GitHub主页是：https://github.com/UniversalDependencies

树库示例


1
DET
NOUN
AUX
VERB
ADP
DET
NOUN
PUNCT
det
nsubj:pass
aux:pass
punct
obl
case
det
The
dog
was
chased
by
the
cat
.
2
NOUN
PRON
VERB
ADP
NOUN
PUNCT
nsubj:pass
expl:pass
punct
obl
case
Кучето
се
преследваше
от
котката
.
3
NOUN
AUX
VERB
NOUN
PUNCT
nsubj:pass
aux:pass
punct
obl
Pes
byl
honěn
kočkou
.
4
NOUN
VERB
ADP
NOUN
PUNCT
nsubj:pass
punct
obl
case
Hunden
jagades
av
katten
.

注释数据的崛起

人们偏好树库多于规则的原因是显而易见的，树库虽然标注难度高，但：

- 每一份劳动都可被复用
  - 可以在其上构建许多解析器，词类标注器等，用于词性标注、命名实体识别等等任务；
  - 语言学的宝贵资源
- 广泛的覆盖面，而不仅仅是一些直觉
- 频率和分配信息
- 评估系统的一种方法

未来工作

走在最前沿的是Google。趋势是:

更大更深调参调得更好(更昂贵)的神经网络;

Beam Search（集束搜索/束搜索）；

在决策序列全局进行类似CRF推断的方法。

Google的SyntaxNet 中的 Parsey McParseFace的效果:

Method	UAS	LAS (PTB WSJ SD 3.3
Chen & Manning 2014	92.0	89.7
Weiss et al. 2015	93.99	92.05
Andor et al. 2016	94.61	92.79

注：Beam Search是一种启发式图搜索算法，通常用在图的解空间比较大的情况下，为了减少搜索所占用的空间和时间，在每一步深度扩展的时候，剪掉一些质量比较差的结点，保留下一些质量较高的结点。这样减少了空间消耗，并提高了时间效率，但缺点就是有可能存在潜在的最佳方案被丢弃，因此Beam Search算法是不完全的，一般用于解空间较大的系统中。

补充

汉语树库

http://www.hankcs.com/nlp/corpus/chinese-treebank.html#h3-6

中文语义依存分析

https://github.com/dingyu-scir/SDP-Specification/blob/master/sdp_guidline.md

依存分析在线演示

- 哈工大LTP语言云：http://ltp.ai/demo.html
- Hanlp: http://hanlp.com/

Thank you!