语法分析实验报告

涂奕腾 2020201018

一、语法规则

首先为了适应 yacc 的语法，改造语法规则如下(尽量使用左递归)：

```
CompUnits: CompUnit | CompUnits CompUnit
CompUnit: Decl | FuncDef
Decl: ConstDecl | VarDecl
ConstDecls: ConstDef | ConstDecls ',' ConstDef
ConstDecl: CONST INT ConstDecls ';'
ConstDefs: '[' ConstExp ']' | ConstDefs '[' ConstExp ']'
ConstDef: ID '=' ConstInitVal| ID ConstDefs '=' ConstInitVal
ConstInitVals: ConstInitVal | ConstInitVals ',' ConstInitVal
ConstInitVal: ConstExp | '{' '}' | '{' ConstInitVals '}'
VarDecls: VarDef | VarDecls ',' VarDef
VarDecl: INT VarDecls ';'
VarDefs: '[' ConstExp ']' | VarDefs '[' ConstExp ']'
VarDef: ID | ID '=' InitVal | ID VarDefs | ID VarDefs '=' InitVal
InitVals: InitVal | InitVals ',' InitVal
InitVal: Exp | '{' '}' | '{' InitVals '}'
FuncDef: VOID ID '(' ')' Block | INT ID '(' ')' Block | VOID ID '(' FuncFParams ')' Block
| INT ID '(' FuncFParams ')' Block
FuncFParams: FuncFParam | FuncFParams ',' FuncFParam
FuncFParam: INT ID | INT ID '[' ']' | INT ID '[' ']' LVals
Blocks: BlockItem | Blocks BlockItem
Block: '{' '}' | '{' Blocks '}'
BlockItem: Decl | Stmt
Stmt: LVal '=' Exp ';' | Exp ';' | ';' | Block | IF '(' Cond ')' Stmt | IF '(' Cond ')'
Stmt ELSE Stmt | WHILE '(' Cond ')' Stmt | BREAK ';' | CONTINUE ';' | RETURN Exp ';' |
RETURN ';'
Exp: AddExp
Cond: LOrExp
LVals: '[' Exp ']' | LVals '[' Exp ']'
LVal: ID | ID LVals
PrimaryExp: '(' Exp ')' | LVal | NUMBER
UnaryExp: PrimaryExp | ID '(' ')' | ID '(' FuncRParams ')' | '+' UnaryExp | '-' UnaryExp
| '!' UnaryExp
FuncRParams: Exp | FuncRParams ',' Exp
MulExp: UnaryExp | MulExp '*' UnaryExp | MulExp '/' UnaryExp | MulExp '%' UnaryExp
AddExp: MulExp | AddExp '+' MulExp | AddExp '-' MulExp
RelExp: AddExp | RelExp '<' AddExp | RelExp '>' AddExp | RelExp LEQ AddExp | RelExp GEQ
AddExp
EqExp:  RelExp | EqExp EQ RelExp  | EqExp UEQ RelExp
LAndExp: EqExp | LAndExp AND EqExp
```

```
LOrExp: LAndExp | LOrExp OR LAndExp
ConstExp: AddExp
```

而对于优先级的设置，运算部分我是利用%left 左结合进行人为控制(实际上似乎并不需要？)，而 if-else 部分则是通过%nonassoc 指定：

```
%left '>' '<' EQ UEQ GEQ LEQ
%left '+' '-'
%left '*' '/' '%'
```

```
%nonassoc WITHOUTELSE
%nonassoc ELSE
IF '(' Cond ')' Stmt %prec WITHOUTELSE
IF '(' Cond ')' Stmt ELSE Stmt
```

我设计了以下数据结构记录分析过程和生成语法树过程，tot 和 num 分别表示了两种不同的编号方式，tot 记录匹配时创建节点时的编号；num 为构造/输出语法树时各个节点的编号。Tag 记录某个节点的标签。Node 结构体记录了语法树的相关信息：某个节点有多少个子节点(siz)，子节点分别是哪些由 tot 记录(ch)，父节点由 num 记录(fa)，该节点是父节点的哪个儿子由 num 记录(which_ch)。结构定义与构造语法树如下：

```
int tot = 0;//number of tree node
int num = 0;//node number
char tag[114514][100];
typedef struct {int ch[15], siz, fa, which_ch; }Node;
Node node[114514];

void dfs(int cur){
    fprintf(f1, "node%d[label = \"", ++num);
    for(int i = 0; i < node[cur].siz; ++i){
        node[node[cur].ch[i]].fa = num, node[node[cur].ch[i]].which_ch = i;
        //输出子节点
    }
    //输出连边
    for(int i = 0; i < node[cur].siz; ++i)  if(node[node[cur].ch[i]].siz > 0)
dfs(node[cur].ch[i]);
}
```

在每一次匹配成功时，需要输出产生式、记录当前节点的标志 tag、记录语法树当前节点的儿子信息，以第一条产生式为例：

```
CompUnits: CompUnit{
        fprintf(f2, "CompUnits -> CompUnit\n");
        node[$$ = ++tot].siz = 0;
        node[$$].ch[node[$$].siz++] = $1;
        strcpy(tag[$$], "CompUnits");
    }
    | CompUnits CompUnit{
        fprintf(f2, "CompUnits -> CompUnits CompUnit\n");
        node[$$ = ++tot].siz = 0;
```

```
        node[$$].ch[node[$$].siz++] = $1;
        node[$$].ch[node[$$].siz++] = $2;
        strcpy(tag[$$], "CompUnits");
    }
    ;
```

而对于终结符，则在词法分析中更新其节点标签 tag 和 siz = 0。

二、错误处理

根据 2.sy 中的代码，我进行了以下的错误处理：

常量定义错误 Constant definition error

ConstDecl: CONST error ';'

变量定义错误 Variable definition error

VarDecl: INT error ';'

函数参数错误 Function patameter error

FuncDef：INT error Block | VOID error Block

块错误 Block error

Block: '{' error '}'

表达式错误 Statement error

Stmt: error ';'

条件错误 Condition error

Cond: error

使用 2.sy 结果如下：

```
stibiumt@ubuntu:~/Desktop/Compilers/Grammar$ ./aaa ./2.sy
syntax error:   (1, 14)  ;
Constant definition error!
syntax error:   (3, 5)   123abc
Variable definition error!
syntax error:   (5, 32)  h
Function parameter error!
syntax error:   (22, 14)         point
Statement error!
syntax error:   (24, 13)         )
Condition error!
syntax error:   (35, 11)         a
Function parameter error!
```

三、实验结果

见 result 文件夹