

第5章 自底向上的分析

一、自顶向下分析方法

二、自底向上分析方法

自底向上分析的一般过程

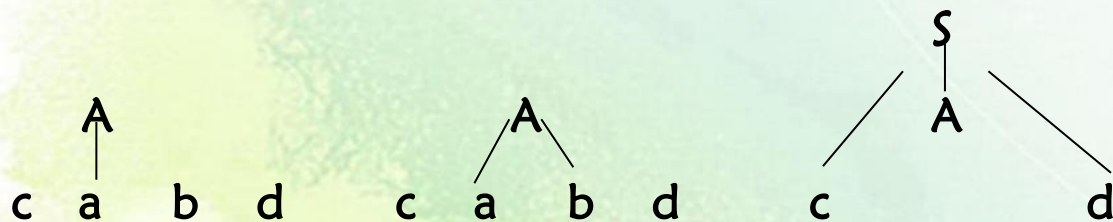
- 从所要分析的**终结字符串**开始进行归约；
- 每一步归约是在当前串中找到与某个产生式的右部相匹配的子串，然后将该子串用这一产生式的左部非终结符进行替换；如果找不到这样的子串，则回退到上一步归约前的状态，选择不同的子串或不同的产生式重试；
- 重复上一步骤，直到**归约至文法开始符号**；
- 如果不存在任何一个这样的归约，则表明该终结字符串存在语法错误

自底向上分析的问题分析

例：文法 $G[S]$ ：

$$S \rightarrow cAd$$
$$A \rightarrow ab$$
$$A \rightarrow a$$

识别输入串 $w=cabd$ 是否该文法的句子。



归约过程构造的推导： $cAd \Rightarrow cabd$ $S \Rightarrow cAd$

自底向上分析的问题分析

自底向上分析中的不确定性

- 在每一步归约中，选择哪一个产生式以及匹配哪一个位置上的子串都可能是不确定的；
- 这些不确定性导致分析过程会有很高的复杂性；

自底向上分析法是如何工作的？

$G[S]=\{$

$S \rightarrow gAhBkC$

$A \rightarrow ab$

$B \rightarrow cd$

$C \rightarrow ef$

$\}$

用自底向上法分析符号串（如串gabhc dkef）

自底向上的分析

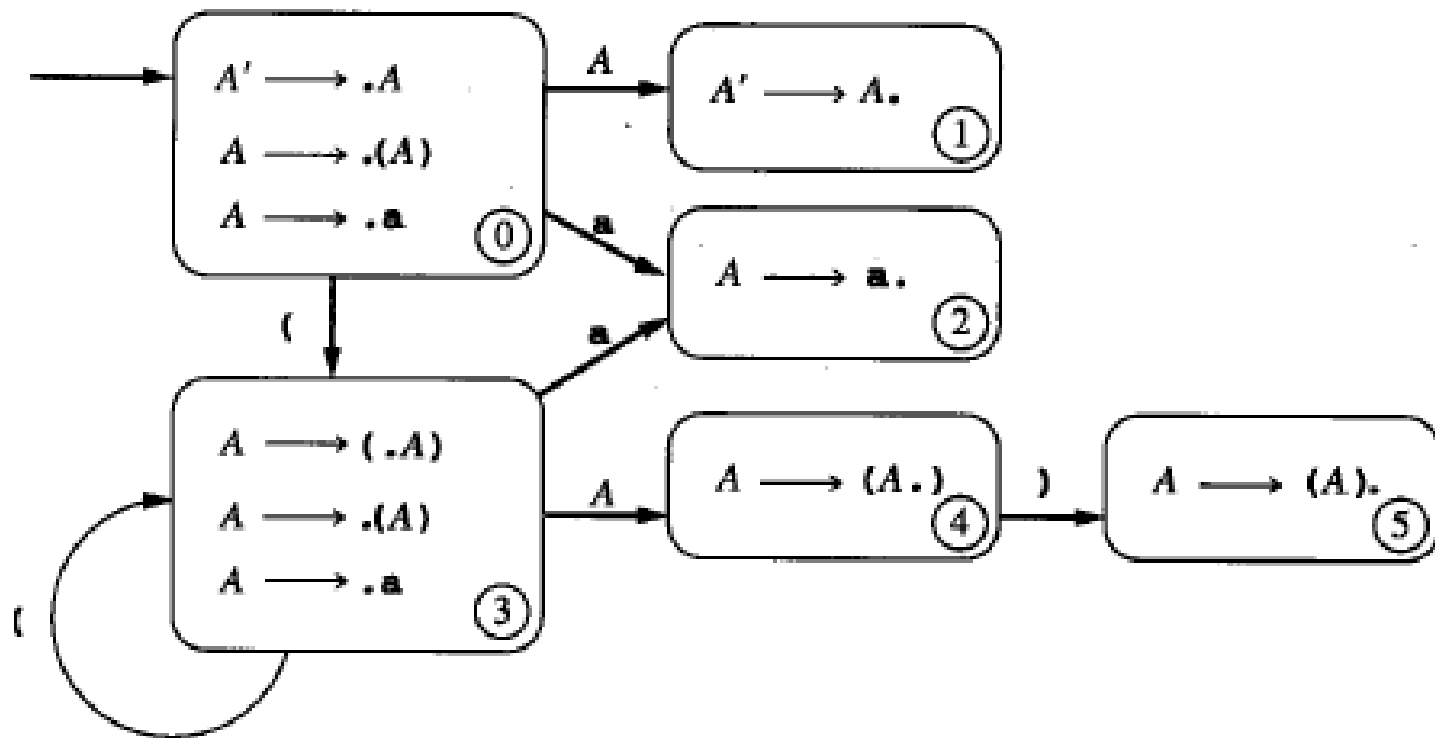
问题:

$G[A']$:

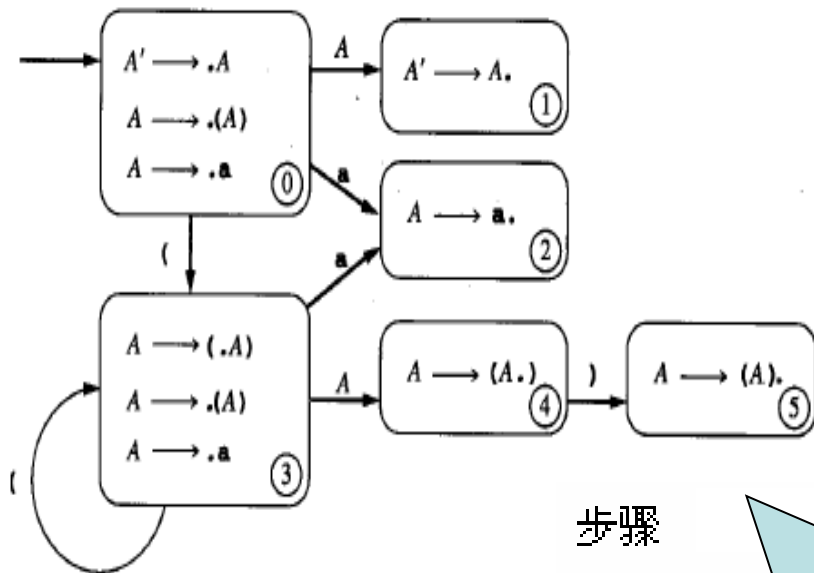
$A' \rightarrow A$

$A \rightarrow (A) \mid a$

分析串: $((a))$



分析串: ((a))



步骤

分析

输入

LR(0)分析法

1						a))	\$
2)))	\$
3						a))	\$
4)))	\$
5)))	\$
6)))	\$
7	\$	0		3	A	4			
8	\$	0	(3	A	4)	5	
9	\$	0	A	1					

LR(k) 分析法

- “L” 是指从左至右扫描输入符号串；
 - “R” 是指构造一个最右推导的逆过程；
 - “k” 是指向前看的输入符号的个数。
- LR分析方法是当前最广义的无回溯的“移进-归约”方法。根据栈中的符号串和向右顺序查看输入串的 $k(k \geq 0)$ 个符号，就能唯一确定分析器的动作是移进还是归约，以及用哪个产生式进行归约。
 - LR(k)分析法knuth于1965年首先提出来的。

LR(0) 分析在计算机上的实现

(1)LR(0)的存储结构

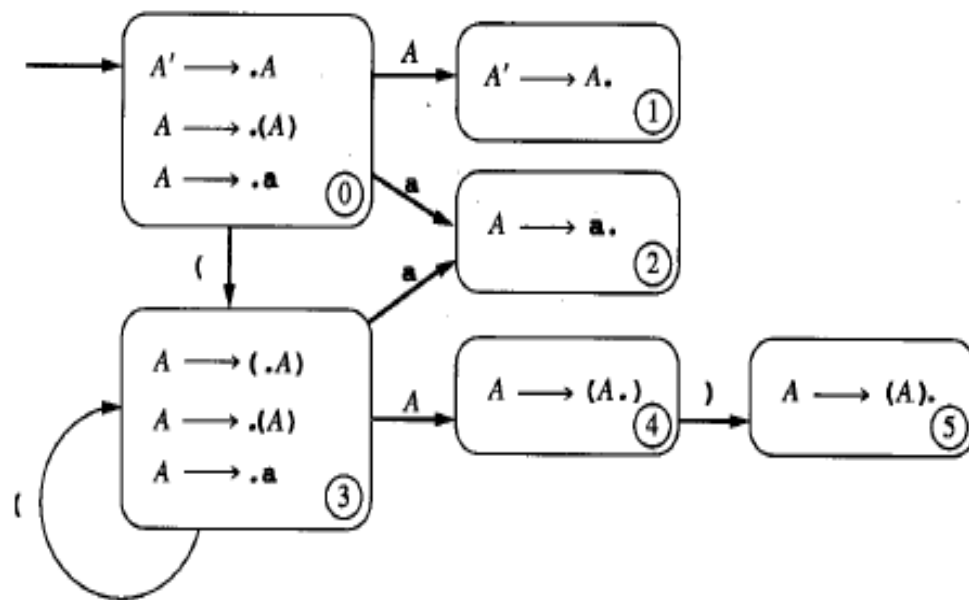
(2)LR(0)的分析算法

LR(0)项DFA的存储结构

(1) 邻接矩阵

(2) 链表表

(3) 新二维表存储——LR(0)分析表



状 态	动 作	规 则	输 入			Goto
			(a)	A
0	移进		3	2		1
1	归约	$A' \rightarrow A$				
2	归约	$A \rightarrow a$				
3	移进		3	2		4
4	移进				5	
5	归约	$A \rightarrow (A)$				

状 态	动 作	规 则	输 入			Goto
			(a)	A
0	移进		3	2		1
1	归约	$A' \rightarrow A$				
2	归约	$A \rightarrow a$				
3	移进		3	2		4
4	移进				5	
5	归约	$A \rightarrow (A)$				

下面讲述LR(0)分析算法是如何工作的。例子：用LR(0)分析方法分析上述文法的一串((a))。

步骤	分析栈	输入	动作
1	\$ 0	((a))\$	移进
2	\$ 0 (3	(a))\$	移进
3	\$ 0 (3 (3	a))\$	移进
4	\$ 0 (3 (3 a 2))\$	用 $A \rightarrow a$ 归约
5	\$ 0 (3 (3 A 4))\$	移进
6	\$ 0 (3 (3 A 4) 5)\$	用 $A \rightarrow (A)$ 归约
7	\$ 0 (3 A 4)\$	移进
8	\$ 0 (3 A 4) 5	\$	用 $A \rightarrow (A)$ 归约
9	\$ 0 A 1	\$	接受

一个新例子

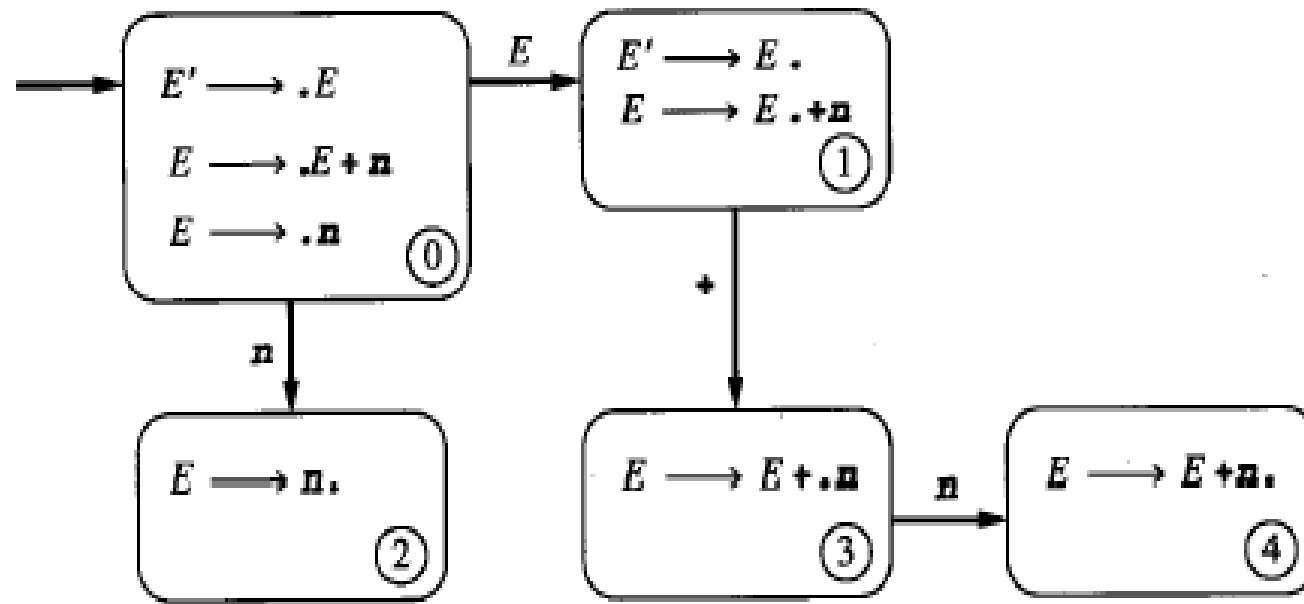
$G[E]:$

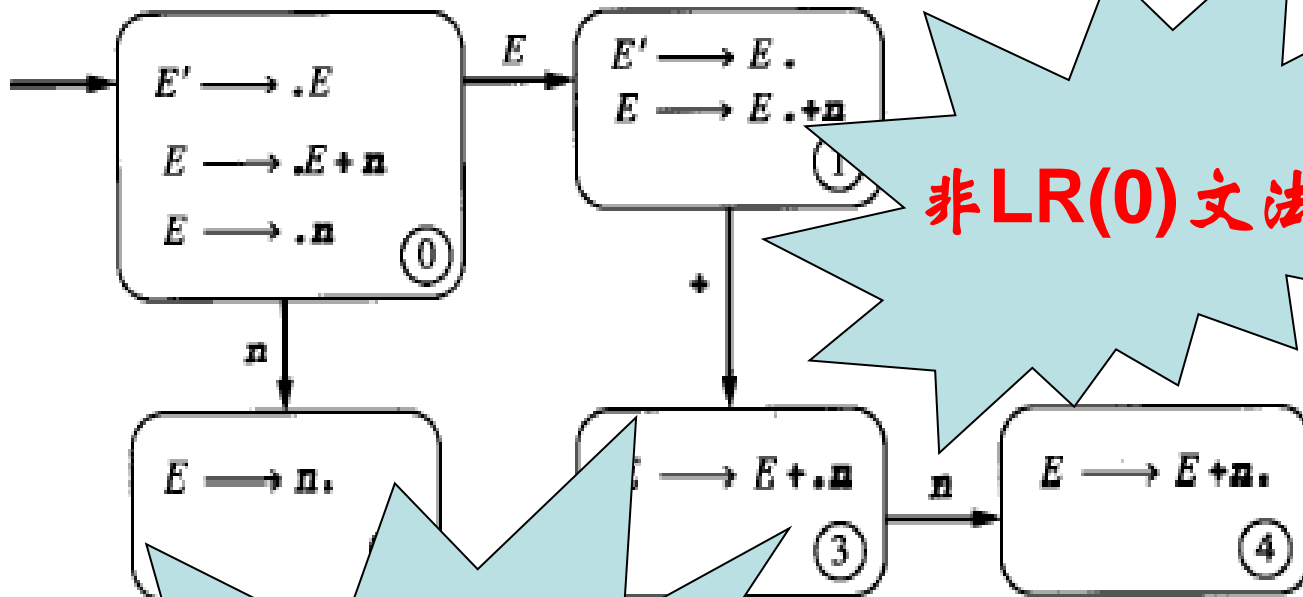
$E \rightarrow E + n \mid n$

试用自底向上分析方法分析输入串: $n+n+n$

(1) 文法的扩充

$$E' \rightarrow E$$
$$E \rightarrow E + n \mid n$$





非LR(0)文法

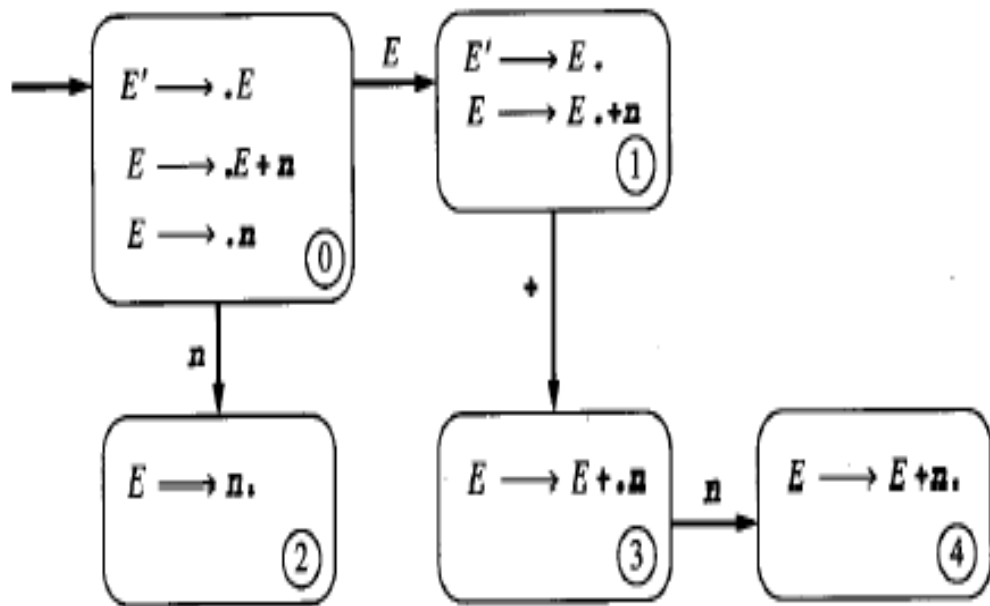
无法分析下去!

如何解决?

输入串:

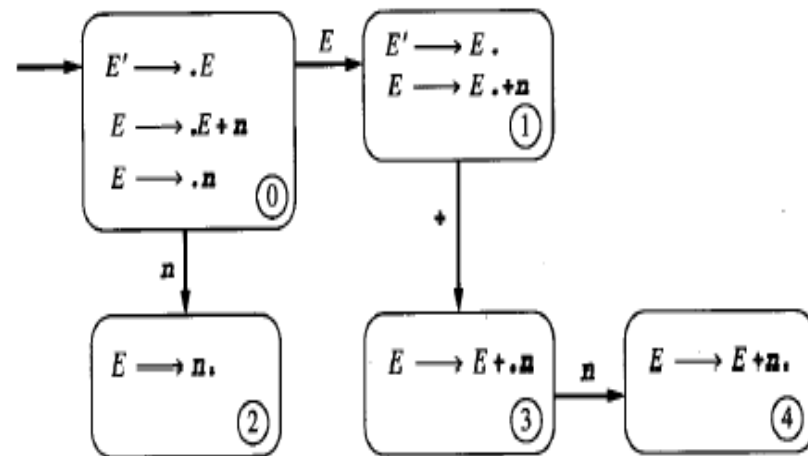
SLR(1) 分析

- **简单LR(1)分析**
- SLR(1)分析还是使用LR(0)项目集DFA。
- 只是在构造分析表时才考虑超前查看的符号。



- 重点考察状态1
- $\text{Follow}(E') = \{ \$ \}$
- $E \rightarrow E.+n$ 分隔符后的符号是终结符号+,因此在状态1时,输入串当前符号为+则移进,而\$则规约.

- SLR(1) 分析表如下表5-5:



状 态	输 入			Goto
	n	+	\$	
0	s2			1
1		s3	接受	
2		r($E \rightarrow n$)	r($E \rightarrow n$)	
3	s4			
4		r($E \rightarrow E + n$)	r($E \rightarrow E + n$)	

注意: 移进由表项中的字母s指出, 归约由字母r指出。

状 态	输 入			Goto
	n	+	\$	E
0	s2			1
1		s3	接受	
2		r($E \rightarrow n$)	r($E \rightarrow n$)	
3	s4			
4		r($E \rightarrow E + n$)	r($E \rightarrow E + n$)	

步骤	分析栈	输入	动作
1	\$ 0	n + n + n \$	移进2
2	\$ 0 n 2	+ n + n \$	用 $E \rightarrow n$ 归约
3	\$ 0 E 1	+ n + n \$	移进3
4	\$ 0 E 1 + 3	n + n \$	移进4
5	\$ 0 E 1 + 3 n 4	+ n \$	用 $E \rightarrow E + n$ 归约
6	\$ 0 E 1	+ n \$	移进3
7	\$ 0 E 1 + 3	n \$	移进4
8	\$ 0 E 1 + 3 n 4	\$	用 $E \rightarrow E + n$ 归约
9	\$ 0 E 1	\$	接受

括号匹配的问题

在一些表达式中通常会允许括号出现嵌套的现象,现在请采用LR方法检验下面式子的括号是否匹配:

$$()((\dots)\dots)(\dots)\dots$$

文法:

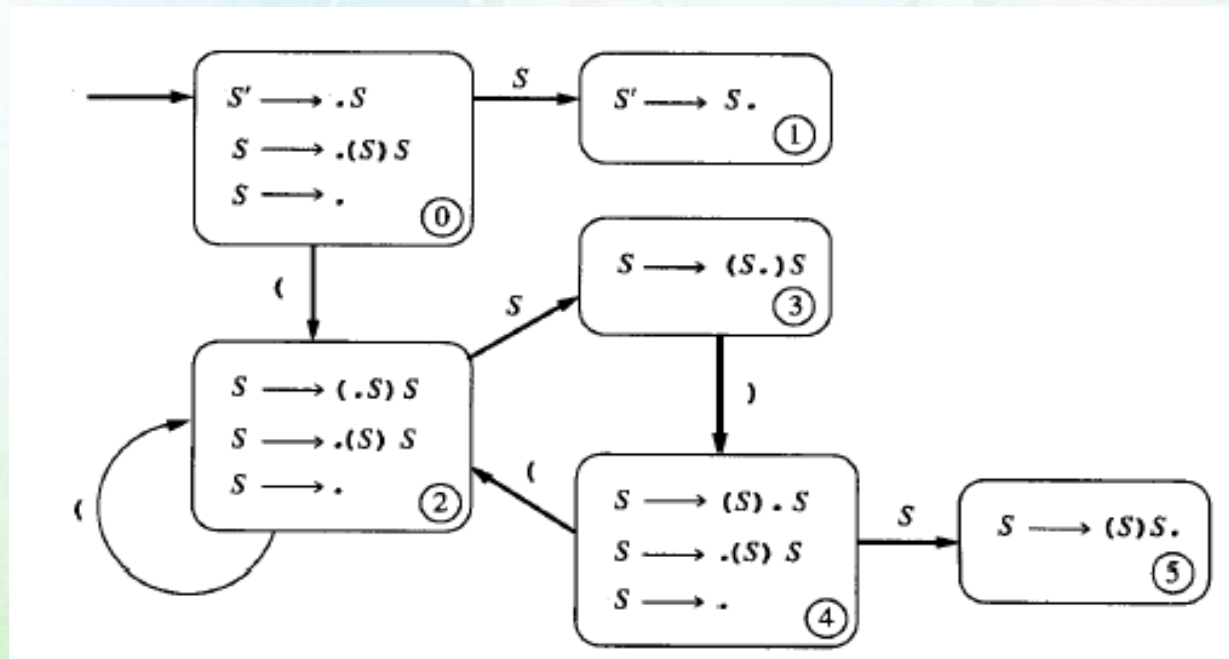
$$S' \rightarrow S$$

$$S \rightarrow (S) S \mid \varepsilon$$

文法:

$$S' \rightarrow S$$

$$S \rightarrow (S) S \mid \varepsilon$$

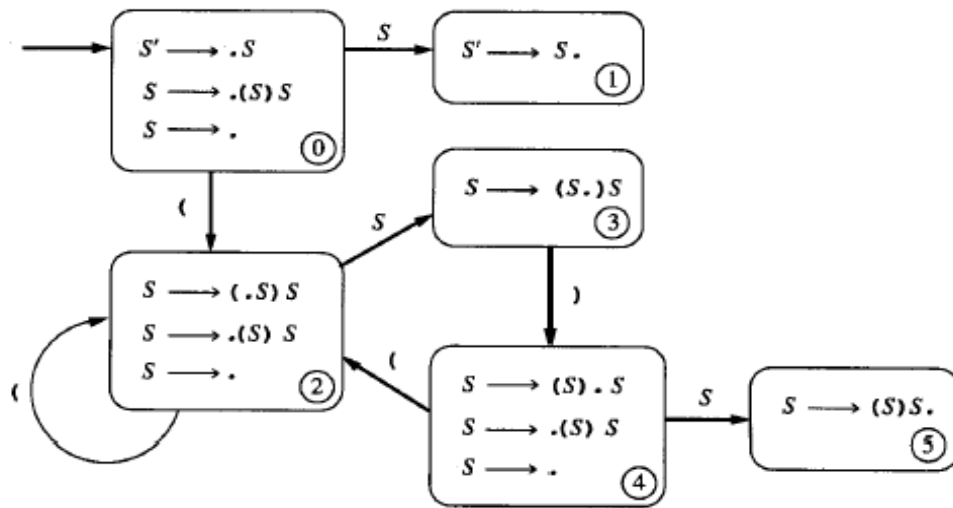


重点考察状态0,状态2,状态4:

(1) 规约项 $S \rightarrow \cdot$ 其 $\text{Follow}(S) = \{ \$,) \}$

(2) 移进项 $S \rightarrow \cdot (S) S$ 的下一个符号为 (

Follow (S) = { \$,) }



状 态	输 入			Goto
	()	\$	S
0	s2	r ($S \rightarrow \bullet$)	r ($S \rightarrow \bullet$)	1
1			接受	
2	s2	r ($S \rightarrow \bullet$)	r ($S \rightarrow \bullet$)	3
3		s4		
4	s2	r ($S \rightarrow \bullet$)	r ($S \rightarrow \bullet$)	5
5		r ($S \rightarrow (S) S$)	r ($S \rightarrow (S) S$)	

状 态	输 入			Goto
	()	\$	S
0	s2	$r(S \rightarrow \mathbf{e})$	$r(S \rightarrow \mathbf{e})$	1
1			接受	
2	s2	$r(S \rightarrow \mathbf{e})$	$r(S \rightarrow \mathbf{e})$	3
3		s4		
4	s2	$r(S \rightarrow \mathbf{e})$	$r(S \rightarrow \mathbf{e})$	5
5		$r(S \rightarrow (S) S)$	$r(S \rightarrow (S) S)$	

步骤	分析栈	输入	动作
1	\$ 0	() () \$	移进2
2	\$ 0 (2) () \$	用 $S \rightarrow e$ 归约
3	\$ 0 (2 S 3) () \$	移进4
4	\$ 0 (2 S 3) 4	() \$	移进2
5	\$ 0 (2 S 3) 4 (2) \$	用 $S \rightarrow e$ 归约
6	\$ 0 (2 S 3) 4 (2 S 3) \$	移进4
7	\$ 0 (2 S 3) 4 (2 S 3) 4	\$	用 $S \rightarrow e$ 归约
8	\$ 0 (2 S 3) 4 (2 S 3) 4 S 5		\$ 用 $S \rightarrow (S) S$ 归约
9	\$ 0 (2 S 3) 4 S 5	\$	用 $S \rightarrow (S) S$ 归约
10	\$ 0 S 1	\$	接受

SLR(1) 分析法的新问题1

例子：

if语句的文法：

$\text{statement} \rightarrow \text{if-stmt} \mid \text{other}$

$\text{if-stmt} \rightarrow \text{if (exp) statement} \mid \text{if (exp)}$
 $\text{statement else statement}$

$\text{exp} \rightarrow 0 \mid 1$

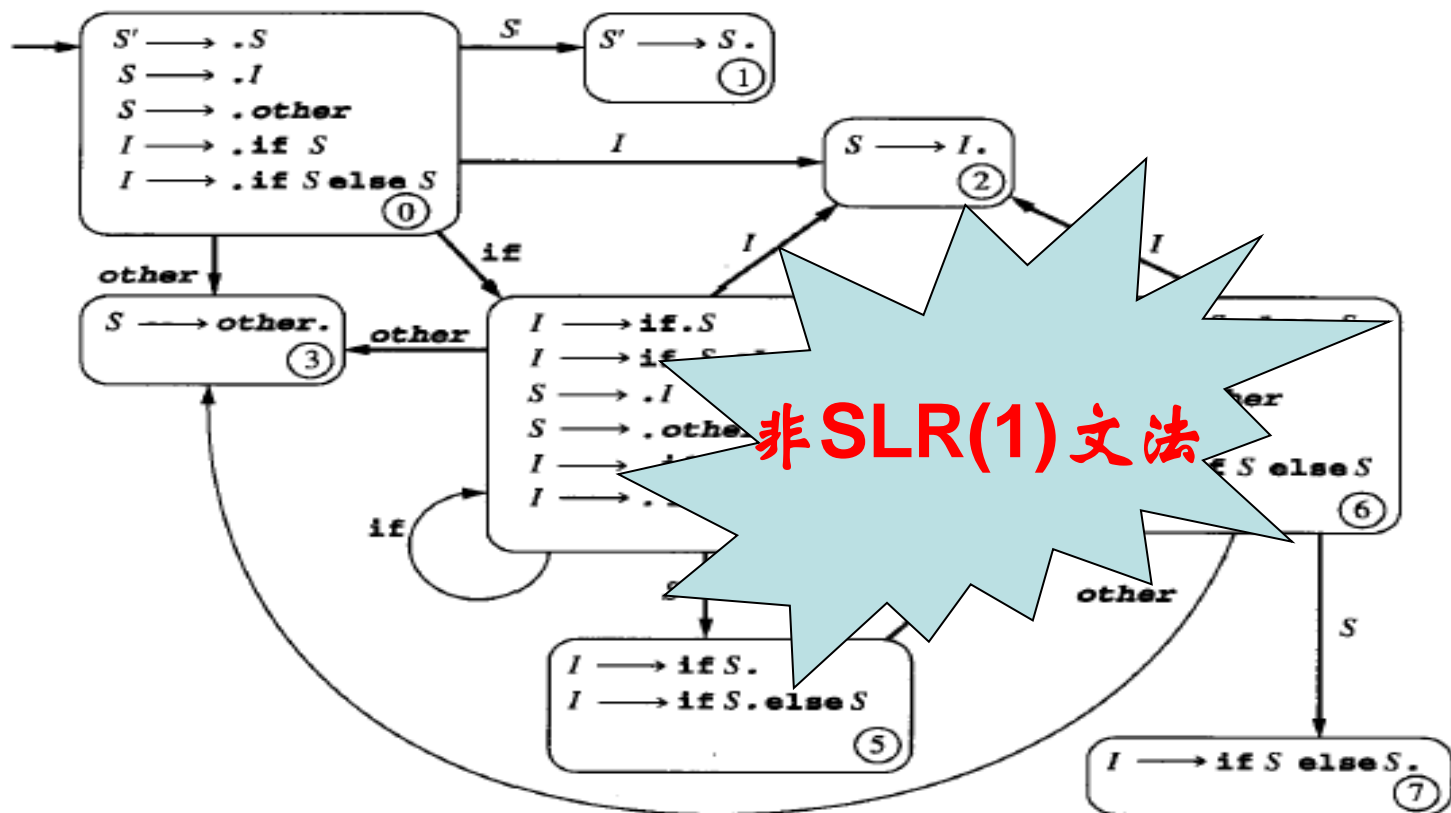
试用SLR(1)分析方法进行输入串的分析。

- 步骤:

(1) 为了处理上的简单, 我们先做文法的简化

$$S \rightarrow l \mid \text{other}$$
$$l \rightarrow \text{if } S \mid \text{if } S \text{ else } S$$

(2) 画出LR(0)项DFA图



- 状态5: 对于规约项: $\text{Follow}(I) = \{\$, \text{else}\}$
- 对于移进项的下一个符号为 **else**
- **问题**: 移进-归约冲突
- **解决方法**: 只做**移进**不做**归约**
- **——消除二义性的新方法**

为了在SLR(1)分析表中描述方便，我们对文法规则进行编号：

(1) $S \rightarrow /$

(2) $S \rightarrow other$

(3) $/ \rightarrow \text{if } S$

(4) $/ \rightarrow \text{if } S \text{ else } S$

[该分析表已删除了分析冲突部分]

状 态	输 入				Goto	
	if	Else	other	\$	S	/
0	s4		s3		1	2
1				接受		
2		r1		r1		
3		r2		r2		
4	s4		s3		5	2
5		<u>s6</u>		<u>r3</u>		
6	s4		s3		7	2
7		r4		r4		

SLR(1) 分析法的新问题2

文法：

$\text{stmt} \rightarrow \text{call-stmt} \mid \text{assign-stmt}$

$\text{call-stmt} \rightarrow \text{identifier}$

$\text{assign-stmt} \rightarrow \text{var} := \text{exp}$

$\text{var} \rightarrow \text{var} [\text{exp}] \mid \text{identifier}$

$\text{exp} \rightarrow \text{var} \mid \text{number}$

试采用SLR(1)分析方法进行分析。

- 步骤:

- (1) 先简化文法:

$S \rightarrow id \mid V := E$

$V \rightarrow id$

$E \rightarrow V \mid n$

- (2) 文法的扩充:

$S' \rightarrow S$

$S \rightarrow id$

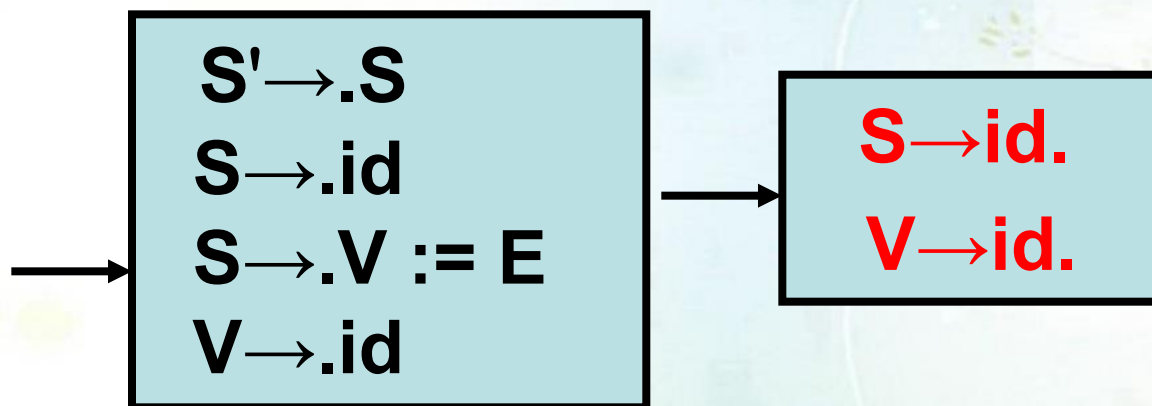
$S \rightarrow V := E$

$V \rightarrow id$

$E \rightarrow V$

$E \rightarrow n$

(3)构造DFA



$\text{Follow}(S) = \{\$ \}$ 和 $\text{Follow}(V) = \{:=, \$ \}$

如何解决? 问题的根源在哪里?

SLR(1)分析法的缺陷在于构造LR(0)项DFA时不考虑先行符号, 而在构造分析表的时候才加以考虑。

总结

(1)

$S \rightarrow I \mid \text{other}$

$I \rightarrow \text{if } S \mid \text{if } S \text{ else } S$

$I \longrightarrow \text{if } S .$

$I \longrightarrow \text{if } S . \text{else } S$

⑤

$\text{Follow}(I) = \{\$, \text{else}\}$

(2)

$S \rightarrow \text{id} \mid V := E$

$V \rightarrow \text{id}$

$E \rightarrow V \mid n$

$S \rightarrow \text{id} .$

$V \rightarrow \text{id} .$

SLR(1) 文法 (SLR(1) grammar)

- 当且仅当对于任何状态 s ，以下的两个条件：
 - 1) 对于在 s 中的任何项目 $A \rightarrow \alpha.X\beta$ ，当 X 是一个终结符，且 X 在 $\text{Follow}(B)$ 中时， s 中没有完整的项目 $B \rightarrow \gamma.$ 。 [移进-归约冲突]
 - 2) 对于在 s 中的任何两个完整项目 $A \rightarrow \alpha.$ 和 $B \rightarrow \beta.$ ， $\text{Follow}(A) \cap \text{Follow}(B)$ 为空。 [归约-归约冲突]
- 均满足时，文法为 SLR(1) 文法。

SLR(1)分析法的总结

- SLR(1)分析是LR(0)分析的一个简单但有效的扩展,
- SLR(1)分析的能力足以处理几乎所有实际的语言结构。
- 某些情况, SLR(1)分析能力并不太强

解决方法一

- 超前查看K个符号——SLR(k)文法
- 当 $k > 1$ 时，SLR(k)分析比SLR(1)分析更强大，但由于分析表的大小将按k的指数倍增长，所以它又要复杂许多。
- 上例不是简单SLR(1)文法而是SLR(2)文法。

解决方法二

在构造DFA的时候就超前考虑一个符号。

$$S \rightarrow id \mid V := E$$
$$V \rightarrow id$$
$$E \rightarrow V \mid n$$

解决方法二

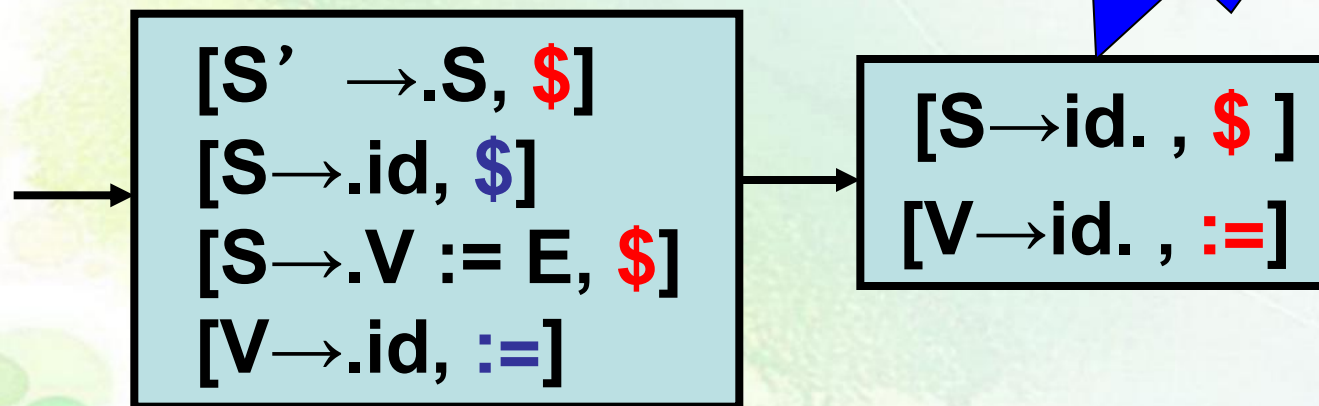
在构造DFA的时候就超前考虑一个符号。

$S \rightarrow id \mid V := E$

$V \rightarrow id$

$E \rightarrow V \mid n$

LR(1)分析法

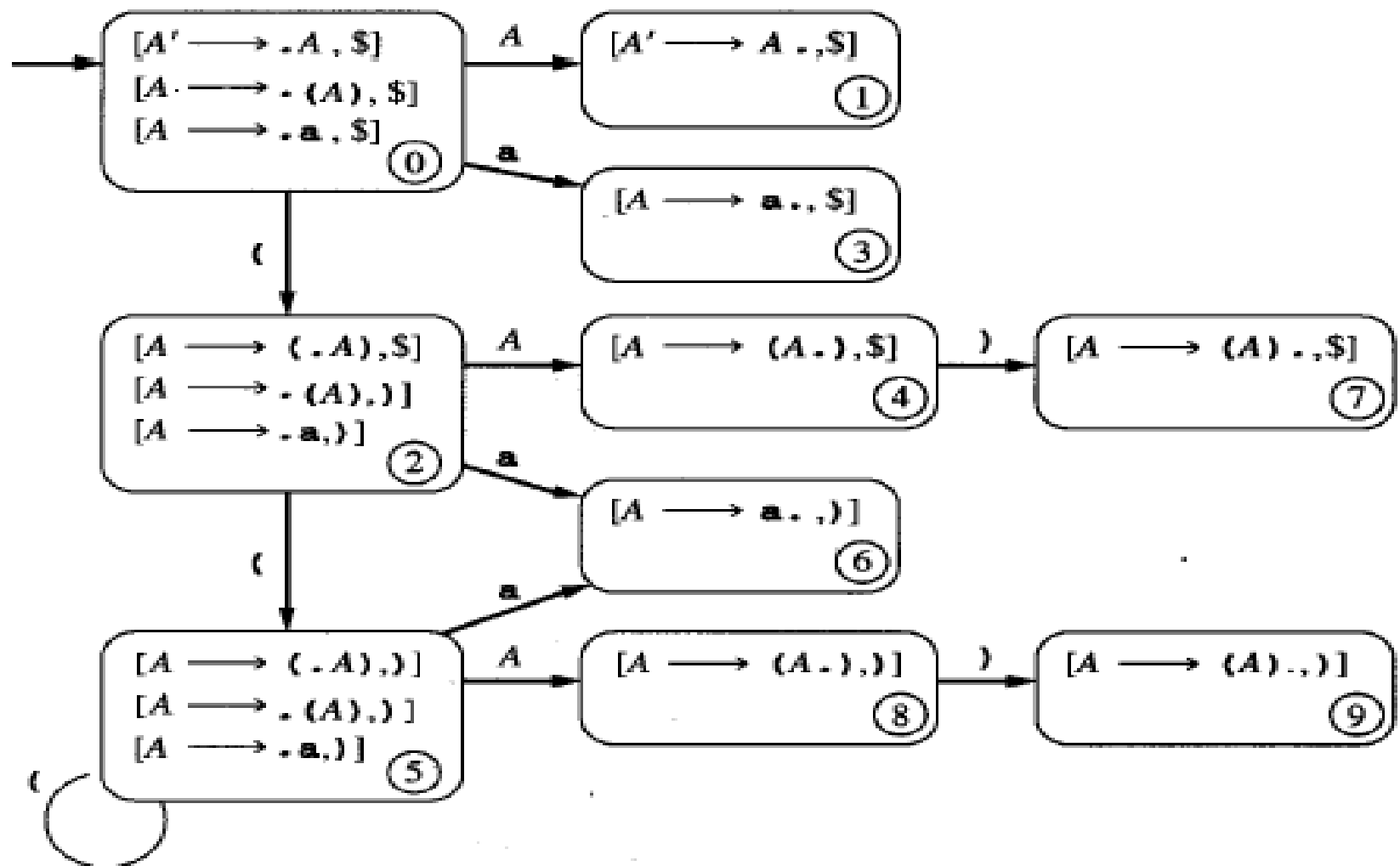


• LR(1)项目DFA的构造示例

文法: $A \rightarrow (A) \mid a$

- (1) 首先改写为扩充文法

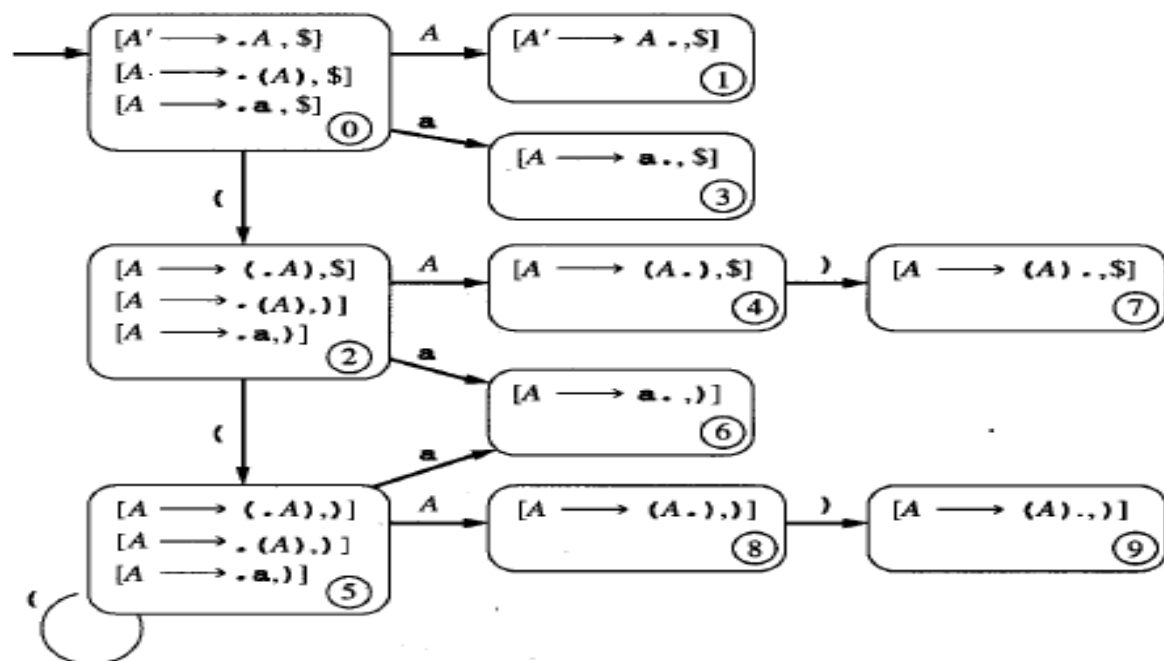
$A' \rightarrow A$ $A \rightarrow (A) \mid a$



- LR(1)项DFA的存储结构

- 二维表

状 态	输 入					Goto
	(a)	\$		A
0	s2	s3		<u>接受</u>		1
1						
2	s5	s6				4
3				<u>r2</u>		
4			s7			
5	s5	s6				8
6			r2			
7				r1		
8			s9			
9			r1			



LR(1) 分析过程

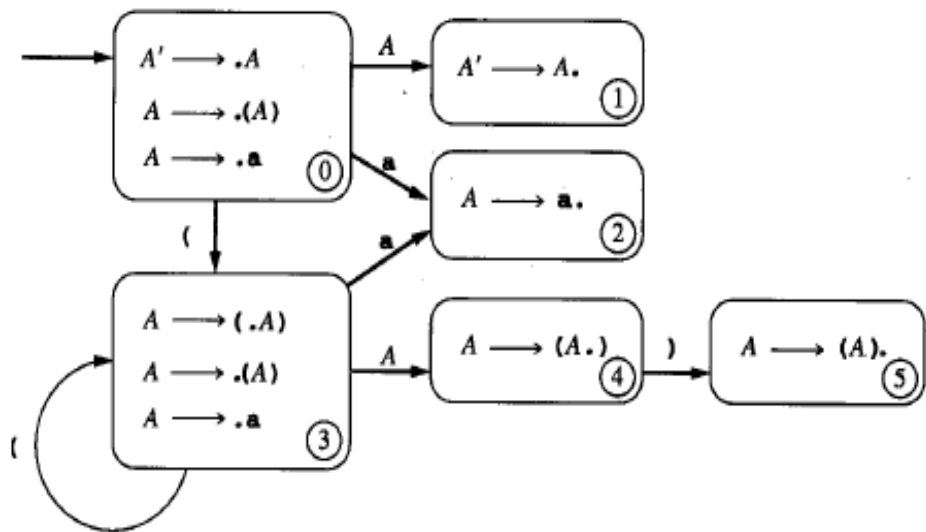
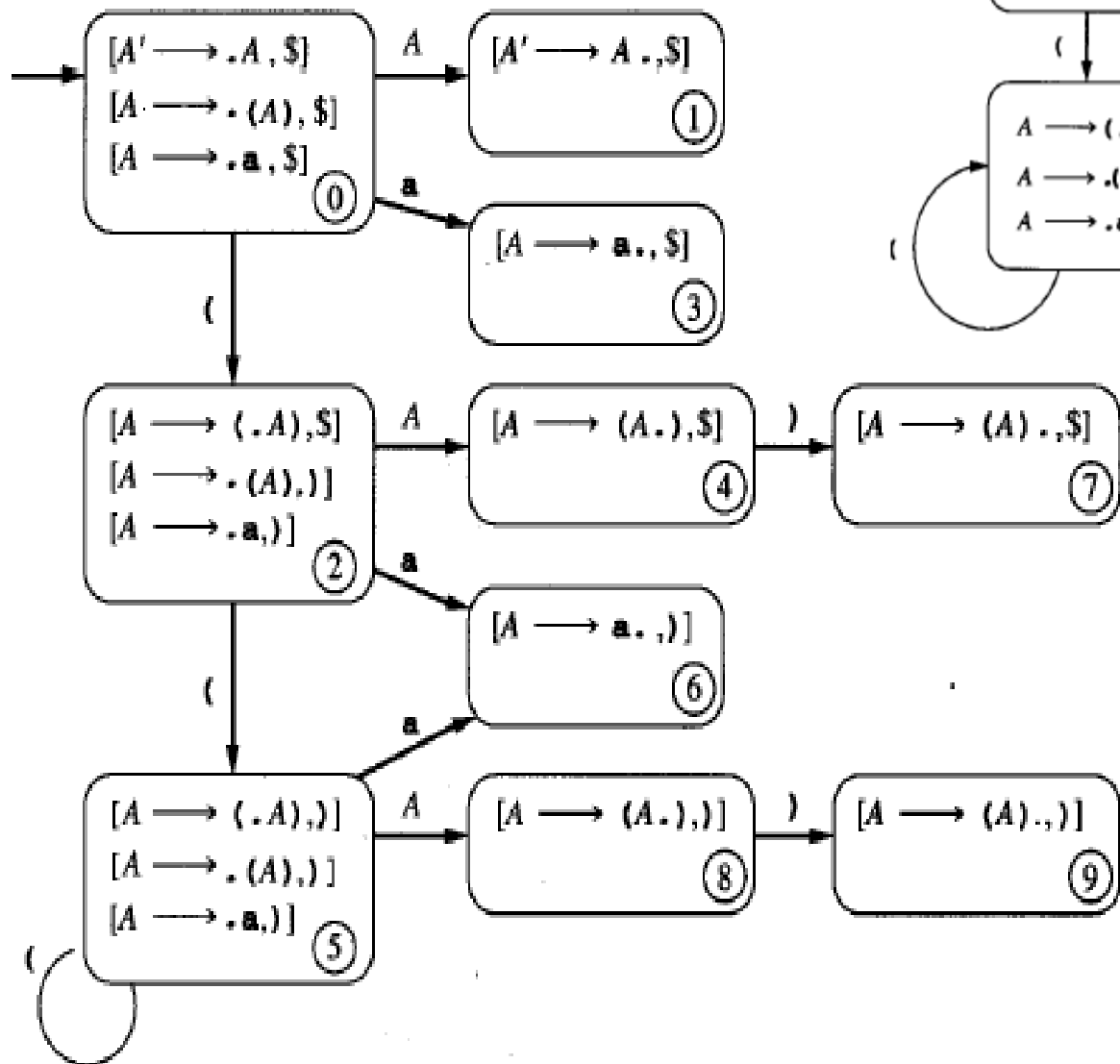
(1) 与SLR(1)分析过程相似。

(2) 步骤：查表——分别处理

状 态	输 入					Goto
	(a)	\$		A
0	s2	s3		<u>接受</u>		1
1						
2	s5	s6		<u>r2</u>		4
3						
4			s7			
5	s5	s6				8
6			r2			
7				r1		
8			s9			
9			r1			

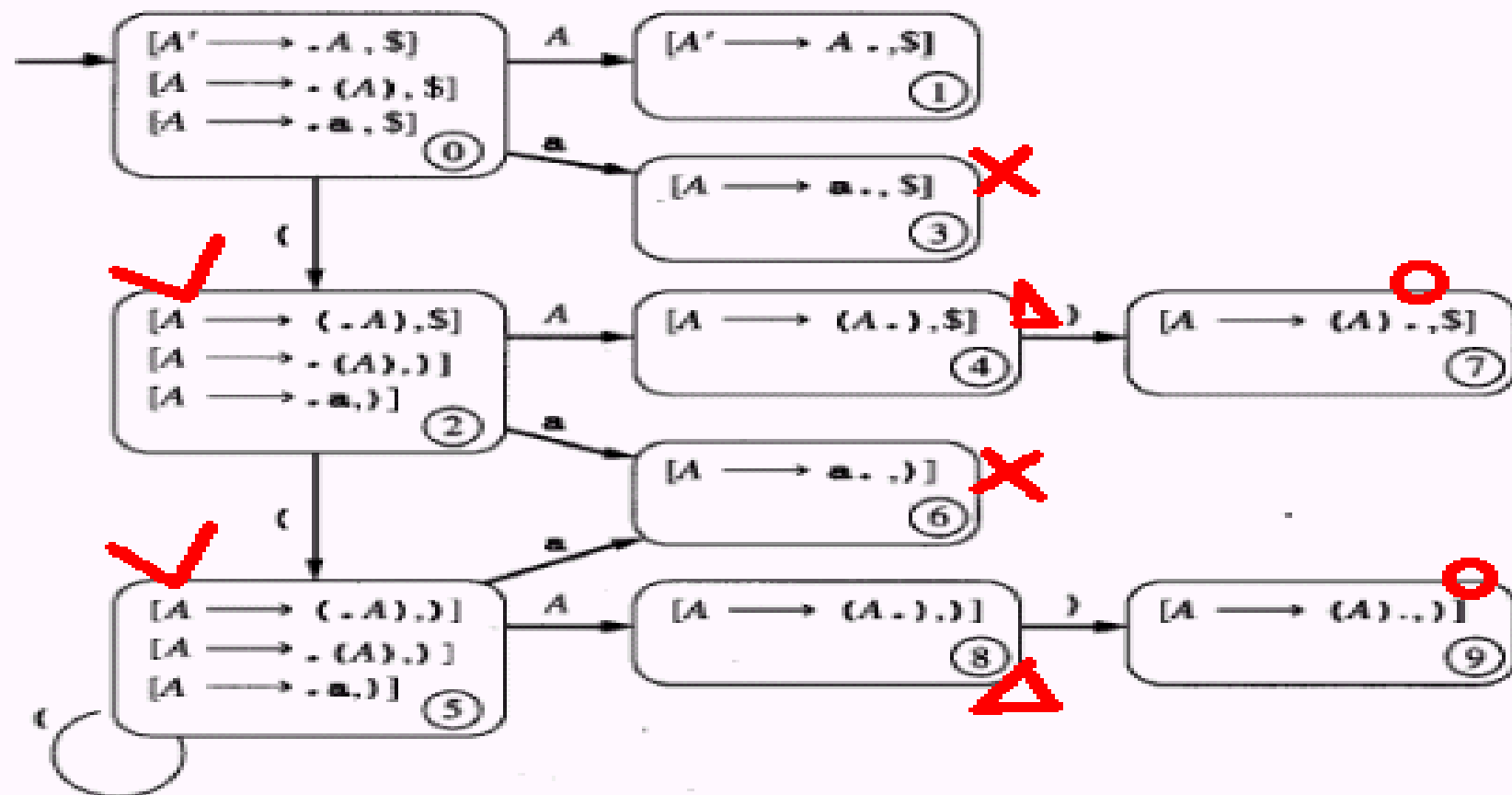
分析串：((a))

LR(1)分析法之问题发现



对比效果

LR(1)分析法之问题



状态2和5、状态4和8、状态7和9、状态3和6，只是先行的符号不一样，而核心是一样的。

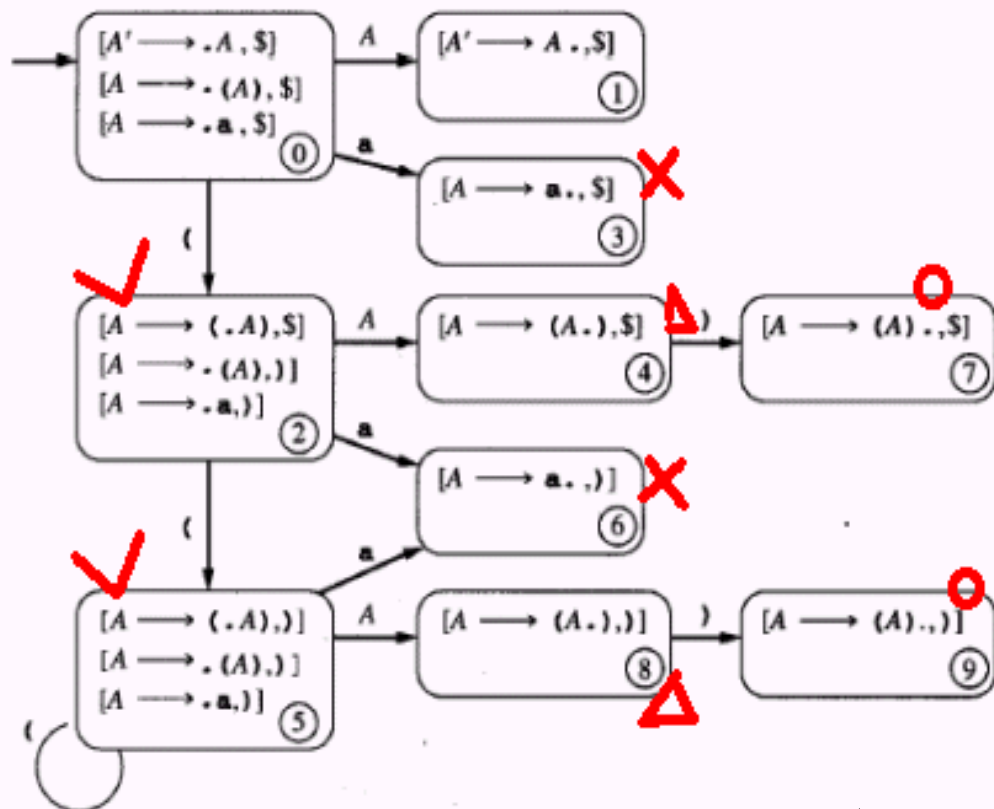
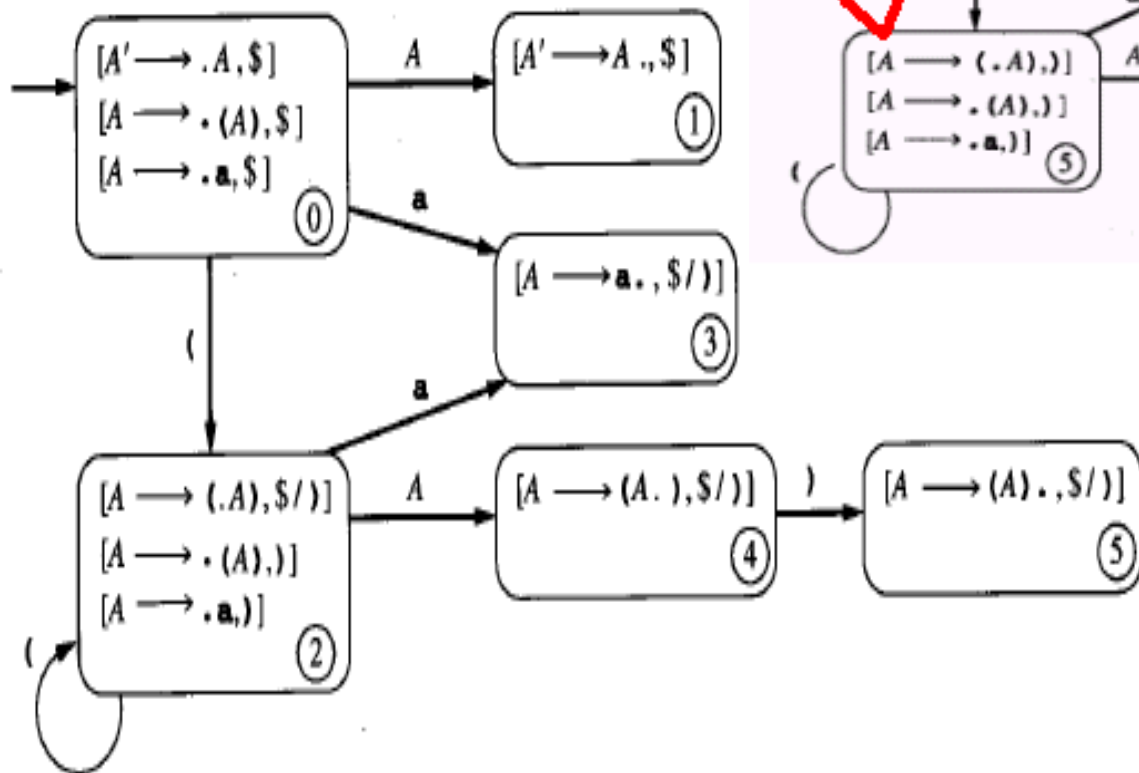
解决方法

- 压缩状态数——数目与LR(0)相仿
- 一个概念：
 - 状态的核心(core)是由状态中的所有LR(1)项目的第1个成分组成的LR(0)项目的集合。
- 压缩方法：
 - 核心一样就合并，先行符号就做并集



LALR(1)
分析法

合并结果



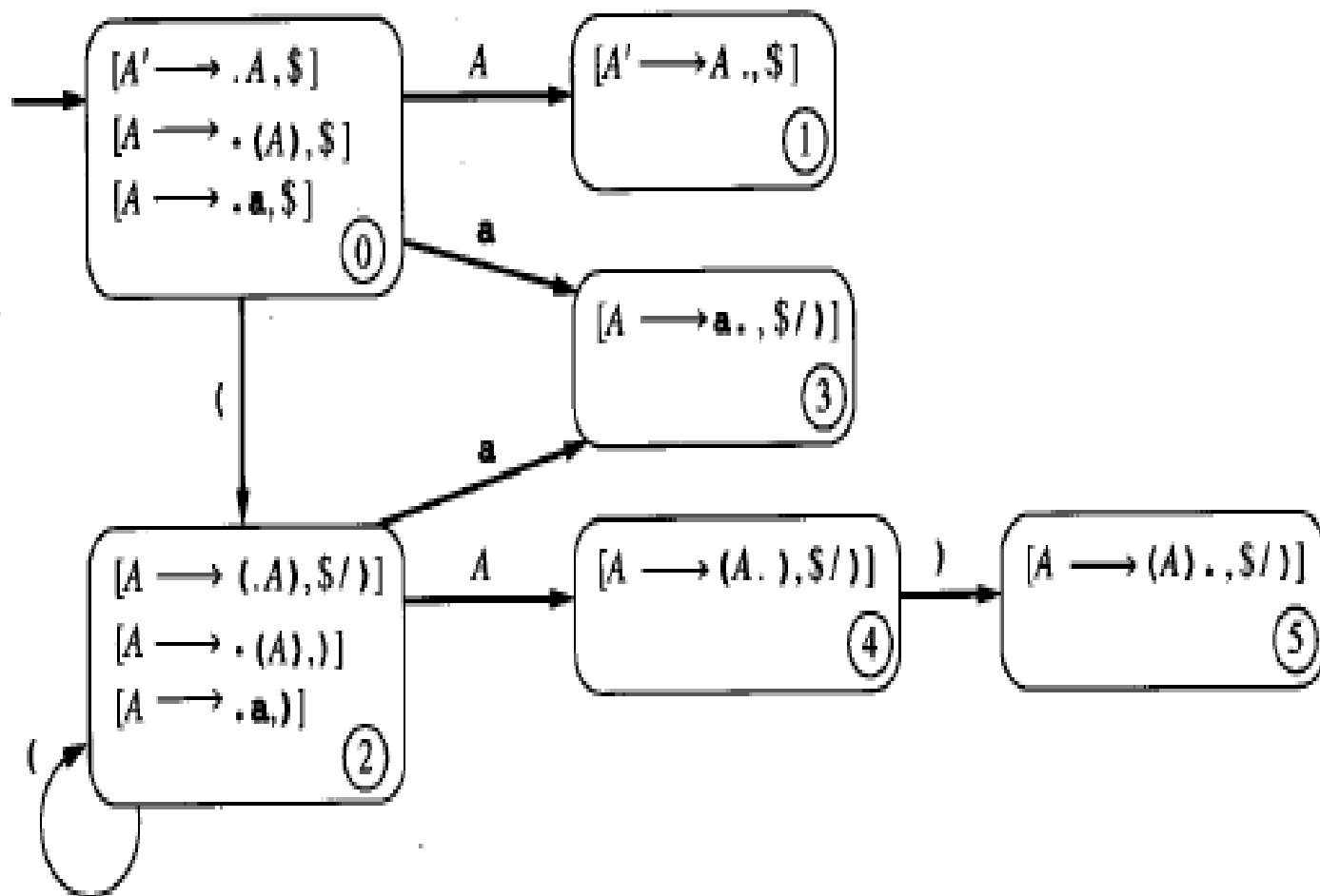
LALR(1) 项
DFA

LALR(1) 分析过程的步骤

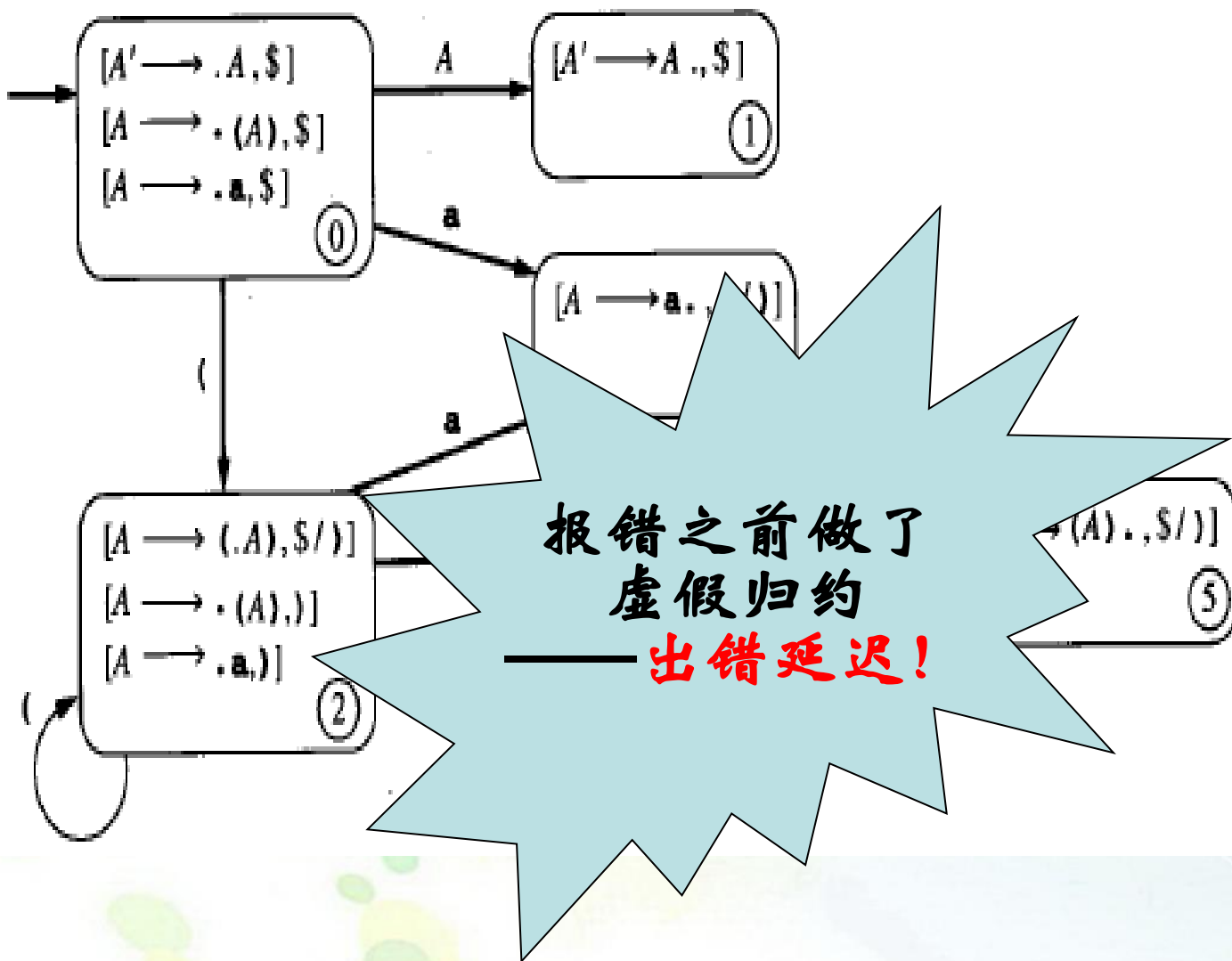
- (1) 先构造LALR(1)DFA,
 - (2) 再构造出LALR(1)分析表,
 - (3) 最后利用LALR(1)分析表对所分析的符号串进行分析。
-
- 例子： 使用LALR(1)分析方法分析符号串(a)

[只用LALR(1)DFA进行分析， 而利用LALR(1)分析表的方法请自行完成。]

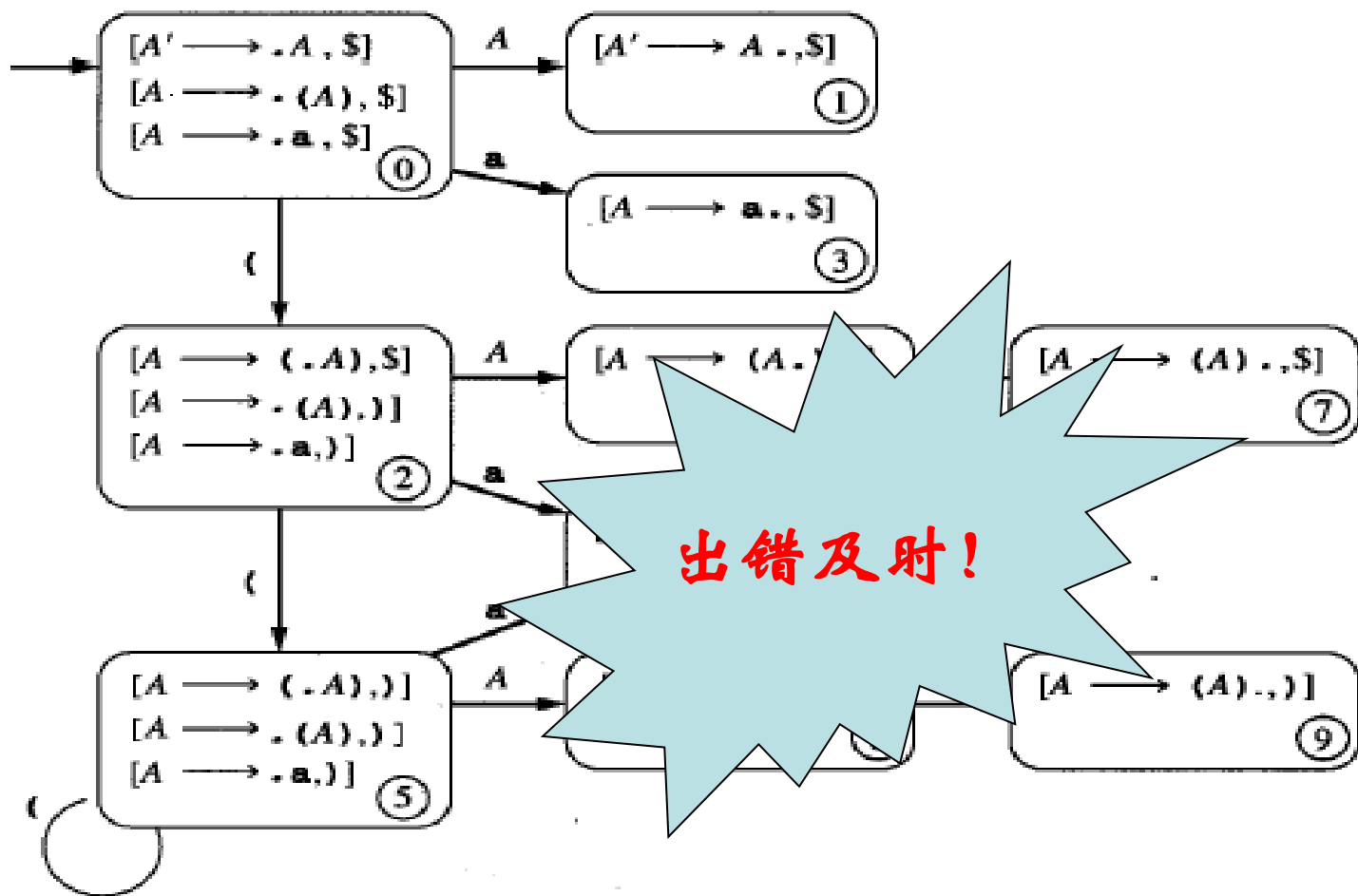
符号串: (a)



用LALR(1)分析方法分析符号串 a)



用LR(1)分析方法分析符号串 a)



总结

能力强

几乎所有CFG的语言结构都能用LR分析
不需要对文法附加条件

难点

几乎不可能用手工编写LR分析器

现实

有LR分析器的生成器

自底向上分析方法

- LR(0) 分析
- SLR(1) 分析
- LR(1) 分析
- LALR(1) 分析

- yacc: LALR(1)语法分析程序的生成器
- Flex: 词法分析程序的生成器