5.1.1 Introduction and recap of parsing

本节简要回顾了人类语言广泛存在的歧义导致的语法分析过程中的困难，这些歧义在之前的课程中已有详细描述。

5.1.2 Parsing noun sequences

接下来介绍了一类具体的语法分析例子，即名词序列的分析（Parsing noun sequences）。从语义角度来看，名词序列中的修饰方式是多样的，例如 Fish tank = tank that holds fish，Fish net = net used to catch fish，Fish soup = soup made with fish，Fish oil = oil extracted from fish。

此外，虽然名词序列的关键词（head of the compound）常出现在末尾，但这不是必须的，例如 attorney general（首席检察官，单词含义律师、总的）。

尽管实际生活中一个由名词序列形成的短语是有唯一含义的。但理论上长度为 n+1 的名词序列具有 $N=\frac{1}{n+1}C_{2n}^{n}$，例如长度为 4（n=3）的序列具有 5 种分析方式。

$N=\frac{1}{n+1}C_{2n}^{n}, for\ n\geq 0$ 定义的序列称为 Catalan number。该序列描述的另外两个有趣的情景如下图所示。

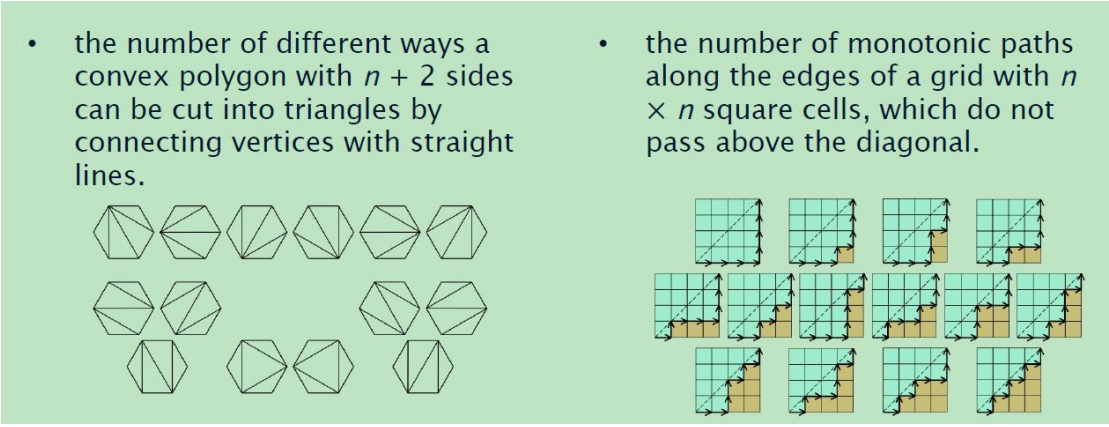

图 5.1.1

5.2 Prepositional Phrase Attachment (1)

本节介绍另一类分析问题，即介词短语附属（prepositional phrase attachment）。一般而言，介词短语既可以修饰前面的名词（the noun that is closest to them），也可以修饰该名词

前面的动词（the verb before that noun）。而自动识别介词短语修饰的目标（the problem of automatically figuring out this attachment），就是介词短语附属问题。

由此可见，介词短语附属分为两类。前者称为 low attachment（or nominal attachment），例如 is chairman of Elsevier 中 of Elsevier 修饰 chairman；所谓 of Elsevier 修饰 chairman（modify chairman），是指 chairman 是与 Elsevier 相关的（the chairman is associated with Elsevier），而不是与 is 相关。后者称为 high attachment（or verbal attachment），例如 join board as director 中 as director 修饰 join；所谓 as director 修饰 join，是指 as director is the way in which the person join the board。

应当指出，这两类介词短语附属情况对应的介词短语在语法分析树中的位置是不同的。例如，对于 Police shoot man with box cutters，两类对应的分析树分别如图 5.2.1 和 5.2.2。

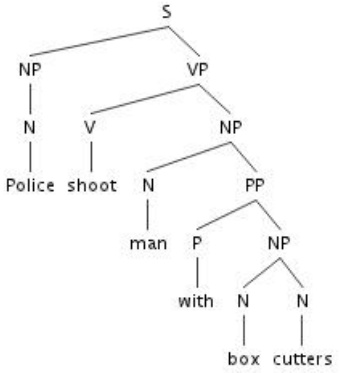

图 5.2.1

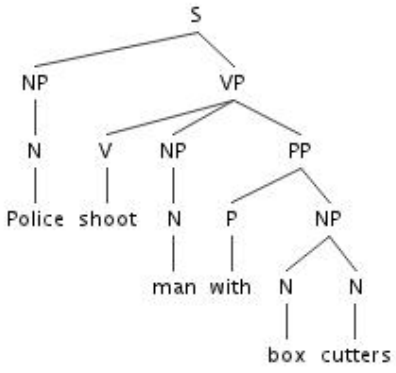

图 5.2.2

一般而言，决定介词短语类型的信息主要包含于 4 个单词中：the preposition, the verb before the preposition, the noun before the preposition, and the noun after the preposition。而且，统一使用这 4 个单词作为特征便于后续分类算法的设计。例如，我们可以将包含介词短语的句子整理为如图 5.2.1 形式的样本用于后续算法的训练或测试。

Sent #	Verb	$Noun_1$	Preposition	$Noun_2$	Class
0	join	board	as	director	V
2	named	director	of	conglomerate	N
3	caused	percentage	of	deaths	N
6	bring	attention	to	problem	V
12	led	team	of	researchers	N
16	including	three	with	cancer	N
24	imposed	ban	on	uses	N
26	made	paper	for	filters	N
28	dumped	sacks	of	material	N
28	dumped	sacks	into	bin	V

图 5.2.1

一个这样的数据集是 RRR 1994，包含 27937 个介词短语样本，从上一章介绍过的 Penn Treebank 数据集提取而来。

5.3 Prepositional Phrase Attachment (2)

首先概括了监督分类的一般过程：

1. Manually label a set of instances.

2. Split the labeled data into training and testing sets.

3. Use the training data to find patterns.

4. Apply these patterns on the testing data set.

5. For evaluation: use accuracy (the percentage of correct labels that a given algorithm has assigned on the testing data).

6. Compare with a simple baseline method.

第六步中的基准方法可以用如下方式选择：Find the more common class (label) in the training data and assign it to all instances of the testing data set. 注意到最终的正确率是要在测试集上执行算法得到的。

为了提高正确率，算法需要利用语言学的知识。例如，介词 of 对应低附属的比例显著更高，在 RRR94 的训练集上，这一比例高达 98.7%，且含有 of 的介词短语占全体比例 27%，以上两点使得“是否含有 of”成为介词短语的重要特征，这两点优势可以概括为：

1. 信息量大（be very informative），98.7% of the time it is linked with the low attachment

class in training set；

2. 出现频率高（be very frequent），27.0% of the entire training set。

与之相对，介词 against 出现频率小于 1%，且对应 48%的低附属，因此频率低且信息量少，难以直接作为有效的特征。

5.4 Prepositional Phrase Attachment (3)

承接上节。通过上节介绍的语言学知识可以提高算法的正确率，但前提是这些知识对于新样本具有泛化能力。一种极端的情况是针对每个训练样本设计一条判断准则，这些准则对应的决策树算法对测试集几乎没有意义。

此外，Radev 还指出对于 RRR 1994 数据集而言，不存在正确率 100%的算法，因为该数据集自身就存在不一致（discrepancy）的情况，即同一样本不止一次出现且人工标记类别不同。出现这种情况的原因主要有二：

1. 人工标记过程中标记者前后的判断标准不一致；

2. 某些情况下只保留 4 个单词损失了一些信息，同样的 4 个单词在更丰富的上下文中可能对应不同的附属类别。

只利用 of 和 to 两条规则的算法过于简单，下面列出几条进一步改进的思路：

1. 使用更多有效的词特征，例如其他介词，甚至名词和动词（但是一般而言名词和东西的频率过低，难以直接将其是否出现作为特征）；

2. 对于不能用显著规则判断的样本，设计更精明的方式进行估计（而不是单一地设置为某个默认值），例如使用训练集合中与其较为相似的样本的标签（所谓相似，一种情况是指 4 个词中有 1 到 3 个是相同的）；

3. 结合词义进行判断，例如同义词、同类词对应的介词短语可能具有同类的附属；

4. 使用额外的上下文信息，例如句子中的其他内容，文档的类型（genre），等等。

Collins 和 Brooks 的利用 Backoff（后退）的方法利用了上述的思路 2。每个介词短语最多通过 5 步完成附属类型的判断，具体描述为图 5.4.1（图中第一句话似乎应当改为 next formula>0，而不是 is 0）。

If the denominator of the next formula is 0, then use the classification for the 4-tuple

$$\hat{p}_4(H|v,n_1,p,n_2) = \frac{f(H,v,n_1,p,n_2)}{f(v,n_1,p,n_2)}$$

Else

If the denominator of the next formula > 0, then use the following estimate:

$$\hat{p}_3(H|v,n_1,p,n_2) = \frac{f(H,v,n_1,p) + f(H,v,p,n_2) + f(H,n_1,p,n_2)}{f(v,n_1,p) + f(v,p,n_2) + f(n_1,p,n_2)}$$

Else

If the denominator of the next formula > 0, then use the following estimate:

$$\hat{p}_2(H|v,n_1,p,n_2) = \frac{f(H,v,p) + f(H,p,n_2) + f(H,n_1,p)}{f(v,p) + f(p,n_2) + f(n_1,p)}$$

Else

If the denominator of the next formula > 0, then use the following estimate:

$$\hat{p}_1(H|v,n_1,p,n_2) = \frac{f(H,p)}{f(p)}$$

Else label as "low" (default):

$$\hat{p}_0(H|v,n_1,p,n_2) = 0$$

图 5.4.1

其中，f 表示取频数。注意到该方法总是要求两样本具有同样的介词才认为其相似。

本节的最后，给出了其他方法的相关参考文献和正确率。

5.5 Statistical Parsing

本节的开始介绍了统计语法分析的必要性。正如前面接触到的一些例子，一个句子可能对应多种可能的分析方式，而其中一些更为合理。因此我们希望对这些可能的分析进行排序，而根据概率作为标准显然是很好地选择（We need to have some way to rank them, based on the score. Since we're using the score, why not just use probabilities. And that turns out to be the best approach）。

首先介绍概率上下文无关文法（Probabilistic Context-free Grammars）。与 deterministic CFG 相似，它也是一个四元组(N, Σ,R,S)，其中转移规则赋予了概率的意义：

- N: non-terminal symbols
- Σ: terminal symbols (disjoint from N)
- R: rules (A→ β) [p]， β is a string from (Σ ∪ N)*， p is the probability P(β |A)
- S: start symbol (from N)

需要注意左侧相同的所有生成式的概率求和为 1（The rules that have the same left hand side, should have their probabilities add up to one）。

因此，任何一个语法分析树 t 的概率为：

$p(t)=\prod_{i}^{n} p(\alpha_i \to \beta_i)$，其中 $\alpha_i \to \beta_i, 1 \le i \le n$ 是该树涉及到的每个转移规则。

一条语句 s 对应的最有可能的分析树为：

$\arg\max_{t \in T(s)} p(t)$，其中 $T(s)$ 是语句 s 对应的所有可能分析树集合（all the parse trees that correspond to s）。

此外，一条语句的概率是 $\sum_{t \in T(s)} p(t)$，即对所有可能分析树的概率求和。

应当指出，当比较同一语句的不同分析树的概率时，因为涉及到的多数转移规则都是相同的，且概率公式为连乘形式，所以多数概率因子可以抵消，常常只需比较少量不同的概率因子。

基于概率上下文无关文法，之前介绍的 Earley 算法和 CKY 算法可以继续使用。不同之处在于，可以结合转移规则的概率来选择更合理的分析结果。当然，也可以算法的中间过程中就计算相应的概率（而不是最后统一计算），对于 CKY 的一个例子是，若语法中包含有下列规则（和其他规则）：

NP -> DT N [p1=.8]

DT -> 'the' [p7=.75]

N -> 'child' [p8=.5]

则 the child 在 CKY 算法中的一步计算如图 5.5.1：

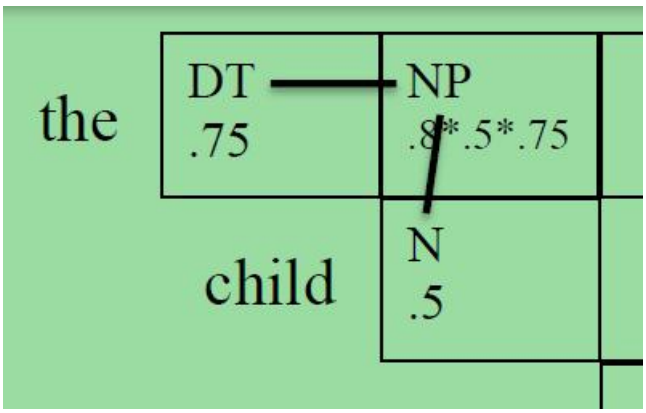

图 5.5.1

标注数据集可用的情况下，可以使用最大似然方法估计转移规则对应的概率：

$P_{ML}(\alpha \to \beta) = Count(\alpha \to \beta)/Count(\alpha)$

还应注意，分析树的概率单独的价值可能是有限的，但它可以与其他处理过程结合，进而应用于自动对话、机器翻译等应用。

5.6 Lexicalized parsing

之所以引入带词汇的语法分析（Lexicalized parsing），是为了改进 PCFG 的局限性。主要的两个局限性是：

1. 转移规则的概率与词汇无关（The probabilities don't depend on the specific words），这是不合理的。例如 give someone something (2 arguments) 与 see something (1 argument)；

2. 无法根据语义来解决分析的歧义（It is not possible to disambiguate sentences based on semantic information）。例如 eat pizza with pepperoni/fork，两种情况下根据 pepperoni 与 fork 词义的不同，介词短语的依附类型明显不同，但 PCFG 无法利用该语义信息。

因此，考虑使用短语的关键词（the head of a phrase）作为附加的信息，形如 VP[ate] -> V[ate]。带词汇的语法分析树的一个例子如图 5.6.1，其中标记的每个黑色单词都是对应短语的关键词。

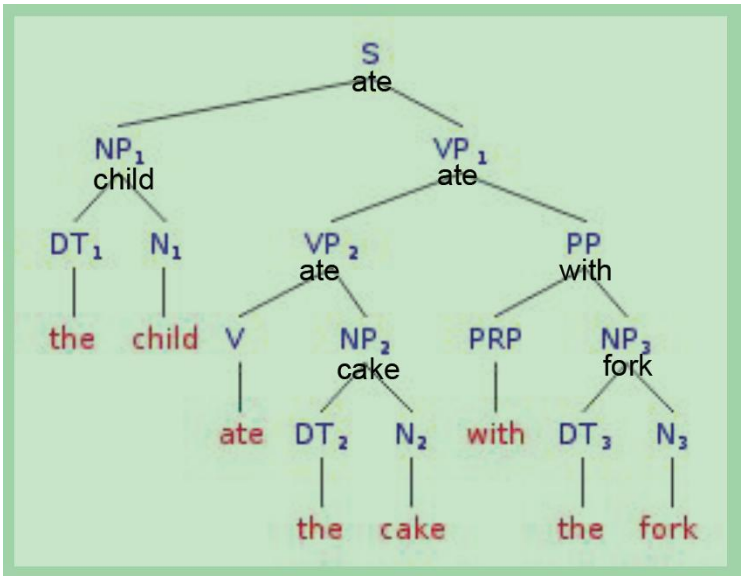

图 5.6.1

Collins Parser (1999)是一个经典的生成式带词汇的语法分析模型。它的生成规则是：

LHS → $L_nL_{n-1}\cdots L_1H\ R_1\cdots R_{m-1}R_m$

其中 H 是关键词，它首先被生成。然后从右到左（1 to n）依次生成 L，最后从左到右（1 to m）依次生成 R。

类似于经典的 PCFG，可以用最大似然方法估计相关的条件概率。例如，当动词短语关

键词 think 的转移规则为 VP[think]->VB 时，估计此时右侧 of 的转移规则为 PP[of]->IN 的概率，可以用下式：

P_{ML} (PPof-IN | VPthink-VB) =

Count (PPof-IN right of the head VPthink-VB) /

Count (symbols right of the head VPthink-VB)

显然，实际操作中的关键问题是数据的稀疏性，一般不会有足够的样本支持我们估计如此细致的条件概率参数，一种解决方案是利用平滑的方式进行粗略的估计：

smoothedP (PPof-IN | VPthink-VB) = λ_1 P (PPof-IN | VPthink-VB) +

+ λ_2 P (PPof-IN | VP-VB) + (1- λ_1- λ_2) P (PPof-IN | VP))

另一个困难是组合爆炸问题，因此我们可能进行额外的处理过程，例如先将词汇归类，对类别的组合估计参数。

用上述方法得到的分析器，对句子得到的多个分析树可能对应相近的概率，不具有判别力。因此有判别力的重排序（Discriminative Reranking）是必要的（例如判别树、或训练重排序分类器等），对每个分析树我们可以考虑其他参考指标：

1. 树的深度（parse tree depth），我们可能不想要太深或太浅的树；

2. 附属类型（left attachment vs. right attachment），英语中右附属居多，因此含有大量左附属的分析树可能不是我们想要的；

3. 叙述结构（discourse structure），例如如果文章中第二次出现同样的一句话，那么它对应的分析可能和第一次出现时有所不同；

4. 分析一致性（consistency across sentences），即你可能希望某个词汇在不同句子中的分析方式是相同的（interpreted exactly the same way）或相似的；

5. 从其他处理过程中获得信息（take into account some information that comes from some of the other stages of the natural language pipeline），例如词性标注等。

到 2005 年，对于长度小于 40 词的语句的分析正确率，可以达到 92%。

5.7 Dependency Parsing

Radev 先从句子的从属关系结构（dependency structure of sentence）引入。例如，对短语 blue house 而言，the entire phrase is a type of house, but not a type of blue。因此，blue 称为 modifier, dependent, child, subordinate，house 称为 head, governor, parent, regent。

利用这种方式可以对更为复杂的句子进行分析，这种语法称为从属关系语法（Dependency Grammars），它是一种 a grammar that capture lexical/syntactic dependencies between words，以句子中顶层的谓语（predicate）作为分析树的根节点。它更加易于分析，且更适合词语顺序相对自由（free word order）的语言（如 Russian、Czech、Latin）。

从属关系分析中识别 head 很重要，直观上 head 有以下特点：

- 决定句法类型，determines the syntactic category of the construct；
- 决定语义类型，determines the semantic category of the construct；
- head 是必须的，修饰词是可选的，H is required; M may be skipped；
- 常常与修饰词有固定的位置关系（如名词和形容词之间），fixed linear position of M with respect to H。

当然，也有一些系统的规则用于识别 head，例如 Collins 的论文中给出的一组给则，如图 5.7.1。注意第二列的 direction 指的是当有多个同类型的候选词时，优先选择靠左还是靠右的。

Parent Non-terminal	Direction	Priority List
ADJP	Left	NNS QP NN $ ADVP JJ VBN VBG ADJP JJR NP JJS DT FW RBR RBS SBAR RB
ADVP	Right	RB RBR RBS FW ADVP TO CD JJR JJ IN NP JJS NN
CONJP	Right	CC RB IN
FRAG	Right	
INTJ	Left	
LST	Right	LS :
NAC	Left	NN NNS NNP NNPS NP NAC EX $ CD QP PRP VBG JJ JJS JJR ADJP FW
PP	Right	IN TO VBG VBN RP FW
PRN	Left	
PRT	Right	RP
QP	Left	$ IN NNS NN JJ RB DT CD NCD QP JJR JJS
RRC	Right	VP NP ADVP ADJP PP
S	Left	TO IN VP S SBAR ADJP UCP NP
SBAR	Left	WHNP WHPP WHADVP WHADJP IN DT S SQ SINV SBAR FRAG
SBARQ	Left	SQ S SINV SBARQ FRAG
SINV	Left	VBZ VBD VBP VB MD VP S SINV ADJP NP
SQ	Left	VBZ VBD VBP VB MD VP SQ
UCP	Right	
VP	Left	TO VBD VBN MD VBZ VB VBG VBP VP ADJP NN NNS NP
WHADJP	Left	CC WRB JJ ADJP
WHADVP	Right	CC WRB
WHNP	Left	WDT WP WP$ WHADJP WHPP WHNP
WHPP	Right	IN TO FW

Table A.1: The head-rules used by the parser. *Parent* is the non-terminal on the left-hand-side of a rule. *Direction* specifies whether search starts from the left or right end of the rule. *Priority* gives a priority ranking, with priority decreasing when moving down the list.

图 5.7.1

接下来介绍了一些具体的分析方法。第一种是动态规划（dynamic programming），类似于 CKY，复杂度为三次方，Jason Eisner 对此有一篇著名的文章。

第二种方法称为 Constraint-based methods，由 Maruyama, Karlsson 等在 1990 年提出。其求解过程是 NP 完全问题，需要启发式搜索。将句子表达为 constraint graph 再进行分析是一种方式。

第三种方法称为 Deterministic parsing，研究者有 Covington、Joakim Nivre 等。Nivre 的方法类似上下文无关语言分析中的 shift/reduce 方法。Graph-based methods 在这种方式中也有使用。

Projectivity 与 non-projectivity 是从属关系分析中重要的问题。如果冀子中出现了如图 5.7.2 的交叉（cross links）情况，就称为 non-projectivity。

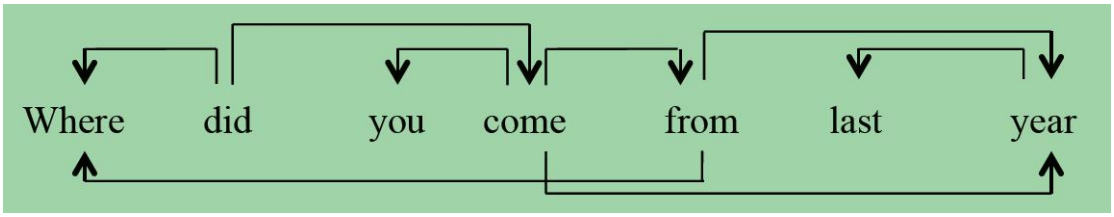

图 5.7.2

这一问题对于 free word order 的语言更为重要，因此对英语而言是不显著的，这也是很多文献主要讨论 projective part。

接下来简要介绍了几个具体的方法。第一个是 McDonald 等 2005 年的方法，将一个句子表达为有向图，节点为单词加上 root 节点，权值根据训练结果得到。利用 MST（maximum spanning tree）的思路求解分析结果，这里指出如果使用 Chu-Liu-Edmonds algorithm 求解则结果是 non-projective tree，即允许 cross links 出现。

第二个是近年来十分著名的方法，即 MaltParser (Joakim Nivre 2008)，且已经有多种变体。这里介绍经典的内容。该方法包括四种操作（action），即 shift, reduce, left-arc, right-arc。句子 People want to be free 的前几步操作如图 5.7.3。

–		[ROOT]	[People, want, to, be, free]	Ø
–	Shift	[ROOT, People]	[want, to, be, free]	
–	LA_{nsubj}	[ROOT]	[want, to, be, free]	A_1 = {nsubj(want, people)}
–	RA_{root}	[ROOT, want]	[to, be, free]	$A_2 = A_1 \cup$ {root(ROOT, want)}

图 5.7.3

该方法的关键在于，训练出一个分类器（classifer）来选择下一步采用何种操作。该方法不涉及搜索问题，因此效率很高。最终 arc 列表可还原出从属关系。

评价分析结果的常用指标是 labeled dependency accuracy，即正确的从属关系数目/全部从属关系数目。分析结果可以表示为图 5.7.4 中的形式，对应句子 Unionized workers are usually better paid than their non-union counterparts 的一种可能的分析结果。

1	Where	Where	WRB	WRB	-	2	SBJ	-	-
2	did	did	VBD	VBD	-	0	ROOT	-	-
3	you	you	PRP	PRP	-	4	SBJ	-	-
4	come	come	VBP	VBP	-	2	VC	-	-
5	from	from	IN	IN	-	4	DIR	-	-
6	last	last	JJ	JJ	-	7	NMOD	-	-
7	year	year	NN	NN	-	5	PMOD	-	-
8	?	?	.	.	-	2	P	-	-

图 5.7.4

介绍了几种方法的复杂度：Projective (CKY)-O(n5)，Projective (Eisner)-O(n3)，Non-projective (MST - Chu-Liu-Edmonds)-O(n2)， Projective (Malt) -O(n)。

下面介绍从属关系分析的应用。第一个是信息提取（Information Extraction）。这里的例子是从医学论文中自动提取相互影响的蛋白质，该过程的示意图为图 5.7.5（左）。5.7.5（右）展示了一种思路，即训练出能够识别这些从属模式的模型。

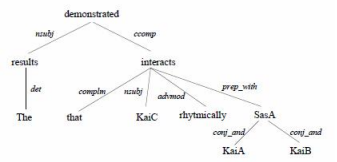

Figure 1: The dependency tree of the sentence "*The results demonstrated that KaiC interacts rhythmically with KaiA, KaiB, and SasA.*"

1. KaiC - nsubj - interacts - prep_with - SasA
2. KaiC - nsubj - interacts - prep_with - SasA - conj_and - KaiA
3. KaiC - nsubj - interacts - prep_with – SasA - conj_and - KaiB
4. SasA - conj_and - KaiA
5. SasA - conj_and - KaiB
6. KaiA – conj_and – SasA - conj_and - KaiB

1. *PROTX1* - nsubj - interacts - prep_with - *PROTX2*
2. *PROTX1* - nsubj - interacts - prep_with - *PROTX0* - conj_and - *PROTX2*
3. *PROTX1* - nsubj - interacts - prep_with – *PROTX0* - conj_and - *PROTX2*
4. *PROTX1* - conj_and - *PROTX2*
5. *PROTX1* - conj_and - *PROTX2*
6. *PROTX1* – conj_and – *PROTX0* - conj_and - *PROTX2*

图 5.7.5

第二个应用称为 Dependency Kernels， it's a technique that decides how similar two sentences are based on how similar their dependency structure are。一个例子是 Bunescu 和 Mooney 在 2005 年给出的识别不同句子中某些关系（relation instance）的方法，例如 A at B 这样的关系。如图 5.7.6 所示。

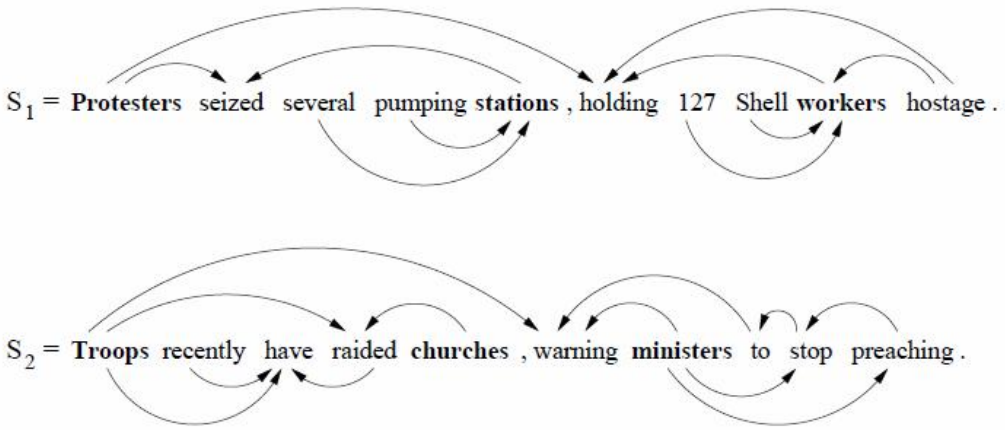

Figure 1: Sentences as dependency graphs.

Relation Instance	Shortest Path in Undirected Dependency Graph
S_1: protesters AT stations	**protesters** ⟶ seized ⟵ **stations**
S_1: workers AT stations	**workers** ⟶ holding ⟵ protesters ⟶ seized ⟵ **stations**
S_2: troops AT churches	**troops** ⟶ raided ⟵ **churches**
S_2: ministers AT churches	**ministers** ⟶ warning ⟵ troops ⟶ raided ⟵ **churches**

Table 1: Shortest Path representation of relations.

图 5.7.6

最后介绍了一些相关链接，包括资源、方法介绍等。

5.8.1 Alternative Syntactic Formalisms

为了介绍的完整性，Radev 简要介绍了几种弱上下文相关语法（Mildly Context-Sensitive Grammars）。它们不如完全上下文相关语法（Full Context-Sensitive Grammars）那样强大，但比标准的上下文无关语法（regular context-free grammar）强大。它们可以处理（capture）一些常见的上下文无关的语句。

首先由树替代语法（TSG, Tree Substitution Grammar）引入，它与上下文无关语法是等价的，但与它相似的 TAG（Tree Adjoining Grammar）则更为强大。组合范畴语法（CCG, Combinatory Categorial Grammar）比 TAG 更强大一些，它涉及到所谓的复杂类型（Complex types），而不仅仅是 CFG 中的那些变元类型。这种类型形如 X/Y 或 X\Y，它的作用是 take an argument of type Y and return an object of type X. 其中 X/Y means that Y should appear on the right. X\Y means that Y should appear on the left. 例如，sleep 的类型可以是 S\NP，说明若 sleep

左侧出现名词（如 He）则返回 S，即形成一条语句。

5.8.2 Semantic parsing

十分简要地介绍了基于语义的句法分析（semantic parsing，相对于 syntactic parsing），并指出这将是近几年研究的热门。这一研究涉及问答生成（question-answering generation）、自动概括（summarization）等应用。

这种思路的出发点很简单，every who in their grammar is going to be associated not just with a syntactic structure, but also with what is known as compositional semantics。在组合语义（in compositional semantics）中，一个单词的含义是它本身（the meaning of a word is the word itself），然后单词组合的含义是描述如何整合单词含义的公式（and then the meaning of a combination of words is a formula that tells you how to combine the meanings of the individual words）。

一种分析方式的例子如图 5.7.1。这里，Javier 和 pizza 的含义是其本身，而 eat 则表示一个逻辑公式（as a verb, is represented as the logical formula which corresponds to a longer function of two arguments, where the predicate is eat and x is the entity doing the eating, and y is the entity receiving the eating）。后续分析过程中，该公式中的两个参数将会被实际单词替换（substitution）。

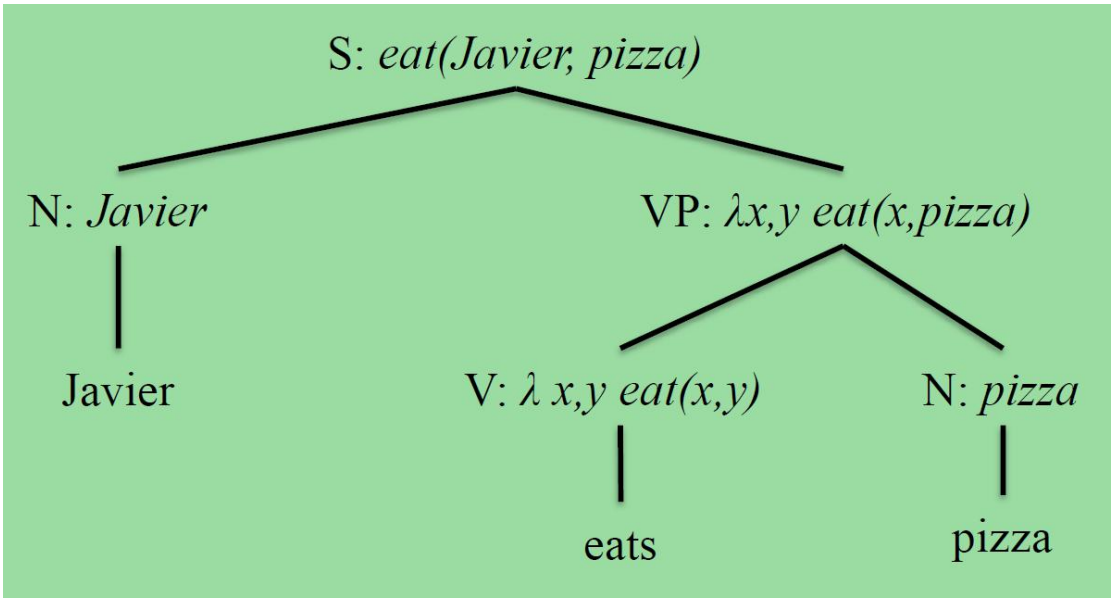

图 5.7.1

最后，推荐了 NACLO 中几个语法分析相关的问题。