中国科学技术大学
University of Science and Technology of China

语法分析 IV

《编译原理和技术(H)》、《编译原理(H)》

张昱

0551-63603804，yuzhang@ustc.edu.cn

中国科学技术大学

计算机科学与技术学院

中国科学技术大学
University of Science and Technology of China

3.5 自下而上解析
(移进–归约分析)

- 归约(最右推导的逆过程)
- 句柄(可归约串),可能不唯一
- 冲突：移进-归约、归约-归约

语法分析的主要方法

- 自上而下（top-down）预测分析
  - 从开始符号出发，为输入串寻找最左推导
    是自上而下形成分析树的过程
  - 即便消除左递归、提取左公因子，仍然存在一些程序语言,它们对应的文法不是LL(1)
- 自下而上（bottom-up）移进-归约
  - 针对输入串，尝试根据产生式归约(reduce，将与产生式右部匹配的串归约为左部符号)，直至归约到开始符号
    是自下而上形成分析树的过程
  - 比top-down方法更一般化

中国科学技术大学
University of Science and Technology of China

3.5 自下而上解析
(移进-归约分析)

- 归约(最右推导的逆过程)
- 句柄(可归约串),可能不唯一
- 冲突：移进-归约、归约-归约

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例 $S \rightarrow aABe$

$A \rightarrow Abc \mid b$

$B \rightarrow d$

输入串： $abbcde$

一步归约：将与某个产生式右部匹配的子串代换成该产生式的左部符号

如 $A \rightarrow b$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

$B \to d$

输入串：$abbcde$

ab（读入ab）　寻找能匹配某产生式右部的子串　$A \to b$

a　b

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

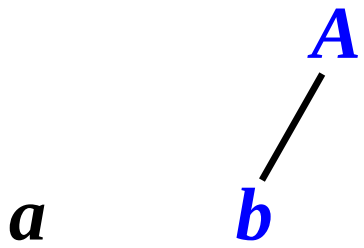
$B \to d$

输入串：$abbcde$

ab

aA（归约）

用产生式 $A \to b$
归约后仍能形成右句型
$aAbcde$ 是右句型



$$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$$

右句型：**按最右推导推出的句型，** $aABe$ 、 $aAde$ 、 $aAbcde$ 、 $abbcde$ 都是右句型

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

$B \to d$

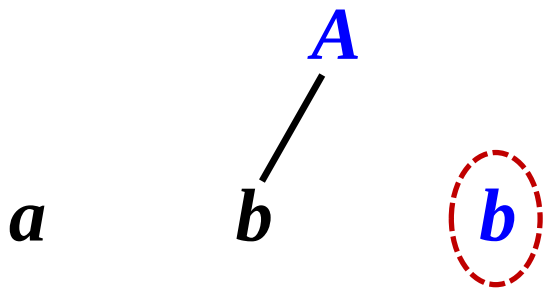
输入串：$abbcde$

ab

aAb（再读入b）

$A \to b$

b可以归约成A吗?



归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

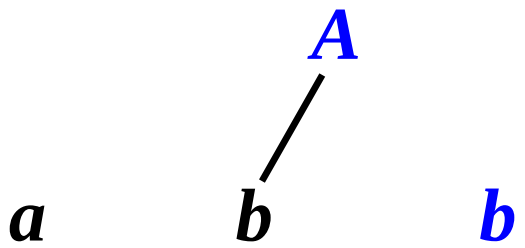
$A \to Abc \mid b$

$B \to d$

输入串：$abbcde$

ab

aAb（再读入b）



$A \to b$

b可以归约成A吗?

归约后能否形成右句型?

$aAAcde$ 不是右句型

故不能将b归约成A

$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \rightarrow aABe$

$A \rightarrow Abc \mid b$

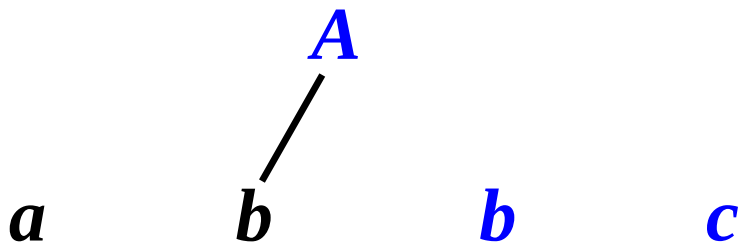
$B \rightarrow d$

输入串：$abbcde$

ab

$aAbc$（再读入c ）

$A \rightarrow Abc$



归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

$B \to d$

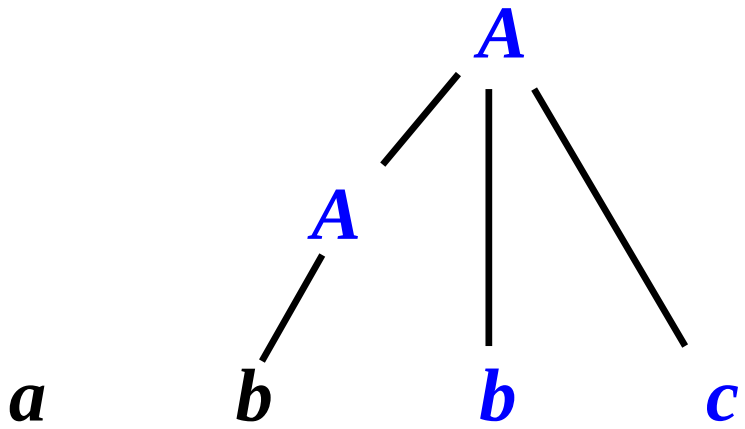
输入串：$abbcde$

ab

$aAbc$

aA（归约）

$A \to Abc$



$$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例 $S \to aABe$

$A \to Abc \mid b$

$B \to d$

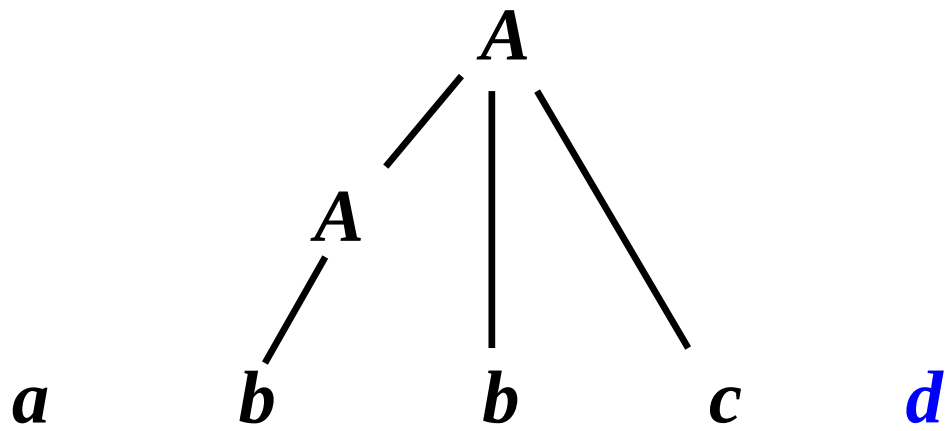
输入串：$abbcde$

ab

$aAbc$

aAd（再读入d）

$B \to d$



$$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

$B \to d$

输入串：$abbcde$

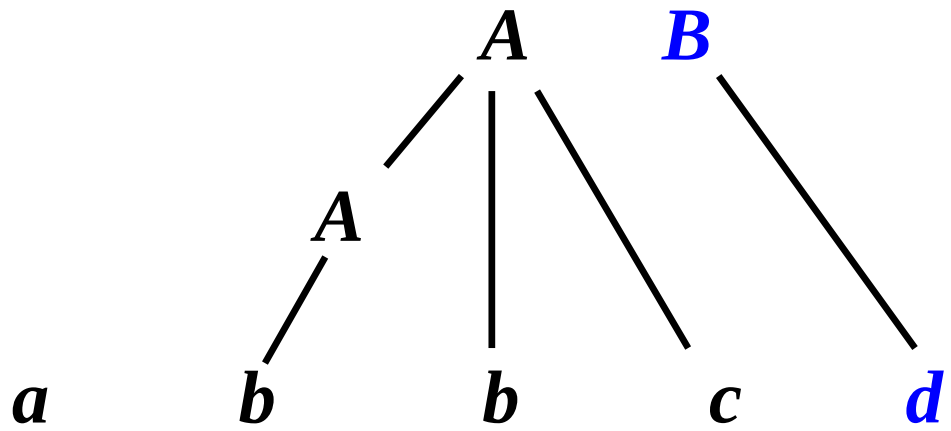
ab

$aAbc$

aAd

aAB（归约）

$B \to d$



$$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例　$S \to aABe$

$A \to Abc \mid b$

$B \to d$

输入串：$abbcde$

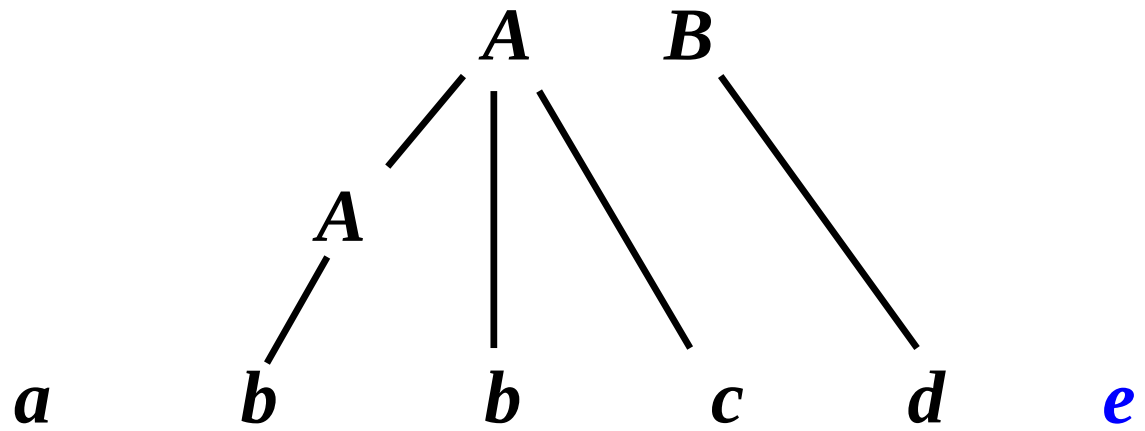
ab

$aAbc$

aAd

aAB

$aABe$(再读入e)



$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$

归约（Reduce）

把输入串归约成文法的开始符号，是最右推导的逆过程

例 $S \to aABe$

$A \to Abc \mid b$

$B \to d$

输入串： $abbcde$

ab

$aAbc$

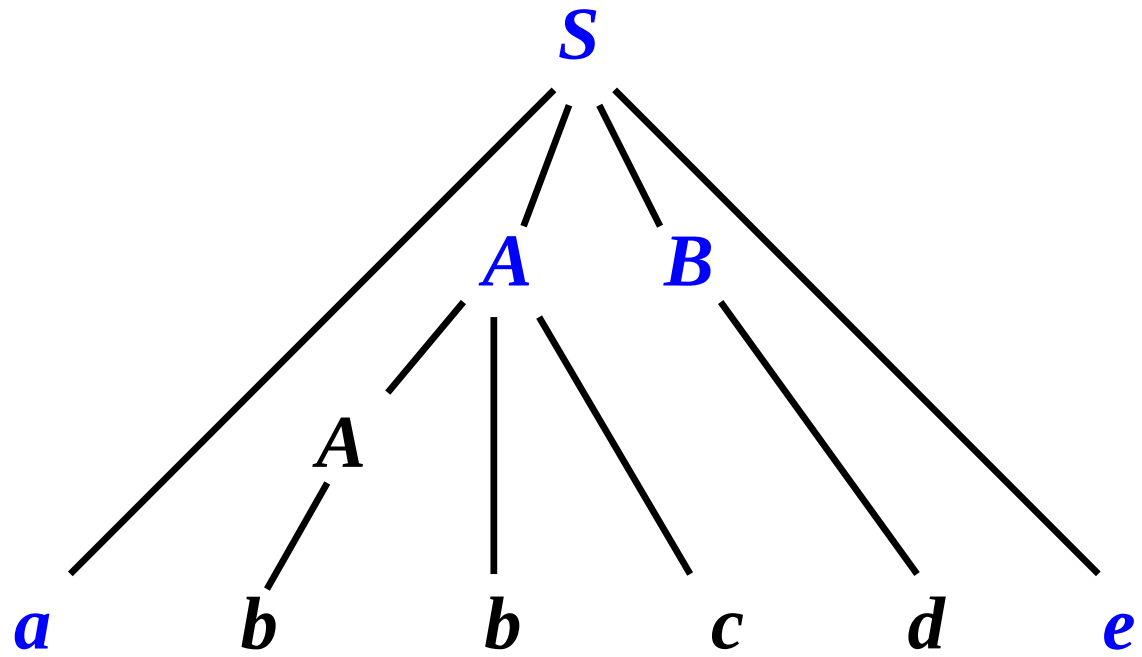
aAd

aAB

$aABe$

S（归约）

$S \to aABe$



$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$

中国科学技术大学
University of Science and Technology of China

3.5 自下而上解析
(移进-归约分析)

- 归约(最右推导的逆过程)
- 句柄(可归约串),可能不唯一
- 冲突：移进-归约、归约-归约

句柄(handles)

右句型的句柄（可归约串）

- 是右句型$\alpha\beta w$中和某产生式 $A \to \beta$右部匹配的最左子串δ, 并且
- 把 β归约成 A 代表了最右推导的逆过程的一步

  右句型$\alpha\beta w$中将子串β归约成A后得到的串$\alpha A w$仍是右句型

  $S \to aABe$

  $A \to Abc \mid b$

  $B \to d$

  * 右句型：最右推导可得的句型

$S \Rightarrow_{rm} aABe \Rightarrow_{rm} aAde \Rightarrow_{rm} aAbcde \Rightarrow_{rm} abbcde$

- 句柄的右边仅含终结符 （是尚未处理输入串）
- 如果文法二义，那么句柄可能不唯一

例 句柄不唯一

$$E \to E + E \mid E * E \mid (E) \mid \mathrm{id}$$

$$\begin{aligned}
E &\Rightarrow_{rm} E * E \\
&\Rightarrow_{rm} E * E + E \\
&\Rightarrow_{rm} E * E + \mathrm{id}_3 \\
&\Rightarrow_{rm} E * \mathrm{id}_2 + \mathrm{id}_3 \\
&\Rightarrow_{rm} \mathrm{id}_1 * \mathrm{id}_2 + \mathrm{id}_3
\end{aligned}$$

例 句柄不唯一

$$E \to E + E \mid E * E \mid (E) \mid \mathrm{id}$$

$$\begin{aligned}
E &\Rightarrow_{rm} E * E \\
&\Rightarrow_{rm} E * E + E \\
&\Rightarrow_{rm} E * E + \mathrm{id}_3 \\
&\Rightarrow_{rm} E * \mathrm{id}_2 + \mathrm{id}_3 \\
&\Rightarrow_{rm} \mathrm{id}_1 * \mathrm{id}_2 + \mathrm{id}_3
\end{aligned}$$

$$\begin{aligned}
E &\Rightarrow_{rm} E + E \\
&\Rightarrow_{rm} E + \mathrm{id}_3 \\
&\Rightarrow_{rm} E * E + \mathrm{id}_3 \\
&\Rightarrow_{rm} E * \mathrm{id}_2 + \mathrm{id}_3 \\
&\Rightarrow_{rm} \mathrm{id}_1 * \mathrm{id}_2 + \mathrm{id}_3
\end{aligned}$$

在右句型$E * E + \mathrm{id}_3$中，句柄不唯一：id_3、$E * E$

* 右句型：最右推导可得的句型

用栈实现移进–归约分析

先通过“移进–归约分析器在分析输入串$id_1 * id_2 + id_3$时的动作序列“来了解移进–归约分析的工作方式。

需要引入**栈**保存移进的文法符号

归约时需要从栈的顶部将形成**句柄**的文法符号串弹出，再将归约成的非终结符入栈

初始格局：

栈	输入
$	*w*$

分析成功后的格局：

栈	输入
$S	$

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	移进

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	移进
$$E*id_2$	$+ id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	移进
$$E*id_2$	$+ id_3$$	按$E \rightarrow$ id归约

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \rightarrow$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	移进
$$E*id_2$	$+ id_3$$	按$E \rightarrow$ id归约
$$E*E$	$+ id_3$$	

移进-归约分析：$id_1 * id_2 + id_3$

栈	输　入	动　作
$	$id_1 * id_2 + id_3$$	移进
$ id_1	$* id_2 + id_3$$	按$E \to$ id归约
$$E$	$* id_2 + id_3$$	移进
$$E*$	$id_2 + id_3$$	移进
$$E*id_2$	$+ id_3$$	按$E \to$ id归约
$$E*E$	$+ id_3$$	移进？归约？

$$E \Rightarrow_{rm} E*E \Rightarrow_{rm} E*E+E \Rightarrow_{rm} E*E+id_3$$ 移进

$$E \Rightarrow_{rm} E+E \Rightarrow_{rm} E+id_3 \Rightarrow_{rm} E*E+id_3$$ 归约

移进-归约分析：$id_1 * id_2 + id_3$

栈	输入	动作
$	$id_1 * id_2 + id_3$	移进
$ id_1	$* id_2 + id_3$	按E → id归约
$E	$* id_2 + id_3$	移进
$E*	$id_2 + id_3$	移进
$ E* id_2	$+ id_3$	按E → id归约
$ E*E	$+ id_3$	移进
$ E*E+	id_3	移进
$ E*E+ id_3	$	按E → id归约
$ E*E+E	$	按E → E+E归约
$ E*E	$	按E → E*E归约
$E	$	接受

中国科学技术大学
University of Science and Technology of China

3.5 自下而上解析
(移进-归约分析)

- 归约(最右推导的逆过程)
- 句柄(可归约串),可能不唯一
- 冲突：移进-归约、归约-归约

移进–归约分析需要解决的一些问题

- 如何决策是选择移进还是归约？面临**移进-归约冲突**
- 进行归约时，怎么确定右句型中要归约的子串(即句柄)
  - 句柄总是出现在栈顶
- 进行归约时，不知该选择哪一个产生式归约？
  面临**归约-归约冲突**
  - 栈顶形成不同的句柄，或
  - 句柄是不止一个产生式的右部

移进-归约分析的冲突

□ 移进–归约冲突(shift/reduce conflict)

例

stmt → if *expr* then *stmt*

| if *expr* then *stmt* else *stmt*

| other

如果移进–归约分析器处于格局(configuration)

栈	输入
… if *expr* then *stmt*	else … $

一般地，优先移进
也满足else的
就近匹配原则

移进-归约分析的冲突

- 归约–归约冲突(reduce/reduce conflict)

stmt → id (*parameter_list*) | *expr* = *expr*　　id(…)是过程调用

parameter_list → *parameter_list*, *parameter* | *parameter*

parameter → id

expr → id (*expr_list*) | id　　id(…)也表示数组元素的引用

expr_list → *expr_list*, *expr* | *expr*

由*A*(*I*, *J*)开始的语句

归约成*expr*还是*parameter*？

栈	输入
… id (id	, id)…

方法1
一般用位于前面的产生式进行归约

移进-归约分析的冲突

- 归约–归约冲突(reduce/reduce conflict)

procid

stmt → id (*parameter_list*) | *expr* = *expr*

parameter_list → *parameter_list*, *parameter* | *parameter*

parameter → id

expr → id (*expr_list*) | id　　id(…)也表示数组元素的引用

expr_list → *expr_list*, *expr* | *expr*

由*A*(*I*, *J*)开始的语句(词法分析查符号表, 区分第一个id)

栈	输入
… procid(id	, id)…

  - 需要修改上面的文法

方法2
改写文法，区分id是否是procid

中国科学技术大学

University of Science and Technology of China

下期预告：LR解析技术