- 一、 词法分析(30 分)

 - 1)(10 分)用 thompson 算法将上述正则表达式转成 NFA 的过程和结果;
 - 3)(10 分)用子集构造法进一步将 NFA 转成 DFA 的过程和结果;
 - 4)(10 分)进一步将 DFA 最小化的过程和结果。

注: 题干正则表达式中符号 "digit" 表示 0-9 中任意一个整数数字,为简化答题过程,可以用 digit 代表 0-9 中任意一个数字; 题干正则表达式中符号 "*"表示该元素出现大于等于 0 次。该正则表达式描述的是整数的表示方法,如+12,又如-5,又如 99。

二、 语法分析(40分)

考虑上下文无关文法 CFG。其非终结符集合为 {S, A, B}, 开始符号为 S, 终结符集合为 {id} (id 是一个整体), \$为表示输入结束的终结符。文法产生式如下

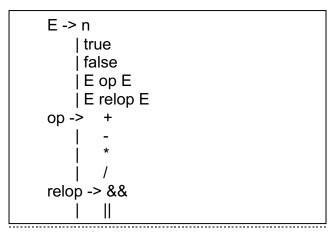
```
1: S→X$
2: X→Y id
3: Y→Y id
4: Y→id
```

回答下列问题:

- 1)(10 分) 计算非终结符 X 和非终结符 Y 的 FOLLOW 集合;
- 2)(20分)计算上述文法的 LRO 项集 DFA, 并画出其 LRO 分析表;
- 3)(10 分)上述 LR0 分析表里面是否由移进规约冲突?如有,画出其 SLR 分析表,看 SLR 分析表中是否还有移进规约冲突。
- 三、 语义分析和语法制导翻译(20分)

考虑描述表达式规则的文法 CFG, 其非终结符集合为 {E, op, relop}, 终结符集合为 {n, true,

false, +, -, *, /, &&, ||}。文法产生式如下



假设其语义规则为:每个表达式 E 都有一个类型,且 E op E 类型的表达式中两个子表达式

都必须是整型的 n, E relop E 类型的表达式中两个子表达式都必须是布尔型的 true 或 false。

回答以下问题:

- 1)(10分)给出一段伪代码,能完成上述语言的语义分析,当输入表达式语义错误时报错,否则返回输入表达式的类型(假设抽象语法树已经建立好,词法分析已完成);
- 2)(10分)用语法制导定义 SDD 的方法描述上述语义分析的语义动作,并指出该 SDD 中的属性哪些是综合属性、哪些是继承属性。
- 四、 中间表示和代码生成(10 分) 考虑语言 C-, 其文法如下

```
S -> id = E
  | if E then S else S endif
E -> n
  | id
  | true
  | false
  | E op E
  | E relop E
```

S 表示 C-语言的语句, 语句形式为赋值或 if 语句; E 表示该语言的表达式, op 表示算术操作(如+-*/), relop 表示逻辑操作(如&& □), 其他所有小写字母为终结符。

假设有一段 C-语言代码片段如下

```
if c<d then
    x=y
else
    x=z
endif
```

- 1)(6分)给出一段伪代码,能将上述 C-语言代码片段翻译成三地址码的中间表示形式(假设抽象语法树已经建立好,词法分析已完成);
- 2)(4分)给出上述代码片段翻译成的三地址码。

注: 三地址码的形式如下