中山大学计算机科学系

《编译原理》期末考试 B 卷（附参考答案与评分标准）

班级：**2004 级 A、B、C、D 班**　　专业：**计算机科学与技术专业、网络工程专业**

考试时间：**共 2 小时**　　考试形式：**闭卷**　　任课教师：**李文军、舒忠梅**

注意：所有答案必须写在答题纸上！考试后请自己保存试卷留作纪念。

警示

《中山大学授予学士学位工作细则》第六条：

“考试作弊不授予学士学位。”

Part I、词法分析（共 4 题，35 分）

1、（10 分）考虑字母表Σ = { a }上的语言，该语言包含的所有串的长度要么是 2 的倍数，要么是 3 的倍数，其中包括了空串ε。

(1) （4 分）试写出该语言的正则表达式。

(2) （3 分）画出识别该语言的不确定有限自动机（NFA）；你可直接画 NFA，不必运用机械算法从上一小题的正则表达式转换得到。

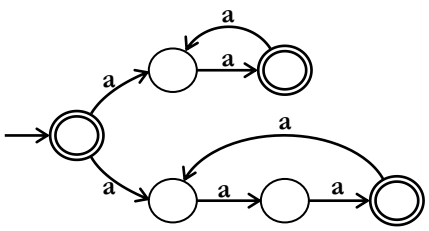
(3) （3 分）画出识别该语言的确定有限自动机（DFA）；你可直接画 DFA，不必运用机械算法从上一小题的 NFA 确定化得到。

〖参考答案〗

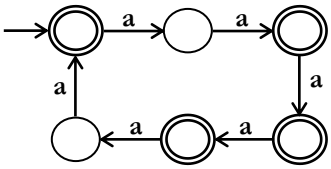
(1) 正则表达式为：

$$(aa)^* \mid (aaa)^*$$

(2) 识别该语言的 NFA 如下：



(3) 识别该语言的 DFA 如下：



〖评分标准〗
注意这一类写正则表达式的题目的答案通常不是惟一的。
(1) 将答案写作$(\mathbf{aa}) \mid (\mathbf{aaa})^*$则扣 2 分；将答案写作上下文无关文法，而不是正则表达式，即使正确也扣 3 分。
(2) 将循环弧指向起始状态导致 NFA 接收过多的字符串，则扣 2 分。
(3) 若存在一个状态有两个射出弧标记为 **a**，则扣 3 分；为一条弧标记了两个终结符号，则扣 3 分；状态数目少于 6 个，则扣 1～2 分。

2、（10 分）在 C++程序设计语言的规格说明中，如此定义 C++语言的整型常量：

> An integer constant consisting of a sequence of digits is taken to be decimal (base ten) unless it begins with **0** (digit zero). A sequence of digits starting with **0** is taken to be an octal integer (base eight). The digits **8** and **9** are not octal digits. A sequence of digits preceded by **0x** or **0X** is taken to be a hexadecimal integer (base sixteen). The hexadecimal digits include **a** or **A** through **f** or **F**…. The type of an integer depends on its … suffix. [An integer may have no suffix.] …If [the integer] if suffixed by **u** or **U**, its type is …. If it is suffixed by **l** or **L**, its type is …. [Either or both suffixes may appear at most once each in either order.]

试为 C++语言的整型常量设计一个正规定义式，要求其中仅使用正则表达式的基本算子“|”、“*”和连接，禁止使用 lex 或 flex 等软件工具扩展的表示；在正则表达式中可以使用括号和空串ε。

〖参考答案〗

$$
\begin{aligned}
&\text{Octal} \rightarrow \mathbf{0} \mid \mathbf{1} \mid \mathbf{2} \mid \mathbf{3} \mid \mathbf{4} \mid \mathbf{5} \mid \mathbf{6} \mid \mathbf{7}\\
&\text{NonZero} \rightarrow \mathbf{1} \mid \mathbf{2} \mid \mathbf{3} \mid \mathbf{4} \mid \mathbf{5} \mid \mathbf{6} \mid \mathbf{7} \mid \mathbf{8} \mid \mathbf{9}\\
&\text{Decimal} \rightarrow \mathbf{0} \mid \text{NonZero}\\
&\text{Hex} \rightarrow \text{Decimal} \mid \mathbf{a} \mid \mathbf{A} \mid \mathbf{b} \mid \mathbf{B} \mid \mathbf{c} \mid \mathbf{C} \mid \mathbf{d} \mid \mathbf{D} \mid \mathbf{e} \mid \mathbf{E} \mid \mathbf{f} \mid \mathbf{F}\\
&\text{OptU} \rightarrow \mathbf{u} \mid \mathbf{U} \mid \varepsilon\\
&\text{OptL} \rightarrow \mathbf{l} \mid \mathbf{L} \mid \varepsilon\\
&\text{Suffix} \rightarrow \text{OptU OptL} \mid \text{OptL OptU}\\
&\text{Number} \rightarrow \mathbf{0}\ \text{Octal}^* \mid \text{NonZero Decimal}^* \mid (\mathbf{0x} \mid \mathbf{0X})\ \text{Hex Hex}^*\\
&\text{Integer} \rightarrow \text{Number Suffix}
\end{aligned}
$$

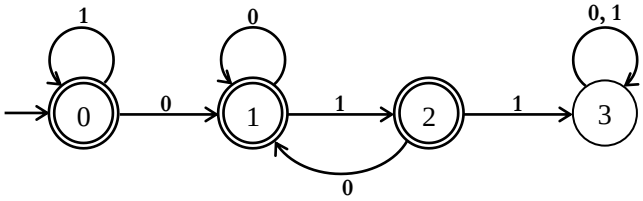
注意，这里允许$(\mathbf{0x}\ \text{Hex}^+)$这种写法。

〖评分标准〗
将正则定义式写成正则文法，则扣 8 分；Octal、NonZero、Decimal、Hex、OptU、OptL、Suffix 的定义错，则每个扣 2 分；Number 和 Integer 定义错，则每个扣 3～5 分。

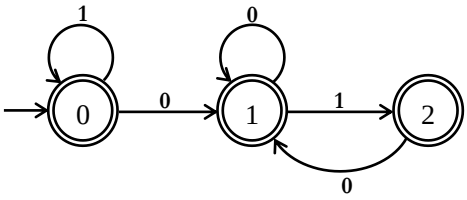
3、（10 分）构造一个 DFA，该 DFA 识别的语言是所有不含子串 **011** 的二进制数。

〖参考答案〗

解题关键：用一个非终结状态表示含有 **011** 子串的输入串，其余状态均为终结状态。下图中，状态 0 表示输入的子串尚未出现 **0**（全是 **1**）；状态 1 表示输入的子串最后一个符号是 **0**；状态 2 表示输入串已经出现了子串 **01**；状态 3 表示输入串形成了子串 **011**。



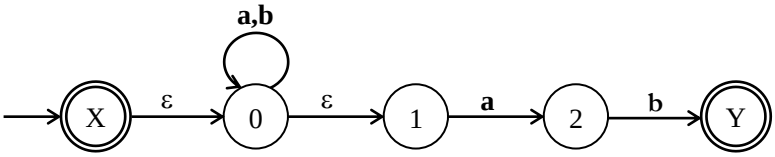
显然，图中的状态 3 是无用状态，可删除之，结果如下：



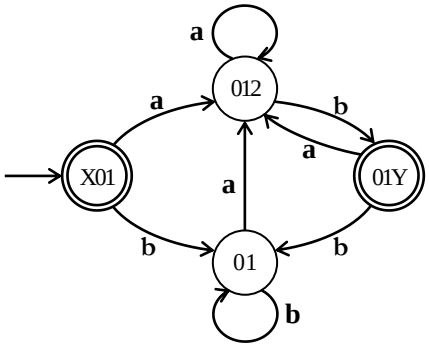
〖评分标准〗

若答案是 NFA 而不是 DFA，则扣 8 分；缺少水平的 0、1 通路，则扣 3～5 分。

4、（5 分）试将以下 NFA 确定化为 DFA，并在 DFA 的状态中标注与原 NFA 对应的状态子集：



〖参考答案〗



〖评分标准〗

未在 DFA 中标注原状态子集则扣 2 分；未标注初始状态则扣 1 分；未正确标注终结状态则每个扣 1 分。

Part II、语法分析（共 3 题，40 分）

5、（8 分）考虑以下文法：

S → **[** S X **]** | **a**
X → ε | **+** S Y | Y **b**
Y → ε | **−** S X **c**

(1) （3 分）计算该文法所有非终结符号的 FIRST 和 FOLLOW 集，并将计算结果填入下表（下表中的第一个单元已经预先填好）：

	FIRST(α)	FOLLOW(α)
S	**[a**	
X		
Y		

(2) （3 分）填写该文法的 LL(1)分析表（其中的第一个单元已经预先填写好）：

	a	**b**	**c**	**+**	**−**	**[**	**]**	**$**
S	**a**							
X								
Y								

(3) （2 分）该文法是否 LL(1)的？请解释你的回答。

〖参考答案〗

(1) 所有非终结符号的 FIRST 和 FOLLOW 集计算结果如下：

α	FIRST(α)	FOLLOW(α)
S	**[a**	**+ − b c] $**
X	**+ − b** ε	**c]**
Y	**−** ε	**b c]**

(2) 该文法的 LL(1)分析表如下：

	a	**b**	**c**	**+**	**−**	**[**	**]**	**$**
S	**a**					**[** S X **]**		
X		Y **b**	ε	**+** S Y	Y **b**		ε	
Y		ε	ε		**−** S X **c**		ε	

(3) 该文法是一个 LL(1)文法，因为在上一小题构造的分析表中未出现冲突。

〖评分标准〗

(1) 每行占 1 分；错一个符号当作整个 FIRST 或 FOLLOW 错，扣 1 分。
(2) 每错一个产生式则扣 1 分，扣完 3 分为止。

(3) 判断错误则扣 2 分；原因未解释清楚则扣 1 分。

6、（8 分）以下(1)～(3)分别描述了 3 个识别字母表{ **a** }上某文法的所有活前缀的 DFA，每一 DFA 状态中包含的是 LR(0)项目，即有效项目不涉及向前看符号。试根据所描述的 DFA 构造出原文法，使得所描述的 DFA 正是识别该文法的所有活前缀的 DFA。注意不必用产生式 S'→S 拓广文法，只须找出一个可行的文法即可。

(1) （2 分）DFA 仅由一个状态组成，且没有任何变迁。

(2) （3 分）DFA 由两个状态 s_1 和 s_2 组成，只有一个变迁从初始状态 s_1 到达 s_2。

(3) （3 分）DFA 由三个状态 s_1、s_2 和 s_3 组成，有一个变迁从初始状态 s_1 到达 s_2，有一个变迁从状态 s_2 到达自身，还有一个变迁从状态 s_2 到达 s_3。

〖参考答案〗

(1) 原文法是

S → ε

该状态仅包含项目 S→•。

(2) 原文法是

S → **a**

状态 s_1 仅包含项目 S→•**a**，状态 s_2 仅包含项目 S→**a**•。

(3) 本小题有几个类似的答案都是正确的。譬如构造原文法为：

S → **a**S

该文法产生的语言是空集∅。状态 s_1 仅包含项目 S→•**a**S，状态 s_2 包含项目 S→**a**•S 和 S→•**a**S，状态 s_3 仅包含项目 S→**a**S•。

亦可回答以下答案：

S → **a**S | ε

该文法产生的语言是由 0 个或多个 **a** 组成的串。状态 s_1 包含项目 S→•**a**S 和 S→•，状态 s_2 包含项目 S→**a**•S 和 S→•**a**S，状态 s_3 仅包含项目 S→**a**S•。

亦可回答以下答案：

S → **a**S | **a**

该文法产生的语言是由 1 个或多个 **a** 组成的串。状态 s_1 包含项目 S→•**a**S 和 S→•**a**，状态 s_2 包含项目 S→**a**•S、S→**a**•、S→•**a**S 和 S→•**a**，状态 s_3 仅包含项目 S→**a**S•。

〖评分标准〗

(1) 错一个产生式扣 2 分。

(2) 错一个产生式扣 2 分。

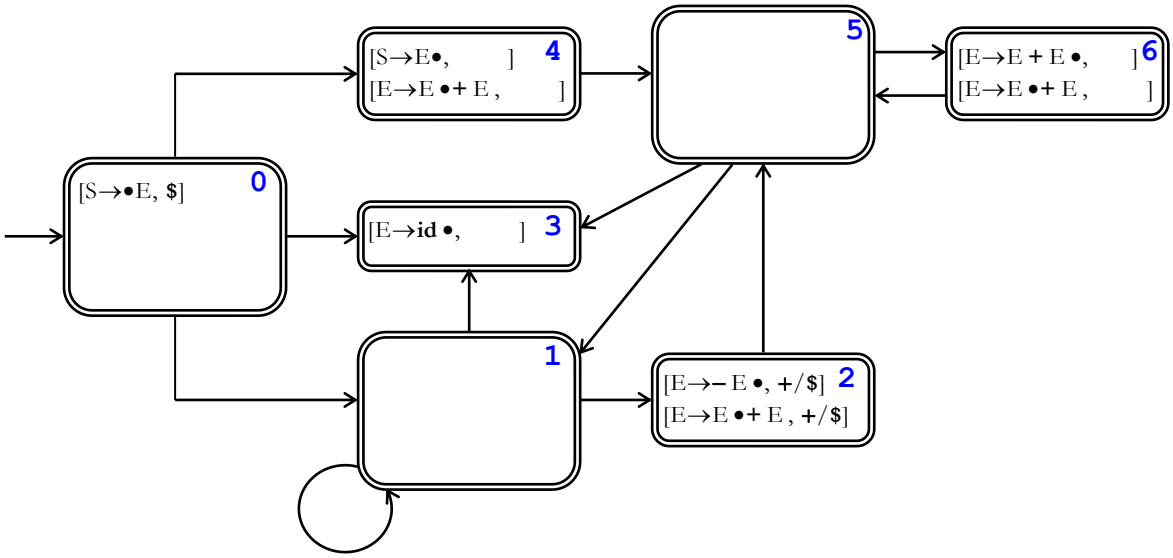
(3) 错一个产生式扣 2 分。

7、（24 分）考虑以下文法：

S → E

E → E + E | − E | **id**

以下是识别该文法所有活前缀的 DFA 的一个局部图：



第 I 部分、LR(1)分析（10 分）

(1) （8 分）完成上述 DFA，具体包括：

(1a) （1 分）计算状态 0 中已有有效项目的闭包并完成状态 0 的填写。

(1b) （4 分）填写状态 1 和状态 5 中的元素；填写状态 3、状态 4 和状态 6 中的向前看符号集。

(1c) （1 分）填写所有变迁上遗漏的符号。

(1d) （2 分）在该 DFA 含有归约项目的状态旁边标识“*reduce by P on x, y, ...*”，表示在该状态见到 *x, y, ...*等向前看符号时用产生式 *P* 归约。

(2) （2 分）对每一个含有冲突的状态，列出状态的编号、引起冲突的输入符号、以及冲突的类型（“移进－归约”冲突、“归约－归约”冲突）。

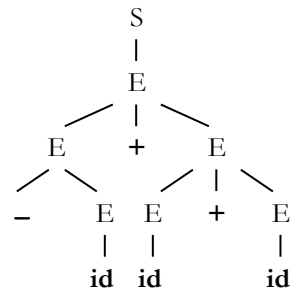
第 II 部分、文法二义性（7 分）

(3) （6 分）画出句子 **id + − id + id** 的所有分析树。

(4) （1 分）该文法是一个二义文法吗？为什么？

第 III 部分、解析二义性（7 分）

假设我们想让句子**– id + id + id** 只有以下一棵分析树是合法的（以下将此称为性质 *P*）：



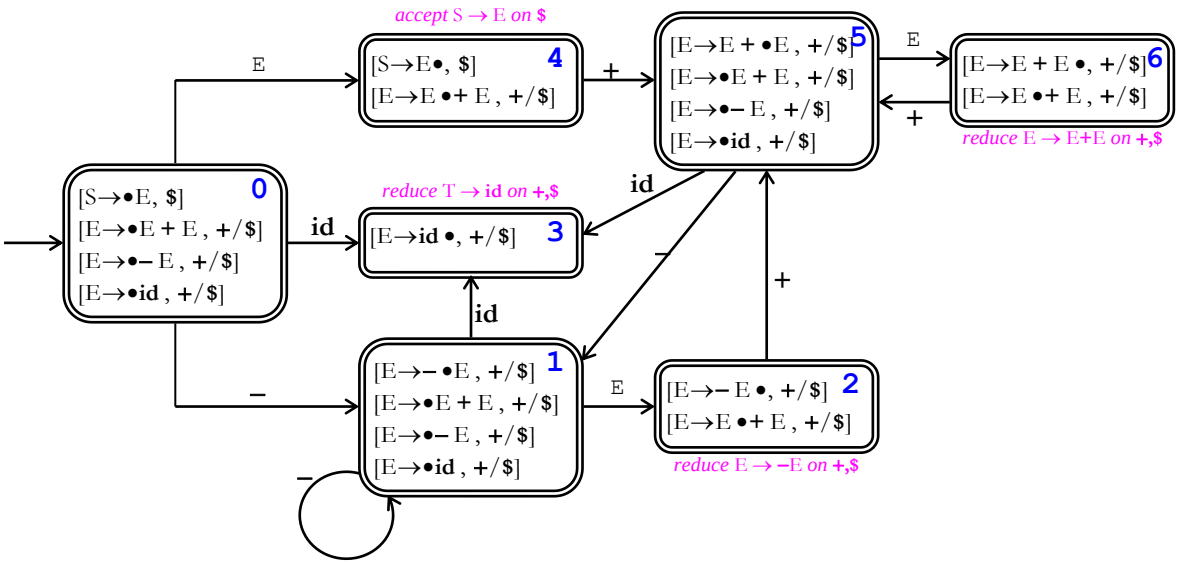
(5) （2 分）用自然语言描述：为保证性质 *P*，相关算符的优先级和结合性质的规则如何？

(6) （2 分）为保证性质 *P*，根据上述 DFA 构造的 LR(1)分析表中的冲突应如何解析，即在“移进－归约”冲突中选择移进还是归约、在“归约－归约”冲突中选择哪一个产生式归约。

(7) （3 分）写出一个与原文法等价、但能够保证性质 *P* 的无二义文法。

〖参考答案〗

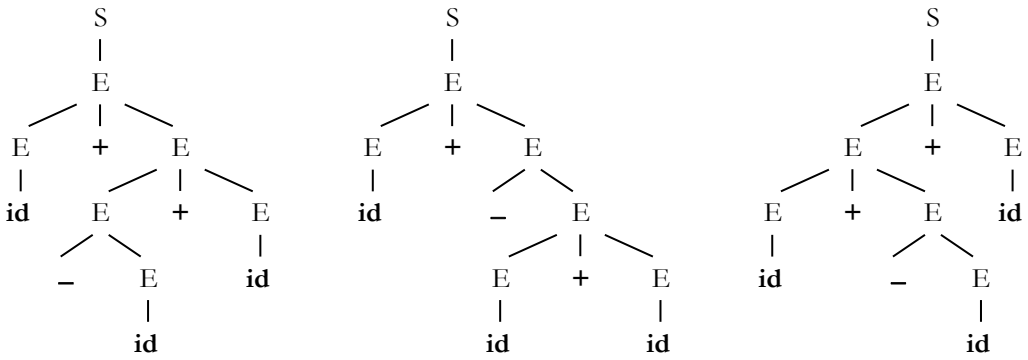
(1) 完整的 DFA 如下：



(2) 存在以下冲突：

状态	输入符号	冲突类型
2	+	“移进－归约”冲突
6	+	“移进－归约”冲突

(3) 句子 **id + – id + id** 有 3 个分析树：



(4) 该文法显然是一个二义文法，因为上一小题中句子 **id + – id + id** 有多个合法的分析树。

(5) 为保证性质 *P*，我们需要规定算符“–”的优先级要高于算符“+”，并且算符“+”是右结合的。

(6) 为保证性质 *P*，按以下方式解析冲突：

状态	输入符号	冲突类型	为解析冲突选择
2	+	“移进－归约”冲突	归约
6	+	“移进－归约”冲突	移进

(7) 与原文法等价且能保证性质 *P* 的无二义文法如下：

S → E
E → T + E | T
T → – T | **id**

〖评分标准〗
(1) 按各小项标准扣分，错一个项目至少扣 1 分，直至小项分数扣完为止。
(2) 每个冲突占 1 分；无论状态错还是向前看符号或冲突类型错均扣 1 分。
(3) 每个分析树占 2 分；分析树画错则扣 1～2 分。
(4) 无论判断错还是解释错均扣 1 分。
(5) 优先级和结合性质均占 1 分。
(6) 每个解析冲突的办法各占 1 分；在第(2)小题未正确列出冲突而导致此处错误也扣分。
(7) 未引入一个类似 T 的非终结符号则扣 3 分；使用左递归而不是右递归则扣 2 分。

Part III、语法制导翻译（共 1 题，15 分）

8、（15 分）设有如下语言：

```
Expr → for id := int to int do Expr
     |  id := Expr
     |  Expr ; Expr
     |  Expr * Expr
     |  Expr + Expr
     |  id
     |  int
```

定义一个表达式的开销是其所有子表达式的开销再加上表达式本身计算的开销。例如，表达式 **a** + E 的开销是 1 加上子表达式 **a** 的开销和子表达式 E 的开销，也可以简单地说加法的额外开销是 1；类似地，定义乘法的额外开销是 2，顺序语句的额外开销是 0，赋值的额外开销是 1，循环的额外开销是 3 再加上循环中每一次迭代的子表达式的额外开销，每一个变量 **id** 与常量 **int** 的额外开销均为 1。

(1) （13 分）给出一个计算上述开销函数定义的语法制导定义。你可假设有一个属性 val 包含单词的词法分析值，此外你也可按自己的需要定义其他属性。

(2) （2 分）请说明你在语法制导定义中引入的每一个属性是综合属性还是继承属性。

〖参考答案〗

注意这一类题目通常存在多种合理的答案。

(1) 计算表达式开销的语法制导定义：

语法规则	语义规则

Expr → **for id :=** int_1 **to** int_2 **do** $Expr_1$	**if** (int_2.val >= int_1.val) **then**
	  Expr.cp := (int_2.val - int_1.val + 1) * $Expr_1$.cp + 3;
	else
	  Expr.cp := 3;
	endif;

Expr → **id :=** $Expr_1$	Expr.cp := $Expr_1$.cp + 1;
Expr → $Expr_1$ **;** $Expr_2$	Expr.cp := $Expr_1$.cp + $Expr_2$.cp;
Expr → $Expr_1$ ***** $Expr_2$	Expr.cp := $Expr_1$.cp + $Expr_2$.cp + 2;
Expr → $Expr_1$ **+** $Expr_2$	Expr.cp := $Expr_1$.cp + $Expr_2$.cp + 1;
Expr → **id**	Expr.cp := 1;
Expr → **int**	Expr.cp := 1;

(2) 上述语法制导定义中，所有属性（包括新引入的 Expr.cp）都是综合属性。

〖评分标准〗

(1) 将语法制导定义写成翻译模式则扣 2 分；第 1 个产生式的语义动作没有判断则扣 2 分，未能计算出循环次数则扣 4 分，其他错则扣 2～3 分；第 2～4 个产生式的语义动作错则每个扣 2 分；最后 2 个产生式的语义动作错则各扣 1 分。

(2) 答错扣 2 分。

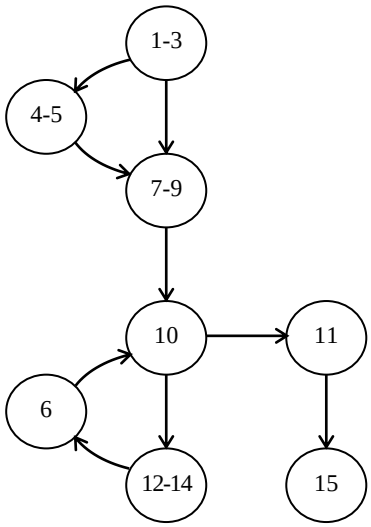
Part IV、代码优化（共 2 题，10 分）

9、（4 分）考虑以下三地址码片断：

```
(1)            b := 1
(2)            b := 2
(3)            if w <= x goto B
(4)            e := b
(5)            jump B
(6)     A:     jump D
(7)     B:     c := 3
(8)            b := 4
(9)            c := 6
(10)    D:     if y <= z goto E
(11)           jump End
(12)    E:     g := g + 1
(13)           h: = 8
(14)           jump A
(15)    End:   h := 9
```

为上述代码片断划分基本块，并画出该代码片断的控制流图（*Control Flow Graph*，简称 CFG）。在答案中你可以直接画出 CFG，但请务必在 CFG 的每一结点中用 *n*～*m* 表示该基本块由第 *n* 至 *m* 条指令组成。

〖参考答案〗



〖评分标准〗

每一基本块（CFG 中的结点）划分错误至少扣 2 分；控制流向每个错误扣 1 分，扣完 4 分为止。

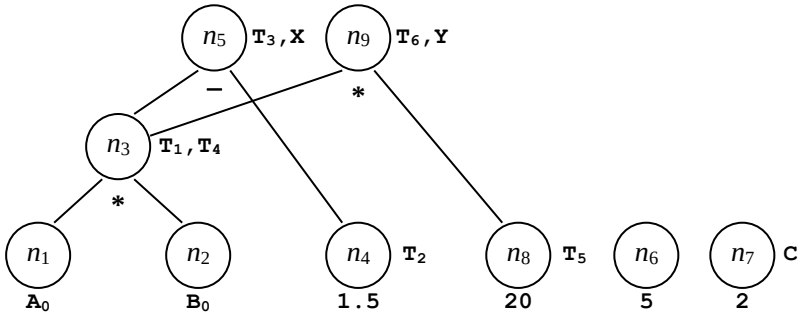
10、（6 分）设某基本块的代码序列如下：

```
(1)  T1 := A * B
(2)  T2 := 3 / 2
(3)  T3 := T1 - T2
(4)  X := T3
(5)  C := 5
(6)  T4 := A * B
(7)  C := 2
(8)  T5 := 18 + C
(9)  T6 := T4 * T5
(10) Y := T6
```

利用 DAG 优化该基本块。要求首先根据基本块产生 DAG，然后将 DAG 转换为基本块的代码序列。

〖参考答案〗

(1) 构造 DAG：



(2) 根据 DAG 生成基本块的代码序列：

```
(1)  T1 := A * B
(2)  T4 := T1           // 这类复写语句的次序可有多种
(3)  T2 := 1.5
(4)  T3 := T1 - 1.5     // 若为 T3 := T1 - T2 则错
(5)  X := T3
(6)  C := 2
(7)  T5 := 20
(8)  T6 := T1 * 20      // 若为 T6 := T1 * T5 则错
(9)  Y := T6
```

〖评分标准〗

根据基本块（BB）转换为 DAG 占 4 分，每一结点结构错则扣 1 分，扣完为止；将 DAG 转换为 BB 占 2 分，每错一条指令则扣 1 分，扣完为止。

Part V、附加题（共 2 题，10 分）

11、（5 分）考察语言 $L = \{ \mathbf{a}^n \mathbf{b}^m \mathbf{c}^n \mathbf{d}^m \mid n \geq 0 \wedge m \geq 0 \}$，该语言是正则表达式 $\mathbf{a}^* \mathbf{b}^* \mathbf{c}^* \mathbf{d}^*$ 定义的语言中满足 **a** 和 **c** 的个数相等、且 **b** 和 **d** 的个数相等的那些串。有学者证明了该语言不是一个上下文无关语言。

(1) （3 分）语言 *L* 是程序设计语言中什么问题的抽象？

(2) （2 分）该语言不是上下文无关的，这对于我们使用 BNF 定义程序设计语言的语法规则有什么启示？

〖参考答案〗

(1) 该语言是关于子程序（过程、函数、方法等）在声明时的形式参数与在调用时的实际参数之间个数保持一致性问题的抽象，$\mathbf{a}^n \mathbf{b}^m$ 代表两个子程序声明的形式参数表中分别有 *n* 个和 *m* 个参数，$\mathbf{c}^n \mathbf{d}^m$ 分别代表这两个子程序调用时的实际参数表。

(2) 由于 BNF 等价于上下文无关文法，这意味着我们使用 BNF 定义一门语言的语法规则时，无法表达形式参数与实际参数的一致性约束，故在语法定义中没有必要涉及参数的个数，无论是子程序的声明还是子程序的调用。因而，形式参数与实际参数的个数一致问题（以及类型匹配问题）是放在语义分析阶段考虑的。

〖评分标准〗
(1) 能回答参数传递个数一致性规则则加 2 分；正确地解释了理由则加 1 分。
(2) 能回答语法规则中无法表达上述规则即加 2 分。

12、（5 分）以下题目考察你对编译原理、技术和工具的课外知识掌握情况。你只需要在以下题目中任意选做 1 题；如果你解答了多题，则以你解答的第 1 题作为评分依据。

(1) （5 分）请从计算机语言及其实现技术的角度讨论 XML 语言以及 XML Parser 的特点。

(2) （5 分）像 ANTLR 和 Coco/R 这一类编译器自动生成工具与 yacc、GNU Bison、JavaCUP 等工具的主要区别是什么？

(3) （5 分）GCC（GNU Compiler Collection）是当前较有影响力的编译器开发项目，试简要介绍 GCC 的体系结构。

〖参考答案〗

(1) XML 的总体语法框架由 SGML 定义，但一个特定 XML 的语法及部分语义约束则由 XML Schema 或 DTD 定义；XML Parser 是关于各种特定 XML 语言的通用分析器，它类似于 yacc 族工具，需要以 XML Schema 或 DTD 作为输入，但 yacc 是编译型处理（会生成一个 Parser

作为输出结果），而 XML Parser 则更像是解释执行。

(2) ANTLR 和 Coco/R 支持 Java、C#等多种语言；它们采用 LL(k)分析技术，而不是 LALR(1)分析技术。

(3) 分为 3 部分：前端为各种语言生成 AST（采用 GIMPLE 表示）；中端围绕 GIMPLE 表示执行多种代码优化；后端生成目标代码。

〖评分标准〗
(1) 写出 XML Schema 或 DTD 作用则加 3 分；写出通用的解释执行方式则加 2 分。
(2) 写出支持多种语言则加 2 分；写出采用 LL(k)分析技术则加 3 分。
(3) 写出 3 个部分则加 2 分；写出中间表示 GIMPLE 则加 3 分。

□