第三章

上下文无关文法与下推自动机

Context Free Grammar (CFG)
and
Push Down Automaton (PDA)

上下文无关文法（**CFG**）在程序设计语言和编译原理中有着重要的应用，因为上下文无关文法可以用来阐述绝大多数的程序设计语言的句法结构。此外上下文无关语言也可以作为描述语言翻译方案的基础。(定义回顾, 语法分析)

本章重点讨论:

CFG的简化

CFG的两种范式

下推自动机（**PDA**）的概念

PDA与**CFG**之间的等价转换

上下文无关语言运算的封闭性

以及**CFL**的有关判定问题。

3.1 上下文无关文法的派生树(推导树)

一个上下文无关文法中的一个句型的派生过程可以用一棵树来描述。

【例3-1.1】 给定文法**G=({S,A},{a,b},P,S)**，其中 **P:**
S→aAS|a, A→SbA|ba|SS。句型**aabbaa**的派生过程就可以用一棵树来描述，如图**3-1.1**所示。将此树的叶结点符号从左到右读取下来构成的符号串就是**aabbaa**。

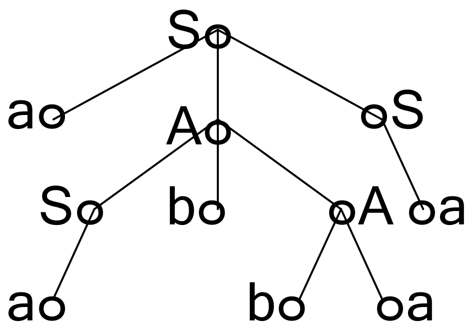


图3-.11 aabbaa的派生树

1．派生树的定义

设文法 **G =(**V_N**,**V_T**,**P **,S)**是上下文无关文法**,**

如果一棵有序树满足下面四个条件，则它是棵派生树**:**

(1) 它的每个结点标记的符号是**(**$V_N \cup V_T \cup$**{** ε **})** 中的符号**;**

(2) 根结点标记开始变元**S;**

(3) 内结点标记的符号是变元，即是V_N中的符号。

(4) 如果一个内结点标记为**A**，且$X_1,X_2,...,X_k$是**A**的从左到右的所有子结点**,** 则$A \rightarrow X_1X_2...X_k$是**P**中一个产生式。

(5) 如果一个结点标记符号是 ε ，则它是其父结点的唯一儿子结点。

其中第(5)条是为了防止下面情况发生：如产生式A→a(a是个终极符)被误认为是A→aε 或A→εa，而在派生树中被画成如图3-2形式。

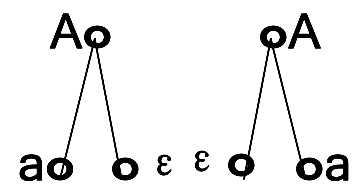


图 3-1.2

2．派生树的结果

设T是棵派生树，将此树的叶结点符号从左到右依次读取下来构成的符号串就是此派生树的结果。

例如，图3-1.1派生树的结果就是aabbaa 。

3．派生树与句型的派生关系

设G =(V_N, V_T, P ,S)是CFG，如果G中有派生$S \Rightarrow^* \alpha$，则在G中必有一棵以α为结果的派生树。反之，如果G中有一棵以α为结果的派生树，则G中也必有派生$S \Rightarrow^* \alpha$。可以说派生与派生树是一一对应的。

4. 最左派生与最右派生

所谓最左派生，就是在一个派生的每一步都是对句型中最左边的变元进行替换。

例如，例**3-1**中aabbaa的派生：

$S \Rightarrow aAS \Rightarrow aSbAS \Rightarrow aabAS \Rightarrow aabbaS \Rightarrow aabbaa$ ，

此派生是最左派生。

所谓最右派生，就是在一个派生的每一步都是对句型中最右边的变元进行替换。

$S \Rightarrow aAS \Rightarrow aAa \Rightarrow aSbAa \Rightarrow aSbbaa \Rightarrow aabbaa$,

此派生是最右派生。

5. 上下文无关文法的二义性

设**G**是个**CFG**，如果它的某个句子有两棵不同构的派生树，则称**G**是二义性的上下文无关文法。

【例3-1.2】 给定CFG G=({S},{a,b},P,S)，其中P：

S→aSbS|bSaS| ε 。

句子abab的两棵不同构的派生树，如图3-1.3所示。

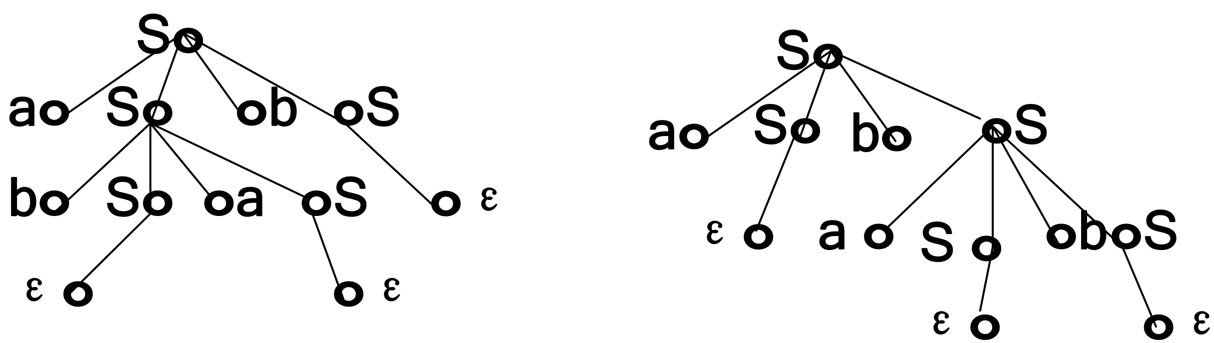


图3-1.3 abab的两棵不同构的派生树

这说明此CFG G是有二义性的。

附加一例:蒋宗礼PPT P537.

3.2 上下文无关文法的简化

一个上下文无关文法有时可以去掉一些符号，或者去掉一些产生式以后，仍然和原来的文法等价，这就是所谓文法的简化。

这里简化文法主要是指：去掉无用符号、去掉ε产生式和去掉单一产生式。

1．去掉无用符号

定义：给定**CFG G=(**V_N**,**V_T**,**P**,S)**，如果在**G**中存在派生**S** $\Rightarrow^* \alpha$ **X** $\beta \Rightarrow^*$**w**，其中**w**$\in V_T^*$，**X**$\in V_N \cup V_T$，则称符号**X**是有用的，否则**X**是无用的。

简单地说，无用符号就是**G**中对任何**w**$\in$**L(G)**的派生中都不会出现的符号。

【例3-2.1】给定文法G=({S,A,B,C},{a,b},P,S)，其中P：

S→AB|a, A→BC|a,C→b 。

G中有派生：

$$S \Rightarrow AB \Rightarrow \left\langle \begin{array}{l} aB \\ BCB \Rightarrow BbB \end{array} \right.$$

可见再往下就无法推导了，因而由**S**只能推出**a**，不能推出其他符号串。所以此文法中，**A**、**B**、**C**、**b**都是无用的符号，只有**S**和**a**是有用符号。

如何去掉无用符号？分两步走，使用两个引理，就可以做到这一点。下面介绍这两个引理。

引理3-2.1 给定CFG $G=(V_N, V_T, P, S)$，且$L(G)\neq\Phi$，可以找到一个与G等价的CFG $G'=(V_N', V_T, P', S)$，使得每个$A\in V_N'$，都有$w\in V_T^*$，且在G′中有$A\Rightarrow^* w$。

证明：1）求V_N'的算法：

begin

(1) $OLDV_N := \Phi$

(2) $NEWV_N := \{A \mid A\rightarrow w\in P$ 且$w\in V_T^*\}$

(3) **While** $OLDV_N \neq NEWV_N$ **do**

  begin

(4) $OLDV_N := NEWV_N$

(5) $NEWV_N := OLDV_N \cup \{A \mid A\rightarrow \alpha \in P$, 且 $\alpha \in (V_T \cup OLDV_N)^*\}$

  end

(6) $V_N' := NEWV_N$,

end

P’：由P中只含有$(V_{N'} \cup V_T)$的符号的产生式构成.

下面证明此算法的有效性。

显然对任何变元$A \in NEWV_N$,不论A是在第⑵步还是在第⑸步加入到$NEWV_N$中的,都有派生$A \Rightarrow^* w$,其中$w \in V_T^*$。

只证明G中任何派生$A \Rightarrow^* w$，$w \in V_T^*$,必有$A \in NEWV_N$。

(对派生的步数归纳证明)

a)若此派生是一步完成的，即有$A \Rightarrow w$，则说明P中有产生式$A \rightarrow w$，于是A在算法的第⑵步被添加到$NEWV_N$中。

b)假设G中派生$A \Rightarrow^* w$是少于k步完成的,则$A \in NEWV_N$。

c)当G中有k步派生$A \Rightarrow X_1X_2 \ldots X_n \Rightarrow^{k-1} w$，不妨设$w=w_1w_2 \ldots w_n$，其中$X_i \Rightarrow^* w_i$，$(i=1,2,\ldots,n)$，而且由于这些派生的步数少于k步，如果$X_i$是变元，则根据假设b）得$X_i$最终会加入到$NEWV_N$中。在执行算法的第⑷步时$OLDV_N:=NEWV_N$,当最后一个$X_i$加入$OLDV_N$时，在执行算法的第⑸步时，就将A加入到$NEWV_N$中。

这说明此算法是有效的，即凡是可以推出终极符串的变元都会添加到NEW V_N中。

于是，最后得到变元集合V_N’。

2）构造文法G’： G’＝(V_N’, V_T, P’, S)，其中

P’：由P中只含有(V_N’∪V_T)的符号的产生式构成的。

3）下面证明L(G)=L(G’)

a） 显然有L(G’)⊆L(G)，因为V_N’⊆V_N，P’⊆P，所以G’中任何派生$S\Rightarrow^{*}w$，在G中也有$S\Rightarrow^{*}w$。所以L(G’)⊆L(G)。

b） 证明L(G)⊆L(G’)，(反证法) 任取w∈L(G)，假设w∉L(G’),则说明在G中w的派生$S\Rightarrow^{*}w$中必用到P－P’中的产生式，即用到了V_N－V_N’中的变元，而这些变元又能推出终极符串，这与上面证明的此算法有效矛盾。所以必有w∈L(G’),从而L(G)⊆L(G’)。

最后得L(G)=L(G’)。

【例3-2.2】 给定CFG G=({S,A,B,C},{a,b},P,S)，其中

P：S→A|B, A→aB|bS|b, B→AB|Ba, C→AS|b

求一个与之等价的文法G′，使得G′中的每个变元都可以推出终极符串。

解：对G应用引理3-2.1，执行上述算法，得到的结果如表3-2.1所示。

循环次数i	初值	1	2	3
OLD V_N	Φ	{A,C}	{A,C,S}	
NEW V_N	{A,C}	{A,C,S}	{A,C,S}	

当算法执行第三次循环时，判定OLD V_N=NEW V_N, 算法终止。最后得G′CFG G'=({S,A,C},{a,b},P',S),

其中 P'：S→A, A→bS|b, C→AS|b

实际上，只去掉了不能推出终极符串的变元B。

引理3-2.2 给定CFG $G=(V_N,V_T,P,S)$， 可以找到一个与G等价的CFG G'， $G'=(V_N',V_T',P',S)$，使得每个$X\in(V_N'\cup V_T')$，都有$\alpha,\beta\in(V_N'\cup V_T')^*$，且在$G'$中有派生$S\Rightarrow^*\alpha X\beta$。

证明：1． 执行下面迭代算法求V_N'和V_T'。

1）置初值： $V_N':=\{S\}$， $V_T':=\Phi$;

2）如果$A\in V_N'$，在P中又有产生式

$A\rightarrow\alpha_1|\alpha_2|\ldots|\alpha_m$，

则可以将$\alpha_1,\alpha_2,\ldots,\alpha_m$中的所有变元加到$V_N'$中，将$\alpha_1,\alpha_2,\ldots,\alpha_m$中的所有终极符加到中$V_T'$中。重复2)。

3）若没有新的符号可加入到V_N'、V_T'中，算法停止。最后得到V_N'、V_T'。

2．构造P'：是由P中只含有($V_N' \cup V_T'$)中的符号的产生式构成的。

3．证明L(G)=L(G')

a） 显然有L(G')⊆L(G)，因为$V_N' \subseteq V_N$， $V_T' \subseteq V_T$，P'⊆P，所以G'中任何派生$S \Rightarrow^* w$，在G中也有$S \Rightarrow^* w$。所以L(G')⊆L(G)。

b） 证明L(G)⊆L(G')，任取w∈L(G)，不妨设w在G中的派生为$S \Rightarrow^* \alpha X \beta \Rightarrow^* w$，其中$\alpha, \beta \in (V_N \cup V_T)^*$，由上述算法可知，在此派生中出现的所有符号，都不会因为对G使用此引理而被去掉，所以这些符号必在$V_N' \cup V_T'$中，此派生中所用到的产生式也在P'中，所以这个派生在G'中也可以实现，因而必有w∈L(G')。故L(G)⊆L(G')。

最后得L(G)=L(G')。

定理3-2.1 设L是一个非空的上下文无关语言，则L可由一个不含无用符号的上下文无关文法产生。

证明：设$G=(V_N, V_T, P, S)$是个CFG，且$L(G)=L\neq\Phi$。先对G用引理3-2.1处理后，得$G'=(V_N', V_T, P', S)$，再将G' 用引理3-2.2处理得$G''=(V_N'', V_T'', P'', S)$，由两个引理得$L(G'')=L(G)$。下面证明G''中不含无用符号。

假设G''中有无用符号Y。根据引理3-2.2得，在G''中必存在派生$S\Rightarrow^{*}\alpha Y\beta$，其中$\alpha,\beta\in(V_N''\cup V_T'')^{*}$，因为G''的符号也都是G'中的符号，所以此派生在G'中也可以实现，又根据引理3-2.1得，α和β中的变元以及Y都可以推出终极符串，于是G'中有派生：

$S\Rightarrow^{*}\alpha Y\beta\Rightarrow^{*}w$，$w\in V_T^{*}$，又因为派生$\alpha Y\beta\Rightarrow^{*}w$中的符号不会因为对G'用引理3-2.2而被去掉，所以在G''中也会实现派生$\alpha Y\beta\Rightarrow^{*}w$，于是G''中也有派生

$S\Rightarrow^{*}\alpha Y\beta\Rightarrow^{*}w$，这与符号Y是无用符号矛盾。所以G''中不含无用符号。

值得注意的是，去掉G中无用符号时，一定要先用引理3-2.1，后用引理3-2.2。应用引理的次序不可颠倒，否则可能遗漏一些无用符号。请看下面例子。

【例3-2.3】 给定CFG G=({S,A,B},{a,b},P,S)，其中

P：S→AB|a, A→a

求一个与之等价的文法G″，使得G″中不含无用符号。

解：先对G应用引理3-2.1方法处理，执行此算法得到的结果如表3-2.2所示。

循环次数i	初值	1	2	3
OLD V_N	Φ	{S,A}		
NEW V_N	{S,A}	{S,A }		

当算法执行第二次循环时，判定**OLD** V_N ＝**NEW** V_N，算法终止。

最后得**G′**：**CFG G′=({S,A},{a,b},P,S)**，

其中 **P′**：**S→a, A→a** 。

再对**G’**用引理**3-2.2**处理，执行算法的结果如表**3-2.3**所示：

循环次数i	初值	1	2	3
V_N’’	{S}	{S}		
V_T’’	Φ	{ a }		

最后得文法**G’’**＝**({S},{a},P’’,S)**，其中**P’’={S→a}**。

但是，如果先对**G**用引理**3-2.2**，后用引理**3-2.1**就得到如下结果：

对G用引理3-2.2执行算法的结果，如表3-2.4所示：

循环次数i	初值	1	2	3
V_N’	{S}	{S,A,B}		
V_T’	Φ	{ a }		

得文法 **G’=({S,A,B},{a},P’,S)，P’：S→AB|a, A→a。**

再对**G′**用引理**3-2.1**执行算法的结果如表**3-2.5**所示：

循环次数i	初值	1	2	3
OLD V_N’’	Φ	{S,A}		
NEW V_N’’	{S,A}	{S,A }		

最后得文法 **G’’=({S,A},{a},P’’,S)，P’’：S→a, A→a。**

显然，这样做，无用符号**A**没有被去掉。可见去掉文法中无用符号时，使用这两个引理的先后次序是很重要的

(可见应做多次反复检查或者一定先用引理1后用引理2)

2．去掉ε产生式

定义：所谓ε产生式，就是形如A→ε的产生式，其中A为变元。

给定CFG G，如果ε∉L(G)，则G中所有ε产生式都可以去掉。如果ε∈L(G)，则除了开始变元S的ε产生式(即S→ε)外，其余ε产生式都可以去掉。原因见定理3-2-2.

为了去掉ε产生式，先定义一个概念——可为零的变元。

定义：设A是个变元，如果$A \Rightarrow^* \varepsilon$，则称A是可为零的。

去掉CFG G中的ε产生式的思路是：

首先，找出G中所有可为零的变元。删除所有$X_1 \rightarrow \varepsilon$产生式.

然后，对P中每个形如$A \rightarrow X_1X_2...X_n$的产生式进行如下处理：要添加一些这样的产生式：这些产生式是通过去掉$X_1X_2...X_n$中某些可为零的变元而得到的。但是，如果所有$X_i(i=1,2,...n)$都是可为零的，则不可全去掉，因为那样会产生新的ε产生式A→ε。

【例3-2.6】有产生式：**S→aSAbB**, 设**A**与**B**都是可为零的，

则由这个产生式变成如下四个产生式：

S→aSAbB,S→aSbB(去掉**A), S→aSAb(**去掉**B), S→aSb(A**和**B**全去掉**)**。注意，要将所有可能的情况均考虑到，才能保证新的文法与原文法等价。

定理3-2.2 给定**CFG G=(**V_N, V_T, P**,S)**，可以找到一个不含无用符号，又无ε产生式的**CFG G′**，使得**L(G′)**＝**L(G)**－**{** ε **}**。

证明：假设**G**已经去掉了无用符号，从**G**中去掉ε产生式后得到文法**G′**，令

G′＝**(**V_N, V_T, P'**,S)**，其中**P′**构成如下：

1. 用下面迭代算法确定G中可为零的变元集合V_0。

begin

(1) **OLD**V_0**:=**Φ

(2) **NEW**V_0**:={A|A→**ε $\in$**P }**

(3) **While OLD**$V_0 \neq$**NEW**V_0 **do**

begin

(4) **OLD**V_0**:=NEW**V_0

(5) **NEW**V_0**:= OLD**$V_0 \cup$**{A|A→**$\alpha \in$**P,**且$\alpha \in$**(OLD**V_0**)*****}**

end

(6) V_0**:=NEW**V_0

end

当**OLD**V_0**=NEW**V_0时，算法终止，最后得到可为零的变元集合V_0。

此算法与引理**3-2.1**中算法相似，类似可证此算法的有效性

2．构造P′：

如果$A \rightarrow X_1X_2...X_n \in P$，则将所有形如$A \rightarrow \alpha_1\alpha_2...\alpha_n$的产生式都加到P′中，其中

(1) 如果X_i不是可为零的，则$\alpha_i = X_i$。

(2) 如果X_i是可为零的，则$\alpha_i = X_i$或者$\alpha_i = \varepsilon$。但是，如果所有$X_i(i=1,2,...n)$都是可为零的，则不可所有$\alpha_i = \varepsilon$。

3．用归纳法证明：对任何$A \in V_N$，任何$w \in V_T^+$，有如果$A \Rightarrow_{G'}^* w$，当且仅当$A \Rightarrow_G^* w$。

1）先证明充分性。设G中有派生$A \Rightarrow_G^* w$，$w \neq \varepsilon$。

(1) 如果此派生是一步完成的，即G中有派生$A \Rightarrow_G w$，则$A \rightarrow w \in P$，因为$w \neq \varepsilon$，所以$A \rightarrow w \in P'$，所以G′中也有派生$A \Rightarrow_{G'}^* w$。

(2) 假设G中派生$A \Rightarrow_G^* w$是少于k步完成的，则G′中有派生$A \Rightarrow_{G'}^* w$。

(3) 当G中有派生$A \Rightarrow_G X_1X_2...X_n \Rightarrow_G^* w = w_1w_2...w_n$是由k步完成的时，其中$X_i \in (V_T \cup V_N)$且有$X_i \Rightarrow_G^* w_i$ (i=1,2,...,n)，

其中有的w_i可能为ε。如果某个w_i＝ε，则对应的X_i是可为零的变元。

由G中派生$A \Rightarrow_G X_1X_2...X_n$得$A \rightarrow X_1X_2...X_n \in P$，根据P′的构成得，必有产生式$A \rightarrow \alpha_1\alpha_2...\alpha_n \in P'$，其中

a）如果$X_i \Rightarrow_G^* w_i \neq \varepsilon$，则$\alpha_i = X_i$，于是有$\alpha_i \Rightarrow_G^* w_i$，且此派生少于k步完成，由假设⑵得G′中有派生

$\alpha_i \Rightarrow_{G'}^* w_i$。

b）如果$X_i \Rightarrow_G^* w_i = \varepsilon$，则$X_i$是可为零的变元，与$X_i$对应

$\alpha_i = \varepsilon$，于是有$\alpha_i \Rightarrow_{G'}^* w_i = \varepsilon$。

最后G′中有派生：

$$A \Rightarrow_{G'} \alpha_1\alpha_2...\alpha_n \Rightarrow_{G'}^* w_1\alpha_2...\alpha_n \Rightarrow_{G'}^* w_1w_2\alpha_3...\alpha_n$$

$$\Rightarrow_{G'}^* w_1w_2...w_n = w,$$

即G′中有派生$A \Rightarrow_{G'}^* w$，充分性成立。

2）再证明必要性。设G′中有派生$A \Rightarrow_{G'}^{*} w$，显然 $w \neq \varepsilon$。

(1). 如果此派生是一步完成的，即G′中有$A \Rightarrow w$，则 $A \to w \in P'$，因为$w \neq \varepsilon$，于是P中有产生式$A \to w$或者 $A \to \alpha$，使得从α中去掉某些可为零的变元后得到w，总之G中有派生$A \Rightarrow_{G} w$，或者$A \Rightarrow_{G} \alpha \Rightarrow_{G}^{*} w$。

(2). 假设G′中有派生$A \Rightarrow_{G'}^{*} w$是少于k步完成的，则G中有派生$A \Rightarrow_{G}^{*} w$。

(3). 当G′中派生$A \Rightarrow_{G'} X_1X_2 \ldots X_n \Rightarrow_{G'}^{*} w_1w_2 \ldots w_n = w$是由k步完成时，此派生的第一步派生是用P′中的产生式 $A \to X_1X_2 \ldots X_n$。根据P′的构成知，P中必存在产生式 $A \to \beta$，使得从β中去掉某些可为零的变元后就得到 $X_1X_2 \ldots X_n$，于是G中有派生：

$A \Rightarrow_{G} \beta \Rightarrow_{G}^{*} X_1X_2 \ldots X_n$,

而在G’的第二步及以后的派生$X_1X_2...X_n \Rightarrow_{G'}^{*} w_1w_2...w_n = w$中，令$X_i \Rightarrow_{G'}^{*} w_i (i=1,2,..,n)$，由于这些派生的步数少于k，由假设⑵得，$X_i \Rightarrow_{G}^{*} w_i$，于是G中也有派生：

$$A \Rightarrow_G \beta \Rightarrow_G^{*} X_1X_2...X_n \Rightarrow_G^{*} w_1X_2...X_n \Rightarrow_G^{*} w_1w_2X_3...X_n$$

$$\Rightarrow_G^{*} w_1w_2...w_n = w。$$

所以必要性成立。

最后得此结论成立。即如果$A \Rightarrow_{G'}^{*} w$，当且仅当 $A \Rightarrow_G^{*} w$ 。

当A=S时，则有$S \Rightarrow_{G'}^{*} w$，当且仅当 $S \Rightarrow_G^{*} w$ 。

于是有L(G′)=L(G)－{ ε }。

3．去掉单一产生式

定义：所谓单一产生式，就是形如$A\rightarrow B$的产生式，其中A和B都是变元。

定理3-2.3 每个不含有ε的上下文无关语言L，都可以由一个不含无用符号、无ε产生式、也无单一产生式的上下文无关文法产生。

证明：令$G=(V_N, V_T, P, S)$是一个不含有无用符号，无ε产生式的上下文无关文法，且$L(G)=L$。构造一个CFG $G'=(V_N, V_T, P', S)$，其中P'的构成如下：

(1) 包含P中所有非单一产生式。

(2) 对任何$A,B\in V_N$，如果有$A\Rightarrow_G^* B$，且$B\rightarrow \alpha_1|\alpha_2|\ldots|\alpha_n$是P中B的所有非单一产生式，则把所有$A\rightarrow \alpha_1|\alpha_2|\ldots|\alpha_n$加到$P'$中。而去掉$A\rightarrow A_2$，$A_2\rightarrow A_3\ (=B)\rightarrow \alpha$。其中，$A\rightarrow A_2$，$A_2\rightarrow A_3$都是单一产生式。

(例如G中有产生式:

$A\to B, B\to C, C\to D, B\to\alpha, C\to\beta, D\to\gamma$，

于是有$A\Rightarrow^{*}B$, $A\Rightarrow^{*}C$, $A\Rightarrow^{*}D$，$B\Rightarrow^{*}C$，$B\Rightarrow^{*}D$, $C\Rightarrow^{*}D$，

则G′中有产生式:

$A\to\alpha\,|\,\beta\,|\,\gamma$，$B\to\alpha\,|\,\beta\,|\,\gamma$，$C\to\beta\,|\,\gamma$，$D\to\gamma$。)

下面证明L(G′)=L(G)

a）首先证明$L(G')\subseteq L(G)$

容易看出任何$A\to\alpha\in P'$，则G中必有派生$A\Rightarrow_G^{*}\alpha$。因为，如果$A\to\alpha\in P$，G中当然有派生$A\Rightarrow_G\alpha$，如果$A\to\alpha\notin P$，则必有变元B，使得G中有派生

$A\Rightarrow_G^{*}B\Rightarrow_G\alpha$。

所以对任何$w\in L(G')$,即$S\Rightarrow_{G'}^{*}w$，此派生中用到的所有产生式$A\to\alpha\in P'$,则G中必有派生$A\Rightarrow_G^{*}\alpha$。所以G中必有派生$S\Rightarrow_G^{*}w$，

所以$w\in L(G)$。因而$L(G')\subseteq L(G)$。

b）再证明$L(G) \subseteq L(G')$

任取$w \in L(G)$，设w在G中的最左派生如下：

$$\mathbf{S} = \alpha_{\mathbf{0}} \Rightarrow \alpha_{\mathbf{1}} \Rightarrow \alpha_{\mathbf{2}} \Rightarrow \alpha_{\mathbf{3}} \Rightarrow \ldots \Rightarrow \alpha_{\mathbf{n-1}} \Rightarrow \alpha_{\mathbf{n}} = \mathbf{w},$$

下面分两种情况讨论：

① 如果上述派生用的都是非单一产生式，则这些产生式在P′中也有，所以这些派生在G′中也可以实现。

② 如果上述派生既用了单一产生式，也用了非单一产生式，不妨取出其中一段：

$$\underbrace{\alpha_{\mathbf{i-1}} \Rightarrow \alpha_{\mathbf{i}}}_{\text{非单一}} \underbrace{\Rightarrow \alpha_{\mathbf{i+1}} \Rightarrow \ldots \Rightarrow \alpha_{\mathbf{j}}}_{\text{单一}} \underbrace{\Rightarrow \alpha_{\mathbf{j+1}}}_{\text{非单一}},$$

设从α_{i-1}到α_i用的是非单一产生式，从$\alpha_{\mathbf{i}}$到$\alpha_{\mathbf{j}}$用的是单一产生式，$\alpha_{\mathbf{j}}$到$\alpha_{\mathbf{j+1}}$用的又是非单一产生式。下面主要考察**G**中从$\alpha_{\mathbf{i}}$到$\alpha_{\mathbf{j+1}}$的派生(此派生是先用单一产生式，后用非单一产生式)，看看这段派生在**G′**中是如何实现的。

先看看从α_i到α_j用的是单一产生式的派生，不妨设

$\alpha_i = yA_i\gamma$，$\alpha_{i+1} = yA_{i+1}\gamma$，… $\alpha_j = yA_j\gamma$，

其中$y \in V_T^*$，$\gamma \in (V_T \cup V_N)^*$

从α_i到α_j的派生写成 $yA_i\gamma \Rightarrow yA_{i+1}\gamma \Rightarrow \dots \Rightarrow yA_j\gamma$，这些派生都是用单一产生式进行的。实际上相当于派生 $A_i \Rightarrow A_{i+1} \Rightarrow \dots \Rightarrow A_j$，所以有$A_i \Rightarrow^* A_j$。

再看看α_j到α_{j+1}的派生$\alpha_j \Rightarrow \alpha_{j+1}$，它用的是非单一产生式，不妨设$\alpha_{j+1} = y\beta\gamma$，$\beta \in (V_T \cup V_N)^*$，于是此派生写成$yA_j\gamma \Rightarrow y\beta\gamma$，实际上此派生用的是非单一产生式$A_j \rightarrow \beta$。于是G中有派生$A_i \Rightarrow^* A_j \Rightarrow \beta$，根据P′的构成，则P′中必有产生式 $A_i \rightarrow \beta$。

于是在G′中有派生：$\alpha_i = yA_i\gamma \Rightarrow y\beta\gamma = \alpha_{j+1}$，即在G′中从$\alpha_i$到$\alpha_{j+1}$的派生是一步完成的。

所以当$S \Rightarrow_G^* w$ 用了两种产生式时，G′中也有派生

$S \Rightarrow_{G'}^* w$。

所以$L(G) \subseteq L(G')$。最后得$L(G) = L(G')$。

作业题

1． 给定CFG G=({S,A,B,C},{a,b,c},P,S)，其中，

P：S→A|B, A→Ab|bS|C|b, B→AB|Ba,

C→AS｜b,

去掉G中的无用符号和单一生成式。

2． 给定CFG G=({S,A,B,C},{a,b},P,S)，其中，

P：S→ABC, A→BB|ε，B→CC|a,

C→AA|b,

去掉G中的ε生成式。

3.3 上下文无关文法的Chomsky范式 (Chomsky Normal Form，CNF)

Chomsky范式形式是：每个产生式的形式要么是$A \to BC$，要么是$D \to a$，其中A，B，C，D是变元，a是终极符。

这种范式在形式语言的理论和应用上都具有重要的意义。

下面介绍如何把一个CFG G写成CNF形式。

定理3-3.1 任何一个不含有ε的CFL L都可由一个具有CNF形式的CFG G产生。

证明：由定理3-2.3可知，对一个不含有ε的CFL L都可由一个不含无用符号、无ε产生式、无单一产生式的CFG G产生。

不妨设CFG $G=(V_N, V_T, P, S)$是一个不含无用符号、无ε产生式、无单一产生式的CFG，且$L(G)=L$。

1. 先对P中产生式做如下处理：

(1)保留右侧只有一个符号的产生式，（此符号是终极符，）

(2) 处理产生式$A \to X_1X_2...X_n$ (n≥2)：

如果其中X_i是终极符a，则引入新的变元Ca，用Ca代替a，同时添加新的产生式$Ca \to a$ 。

经过此步处理后，使得产生式的形式只有两种：一种是$A \to a$，a是终极符；另一种是$A \to B_1B_2...B_n$，其中每个B_i都是变元(包括新引入的变元)。

(3) 处理产生式$A \to B_1B_2...B_n$ (n≥3) (因为n=2时符合要求)，引进新的变元$D_1D_2...D_{n-2}$，此产生式用下面产生式替换：

$A \to B_1D_1$, $D_1 \to B_2D_2$, $D_2 \to B_3D_3$,…, $D_{n-3} \to B_{n-2}D_{n-2}$, $D_{n-2} \to B_{n-1}B_{n-1}$ 。

显然用这n－1个产生式代替$A \to B_1B_2...B_n$，这是个等价变换。经过此步处理后，所有产生式的形式就具有CNF形式了：$A \to BC$, $D \to a$, 其中A, B, C, D是变元，a是终极符。

2. 构造新的CFG $G'=(V_N', V_T, P', S)$，其中V_N'中除了原来V_N中的变元外，还含有新增加的所有变元。P′就是由经过上述处理后得到的具有CNF形式的产生式构成。

3. 容易证明L(G′)=L(G)，这里证明从略。

【例3-3.1】 G是个定义算术表达式的CFG

G=({E,T,F},{a,b,+,*,(,)},P,E),其中E表示算术表达式；T表示项；F表示因子。P含有下面这些产生式：

E→E+T|T　T→T*F|F　F→(E)|a|b

求一个具有**CNF** 形式的**CFG G′,**使得**L(G′)=L(G)**。

解：**1.** 先简化**G，** 因为**G**中不含无用符号，也无ε产生式，所以只去掉单一产生式。

1）去掉**T→F，**添加如下产生式： **T→(E)|a|b**

2）去掉**E→T，**添加如下产生式： **E→T*F|(E)|a|b**

2. 保留符合要求的产生式：**E→a|b, T→a|b, F→a|b**

3．处理产生式$A \to X_1X_2...X_n$ ($n \geq 2$)：终极符X_i变成变元。

$E \to E+T$ 变成 $E \to EC_+T$ $C_+ \to +$

$E \to T*F$ 变成 $E \to TC_*F$ $C_* \to *$

$E \to (E)$ 变成 $E \to C_(EC_)$ $C_(\to ($ $C_) \to)$

$T \to T*F$ 变成 $T \to TC_*F$

$T \to (E)$ 变成 $T \to C_(EC_)$

$F \to (E)$ 变成 $F \to C_(EC_)$

4．处理产生式$A \to B_1B_2...B_n$ (B_i都是变元, $n \geq 3$)：引进新的变元$D_1, D_2, ..., D_{n-2}$。

$E \to EC_+T$ 变成 $E \to ED_1$ $D_1 \to C_+T$

$E \to TC_*F$ 变成 $E \to TD_2$ $D_2 \to C_*F$

$E \to C_(EC_)$ 变成 $E \to C_(D_3$ $D_3 \to EC_)$

$T \to TC_*F$ 变成 $T \to TD_2$

$T \to C_(EC_)$ 变成 $T \to C_(D_3$

$F \to C_(EC_)$ 变成 $F \to C_(D_3$

5．最后得具有CNF形式的CFG $G'=(V_N', V_T, P', E)$，

其中 $V_N' = \{E, T, F, C_+, C_*, C_(, C_), D_1, D_2, D_3\}$

$V_T = \{a, b, +, *, (,)\}$

P'： $E \to ED_1 \mid TD_2 \mid C_(D_3 \mid a \mid b$

$T \to TD_2 \mid C_(D_3 \mid a \mid b$

$F \to C_(D_3 \mid a \mid b$

$D_1 \to C_+ T$

$D_2 \to C_* F$

$D_3 \to EC_)$

$C_+ \to +$

$C_* \to *$

$C_(\to ($

$C_) \to)$

3.4 CFG 的Greibach范式 (GNF)

下面我们在CNF的基础上讨论CFG的另外一种范式－－Greibach范式 (GNF)。

定义：CFG $G=(V_N, V_T, P, S)$，P中产生式形式都是：$A \to a\alpha$，其中a是一个终极符，$a \in V_T$，α是变元符号串，$\alpha \in V_N^*$，则称此文法具有GNF形式。

任何一个不含有ε的CFL，都可以由一个具有GNF形式的CFG产生。那么如何将一个CFG变成具有GNF形式的CFG，应用下面两个引理可以实现。

例子见P48.

引理3-4.1 设$G=(V_N,V_T,P,S)$是个CFG，$A\to\alpha_1B\alpha_2$是P中一个产生式，又$B\to\beta_1|\beta_2|\beta_3\ldots|\beta_m$是P中变元B的所有产生式。又令$G'=(V_N,V_T,P',S)$，其中$P'$构成如下：从P中删除$A\to\alpha_1B\alpha_2$，再添加产生式
$A\to\alpha_1\beta_1\alpha_2|\alpha_1\beta_2\alpha_2|\alpha_1\beta_3\alpha_2|\ldots|\alpha_1\beta_m\alpha_2$，则
$L(G')=L(G)$。

证明：显然有$L(G')\subseteq L(G)$。因为如果
$A\to\alpha_1\beta_i\alpha_2(1\leqslant i\leqslant m)$用在$G'$的一个派生中，虽然在G中没有此产生式，但是在G中有如下派生：
$A\Rightarrow\alpha_1B\alpha_2\Rightarrow\alpha_1\beta_i\alpha_2(1\leqslant i\leqslant m)$。所以$G'$中的任何派生在G中也可以实现。所以$L(G')\subseteq L(G)$。

再证明$L(G) \subseteq L(G')$

这里只需注意到G与G′的差别是：只有$A \to \alpha_1 B \alpha_2$是在G中而不在G′中的产生式，而G中其余的产生式G′中也有。所以只要$A \to \alpha_1 B \alpha_2$用于G的一个派生中，在其后的某一步推导肯定用到产生式$B \to \beta_i$，用以改变变元B，而这二步推导在G′中用一步派生$A \Rightarrow \alpha_1 \beta_i \alpha_2 (1 \leqslant i \leqslant m)$以代之。所以 $L(G) \subseteq L(G')$。

最后得$L(G') = L(G)$。

下面介绍的引理是处理带有左递归的产生式。

这里所谓左递归的产生式形式： $A \to A\alpha$,

显然左递归不去掉，就无法变成GNF形式。因为要求产生式的右侧第一个符号是终极符。当然这不像上面引理那么简单。

应用下面引理解决这个问题。

引理3-4.2 设$G=(V_N, V_T, P, S)$是个CFG，$A\to A\alpha_1|A\alpha_2|\dots|A\alpha_m$是P中变元A的所有带有左递归的产生式。又$A\to\beta_1|\beta_2|\beta_3|\dots|\beta_n$是P中变元A的其余的产生式。构造CFG $G'=(V_N\cup\{Z\}, V_T, P', S)$，其中$P'$构成如下：将P中变元A的产生式用下面产生式替换：

$$A\to\beta_i|\beta_i Z \quad (1\leqslant i\leqslant n)$$

$$Z\to\alpha_j|\alpha_j Z \quad (1\leqslant j\leqslant m)$$

则$L(G')=L(G)$。

证明：此引理的处理思路，就是用变元Z的右递归代替变元A的左递归。

任取$x\in L(G)$,考虑x的最左派生，

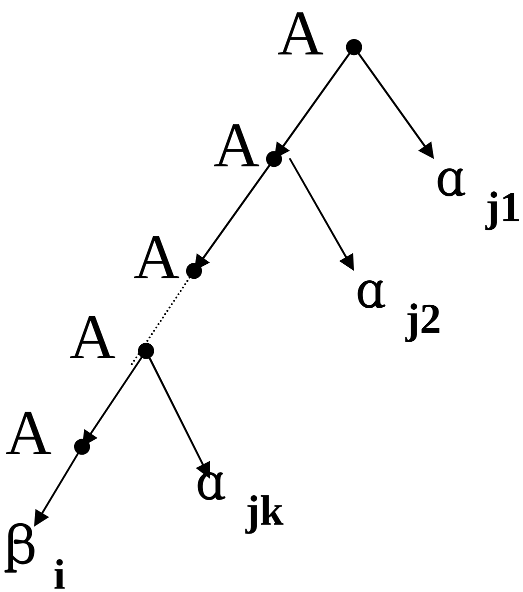
1.如果此派生只使用$A\to\beta_i$型的产生式，则G'中也有这种产生式，所以x的派生在G'中也可以实现。

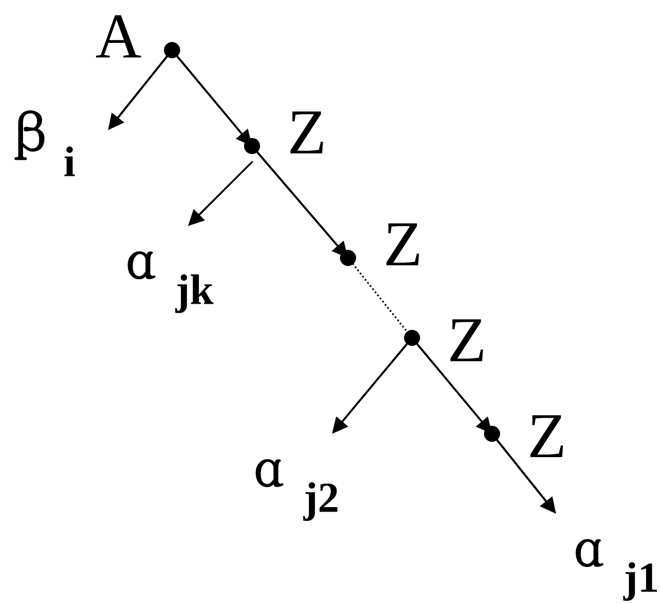
2.如果此派生先用$A\to A\alpha_j$，最后必用$A\to\beta_i$，才可以消去变元A，例如有如下派生：

$A \Rightarrow A\alpha_{j1} \Rightarrow A\alpha_{j2}\alpha_{j1} \Rightarrow \dots \Rightarrow A\alpha_{jk}\alpha_{jk-1}\cdots\alpha_{j2}\alpha_{j1}$
$\Rightarrow \beta_i\alpha_{jk}\alpha_{jk-1}\cdots\alpha_{j2}\alpha_{j1}$

这个派生在G′中也可以实现，如下所示：

$A \Rightarrow \beta_i Z \Rightarrow \beta_i\alpha_{jk}Z \Rightarrow \dots \Rightarrow \beta_i\alpha_{jk}\alpha_{jk-1}\cdots\alpha_{j2}Z$
$\Rightarrow \beta_i\alpha_{jk}\alpha_{jk-1}\cdots\alpha_{j2}\alpha_{j1}$





所以$x \in L(G')$。

反之$x \in L(G')$，x在G'中的派生，类似可证在G中也可以实现，所以$x \in L(G)$。

最后得$L(G')=L(G)$。

定理3-4.1(Greibach定理) 每个不含有ε的CFL L，都可以由一个具有GNF形式的CFG产生。

证明：令$G=(V_N, V_T, P, A_1)$是个CFG，$L(G)=L$，$\varepsilon \notin L(G)$，假设G已经具有CNF形式。不妨设$V_N=\{A_1,A_2,A_3,\ldots,A_n\}$，按照如下方法处理P中产生式：

1. 先暂时保留形如$A_i \to A_jA_k$的产生式，其中$i<j$。
2. 用引理3-4.1处理所有形如$A_i \to A_jA_k$的产生式，其中$i>j$，即用所有A_j产生式的右侧符号串代替该产生式中的A_j。
3. 用引理3-4.2处理带有左递归的产生式$A_k \to A_k\alpha_j \mid \beta_i$，$(1 \leqslant j \leqslant m)$ $(1 \leqslant i \leqslant n)$，用下面产生式替换：$A \to \beta_i \mid \beta_i Z_k$，$Z_k \to \alpha_j \mid \alpha_j Z_k$。
4. 经过上述处理后，再依次将$A_n,A_{n-1},\ldots,A_2,A_1$产生式以及所有新增加的变元$Z_k$ $(1 \leqslant k \leqslant n)$的产生式用引理3-4.1方法处理，使之都变成GNF形式。

由于我们对G的产生式的处理都是使用上面两个引理，所以最后得到的具有GNF形式的文法G′必与G等价，即L(G′)=L(G)。

下面我们通过一个例子说明如何将G变成具有GNF形式的文法G′的过程。

【例3-4.1】. 令G=({A_1,A_2,A_3},{a,b},P,A_1)，其中

P：(1)$A_1 \to A_2A_3$, (2)$A_2 \to A_3A_1$, (3) $A_2 \to b$, (4) $A_3 \to A_1A_2$,
(5)$A_3 \to a$,

将G写成GNF形式。

解：1. 先暂时保留产生式(1)、(2)。(3)、（5）保留.

2．处理产生式(4) $A_3 \to A_1A_2$：用(1)的A_1产生式右侧的A_2A_3替换(4)中A_1，得新产生式(6)$A_3 \to A_2A_3A_2$。(6)仍然不满足要求，再次对(6)处理：用(2)(3)的A_2产生式右侧符号串分别替换A_2，得(7)$A_3 \to A_3A_1A_3A_2$，(8)$A_3 \to bA_3A_2$。这两个产生式符合要求。

3.处理有左递归的产生式(7) $A_3 \to A_3A_1A_3A_2$, (8)$A_3 \to bA_3A_2$,

(5)$A_3 \to a$。用引理3-4.2得：$A_3 \to bA_3A_2|a|bA_3A_2Z_3|aZ_3$，

$Z_3 \to A_1A_3A_2|\ A_1A_3A_2Z_3$。

可以看出，到此所有A_3的产生式都已经符合GNF形式了。

4. 用引理3-4.1 将A_2、A_1产生式变成GNF 形式。

对(2)$A_2 \to A_3A_1$, (3) $A_2 \to b$, 应用所有A_3产生式的右侧符号串替换A_3得：$A_2 \to bA_3A_2A_1|aA_1|bA_3A_2Z_3A_1|aZ_3A_1|b$,

到此A_2已经满足GNF形式了。

对(1)$A_1 \to A_2A_3$, 应用所有A_2产生式的右侧符号串替换A_2得:

$A_1 \to bA_3A_2A_1A_3|aA_1A_3|bA_3A_2Z_3A_1A_3|aZ_3A_1A_3|bA_3$

到此A_1已经满足GNF形式了。

5．将所有Z_3产生式变成GNF形式，用所有A_1产生式得：

$Z_3 \to A_1A_3A_2$ 变成

$Z_3 \to bA_3A_2A_1A_3A_3A_2 | aA_1A_3A_3A_2 | bA_3A_2Z_3A_1A_3A_3A_2$

$| aZ_3A_1A_3A_3A_2 | bA_3A_3A_2$

$Z_3 \to A_1A_3A_2Z_3$ 变成

$Z_3 \to bA_3A_2A_1A_3A_3A_2Z_3 | aA_1A_3A_3A_2Z_3$

$| bA_3A_2Z_3A_1A_3A_3A_2Z_3 | aZ_3A_1A_3A_3A_2\ Z_3 | bA_3A_3A_2Z_3$

最后得文法$G'=(\{A_1,A_2,A_3,Z_3\},\{a,b\},P',A_1)$，其中$P'$：

$A_1 \to bA_3A_2A_1A_3 | aA_1A_3 | bA_3A_2Z_3A_1A_3 | aZ_3A_1A_3 | bA_3$

$A_2 \to bA_3A_2A_1 | aA_1 | bA_3A_2Z_3A_1 | aZ_3A_1 | b$

$A_3 \to bA_3A_2 | a | bA_3A_2Z_3 | aZ_3$

$Z_3 \to bA_3A_2A_1A_3A_3A_2 | aA_1A_3A_3A_2 | bA_3A_2Z_3A_1A_3A_3A_2$
$\quad | aZ_3A_1A_3A_3A_2 | bA_3A_3A_2$

$Z_3 \to bA_3A_2A_1A_3A_3A_2Z_3 | aA_1A_3A_3A_2Z_3$
$\quad | bA_3A_2Z_3A_1A_3A_3A_2Z_3 | aZ_3A_1A_3A_3A_2\ Z_3 | bA_3A_3A_2Z_3$

可见G'具有GNF形式。

顺便说明一下，上边定理中介绍的方法是个将具有CNF形式的文法变成GNF形式的一般的方法，有时不一定严格按照此法去做，即不一定都得先将给定CFG G变成CNF形式后，再写成GNF形式。

【例3-4.2】将下面CFG G写成GNF形式。

$G=(\{S\},\{0,1\},P,S)$，其中 P：$S\rightarrow 0S0|1S1|00|11$。

解：因为此文法所有产生式的右侧的第一个符号已经是终极符，所以问题就简单了，就不必先将G变成CNF形式，可以直接写成GNF形式。令

$G'=(\{S,A,B\},\{0,1\},P',S)$，

P'：$S\rightarrow 0SA|1SB|0A|1B,\ A\rightarrow 0,\ B\rightarrow 1$

G'就是与G等价的具有GNF 形式的文法。

容易看出$L(G)=\{ww^R|w\in\{0,1\}^+\}$

下面用例子验证它们是等价的。

例如，看看它们派生00100100的过程。

在G中的派生：$S\Rightarrow 0S0\Rightarrow 00S00\Rightarrow 001S100\Rightarrow 00100100$

在G'中的派生（最左派生）：

$S\Rightarrow 0SA\Rightarrow 00SAA\Rightarrow 001SBAA\Rightarrow 0010ABAA\Rightarrow 00100BAA\Rightarrow 001001AA\Rightarrow 0010010A\Rightarrow 00100100$

作业题：

1．给定CFG G=({S,A },{0,1}, P, S)，其中，

P：S→AA｜0, A→SS｜1,

将G写成GNF形式。

3.5 下推自动机 (Push Down Automaton, PDA)

下面介绍识别CFL的自动机——下推自动机。

1. PDA的结构

输入带：

带上存放被识别的符号串。

有限控制器：

存放PDA的状态转移函数。

下推栈：

后进先出的下推栈。

两个只读头：

分别用来读取　带上符号和栈内符号

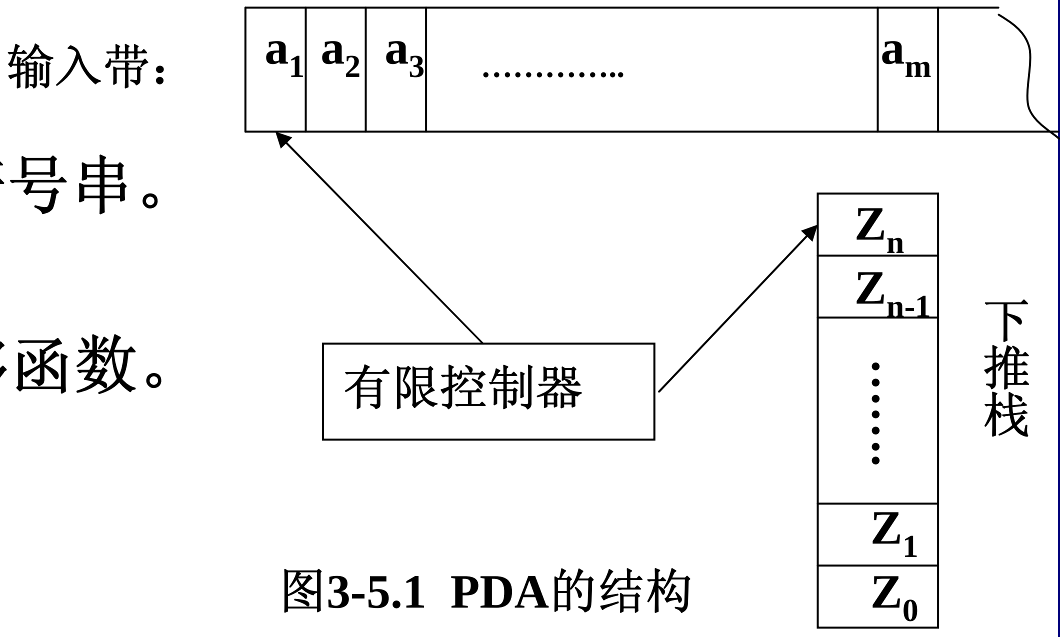


图3-5.1 PDA的结构

2. PDA的形式定义

下推自动机$M=(K,\Sigma,\Gamma,\delta,q_0,Z_0,F)$，其中

K——状态的有限集合。

Σ——输入带上的符号集合。

Γ——栈内符号集合。

q_0——$q_0\in K$，开始状态。

Z_0——$Z_0\in\Gamma$，开始时栈顶符号。

F——$F\subseteq K$，终止状态集合。

δ：状态转移函数，是映射

$K\times(\Sigma\cup\{\varepsilon\})\times\Gamma\rightarrow 2^{(K\times\Gamma^*)}$。

有两种状态转移函数：

（1）扫描输入带上符号a，

$\delta(q,a,Z)=\{(p_1,\gamma_1),(p_2,\gamma_2),\ldots,(p_m,\gamma_m)\}$， 其中

$q\in K$， $a\in\Sigma$， $Z\in\Gamma$， $p_1,p_2,\ldots,p_m\in K$,

$\gamma_1,\gamma_2,\ldots,\gamma_m\in\Gamma^*$。

此动作表示：在q状态下，当前栈顶符号为Z，读取带上符号a后，状态改成p_i，并且用γ_i代替栈顶符号Z($1\leqslant i\leqslant m$)，然后读头右移一个单元，准备读带上下一个符号。

（2）ε–动作，一般用于读带上符号串的开头或者是末尾

$\delta(q,\varepsilon,Z)=\{(p_1,\gamma_1),(p_2,\gamma_2),\ldots,(p_m,\gamma_m)\}$

此动作表示：在q状态下，当前栈顶符号为Z，读取带上符号ε(实际上没有读符号，或者带上无符号可读)后，状态改成p_i,，并且用γ_i($1\leqslant i\leqslant m$)代替栈顶符号Z, 然后，但是读头不动。

注意：假设栈内符号串γ从底到顶依次是**ABCD**，则γ要写成 γ **=DCBA,** 即栈顶符号写在左边，栈底符号写在右边。

3．通常使用符号的约定：

1）用小写英文字母**a**、**b**、**c...**表示输入带上的符号，即 **a,b,c,...**∈Σ。

2）用小写英文字母**x**、**y**、**z...**表示输入带上带符号串，即 **x,y,z,...**∈Σ*。

3）用大写英文字母**A**、**B**、**C...**表示栈内符号，即 **A,B,C,...**∈Γ。

4）用希腊字母α、β、γ**...**表示栈内符号串，即α**,** β**,** γ**,...**∈Γ*。

4.【例3-5.1】 设计一个**PDA M**，使之接收语言

$L=\{wcw^R|w\in\{0,1\}^*\}$，例如$0101c1010\in L$。

解：设计的思想是：PDA M 有两个状态，即$K=\{q_1,q_2\}$，有三个栈内符号，即$\Gamma=\{R,B,G\}$。它的动作设想：

1). 开始时，**PDA M**的状态为q_1，栈内符号为**R**。

2). 当读c左边的符号时，**M**是在q_1状态，此时

如果读**0**，则向栈内压入符号**B**；

如果读**1**，则向栈内压入符号**G**。

例如，带上符号串是**0101c1010**，

读完c左边的**0101**后，栈内符号串为：**GBGB**。

3). 当读符号c时，状态变成q_2。

4). 在此状态下之后读c右边的符号：

如果读**0**，此时栈顶应该是**B**，则**B**退栈；

如果读**1**，此时栈顶应该是**G**，则**G**退栈。

0 1 0 1 c 1 0 1 0	
入 栈	出 栈
q_1	q_2

这样，当带上符号都读完时，栈内应该是符号**R**。再将**R**清除，使得栈内为空，进而接收输入带上的符号串。具体动作如下表所示：

栈顶符号	当前状态	输入符号 0	输入符号 1	输入符号 c
B	q_1	B入栈，状态为q_1	G入栈，状态为q_1	状态改成q_2
	q_2	退栈顶，状态为q_2	—	—
G	q_1	B入栈，状态为q_1	G入栈，状态为q_1	状态改成q_2
	q_2	—	退栈顶，状态为q_2	—
R	q_1	B入栈，状态为q_1	G入栈，状态为q_1	状态改成q_2
	q_2	如果带上无符号可读，则退栈顶R，否则，无动作。		

表中符号“—”表示无动作(拒绝)。栈底为空（ε）读任何符号（非ε 串）都拒绝。

当带上符号串都读完后，栈内为空，就说明带上符号串正是**L**中的句子。**M**的形式定义如下：

M的形式定义：$M=(\{q_1,q_2\},\{0,1,c\},\{R,B,G\},\delta,q_1,R,\Phi)$

δ： $\delta(q_1,0,R)=\{(q_1,BR)\}$ $\delta(q_1,1,R)=\{(q_1,GR)\}$

$\delta(q_1,0,B)=\{(q_1,BB)\}$ $\delta(q_1,1,B)=\{(q_1,GB)\}$

$\delta(q_1,0,G)=\{(q_1,BG)\}$ $\delta(q_1,1,G)=\{(q_1,GG)\}$

$\delta(q_1,c,R)=\{(q_2,R)\}$ $\delta(q_1,c,B)=\{(q_2,B)\}$

$\delta(q_1,c,G)=\{(q_2,G)\}$ $\delta(q_2,0,B)=\{(q_2,\varepsilon)\}$

$\delta(q_2,1,G)=\{(q_2,\varepsilon)\}$ $\delta(q_2,\varepsilon,R)=\{(q_2,\varepsilon)\}$

$\delta(q_2,0,R)=\delta(q_2,1,R)=\delta(q_2,c,R)=\Phi$

$\delta(q_2,1,B)=\delta(q_2,c,B)=\Phi$

$\delta(q_2,0,G)=\delta(q_2,c,G)=\Phi$

这里$\delta(q_2,\varepsilon,R)=\{(q_2,\varepsilon)\}$的作用是：当读完带上所有符号时，清除下推栈，使之为空，接收带上符号串。

5．PDA的瞬间描述ID (Instantaneous Description)

PDA M的当前格局，称之为**M**的瞬间描述**ID**。用有序三元组**(q,x, γ)**表示瞬间描述**ID**，其中

q---- q∈**K, M**的当前状态，

x----x∈Σ*, 输入带上还没有被读到的符号串，

γ ---- γ ∈ Γ*, 当前栈内符号串。

例如上例中，当带上符号串为**0011c1100**时，

开始的ID_0：（q_1,0011c1100,R），

当读完第一个0后，变成ID_1：（q_1,011c1100,BR）。

├：表示由一个ID，经过一个动作变成另一个ID时，ID之间的转变关系。

例如上例中有 $ID_0 \vdash ID_1$，即

（q_1,0011c1100,R）├（q_1, 011c1100,BR）。

├*：表示经过多个动作后, ID之间的转变关系。

如上例中（q_1,0011c1100,R）$\vdash$（q_1, 011c1100,BR）$\vdash$（q_1,11c1100,BBR）$\vdash$（q_1,1c1100,GBBR）$\vdash$（q_1,c1100,GGBBR）$\vdash$（q_2,1100,GGBBR）

可以写成 (q_1,0011c1100,R) $\vdash^*$(q_2,1100,GGBBR)。

6. PDA M所接收的语言

PDA M接收语言有两种方式：

1).空栈接收的语言,记作N(M)：令 $M=(K,\Sigma,\Gamma,\delta,q_0,Z_0,\Phi)$,

$$N(M)=\{w|w\in\Sigma^*,(q_0,w,Z_0)\vdash^*(q,\varepsilon,\varepsilon),q\in K\}$$

即当M读完带上符号串w后，栈内为空，则接收w 。

上例中M识别0011c1100时，ID的变化过程：

(q_1,0011c1100,R) $\vdash$ (q_1,011c1100,BR) $\vdash$ (q_1,11c1100,BBR) $\vdash$ (q_1,1c1100,GBBR) $\vdash$ (q_1,c1100,GGBBR) $\vdash$ (q_2,1100,GGBBR) $\vdash$ (q_2,100,GBBR) $\vdash$ (q_2,00,BBR) $\vdash$ (q_2,0,BR) $\vdash$ (q_2, ε ,R) $\vdash$ (q_2, ε , ε)　　接收0011c1100。

2). 用终止状态接收的语言，记作T(M)：令

$\mathbf{M=(K,\Sigma,\Gamma,\delta,q_0,Z_0,F)}$,

$\mathbf{T(M)=\{w|w\in\Sigma^*,(q_0,w,Z_0)\vdash^*(q,\varepsilon,\gamma),\ q\in F,\ \gamma\in\Gamma^*\}}$

即当M读完带上符号串w后，进入终止状态q，而栈内不一定为空，则接收输入符号串w 。

后面我们将证明这两种接收方式可以互相转换。

上面介绍的PDA M中，每个状态转移函数的函数值最多有一个序偶。这相当于是个确定的PDA。

下面的例子就不同了。

7.【例3-5.2】 设计一个PDA M ，使之接收语言L如下：

$\mathbf{L=\{ww^R|w\in\{0,1\}^*\}}$，例如$\mathbf{01011010\in L}$。

此例与前例不同，前例w与w^R之间有一个标志c，而此例的w与w^R之间无标志，识别起来就麻烦一些。

解：设计的思想是：**PDA M** 有两个状态**K={q_1,q_2}**，有三个栈内符号，即 Γ＝**{R,B,G}**。

它的动作设想：

1．开始时，**PDA M**的状态为q_1，栈顶符号为**R**。

2．(1) 当读左半边的符号串**w**时，**M**是在q_1状态，这时

如果读**0**，则向栈内压入符号**B**;

如果读**1**，则向栈内压入符号**G**。

(2) 当读完左半部分符号串**w**后，状态变成q_2, 之后，要读右半边的符号串w^R，这时

如果读**0**，此时栈顶应该是**B**，则**B**退栈；否则停机(拒绝该串);

如果读**1**，此时栈顶应该是**G**，则**G**退栈;

当带上符号都读完时，栈内应该是符号**R**。

(3) 问题是：如何判断w与w^R的分界线。我们注意到w的末尾符号与w^R开始符号相同。因此当连续读的两个符号相同时，就有两种可能：一种情况是这两个符号都是w中的符号，此时状态不变；另一种情况是前一个符号是w末尾的符号，而后一个符号是w^R的第一个符号，此时就要改变状态。

构造PDA $M=(\{q_1,q_2\},\{0,1\},\{R,B,G\},\delta,q_1,R,\Phi)$

M的动作 δ 如下：

(1) $\delta(q_1,0,R)=\{(q_1,BR)\}$ (2) $\delta(q_1,1,R)=\{(q_1,GR)\}$

(3) $\delta(q_1,0,B)=\{(q_1,BB),(q_2,\varepsilon)\}$ (4) $\delta(q_1,1,B)=\{(q_1,GB)\}$

(5) $\delta(q_1,0,G)=\{(q_1,BG)\}$ (6) $\delta(q_1,1,G)=\{(q_1,GG),(q_2,\varepsilon)\}$

(7) $\delta(q_2,0,B)=\{(q_2,\varepsilon)\}$ (8) $\delta(q_2,1,G)=\{(q_2,\varepsilon)\}$

(9) $\delta(q_2,\varepsilon,R)=\{(q_2,\varepsilon)\}$ (10) $\delta(q_1,\varepsilon,R)=\{(q_2,\varepsilon)\}$

$\delta(q_2,0,R)=\delta(q_2,1,R)=\delta(q_2,1,B)=\delta(q_2,0,G)=\Phi$

下面看看M识别110011的ID变换过程：

注：符号“$\vdash$⑽”表示执行动作⑽得到后面的ID；

符号“$\top$⑵”表示执行动作⑵得到下面的ID。

$(q_1,110011,R) \vdash_{(10)} (q_2,110011,\varepsilon)$

$\top\ (2)$

$(q_1,10011,GR) \vdash_{(6)} (q_2,0011,R) \vdash_{(9)} (q_2,0011,\varepsilon)$

$\top\ (6)$

$(q_1,0011,GGR)$

$\top\ (5)$

$(q_1,011,BGGR) \vdash_{(3)} (q_2,11,GGR) \vdash_{(8)} (q_2,1,GR)$

$\top\ (3)$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $\top\ (8)$

$(q_1,11,BBGGR)$ $\qquad\qquad\qquad\qquad\qquad\qquad$ (q_2,ε,R)

$\top\ (4)$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $\top\ (9)$

$(q_1,1,GBBGGR) \vdash_{(6)} (q_2,\varepsilon,BBGGR)$ $\qquad$ $(q_2,\varepsilon,\varepsilon)$

$\top\ (6)$

$(q_1,\varepsilon,GGBBGGR)$

作业题

1. 构造一个**PDA M**，使得

T(M)= {w| w$\in$**{a,b}* $\wedge$ w** 中的**a,b**的个数相等**}**。

3.6 PDA与CFG之间的等价性

本节介绍PDA与CFG之间的等价性。

在介绍它们相互转换之前，先介绍PDA的两种接收语言方式可以互相转换。

1．PDA的两种接收语言方式互相转换

定理3-6.1 设一个PDA M用空栈接收语言L，即N(M)=L，当且仅当存在一个PDA M′,使得M′用终止状态接收L，即T(M′)=L。

证明：1．必要性：设$M=(K,\Sigma,\Gamma,\delta,q_0,Z_0,\Phi)$，用空栈接收语言L，N(M)=L。

构造 $M'=(K\cup\{q_0',q_f\},\Sigma,\Gamma\cup\{Z_0'\},\delta',q_0',Z_0',\{q_f\})$，其中$q_0',q_f\notin K$，$Z_0'\notin\Gamma$。

假设M接收句子w，即$w \in N(M)$，则M必有如下ID变换：

M： $(q_0,w,Z_0) \vdash^* (q, \varepsilon, \varepsilon)$ 其中$q \in K$

M′要接收w，必须能够从它的开始ID（q_0',w,Z_0'）进入到M的开始ID（q_0,w,Z_0），以后M′就能够模拟M的所有动作，当达到M的终止时的ID（$q, \varepsilon, \varepsilon$）时，最后进入M′的终止状态$q_f$，从而接收w。于是M′的动作序列为：

M′： $(q_0',w,Z_0') \vdash (q_0,w,Z_0Z_0') \vdash^* (q, \varepsilon, \varepsilon Z_0') \vdash (q_f, \varepsilon, \varepsilon)$ 其中$q \in K$

于是δ′有下面三种类型的动作：

1） $\delta'(q_0', \varepsilon, Z_0')=\{(q_0,Z_0Z_0')\}$ 此动作可使M′进入M的开始ID。

2）对任何$q \in K$，任何$a \in \Sigma \cup \{\varepsilon\}$，任何$X \in \Gamma$，

$\delta'(q,a,X)= \delta(q,a,X)$ 这类动作使得M′模拟M的所有动作。

3）对任何$q\in K$,

$\delta'(q,\varepsilon,Z_0')=\{(q_f,\varepsilon)\}$　　使得M′进入自己的终止状态。

根据上面分析，可以看到：任何$w\in N(M)$，必有$w\in T(M')$，(这里不作详细证明)。反之亦然。故有$T(M')=N(M)$。

2．充分性：设$M'=(K,\Sigma,\Gamma,\delta',q_0',Z_0',F)$，用终止状态接收L，且$T(M')=L$。

构造$M=(K\cup\{q_0,q_e\},\Sigma,\Gamma\cup\{Z_0\},\delta,q_0,Z_0,\Phi)$，其中$q_0,q_e\notin K$，$Z_0\notin\Gamma$。

假设M′接收句子w，即$w\in T(M')$，则M′必有ID序列：

M'：　$(q_0',w,Z_0')\vdash^*(q,\varepsilon,\gamma)$　其中$q\in F$，$\gamma\in\Gamma^*$。

M要接收w，必须能够从它的开始ID（q_0,w,Z_0）进入M′的开始ID（q_0',w,Z_0'），以后能够模拟M′的所有动作，达到M′的终了时的ID（q,ε,γ），最后进入M的清栈状态q_e，再进行清栈，使栈内为空。于是M的动作序列为：

M: $(q_0,w,Z_0) \vdash (q_0',w,Z_0'Z_0) \vdash^* (q, \varepsilon, \gamma Z_0) \vdash^*$
$(q_e, \varepsilon, \varepsilon)$ 其中$q \in K$

于是δ有下面四种类型的状态转移函数：

1） $\delta(q_0, \varepsilon, Z_0)=\{(q_0', Z_0'Z_0)\}$ 可使**M**进入**M′**的开始**ID**。

2）对任何$q \in K$，任何$a \in \Sigma \cup \{\varepsilon\}$，任何$X \in \Gamma$，
$\delta(q,a,X)=\delta'(q,a,X)$ 使得**M**模拟**M′**的所有动作。

3）对任何$q \in F$，任何$X \in \Gamma \cup \{Z_0\}$，
$\delta(q, \varepsilon, X)=\{(q_e, \varepsilon)\}$ 此动作使得**M**进入清栈状态q_e。

4）对任何$X \in \Gamma \cup \{Z_0\}$，
$\delta(q_e, \varepsilon, X)=\{(q_e, \varepsilon)\}$ 使得**M**反复清栈，直至栈内为空。

根据上面分析，可得：任何$w \in T(M')$，必有$w \in N(M)$，反之亦然。(这里不作详细证明)。故有$N(M)=T(M')$。

从这个定理看到，**PDA**的两种接收语言的方式可以互相转换。因此以后我们讨论的**PDA**，对它接收语言的方式不加限制。

2. 由CFG 转换成PDA

下面介绍根据给定的CFG G，求接收L(G)的PDA M。

为了理解定理的证明，我们先举一个例子。

【例3-6.1】给定CFG G=({S},{0,1},P,S)，其中P：

$S \to 0S0|1S1|00|11$。显而易见，

$L(G)=\{ww^R|w\in\{0,1\}^*\}$

求一个PDA M，使得N(M)=L(G)。

解：1. 先将G变成GNF形式的G′，得

G′=({S,A,B},{0,1},P′,S)

P′： $S\to 0SA|1SB|0A|1B$, $A\to 0$, $B\to 1$。

2. 构造PDA M=({q},{0,1},{S,A,B}, δ ,q,S, Φ)，其中

δ 中动作的确定：如果$A\to a\gamma \in P'$，则有 $\delta(q,a,A)$中含有(q,γ)。

因而，由$S\to 0SA|0A$ 得：$\delta(q,0,S)=\{(q,SA),(q,A)\}$。

由$S\to 1SB|1B$ 得：$\delta(q,1,S)=\{(q,SB),(q,B)\}$。

由$A\to 0$ 得：$\delta(q,0,A)=\{(q,\varepsilon)\}$。

由$B\to 1$ 得：$\delta(q,1,B)=\{(q,\varepsilon)\}$。

下面验证一下它接收的语言与G产生的语言是相同的，并看一看M是如何模拟G的派生的。

G中对句子011110的最左派生过程：

$S\Rightarrow 0SA\Rightarrow 01SBA\Rightarrow 011BBA\Rightarrow 0111BA\Rightarrow 01111A\Rightarrow 011110$

M识别011110的过程：

$(q,011110,S)\vdash(q,11110,SA)\vdash(q,1110,SBA)\vdash(q,110,BBA)$
$\vdash(q,10,BA)\vdash(q,0,A)\vdash(q,\varepsilon,\varepsilon)$

PDA M在识别011110时的每个ID中的栈内符号串，分别与文法G的相应派生的句型中的变元串相同。从这里看出PDA M是模拟G的最左派生的。

下面给出有关定理。

定理3-6.2如果L是CFL，则存在PDA M，使得N(M)=L。

证明：如果 $\varepsilon \notin L$，令$G=(V_N, V_T, P, S)$是具有GNF形式的CFG，且$L(G)=L$。

构造PDA $M=(\{q\}, V_T, V_N, \delta, q, S, \Phi)$ 其中

δ 中动作确定：如果$A \to a\gamma \in P$，则有 $\delta(q,a,A)$中含有(q, γ)，$a \in V_T$， $\gamma \in V_N^*$。

PDA M模拟G的最左派生。由于G具有GNF形式，所以G的最左派生中的每个句型都是$x\alpha$ 形式，其中$x \in V_T^+$，$\alpha \in V_N^*$。M总是在识别完前部分终极符串x之后，将句型中余下的变元串α 存在栈内。即用归纳法证明如下结论：

G中最左派生$S \Rightarrow^* x\alpha$，当且仅当 M中有$(q,x,S) \vdash^* (q, \varepsilon, \alpha)$。

1．充分性。已知M中有$(q,x,S) \vdash^* (q, \varepsilon, \alpha)$，对M的动作次数归纳，证出G中有最左派生$S \Rightarrow^* x\alpha$。

(1) 当M动作次数为0时，有$(q,x,S) \vdash (q, \varepsilon, \alpha)$，显然此时$x=\varepsilon, \alpha=S$，于是G中有$S \Rightarrow x\alpha$。

(2) 假设$(q,y,S) \vdash^* (q, \varepsilon, \beta)$是由少于**k**个动作完成，则文法**G**中有派生：$S \Rightarrow^* y\beta$ (这里$x=y$，$\alpha=\beta$)。

(3) 当$(q,ya,S) \vdash^* (q,a,\beta) \vdash (q,\varepsilon,\alpha)$是由**k**个动作完成时(这里$x=ya$)，前**k-1**个动作根据假设⑵得：在**G**中有派生$S \Rightarrow^* y\beta$ 。

下面通过分析**M**的栈内符号串得出如下结论：

令$\beta = A\gamma$，$A \in \Gamma$ ($\Gamma = V_N$)，$\alpha = \eta\gamma$，于是**M**的第**k**个动作可以写成：

$(q,a,A\gamma) \vdash (q,\varepsilon,\eta\gamma)$，由此说明**M**有如下状态转移函数：$\delta(q,a,A)$中有$(q,\eta)$,于是**G**中有产生式：$A \rightarrow a\eta$，于是**G**中有派生：

$S \Rightarrow^* y\beta = yA\gamma \Rightarrow ya\eta\gamma = x\alpha$，即$S \Rightarrow^* x\alpha$ 。

可见结论成立。

2．必要性。设G中有最左派生$S\Rightarrow^* x\alpha$，（通过对G中派生步数归纳证明M中有：（q,x,S）$\vdash^*$（q, ε , α ））。

(1) 如果G中此派生是由0步完成的，则此派生为：$S\Rightarrow S$，令x= ε , α =S，所以此派生也可以写成 $S\Rightarrow x\alpha$。而M中显然有(q, ε ,S) $\vdash$ (q, ε ,S)，于是也可以写成

(q,x,S) $\vdash$ (q, ε , α)。结论成立。

(2) 假设G中派生$S\Rightarrow^* x\alpha$是由少于k步完成时，则M中有

(q,x,S) $\vdash^*$(q, ε , α)。

(3) 若G中有k步派生：$S\Rightarrow^* yA\gamma \Rightarrow ya\eta\gamma = x\alpha$ 其中$y\in V_T^*$，$A\in V_N$， $\gamma, \eta \in V_N^*$，$a\in V_T$，ya=x， $\alpha = \eta\gamma$。

由前k-1步派生$S\Rightarrow^* yA\gamma$推出的句型，是由S推导出的前面是

终极符串y的句型。由假设(2)得，M中有

(q,ya,S) $\vdash^*$(q,a,Aγ),

即当M识别完y之后，栈内符号串为Aγ 。

再看看G的第k步派生：此派生是应用产生式$A\to a\eta$。由δ的定义得M中有动作：$\delta(q,a,A)$中含有(q,η)，于是M有如下的ID变换：

$(q,ya,S)\vdash^{*}(q,a,A\gamma)\vdash(q,\varepsilon,\eta\gamma)=(q,\varepsilon,\alpha)$，即

$(q,x,S)\vdash^{*}(q,\varepsilon,\alpha)$，

必要性成立。

最后得结论：**G中有最左派生$S\Rightarrow^{*}x\alpha$，当且仅当M中有$(q,x,S)\vdash^{*}(q,\varepsilon,\alpha)$**。

为了完成本定理的证明，只要令上述结论中$\alpha=\varepsilon$时，

则得：G中一个最左派生$S\Rightarrow^{*}x$，当且仅当M中有$(q,x,S)\vdash^{*}(q,\varepsilon,\varepsilon)$。

于是得：$x\in L(G)$，当且仅当$x\in N(M)$。

所以$N(M)=L(G)$。

3. 由PDA 转换成CFG

定理3-6.3如果有一个PDA M使得N(M)=L，则L是CFL。

证明：令$M=(K,\Sigma,\Gamma,\delta,q_0,Z_0,\Phi)$，$N(M)=L$。

构造一个CFG $G=(V_N,V_T,P,S)$，其中

$V_N=\{[q,A,p]|q,p\in K,A\in\Gamma\}\cup\{S\}$ $V_T=\Sigma$

P中产生式有三种类型：

1．对任何$q\in K$,有 $S\to[q_0,Z_0,q]$。先设置S到代表初始ID的“变量”的语法规则。

2．对K中任何$q,q_1,q_2,\ldots,q_m,q_{m+1}=p$，任何$a\in\Sigma\cup\{\varepsilon\}$,任何$A,B_1,B_2,\ldots,B_m\in\Gamma$,

只要$\delta(q,a,A)$中含有$(q_1,B_1B_2\ldots B_m)$，则有产生式

$[q,A,p]\to a[q_1,B_1,q_2][q_2,B_2,q_3]\ldots[q_m,B_m,p]$。

3．对任何$q,p\in K$，任何$a\in\Sigma\cup\{\varepsilon\}$,任何$A\in\Gamma$,如果有$\delta(q,a,A)$中含有$(p,\varepsilon)$，则有产生式$[q,A,p]\to a$。

（当PDA动作是让变量消失，则加让变量消失的规则）

按照上述规则得到的产生式，有时需要进行简化，去掉无用的产生式，去掉引起推导死循环进而无法推出终极符串的产生式。

可以证明 **L(G)=L**。（证明从略）

为了清楚地了解上述规则，以及看看**CFG G**文法是如何模拟**PDA M**的，下面举例说明。

【例3-6.2】给定PDA $M=(\{q_0,q_1\},\{0,1\},\{X,Z_0\},\delta,q_0,Z_0,\Phi)$

δ：(1) $\delta(q_0,0,Z_0)=\{(q_0,XZ_0)\}$ (2) $\delta(q_0,0,X)=\{(q_0,XX)\}$

(3) $\delta(q_0,1,X)=\{(q_1,\varepsilon)\}$ (4) $\delta(q_1,1,X)=\{(q_1,\varepsilon)\}$

(5) $\delta(q_1,\varepsilon,Z_0)=\{(q_1,\varepsilon)\}$

M: 读一个**0**，一个**X**入栈。

读一个**1**，一个**X**退栈。

当**X**都退出后，再退掉Z_0，栈为空，接收输入符号串。

所以**M**识别的语言$L=T(M)=\{0^i1^i|i\geq 1\}$。

求一个**CFG G**，使得**L(G)=L**。

解. 令$G=(V_N, V_T, P, S)$　$V_T=\{0,1\}$

$V_N = \{S,[q_0,Z_0,q_0],[q_0,Z_0,q_1],[q_1,Z_0,q_0],[q_1,Z_0,q_1],$
$[q_0,X,q_0],[q_0,X,q_1],[q_1,X,q_0],[q_1,X,q_1]\}$

P构成如下：按照上面规则，得三种类型产生式。

1. 对任何$q \in K$,有 $S \rightarrow [q_0,Z_0,q]$。

   1）$S \rightarrow [q_0,Z_0,q_0]$

   2）$S \rightarrow [q_0,Z_0,q_1]$

2. 若$\delta(q,a,A)$中含有$(q_1,B_1B_2 \ldots B_m)$，则有产生式

   $[q,A,p] \rightarrow a[q_1,B_1,q_2][q_2,B_2,q_3]\ldots[q_m,B_m,p]$。$q_{m+1}=p$

   由(1) $\delta(q_0,0,Z_0)=\{(q_0,XZ_0)\}$得

   3）$[q_0,Z_0,q_0] \rightarrow 0[q_0,X,q_0][q_0,Z_0,q_0]$

   4）$[q_0,Z_0,q_0] \rightarrow 0[q_0,X,q_1][q_1,Z_0,q_0]$

   5）$[q_0,Z_0,q_1] \rightarrow 0[q_0,X,q_0][q_0,Z_0,q_1]$

   6）$[q_0,Z_0,q_1] \rightarrow 0[q_0,X,q_1][q_1,Z_0,q_1]$

由(2) $\delta(q_0,0,X)=\{(q_0,XX)\}$得

7) $[q_0,X,q_0]\to 0[q_0,X,q_0][q_0,X,q_0]$

8) $[q_0,X,q_0]\to 0[q_0,X,q_1][q_1,X,q_0]$

9) $[q_0,X,q_1]\to 0[q_0,X,q_0][q_0,X,q_1]$

10) $[q_0,X,q_1]\to 0[q_0,X,q_1][q_1,X,q_1]$

3. 任何$a\in\Sigma\cup\{\varepsilon\}$,任何$A\in\Gamma$,

如果$\delta(q,a,A)$中含有(p,ε),

则有产生式 $[q,A,p]\to a$ 。

由(3) $\delta(q_0,1,X)=\{(q_1,\varepsilon)\}$得

11) $[q_0,X,q_1]\to 1$

由(4) $\delta(q_1,1,X)=\{(q_1,\varepsilon)\}$得

12) $[q_1,X,q_1]\to 1$

由(5) $\delta(q_1,\varepsilon,Z_0)=\{(q_1,\varepsilon)\}$得

13) $[q_1,Z_0,q_1]\to\varepsilon$

δ: ×1） $S \to [q_0,Z_0,q_0]$ 此变元无产生式

2） $S \to [q_0,Z_0,q_1]$

× 3） $[q_0,Z_0,q_0] \to 0[q_0,X,q_0][q_0,Z_0,q_0]$ 出现死循环

简化产生式

× 4） $[q_0,Z_0,q_0] \to 0[q_0,X,q_1][q_1,Z_0,q_0]$ 此变元无产生式

× 5） $[q_0,Z_0,q_1] \to 0[q_0,X,q_0][q_0,Z_0,q_1]$ 此变元无产生式

6） $[q_0,Z_0,q_1] \to 0[q_0,X,q_1][q_1,Z_0,q_1]$

× 7） $[q_0,X,q_0] \to 0[q_0,X,q_0][q_0,X,q_0]$ 出现死循环

× 8） $[q_0,X,q_0] \to 0[q_0,X,q_1][q_1,X,q_0]$ 此变元无产生式

× 9） $[q_0,X,q_1] \to 0[q_0,X,q_0][q_0,X,q_1]$ 此变元无产生式

10） $[q_0,X,q_1] \to 0[q_0,X,q_1][q_1,X,q_1]$

11） $[q_0,X,q_1] \to 1$

12） $[q_1,X,q_1] \to 1$

13） $[q_1,Z_0,q_1] \to \varepsilon$

最后，得P中产生式有：

2） $S \to [q_0,Z_0,q_1]$

6） $[q_0,Z_0,q_1] \to 0[q_0,X,q_1][q_1,Z_0,q_1]$

10） $[q_0,X,q_1] \to 0[q_0,X,q_1][q_1,X,q_1]$

11） $[q_0,X,q_1] \to 1$

12） $[q_1,X,q_1] \to 1$

13） $[q_1,Z_0,q_1] \to \varepsilon$

下面先考察M是如何识别0011的，再看看G是如何模拟M而最左派生出0011的，从而验证L(G)=N(M)。

M识别0011时ID的变换过程：

$(q_0,0011,Z_0) \vdash (q_0,011,\underline{XZ_0}) \vdash (q_0,11,\underline{XXZ_0}) \vdash (q_1,1,\underline{XZ_0})$
$\vdash (q_1,\varepsilon,Z_0) \vdash (q_1,\varepsilon,\varepsilon)$

G最左派生0011的句型变换过程： $S \Rightarrow [q_0,Z_0,q_1] \Rightarrow$
$0[q_0,X,q_1][q_1,Z_0,q_1] \Rightarrow 00[q_0,X,q_1][q_1,X,q_1][q_1,Z_0,q_1]$
$\Rightarrow 001[q_1,X,q_1][q_1,Z_0,q_1] \Rightarrow 0011[q_1,Z_0,q_1] \Rightarrow 0011$

可以看出**G**的每个句型中以三元组形式的各个变元里的第二个元素构成的符号串分别是:

Z_0、XZ_0、XXZ_0、XZ_0、Z_0、 ε 。

这些符号串刚好与**M**中每个**ID**中栈内符号串相匹配。这就说明**G**的最左派生是模拟**M**的动作的。

作业题：

1．给定CFG G=（{S,A,B},{a,b,c},P,S），其中

P为：$S \to aAB \mid aA \quad A \to bSa \mid Ab \mid Bc \mid b$，

求一个PDA M，使得T(M)=L(G)。

2．给定PDA M=（$\{q_0,q_1\},\{0,1\},\{z_0, x\}, \delta, q_0, \Phi$），其中

δ如下：

$\delta(q_0,1,z_0)=\{(q_0,xz_0)\}$　$\delta(q_0,1,x)=\{(q_0,xx)\}$

$\delta(q_0,0,x)=\{(q_1,x)\}$　$\delta(q_0,\varepsilon,z_0)=\{(q_0,\varepsilon)\}$

$\delta(q_1,1,x)=\{(q_1,\varepsilon)\}$　$\delta(q_1,0,z_0)=\{(q_0,z_0)\}$

求一个CFG G 使得L(G)=N(M).

3.7 CFL的有关判定问题

本节讨论以下判定问题：

判定某个语言**L**不是**CFL**；

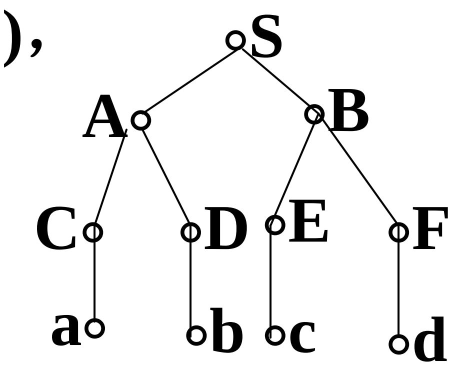
判定给定**CFL L**是有穷的还是无穷的；

判定一个输入符号串**w**是否是给定一个**CFG G**产生的句子。

由于判定中要用到**CFL**的泵作用引理，所以先介绍此引理。

1. CFL的泵作用引理

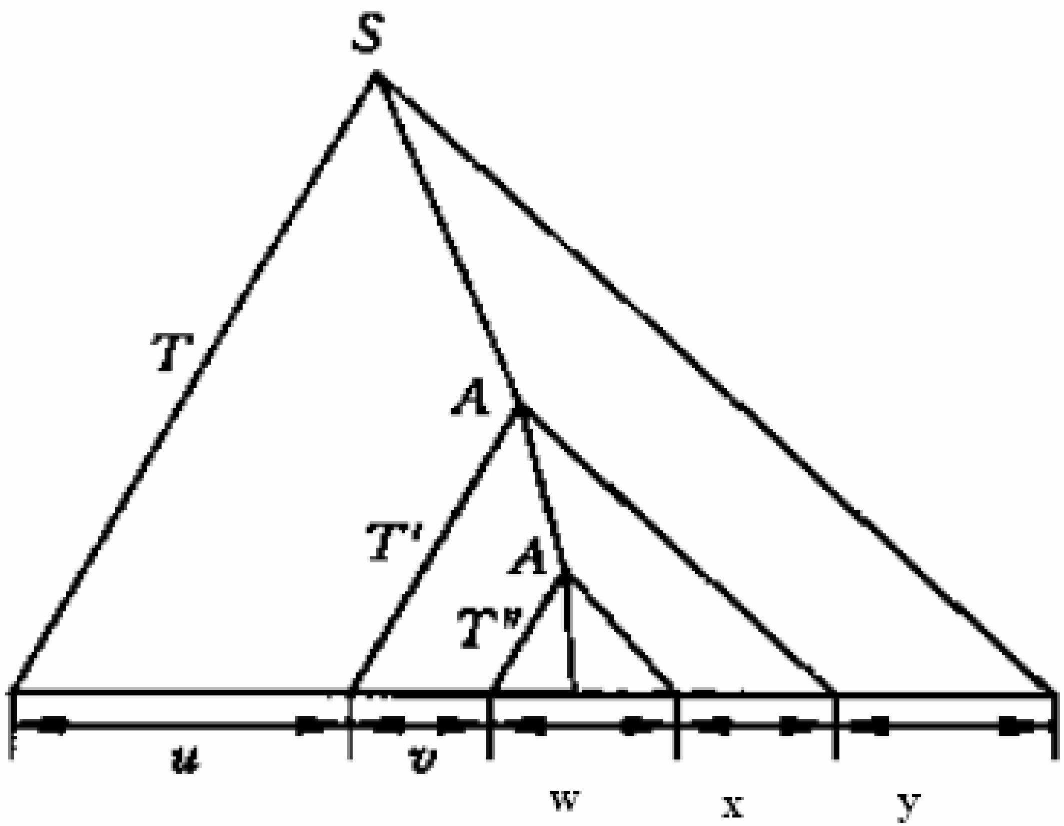
给定有**CNF**形式的CFG G=(V_N, V_T, P, S)，设$|V_N|=k$，



(1) 任取$z\in L(G)$，z的派生树为二叉树，如果该树所有从根到叶的路径中最长的路径长度为i时，则叶结点个数最多为2^{i-1}，所以$|z|\leq 2^{i-1}$，($|z|$最大为2^{i-1}，如右图 i=3的情形)。

(2) 如果$z\in L(G)$，且$|z|>2^{k-1}$ 时，在z的派生树中找一条从根到叶的最长路径，则此路径的长度至少为k+1，那么此路径有 k+2个结点，去掉一个标有终极符的叶结点，余下k+1个结点是标有变元的结点，而G中只有k个变元，所以此路径中至少有两个结点标有相同变元，设此变元为A，于是z的派生树如下图所示。

z的派生树示意图：



从此派生树得G中有派生：

$S \Rightarrow^* uAy$，　$A \Rightarrow^* vAx$，　$A \Rightarrow^* w$。

其中$u,v,w,x,y \in V_T^*$，

$z=uvwxy$

$S\Rightarrow^* uAy$,　$A\Rightarrow^* vAx$,　$A\Rightarrow^* w$。

于是可以进一步得到如下派生：

$S\Rightarrow^* uAy\Rightarrow^* uvAxy\Rightarrow^* uvvAxxy\Rightarrow^* uv^3Ax^3y\Rightarrow^* uv^iwx^iy\in L(G)\ (i\geq 0)$

从而得到下面引理——CFL的泵作用引理。

引理3-7.1（CFL泵作用引理） 设L是CFL，则存在常数n，如果$z\in L$且$|z|\geq n$，则可将z写成$z=uvwxy$形式，其中$|vx|\geq 1$，$|vwx|\leq n$，且对任何$i\geq 0$，有$uv^iwx^iy\in L$。

(此常数n 与接收**L**的**CFG G**中的变元个数**k**有关，$\mathbf{n=2^k}$)

下面介绍应用这个引理判定某个语言**L**不是**CFL**。

2．判定某个语言L不是CFL

【例3-7.1】 证明$L=\{a^ib^ic^i \mid i\geq 1\}$不是CFL。

证明：假设L是CFL。设n是L满足CFL泵作用引理的常数。取$z=a^nb^nc^n$，$|z|=3n>n$，于是根据泵作用引理，可将z写成$z=uvwxy$形式，其中$|vx|\geq 1$，$|vwx|\leq n$，且对任何$i\geq 0$，有$uv^iwx^iy\in L$。下面分五种情况讨论，看看如何选取v和x。

(1) vx中不可能包含符号a和c，因z中最右边的a与最左边的c中间有n个b，受$|vwx|\leqslant n$限制，故不能出现此情况。

(2) 如果v和x中只有符号a，取i=0，$uv^iwx^iy=uwy$，因为$|vx|\geqslant 1$, 所以uwy中a的个数要少于b的个数，所以$uwy\notin L$，这与$i\geqslant 0$, 有$uv^iwx^iy\in L$矛盾。

(3) 如果v和x中只有符号b或者只有符号c，类似可以推出矛盾。

(4) 如果vx中含有符号a和b，取i=0，$uv^iwx^iy=uwy$，因为$|vx|\geqslant 1$, 所以uwy中a和b的个数要少于c的个数，所以$uwy\notin L$，这与$i\geqslant 0$, 有$uv^iwx^iy\in L$矛盾。

(5) 如果vx中含有符号b和c，取i=0，$uv^iwx^iy=uwy$，因为$|vx|\geqslant 1$, 所以uwy中b和c的个数要少于a的个数，所以$uwy\notin L$，这与$i\geqslant 0$, 有$uv^iwx^iy\in L$矛盾。

可见，不论v和x取什么样的符号串，都产生矛盾。

所以L不是CFL。

3．CFL L的有穷性判定

定理3-7.1 存在算法用以判定给定CFL L是否为空集、有穷集还是无穷集。

证明：设有CFG $G=(V_N, V_T, P, S)$，且$L(G)=L$。

1)．判定L是否为空集。

可对G用引理3-2.1的算法处理，使得任何$A\in V_N$，都有派生$A\Rightarrow^* w$，$w\in V_T^*$。

显然 $L(G)\neq\Phi$，当且仅当 存在$w\in V_T^*$，有$S\Rightarrow^* w$。

所以用此算法判定：

$L(G)\neq\Phi$，当且仅当 $S\in NEWV_N$。

$L(G)=\Phi$，当且仅当 $S\notin NEWV_N$。

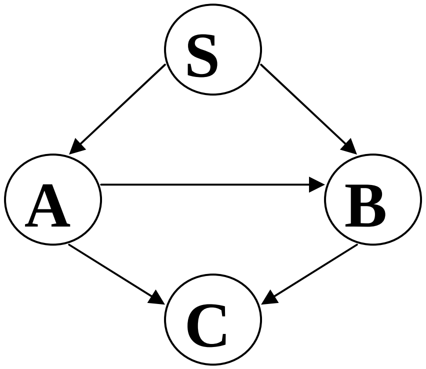
2)．判定L(G)是有穷还是无穷集合

先定义文法G的有向图。

定义：设$G=(V_N, V_T, P, S)$是具有CNF形式的CFG，按照下面方法画出的有向图称为G的有向图：以变元为图的结点，如果有产生式$A\rightarrow BC$或者$A\rightarrow CB$，则从A到B、从A到C画有向边。

【例3-7.2】如G中有产生式：$S\rightarrow AB$，$A\rightarrow BC|a$，$B\rightarrow CC|b$，$C\rightarrow a$，

G 的有向图如右图所示。



对L(G)的有穷性判定，转换成判定G的有向图中是否有回路。

设CFG G是具有CNF形式的文法，若G的有向图中没有回路，则L(G)是有限的。否则L(G)是无限的。

因为如果G的有向图没有回路，则G的任何句子的派生树中从树根到树叶的任何一条路径中不会有两个结点标有相同的变元，所以这样的派生树的高度是有限的，于是此树的结果(派生的句子)长度是有限的，因此G派生出的句子个数是有限的，故L(G)是有限集合。

若G的有向图有回路，假设此回路为$A_0A_1A_2A_3 \ldots A_nA_0$，则G中必有如下派生：

$A_0 \Rightarrow \alpha_1 A_1 \beta_1 \Rightarrow \alpha_2 A_2 \beta_2 \Rightarrow \alpha_3 A_3 \beta_3 \Rightarrow \ldots \Rightarrow \alpha_n A_n \beta_n \Rightarrow \alpha_{n+1} A_0 \beta_{n+1}$，

其中$\alpha_i, \beta_i \in V_N^*$，且$|\alpha_i \beta_i| = i$，$(1 \leqslant i \leqslant n+1)$

(因为G是具有CNF形式的，上述每步派生都是一个变元变成两个变元的派生。比如其中第一个派生，一定是应用产生式$A_0 \rightarrow \alpha_1 A_1$或者$A_0 \rightarrow A_1 \beta_1$，所以$|\alpha_1 \beta_1| = 1$。)

因为G中没有无用符号，所有变元都可以推出终极符串，所以必存在$v,x \in V_T^*$，使得$\alpha_{n+1} \Rightarrow^* v$，$\beta_{n+1} \Rightarrow^* x$，且 因为 $|v| \geqslant |\alpha_{n+1}|$，$|x| \geqslant |\beta_{n+1}|$，所以，
$|v|+|x| \geqslant |\alpha_{n+1}|+|\beta_{n+1}|=n+1$，而$n \geqslant 0$，所以$|v|+|x| \geqslant 1$，即表明v与x不可能同时为ε。于是有派生：$A_0 \Rightarrow^* vA_0x$。
同样因为G中没有无用符号，必存在$u,y,w \in V_T^*$，使得G中有派生：

$S \Rightarrow^* uA_0y$，$A_0 \Rightarrow^* w$，于是G中有派生：

$$S \Rightarrow^* uA_0y \Rightarrow^* uvA_0xy \Rightarrow^* uvvA_0xxy = uv^2A_0x^2y \Rightarrow^* uv^iA_0x^iy \Rightarrow^* uv^iwx^iy,$$

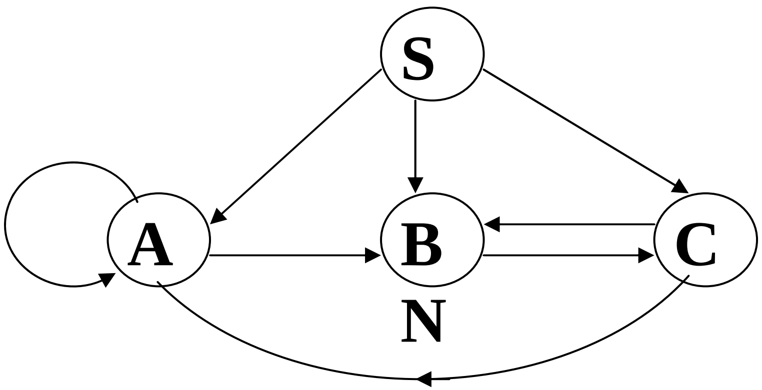
所以$uv^iwx^iy \in L(G)(i \geqslant 0)$。因为$|vx| \geqslant 1$，i可以取无穷多个非负整数，于是这样的句子有无穷多个，所以L(G)是无穷集合。

这样，判定L(G)是无限集合还是有限集合，就是看G的有向图是否有回路。

【例3-7.3】给定CFG G=({S,A,B,C},{a,b},P,S)，其中P为：

S→AB|BC， A→BA|a， B→CC|b， C→AB|a

则G的有向图如下图所示。



G的有向图

由此图看出，图中有回路，所以L(G)是无限集合。显然这种判定方法也适用于正规文法产生的语言的判定。

4．CFL L的成员资格判定

所谓CFL L的成员资格判定，就是给定一个CFG $G=（V_N,V_T,P,S）$， 给定句子$x\in V_T^*$，判定x是否是L(G)中的句子。

如果$x=\varepsilon$，则比较容易判定，看是否有派生 $S\Rightarrow^*\varepsilon$，即判定S是否为可为零的(详见定理3-2.2)。

如果$x\neq\varepsilon$，可以用下面的**CYK算法**进行判定。设CFG $G=(V_N,V_T,P,S)$是具有CNF 形式的文法，任取$x\in V_T^*$，设$|x|=n$，$n\geq 1$，判定是否有$x\in L(G)$。

CYK算法的判断思路是：

（1）如果$|x|=1$，看是否有产生式$S\to x$。

CFL的封闭性

(2)如果$|x|\geqslant 2$，就将x分成两个子串x_1和x_2，$x = x_1x_2$，就看是否有产生式$S\to A_1A_2$，使得$A_1 \Rightarrow^* x_1$, $A_2 \Rightarrow^* x_2$。

如果$|x_1|\geqslant 2$，就将x_1分成两个子串x_{11}和x_{12}，$x_1 = x_{11}x_{12}$，再看是否有产生式$A_1\to A_{11}A_{12}$，使得$A_{11}\Rightarrow^* x_{11}$，

$A_{12}\Rightarrow^* x_{12}$。

对x_2也类似考虑，再对x_{11}、x_{12}如前考虑，直到各个子串的长度为1为止。**CYK算法的处理过程，刚好是上面过程的逆过程，即从各个最短的子串开始，找出可以推出这些子串的所有可能的变元**。为此先定义变元的集合V_{ij}。

定义：变元集合V_{ij}定义如下：

$V_{ij}=\{A|A\Rightarrow^* x_{ij}$，$x_{ij}$是$x$中的第$i$位起长度为$j$的子串$\}$

例如，$x=abcdef$，则$x_{24}=bcde$。

显然，如果$S\in V_{1n}$ $(|x|=n)$，则$x\in L(G)$，下面就是如何求V_{ij}的算法。

CYK(Cocke_Younge_Kasami)算法：

begin

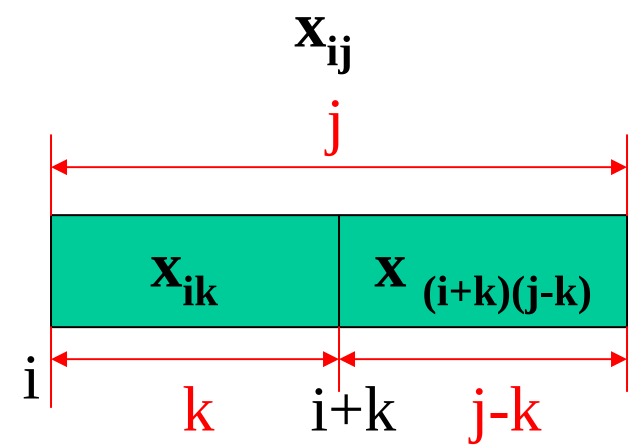
1. for i=1 to n do　　　/*|x|＝n*/
2. V_{i1} :={A|A→a∈P, x_{i1}=a}
3. for j=2 to n do
4. for i=1 to n-j+1 do

   begin

5. V_{ij}:=Φ
6. for k=1 to j-1 do　/*k是将符号串分成两个子 串的分割点*/
7. $V_{ij}:=V_{ij}\cup\{A|A\rightarrow BC\in P, BC\in V_{ik}V_{(i+k)(j-k)}\}$

   end

end



其中$V_{ik}V_{(i+k)(j-k)}$是集合V_{ik}与$V_{(i+k)(j-k)}$的乘积，实际上,若$BC\in V_{ik}V_{(i+k)(j-k)}$，就是$B\in V_{ik}$，$C\in V_{(i+k)(j-k)}$。

执行完该算法，最后求出V_{1n}，如果$S\in V_{1n}$，则$x\in L(G)$。

【例3-7.4】给定CFG G=({S,A,B,C},{a,b},P,S)，其中P为：

S→AB|BC， A→BA|a， B→CC|b， C→AB|a

用CYK算法判定baaba是否属于L(G)。

解：令x=baaba，|x|=5，

执行CYK算法的结果是将各个V_{ij}填入下表中。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2					
3					
4					
5					

执行第1,2步：

即j＝1的情形，求V_{i1}(i=1,2,...,5)，这些集合分别是产生b, a, a, b, a的变元的集合

S→AB|BC， A→BA|a， B→CC|b， C→AB|a

j＝2 i=1,2,3,4。

i=1 求V_{12}： 即推出ba的变元集合。

k=1 b,a 这取决于 $V_{11}V_{21}$＝{B}{A,C}={BA,BC}。

因为有产生式A→BA S→BC， 所以 V_{12}＝{S,A}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}				
3					
4					
5					

S→AB|BC, A→BA|a, B→CC|b, C→AB|a

i=2 求V_{22}：即推出aa的变元集合。

k=1 a,a 取决于 $V_{21}V_{31}$＝{A,C}{A,C}={AA,AC,CA,CC}。

因为只有产生式 B→CC, 所以 V_{22}＝{B}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}			
3					
4					
5					

S→AB|BC, A→BA|a, B→CC|b, C→AB|a

i=3 求V_{32}: 即推出ab的变元集合。

k=1 a,b 这取决于 $V_{31}V_{41}$＝{A,C}{B}={AB,CB}。

因为有产生式S→AB C→AB， 所以 $V_{3\,2}$＝{S,C}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}		
3					
4					
5					

$S \to AB \mid BC$， $A \to BA \mid a$， $B \to CC \mid b$， $C \to AB \mid a$

i=4 求V_{42}：即推出**ba**的变元集合。

k=1 b,a 这取决于 $V_{41}V_{51}$＝{B}{A,C}={BA,BC}。

因为有产生式$A \to BA$ $S \to BC$， 所以 V_{42}＝{S,A}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3					
4					
5					

S→AB|BC, A→BA|a, B→CC|b, C→AB|a

j=3　i=1,2,3。

i=1　求V_{13}：　即推出baa的变元集合。　k＝1,2

k=1　b,aa 取决于 $V_{11}V_{22}$＝{B}{B}={BB}。无变元推出BB

k=2　ba,a 取决于 $V_{12}V_{31}$＝{S,A}{A,C}={SA,SC,AA,AC}，

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ				
4					
5					

无推出这些符号的产生式。

所以最后的V_{13}＝Φ。

S→AB|BC, A→BA|a, B→CC|b, C→AB|a

i=2 求V_{23}：即推出aab的变元集合。 k＝1,2

k=1 a,ab 取决于 $V_{21}V_{32}$＝{A,C}{S,C}= {AS,AC,CS,CC}

因为只有B可以推出CC。所以B∈V_{23}。

k=2 aa,b 取决于 $V_{22}V_{41}$＝{B}{B}={BB} 无变元推出BB。

所以最后的V_{23}＝{B}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ	{B}			
4					
5					

$S \to AB|BC,\ A \to BA|a,\ B \to CC|b,\ C \to AB|a$

i=3 求V_{33}：即推出**aba**的变元集合。 **k＝1,2**

k=1 a,ba 取决于 $V_{31}V_{42}=\{A,C\}\{S,A\}=\{AS,AA,CS,CA\}$

没有可以推出这些符号的产生式。

k=2 ab,a取决于 $V_{32}V_{51}=$ $\{S,C\}\{A,C\}=\{SA,SC,CA,CC\}$。

因为只有B可以推出CC。所以$B \in V_{33}$

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ	{B}	{B}		
4					
5					

所以最后得：

$V_{33}=\{B\}$。

$S \to AB|BC$， $A \to BA|a$， $B \to CC|b$， $C \to AB|a$

j=4 i=1,2 x=baaba

i=1 求V_{14}： 即推出**baab**的变元集合。 **k＝1,2,3**

k=1 baab被分成**b,aab**两个子串，

这取决于 $V_{11}V_{23}$＝{B}{B}={BB}。

没有可以推出**BB**的产生式。

k=2 baab被分成**ba,ab**两个子串，

这取决于 $V_{12}V_{32}$＝{S,A}{S,C} ={SS,SC,AS,AC}。

没有可以推出这些符号的产生式。

k=3 baab被分成**baa,b**两个子串，

这取决于 $V_{13}V_{41}$＝Φ{S,A}=Φ。

所以最后的V_{14}＝Φ

所以最后的$V_{1\,4} = \Phi$

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ	{B}	{B}		
4	Φ				
5					

$S \to AB|BC$, $A \to BA|a$, $B \to CC|b$, $C \to AB|a$

x=baaba

i=2 求V_{24}： 即推出**aaba**的变元集合， **k＝1,2,3**

k=1 aaba 被分成**a,aba**两个子串，

这取决于 $V_{21}V_{33}=\{A,C\}\{B\}=\{AB,CB\}$。

因为只有**S,C**都可以推出**AB**。所以$S,C \in V_{24}$。

k=2 aaba被分成 **aa,ba**两个子串，

这取决于 $V_{22}V_{42}=\{B\}\{S,A\}=\{BS,BA\}$。

因为只有**A**都可以推出**BA**。 所以$A \in V_{24}$。

k=3 aaba被分成**aab,a**两个子串，

这取决于 $V_{23}V_{51}=\{B\}\{A,C\}=\{BA,BC\}$。

因为有产生式$A \to BA$ $S \to BC$, 所以$A,S \in V_{24}$。

所以最后的$V_{24}=\{S,A,C\}$。

最后的$V_{2\,4}=\{S,A,C\}$。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ	{B}	{B}		
4	Φ	{S,A,C}			
5					

S→AB|BC， A→BA|a， B→CC|b， C→AB|a　　x=baaba

j=5 i=1 求V_{15}： 即推出baaba的变元集合， k＝1,2,3,4

k=1 baaba被分成b,aaba两个子串，

这取决于 $V_{11}V_{24}$＝{B}{S,A,C}={BS,BA,BC}。

因为有产生式A→BA S→BC, 所以A,S∈V_{15}。

k=2 baaba被分成ba,aba两个子串，

这取决于 $V_{12}V_{33}$＝{S,A}{B} ={SB,AB}。

因为有产生式S→AB C→AB, 所以S,C∈V_{15}。

k=3 baaba被分成baa,ba两个子串，

这取决于 $V_{13}V_{42}$＝Φ{S,A}=Φ。

k=4 baaba被分成baab,a两个子串，

这取决于 $V_{14}V_{51}$＝Φ{A,C}=Φ。

所以最后的V_{15}＝{S,A,C}。

由于S∈V_{15}，所以baaba∈L(G)。

最后得V_{15}＝{S,A,C}。

x	b	a	a	b	a
V_{ij} i / j	1	2	3	4	5
1	{B}	{A,C}	{A,C}	{B}	{A,C}
2	{S,A}	{B}	{S,C}	{S,A}	
3	Φ	{B}	{B}		
4	Φ	{S,A,C}			
5	{S,A,C}				

由于S∈V_{15}，所以baaba∈L(G)。

作业题

1．求证下面语言L不是CFL，

$L=\{a^k \mid k$是个素数$\}$。

3.8 CFL运算的封闭性

CFL对并、乘积、闭包运算满足封闭性，但是对交运算和补运算就不满足封闭性。

定理3-8.1 CFL对并、乘积、闭包运算满足封闭性。即如果L_1、L_2是CFL，则$L_1 \cup L_2$、L_1L_2以及L_1^*也是CFL。

证明：设L_1、L_2分别是由CFG G_1=（V_{N1}, V_{T1}, P_1, S_1）、G_2=（V_{N2}, V_{T2}, P_2, S_2）产生的语言。

并假设$V_{N1} \cap V_{N2} = \Phi$，$S_3$, S_4, $S_5 \notin (V_{N1} \cup V_{N2})$。

1．为了接收语言$L_1 \cup L_2$，构造文法

G_3=（$V_{N1} \cup V_{N2} \cup \{S_3\}$, $V_{T1} \cup V_{T2}$, P_3, S_3),

其中$P_3 = P_1 \cup P_2 \cup \{S_3 \to S_1, S_3 \to S_2\}$。

下面证明$L(G_3)=L(G_1)\cup L(G_2)=L_1\cup L_2$。

1）先证明$L(G_1)\cup L(G_2)\subseteq L(G_3)$。

任取 $w\in L(G_1)\cup L(G_2)$，则$w\in L(G_1)$或者 $w\in L(G_2)$。

如果$w\in L(G_1)$，则G_1中有派生$S_1\Rightarrow^* w$。 且此派生使用的都是P_1中的产生式。而P_3中有$S_3\rightarrow S_1$，又$P_1\subseteq P_3$，所以G_3中有派生：$S_3\Rightarrow S_1\Rightarrow^* w$， 于是$w\in L(G_3)$。

如果$w\in L(G_2)$，类似可以推出$w\in L(G_3)$。

所以$L(G_1)\cup L(G_2)\subseteq L(G_3)$。

2）再证$L(G_3) \subseteq L(G_1) \cup L(G_2)$。

任取$w \in L(G_3)$，即G_3有派生$S_3 \Rightarrow^* w$，根据$P_3 = P_1 \cup P_2 \cup \{S_3 \to S_1, S_3 \to S_2\}$，w的第一步派生用的产生式必是$S_3 \to S_1$或者$S_3 \to S_2$，于是$S_3 \Rightarrow S_1 \Rightarrow^* w$，或者$S_3 \Rightarrow S_2 \Rightarrow^* w$，又因为$S_3 \notin V_{N1} \cup V_{N2}$，$V_{N1} \cap V_{N2} = \Phi$，所以w的第二步及其以后的派生要么用的都是$P_1$中的产生式，要么用的都是$P_2$中的产生式，即有$S_1 \Rightarrow^* w$，或者$S_2 \Rightarrow^* w$，所以$w \in L(G_1)$或者$w \in L(G_2)$。

故$w \in L(G_1) \cup L(G_2)$，

所以$L(G_3) \subseteq L(G_1) \cup L(G_2)$。

最后得 $L(G_3) = L(G_1) \cup L(G_2)$。

2. 为接收语言L_1与L_2的乘积L_1L_2， 构造文法

$G_4=(V_{N1}\cup V_{N2}\cup\{S_4\}, V_{T1}\cup V_{T2}, P_4, S_4)$，

其中$P_4=P_1\cup P_2\cup\{S_4\to S_1S_2\}$。

容易证明$L(G_4)=L(G_1)L(G_2)=L_1L_2$。（留给读者证明）

3. 为接收语言$L_1{}^*$，构造文法

$G_5=(V_{N1}\cup\{S_5\}, V_{T1}, P_5, S_5)$，

其中$P_5=P_1\cup\{S_5\to S_1S_5，S_5\to\varepsilon\}$。

下面证明$L(G_5)=(L(G_1))^*=L_1{}^*$。

1） 先证明$L_1{}^*\subseteq L(G_5)$

任取$w\in L_1{}^*$，因为$L_1{}^*=L_1{}^0\cup L_1\cup L_1{}^2\cup\ldots\cup L_1{}^n\cup\ldots$，

其中$L_1{}^0=\{\varepsilon\}$。

如果$w\in L_1{}^0$，则$w=\varepsilon$ 。因为P_5中有$S_5\to\varepsilon$，所以

$\varepsilon\in L(G_5)$。

如果$w \in L_1^n$，$(n \geqslant 1)$，则可将w写成$w=w_1w_2 \ldots w_n$，使得每个$w_i \in L_1 (1 \leqslant i \leqslant n)$， 于是$G_1$中有$S_1 \Rightarrow^* w_i$。又因为$P_1 \subseteq P_5$，所以这个派生在$G_5$中也可以实现，即$G_5$中有派生$S_1 \Rightarrow^* w_i$。于是$G_5$中有派生：

$$S_5 \Rightarrow S_1S_5 \Rightarrow^* S_1^nS_5 \Rightarrow S_1^n \Rightarrow^* w_1S_1^{n-1} \Rightarrow^* w_1w_2S_1^{n-2} \Rightarrow^* w_1w_2 \ldots w_{n-1} S_1 \Rightarrow^* w_1w_2 \ldots w_{n-1}w_n = w,$$

所以$w \in L(G_5)$。因而$L_1^* \subseteq L(G_5)$。

2）再证明$L(G_5) \subseteq L_1^*$

任取$w \in L(G_5)$，如果w的派生用的是$S_5 \to \varepsilon$，则$w=\varepsilon$，而$\varepsilon \in L_1^0$，$L_1^0 \subseteq L_1^*$，所以$\varepsilon \in L_1^*$。

如果$w \neq \varepsilon$，则w的派生必先用$S_5 \to S_1S_5$，而$S_5 \notin V_{N1}$，所以以后必用$S_5 \to \varepsilon$以消去变元S_5，故G_5中有派生：

$S_5 \Rightarrow S_1S_5 \Rightarrow^* S_1^nS_5 \Rightarrow S_1^n \Rightarrow^* w$，于是可将w写成$w=w_1w_2...w_n$，使得每个$w_i$有$S_1 \Rightarrow^* w_i$ $(1 \leqslant i \leqslant n)$，根据$P_5$的构成可知，这些派生所用的产生式实际上都是$P_1$中的产生式，所以$G_1$中有派生$S_1 \Rightarrow^* w_i$，所以$w_i \in L_1$ $(1 \leqslant i \leqslant n)$，

于是$w_1w_2...w_n \in L_1^n$，而$L_1^n \subseteq L_1^*$，所以$w_1w_2...w_n \in L_1^*$，即$w \in L_1^*$。所以$L(G_5) \subseteq L_1^*$。

最后得$L(G_5) = L_1^*$。

由于G_3、G_4、G_5都是CFG，所以$L_1 \cup L_2$、L_1L_2以及L_1^*都是CFL。 **CFL对并、乘积、闭包运算满足封闭性**

但是，CFL对交运算和补运算则不满足封闭性。

请看下面例子，在3.7节中，曾经用CFL的泵作用引理证明语言$L=\{a^ib^ic^i|i\geqslant 1\}$不是CFL。现给定CFL L_1、L_2如下：$L_1=\{a^ib^ic^j|i,j\geqslant 1\}$

$L_2=\{a^ib^jc^j|i,j\geqslant 1\}$

因为有CFG G_1和G_2，使得$L(G_1)=L_1$ $L(G_2)=L_2$，其中

$G_1=(\{A,B,S_1\},\{a,b,c\},P_1,S_1)$，

$P_1=\{S_1\to AB, A\to aAb, A\to ab, B\to cB, B\to c\}$。

$G_2=(\{D,E,S_2\},\{a,b,c\},P_2,S_2)$，

$P_2=\{S_2\to DE, D\to aD, D\to a, E\to bEc, E\to bc\}$。

显然，在G_1中有派生：

$S_1\Rightarrow AB\Rightarrow^* a^{i-1}Ab^{i-1}B\Rightarrow a^ib^iB\Rightarrow^* a^ib^ic^{j-1}B \Rightarrow a^ib^ic^j$，

G_2中有派生：$S_2\Rightarrow DE\Rightarrow^* a^{i-1}DE\Rightarrow a^iE\Rightarrow^* a^ib^{j-1}Ec^{j-1}\Rightarrow a^ib^jc^j$，

所以$L(G_1)=L_1$ $L(G_2)=L_2$。

$L=\{a^ib^ic^i|i\geqslant1\}$不是CFL。

$L_1=\{a^ib^ic^j|i,j\geqslant1\}$ 是CFL。

$L_2=\{a^ib^jc^j|i,j\geqslant1\}$ 是CFL。

而$L_1\cap L_2=L$，而L不是CFL，所以$L_1\cap L_2$也不是CFL，这说明了**CFL对交运算不满足封闭性**。

进而，我们可以应用集合的底－摩根定律，说明CFL对补运算也不封闭。

设L_1与L_2是CFL，如果CFL对补运算封闭，又CFL对并运算封闭，则因为

$$L_1\cap L_2=\overline{\overline{L_1}\cup\overline{L_2}}$$

可以得到$L_1\cap L_2$也是CFL，这与前边的结论矛盾。

所以**CFL对补运算不封闭**。

小结

第3章 上下文无关文法与下推自动机

3.1 上下文无关文法**(CFG)**的派生树**(**推导树**)**
3.2 上下文无关文法的简化

3.3 上下文无关文法的**Chomsky**范式
3.4 CFG的**Greibach**范式**(GNF)**
3.5下推自动机**(PDA)**

3.6 PDA与**CFG**之间的等价性
3.7 CFL的有关判定问题
3.8 CFL运算的封闭性

习题．求证下面语言L不是CFL，

$L=\{a^{k} \mid k$是个素数$\}$。

证明：(1) 假设L是CFL。

(2)令n是L满足CFL泵作用引理常数。

(3)取$z=a^{m}$，$m\geqslant n$ 且m是个素数。$|z|=m\geqslant n$, 根据CFL的泵作用引理，可将z写成 $z=uvwxy$ 形式，其中$|vwx|\leqslant n$, $|vx|\geqslant 1$, 且对任何$i\geqslant 0$ 有 $uv^{i}wx^{i}y\in L$。

(4)令$u=a^{n1}$, $v=a^{n2}$, $w=a^{n3}$, $x=a^{n4}$, $y=a^{n5}$,于是$|vx|=n_2+n_4\geqslant 1$,

$n_1+n_2+n_3+n_4+n_5=m$， $z=uvwxy=a^{n1+n2+n3+n4+n5}=a^{m}$,

$uv^{i}wx^{i}y=a^{n1+in2+n3+in4+n5}=a^{m+(i-1)n2+(i-1)n4}=a^{m+(i-1)(n2+n4)}$

取$i=m+1$,则

$uv^{m+1}wx^{m+1}y=a^{m+(m+1-1)(n2+n4)}=a^{m+m(n2+n4)}=a^{m(1+n2+n4)}$

由于$n_2+n_4\geqslant 1$，故$1+n_2+n_4\geqslant 2$, 而$m\geqslant 2$, 所以$m(1+n_2+n_4)$不是素数，故 $uv^{m+1}wx^{m+1}y\notin L$，产生矛盾。所以L不是CFL。

例 3.1.2 设 $G=(W,\Sigma,R,S)$，其中

$$W=\{S,A,N,V,P\}\cup\Sigma$$
$$\Sigma=\{\text{Jim},\text{big},\text{green},\text{cheese},\text{ate}\}$$
$$\begin{aligned}R=\{&P\to N,\\ &P\to AP,\\ &S\to PVP,\\ &A\to \text{big},\\ &A\to \text{green},\\ &N\to \text{cheese},\\ &N\to \text{Jim},\\ &V\to \text{ate}\}\end{aligned}$$

在这里，把 G 设计成一个表示部分英语的文法，S 代表句子，A 代表形容词，N 代表名词，V 代表动词，P 代表短语。下面是 $L(G)$ 中的几个字符串：

Jim ate cheese

big Jim ate green cheese

big cheese ate Jim

不幸的是，下面也是 $L(G)$ 中的字符串

big cheese ate green green big green big cheese

green Jim ate green big Jim ◇

BACK