

合肥工业大学

系统软件综合设计报告

编译原理分册

设计题目	41
学生姓名	
学 号	
专业班级	计算机科学与技术
指导教师	
完成日期	2020.8.22

报告目录

一、设计目的及设计要求.....	3
二、开发环境描述	3
三、设计内容、主要算法描述.....	4
四、设计的输入和输出形式.....	7
五、程序运行（测试、模拟）的结果（屏幕拷贝、生成结果的打印输出）	8
六、总结（体会）	11
七、源程序清单（部分核心代码）作为报告的附件。	12

一、设计目的及设计要求

设计目的

为了进一步掌握对编译原理这门课程所学知识以及将这些知识运用在项目中。

本次之所以选择 LALR (1) 文法是因为当初实验已经实现了 LL(1),以及 LR(1)等方法, 就是 LALR (1) 未实现, 因此在这基础上进行一个尝试

设计要求

假设对于给定文法, 识别文法活前缀的 DFA、LR(1)项目集族已构造出来了。
构造一程序, 检查两个 LR(1)项目集是否为同心集 (可任意输入), 若是, 则输出合并后的同心集, 并检查合并后的集合是否含有冲突项目 (指出存在何种冲突), 输出合并同心集后的识别文法活前缀的 DFA, 及 LALR(1)项目集规范族

二、开发环境描述

Windows 版本

Windows 10 家庭中文版

© 2019 Microsoft Corporation。保留所有权利。



系统

制造商:	ASUSTek Computer Inc.
处理器:	Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 2.20 GHz
已安装的内存(RAM):	24.0 GB (23.9 GB 可用)
系统类型:	64 位操作系统, 基于 x64 的处理器
笔和触控:	没有可用于此显示器的笔或触控输入



ASUSTek Computer Inc. 支持

网站: [联机支持](#)

windows 10 家庭中文版

处理器: Intel i7-8750H

内存: 24GB

开发软件: IntelliJ IDEA 2019.1.3 x64

使用语言: Java

三、设计内容、主要算法描述

设计内容

一个 LALR 文法分析程序，可以有效地对输入文法进行分析，并生成识别文法活前缀的 DFA 和 LALR (1) 项目集族，并且可以通过先前生成的 LR (1) 项目集族和 DFA 对两者进行一个比较输出

主要算法描述

※题目 41 LALR(1)的项目集族与识别活前缀的 DFA 的生成

这个题目实际上也是本人为了弥补实验未能完成的遗憾。实际上 LALR(1)方法就是 LR(1)的方案上进行了进一步改进，在通过 LR(1)求出了项目集族后，将其中具有同心项的项目集族进行合并即可，因此我们首先对 LR(1)方法进行分析，之后再对 LALR(1)进行进一步分析以及阐述实现方法

●LR(1)思想

LR (1) 属于自下而上分析方法，通过分析表，结合输入串进行规约，并且它的每一个入口都是唯一的，LR(1)文法每次向前查看一个符号，比起 LR (0) 它多了一个展望符，因此相比而言 LR (1) 的项目集族会更多，项目集族从第一个开始依次求闭包，直到项目集族不再增大为止

◎项目集族构建

首先将文法拓广，随后对文法进行处理

依次遍历将点添加至各个产生式中形成项目备选集

$A \rightarrow E + F$ 可形成四个项目: $A \rightarrow \cdot E + F$, $A \rightarrow E \cdot + F$, $A \rightarrow E + \cdot F$, $A \rightarrow E + F \cdot$

随后将第一个项目添加到第一个项目集族中，对该项目集族进行完善，同时

还要注意将展望符一同添加,按照法则求出扩展符并且添加,待项目集族完善后,开始求导其他项目集族,依次遍历项目集族中的所有项目,并且遍历到当前的点后的非终结符还要在余下的项目中寻找相同的非终结符一同处理,至于如何将圆点·向后移动,刚刚构造的项目备选集就起了作用,在备选集中找到当前项目并向后搜寻一位即可,如果圆点·已经位于末尾则不必搜寻

构造的项目集族数据结构如下,其中的很多定义都是为了 LALR(1)做准备,比如 OwnNum 就是代表了集族的项数量

//单项的数据结构

```
class Own {  
    String Sen;                //项的具体内容, 比如 ab  
  
    int PreNum=0;              //单项的展望符数量  
  
    String[] Pre = new String[50]; //存储单项的所有展望符  
}
```

//项目集族的数据结构

```
class Sentence{  
    Map OwnSet = new HashMap(); //存储该项目集族的所有项的哈希 Map  
  
    String No;                  //存储该项目集族的标号, 使用 string 是为了便于 LALR(1)合并  
  
    int OwnNum=0;               //存储该项目集族的所有项数量  
  
    Own[] OwnDetail= new Own[50]; //存储该项目所有项  
}
```

●LALR(1)思想

在完成 LR(1)的实现后,接下来要做的就是修改成 LALR(1)项目集族,到这一步已经拥有了 LR(1)项目集族,并且因为数据结构的先前定义使得这一步更容易实现,本人在此基础上构建了三个函数

```
boolean EqualOrInvolve(int MAIN, int Com);

boolean NotIn(String [] TA, int TN, String Insert);

void Melt();
```

Melt 函数调用了另外两个函数从而实现了合并相同同心项的项目集族

◎如何找出相同同心项的集族？

首先设置两个变量(哨兵),第二个变量从第一个变量+1 的数值往后依次遍历,如果遇到同心项相同的集族就开始融合操作,因此判断是否有同心项是本次功能实现的关键,同心项相同就是指两个项目集族中具有相同的项(包含不算)

这就要阐述一下 EqualOrInvolve 函数的作用,两个参数分别是要进行对比的集族标号。在先前 LR(1)项目集族的构造时,都会计算相应的所有单项的数目,因此第一部就是对这两项的单项数目进行对比,如果两个集族项数目不相同,那么直接视为不相同,进入下一个比较。在项目集族项目数目相同后,再依次比较每个项。这里要提的是在 LR(1)项目集族的形成时就已经提前对项进行特殊排序,因此如果项相同的话排序后每个序号对应的项都会相同。一旦比较全部相同后,就说明可以进行合并操作了,如下图所示

项目集族 “L1”

E->T G	#)
G-> +TG	#)
G-> -TG	#)
G-> ·	#)

项目集族 “L2”

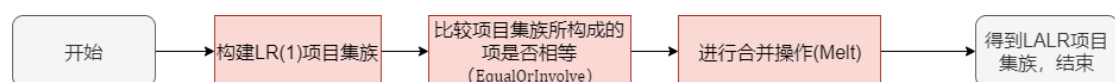
E->T G)
G-> +TG)
G-> -TG)
G-> ·)

本次功能的实现是将后一个集族合并到前一个集族中，这样便于扫描。因为项经过排序的缘故，因此只需要扫描一遍项数组，然后将后一个集族中的展望符加入前一个集族中的展望符数组即可，NotIn 函数就是用于判断展望符是否存在。待全部合并完成过后，将后面的那一个集族整体删除，并且后续的集族集体前移。最终只需要将结果集族的编号进行修改就行了，基本格式是将第一个集族的编号+第二个集族的编号

项目集族 “L1+2”

E->T G	#)
G-> +TG	#)
G-> -TG	#)
G-> ·	#)

合并完成后第二个哨兵从原地继续向后移动直到末尾，从而完成一次遍历再将第一个哨兵向后一步，并将第二个哨兵置为第一个哨兵+1，直到第一个哨兵到达末尾为止，这便是 Melt 函数的全部功能。一遍流程下来就得到了 LALR(1) 的项目集族，至此我们就完成了对 LALR(1)项目集族的构建，算法大致流程图如下



题目41：LALR（1）项目集族构建算法基本流程图

四、设计的输入和输出形式

设计输入：原始文法（字符串），在 GUI 中专门提供一个输入框可以输入完整的文法，单个产生式的格式为 A->Bab，通过->连接，非终结符限定为 26 个大写字母

设计输出：可选择性输出，输出 LR(1)也就是融合前的识别活前缀的 DFA 和项目

集族, 以及 LALR(1)的识别活前缀的 DFA 和项目集族, 分别可以通过文本形式, 以及 HTML 可视化表格形式进行输出, 方便进行比较

五、程序运行（测试、模拟）的结果（屏幕拷贝、生成结果的打印输出）

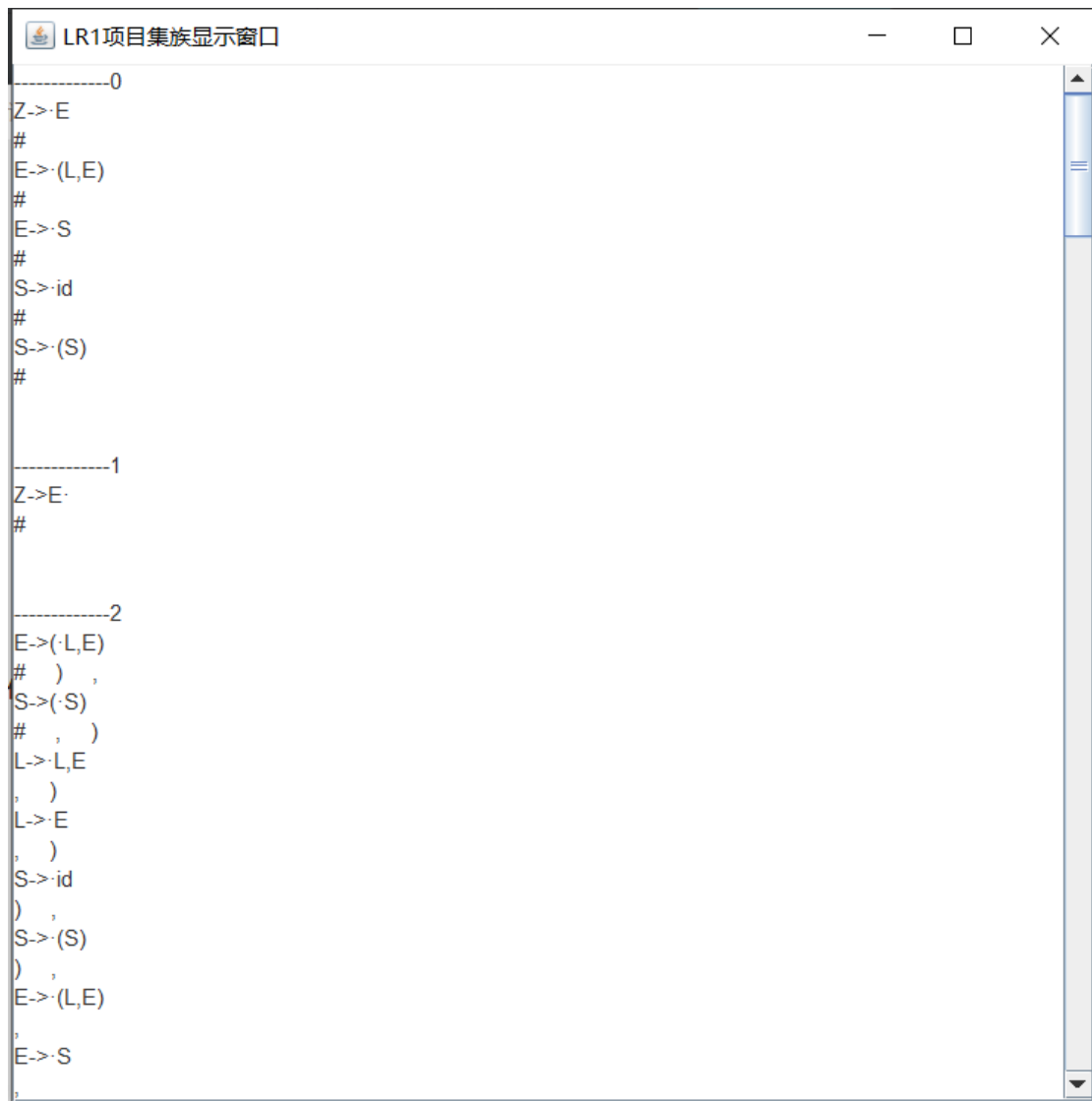
①项目 UI 如下图所示, 在上方框框输入文法即可



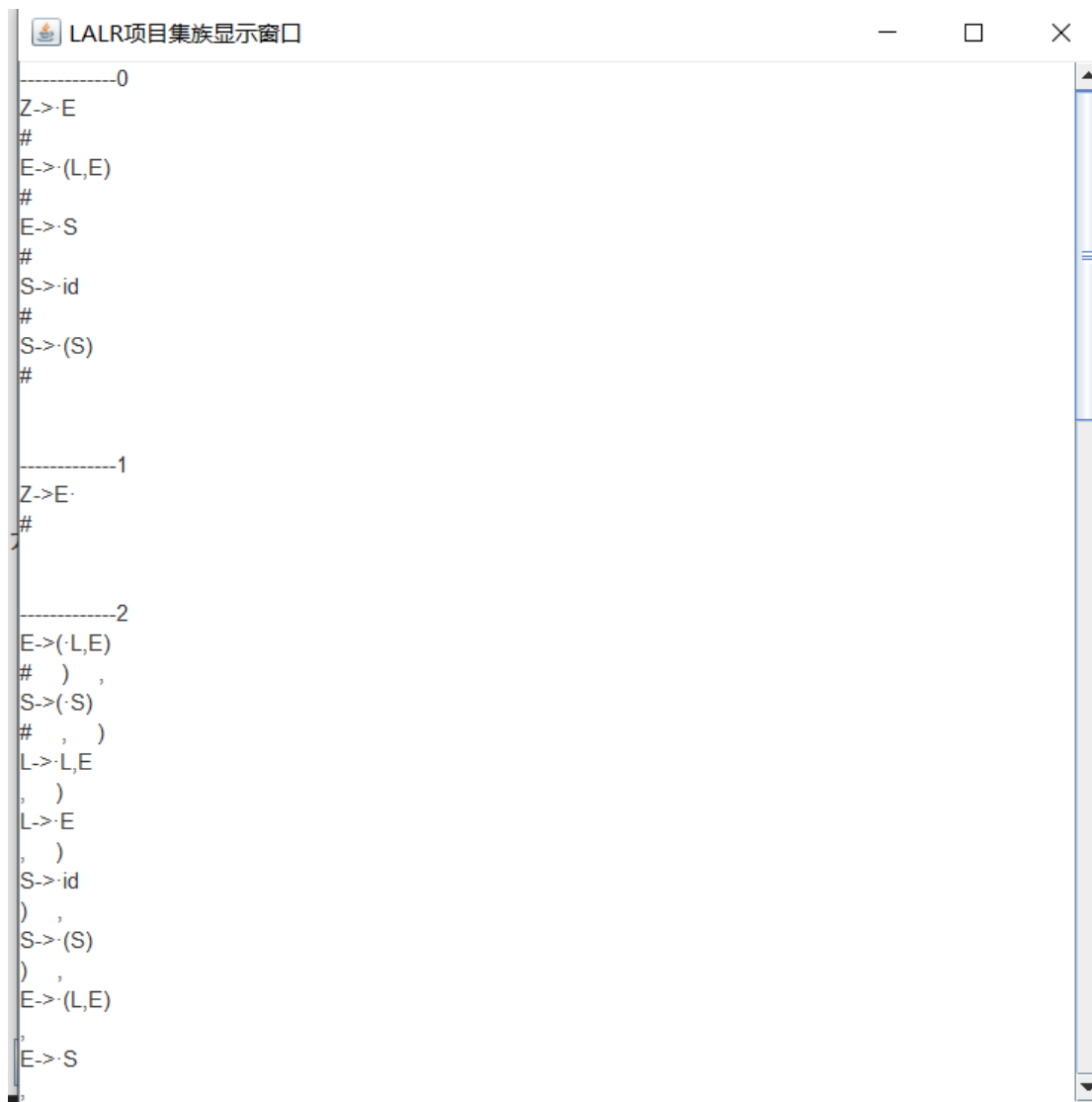
本次测试采用以下文法进行测试

Z- > E
E- > (L,E)
E- > S
L- > L,E
L- > E
S- > id
S- > (S)

②输入文法，点击“文法分析”后，分析过程开始，此时点击“显示 LR1 集族”按钮进行 LR(1)项目集族的文本展示，如下图所示（仅截取部分）



③此时点击“显示 LALR 集族”按钮进行 LALR 项目集族的文本展示，如下图所示（仅截取部分）



The screenshot shows a window titled "LALR项目集族显示窗口" (LALR Item Set Display Window). It displays the LR(0) and LR(1) item sets for a grammar. The LR(0) item sets are shown at the top, and the LR(1) item sets are shown below them, separated by a horizontal line. The LR(1) item sets are labeled with their look-ahead symbols (L, E, S, id, (,), ,).

```
-----0
Z->·E
#
E->·(L,E)
#
E->·S
#
S->·id
#
S->·(S)
#

-----1
Z->E·
#

-----2
E->(·L,E)
# ) ,
S->(·S)
# , )
L->L·E
, )
L->E·
, )
S->id
) ,
S->(S)
) ,
E->(L,E)
,
E->S·
```

④最后进行可视化项目集族延时，点击“HTML 显示集族（前后对比）”，程序自动生成 html 代码文件并以网页的形式打开，同时有两个网页打开分别进行 LR(1)和 LALR 的项目集族可视化展示, LR (1) 项目集族展示如下 (仅截取部分)

LR(1)项目集族											
L0		L1		L2		L3		L4		L5	
Z->E	#	Z->E	#	E->(L,E)	# ,	E->S	# ,	S->id	# ,	E->(L,E)	# ,
E->(L,E)	#			S->(S)	# ,					L->L,E	,
E->S	#			L->L,E	,						
S->id	#			L->E	,						
S->(S)	#			S->id	,						
				S->(S)	,						
				E->(L,E)	,						
				E->S	,						

L6		L7		L8		L9		L10		L11	
S->(S)	# ,	L->E	,	S->id	,	S->(S)	,	S->id	# ,	E->(L,E)	# ,
E->S	,					E->(L,E)	,			L->L,E	,
						S->id	,			E->(L,E)	,
						S->(S)	,			E->S	,
						L->L,E	,			S->id	,
						L->E	,			S->(S)	,
						E->(L,E)	,				
						E->S	,				

⑤LALR 项目集族可视化展示如下（仅截取部分）

LALR(1)项目集族											
L0		L1		L2+9+17		L3+18		L4+8		L5+15+22	
Z->E	#	Z->E	#	E->(L,E)	# ,	E->S	# ,	S->id	# ,	E->(L,E)	# ,
E->(L,E)	#			S->(S)	# ,					L->L,E	,
E->S	#			L->L,E	,						
S->id	#			L->E	,						
S->(S)	#			S->id	,						
				S->(S)	,						
				E->(L,E)	,						
				E->S	,						

六、总结（体会）

本次题目补全了本人在课程实验阶段没有实现的 LALR（1）方法，也让我对

项目集族和程序的语法分析细节有了进一步的掌握。并且项目功能的实现也让我对使用 java 中的部分数据结构有了进一步的提升。项目集族的可视化输出让我进一步巩固了对 HTML 网页代码的编写以及 Java 对网页的功能操作

总而言之，本次项目让我对编译原理的理解有了进一步的加深，这也为我的第三个自拟题目有了思路上的帮助

七、源程序清单（部分核心代码）作为报告的附件。

路径下的 src 文件夹包含三个源代码文件

GUI.java：项目的 GUI 界面代码实现

LALR.java 项目的功能函数定于与实现

HTML.java 用于将项目集族用网页的形式可视化输出

路径中的 LALRPrint.html 就是 LALR 项目集族通过网页形式输出的 html 文件，

LR1Print.html 同理