北京航空航天大学

2019－2020 学年 第一学期期末

《编译技术》

B卷

班 级\_\_\_\_\_\_\_\_\_\_\_\_\_\_学 号 \_\_\_\_\_\_\_\_\_

姓 名\_\_\_\_\_\_\_\_\_\_\_\_\_\_成 绩 \_\_\_\_\_\_\_\_\_

2020年5月25日

班号 学号 姓名 成绩

《 编译技术 》期末考试卷

注意事项：

1. 试卷共 6 页（不含封面和目录），请仔细检查。

2．在监考老师统计完试卷后，再离开考场；

题目：

一、简答题…………………………………………………………… ( 20 分)

二、正则文法与自动机………………………………………………… ( 20 分)

三、算符优先分析法……………………………………………………( 15 分)

四、SLR分析法………………………………………………………… ( 15 分)

五、符号表构造与运行时存储分析…………………………………… ( 15 分)

六、代码优化……………………………………………………………( 15 分)

|  |  |  |
| --- | --- | --- |
| **题号** | **得分** | **教师签字** |
| **1** |  |  |
| **2** |  |  |
| **3** |  |  |
| **4** |  |  |
| **5** |  |  |
| **6** |  |  |
| **总分** |  |  |

1. 简答题：（20分）
2. 简述什么是错误的局部化处理，主要作用是什么？

错误局部化处理:指当编译程序发现错误后，尽可 能把错误的影响限制在一个局部

的范围，避免错误扩散和影响 程序其他部分的分析。

1. 分别简述语法分析的任务和语义分析的任务。

语法分析功能:根据文法规则，从源程序单词符号串中识别出语法 成分，并进行语法检查。基本任务:识别符号串S是否为某语法成分。

语义分析是审查源程序有无语义错误，为代码生成阶段收集类型信息。

1. 分别简述静态存储分配和动态存储，二者的使用场景有什么不同。

静态存储分配

在编译阶段由编译程序实现对存储空间的 管理和为源程序中的变量分配存储的方法。

条件

如果在编译时能够确定源程序中变量在运 行时的数据空间大小，且运行时不改变，那么 就可以采用静态存储分配方法。

动态存储分配

在目标程序运行阶段由目标程序实现对存 储空间的组织与管理，和为源程序中的变量 分配存储的方法。

特点

• 在目标程序运行时进行变量的存储分配。 • 编译时要生成进行动态分配的目标指令。

1. 分别说明什么是局部优化、全局优化和循环优化。

• 局部优化技术

– 指在基本块内进行的优化

– 例如，局部公共子表达式删除

• 全局优化技术

– 函数/过程内进行的优化 – 跨越基本块

– 例如，全局数据流分析

• 跨函数优化技术 – 整个程序

– 例如，跨函数别名分析 ，逃逸分析 等

1. 说明什么是交叉编译，什么时候需要使用交叉编译？

在一种计算机环境中运行的编译程序，能编译出在另外一种环境下运行的代码，我们就称这种编译器支持交叉编译。这个编译过程就叫交叉编译。

有时是因为目的平台上不允许或不能够安装我们所需要的编译器，而我们又需要这个编译器的某些特征；有时是因为目的平台上的资源贫乏，无法运行我们所需要编译器；有时又是因为目的平台还没有建立，连操作系统都没有，根本谈不上运行什么编译器。

1. 在输入字母表Σ={0, 1} 上定义语言L = {x|x∈Σ\*，且x以0开头，以1结尾}。（20分）
   1. 给出识别该语言L的正则表达式。

0(0|1)\*1

* 1. 根据正则表达式构造NFA。

X

* 1. 设计识别该语言的极小化DFA。（给出求解过程：①对应的）

1. 有如下文法G[S]：（15分）

S → V

V → T | V i T

T → F | T + F

F → b V\* | a

1. 判断该文法是否是算符文法，并说明理由。

是，因为文法中无形如U∷= ···VW···的规则

1. 求每个非终结符的FIRSTVT和LASTVT集合。

F(S) = {i, +, b, a}

L(S) = {i, +, \*, a}

F(V) = {i, +, b, a}

L(V) = {i, +, \*, a}

F(T) = { +, b, a}

L(T) = {+, \*, a}

F(F) = {b, a}

L(F) = {\*, a}

1. 构造算法优先关系矩阵。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | i | + | b | \* | a | # |
| i | > | < | < | > | < | > |
| + | > | > | < | > | < | > |
| b | < | < |  | < | < |  |
| \* | > | > |  | > |  | > |
| a | > | > |  | > |  | > |
| # | < | < | < |  | < |  |

1. 判断该文法是否为算符优先文法。

是，因为算符的优先关系唯一

1. 有如下文法G[S]：（15分）

        S→ SaBc| aBc

B→ bAc

A→ a | bS

SLR分析表（Action表和GOTO表）。

1. S→ SaBc
2. S→ aBc
3. B→ bAc
4. A→ a
5. A→ bS

I0:

S = .SaBc

S = .aBc

I1: I0 + S

S = S.aBc

I2: I0 + a

S = a.Bc

B = .bAc

I3: I1 + a

S = Sa.Bc

B = .bAc

I4: I2 + B

S = aB.c

I5: I2 + b

B = b.Ac

A = .a

A = .bS

I6: I3 + B

S = SaB.c

\*I7: I3 + b

goto I5

I7: I4 + c

S = aBc.

I8: I5 + A

B = bA.c

I9 : I5 + a

A = a.

I10 : I5+b

A = b.S

S = .SaBc

S = .aBc

I11: I6 +c

S = SaBc.

I12: I8 +c

B = bAc.

I13: I10 + S

A = bS.

S = S.aBc

\*I14: I10 + a

goto I2

\*I14: I13 + a

goto I3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | a | b | c | # | S | A | B |
| 0 | S2 |  |  |  | 1 |  |  |
| 1 | S3 |  |  |  |  |  |  |
| 2 |  | S5 |  |  |  |  | 4 |
| 3 |  | S5 |  |  |  |  | 6 |
| 4 |  |  | S7 |  |  |  |  |
| 5 | S9 | S10 |  |  |  | 8 |  |
| 6 |  |  | S11 |  |  |  |  |
| 7 |  |  |  | r2 |  |  |  |
| 8 |  |  | S12 |  |  |  |  |
| 9 |  |  |  |  |  |  |  |
| 10 | S2 |  |  |  | 13 |  |  |
| 11 |  |  |  | r1 |  |  |  |
| 12 |  |  |  | r3 |  |  |  |
| 13 | S3 |  |  |  |  |  |  |

识别句型SabAcc活前缀SabAc的有效项目集 。

1. 有如下程序段：（15分）

program paser;

var filename:string;

curchar:char;

procedure getsym;

var symsize:integer;

procedure getchar;

var errorinfo:string;

begin

… ------------（1）

end;

begin

…

call getchar;

…

end;

procedure lexer;

var linelength:integer;

begin

…

call getsym;--------------（2）

…

end;

begin

…

call lexer;

…

end.

1. 请分别画出当编译到位置（1）和（2）时的符号表。

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 维数 |
| filename | string | 0 |
| curchar | char | 0 |
| getsym | proc |  |
| symsize | integer | 0 |
| getchar | proc |  |
| errorinfo | string | 0 |

|  |  |  |
| --- | --- | --- |
| 变量名 | 类型 | 维数 |
| filename | string | 0 |
| curchar | char | 0 |
| getsym | proc |  |
| symsize | integer | 0 |
| getchar | proc |  |
| errorinfo | string | 0 |
| lexer | proc |  |
| linelength | integer | 0 |
|  |  |  |

1. 运行到位置（1）时的运行栈。

|  |
| --- |
|  |
|  |
| errorinfo |
| prev abp(3) |
| ret addr(4) |
| abp(4) |
| abp(3) |
| abp(2) |
| abp(1) |
| symsize |
| prev abp(2) |
| ret addr(3) |
| abp(3) |
| abp(2) |
| abp(1) |
| linelength |
| prev abp(1) |
| ret addr(2) |
| abp(2) |
| abp(1) |
| AR1 |

1. 有如下中间代码序列，其中{a,b,c,d}为局部变量，{i,j}为形参，{t1,t2,t3,t4,t5,t6}为临时变量（15分）：

a = i

b = 0

c = 0

d = 0

\_loop\_begin:

cmp a, j

bge \_loop\_end //a>=j

t1 = a \* a

b = t1 + j

t3 = b + c

c = t3

t4 = a \* a

t5 = t4 + i

d = t5

c = d + c

a = a + 1

br \_loop\_begin // jmp \_loop\_begin

\_loop\_end:

return c

* 1. 对图中代码序列划分基本块，构建流图。
  2. 对循环体所在基本块利用DAG图做“局部公共子表达式删除”优化，并根据启发式算法给出优化后的中间代码序列。