PL/0 编译器的语法

PB09210183 何春晖

2012.03.04

以下是根据程序流程翻译而来的语法：

```
program -> block "."

block -> blockloop statement
blockloop ->
    constblock blockloop |
    varblock blockloop |
    procblock blockloop |
    E

constblock -> "const" constloop
constloop -> constcomm constopt
constcomm -> constdeclaration constloop0 ";"
constopt  -> constloop | E
constloop0 ->
    "," constdeclaration constloop0 | E
constdeclaration -> ident "=" number

varblock -> "var" varloop
varloop -> varcomm varopt
varcomm -> vardeclaration varloop0 ";"
varopt -> varloop | E
varloop0 ->
    ", " vardeclaration varloop0 | E
vardeclaration -> ident

procblock -> "procedure" ident ";" block ";"

statement ->
    ident ":=" expression |
    "call" ident |
    "if" condition "then" statement |
    "begin" statements "end" |
    "while" condition "do" statement

statements -> statement ";" statements | ";" statements | E

condition -> "odd" expression | expression relop expression
relop -> "=" | "<>" | "<" | ">" | "<=" | ">="

expression ->
    "+" term termloop |
    "-" term termloop |
        term termloop
termloop ->
    "+" term termloop |
    "-" term termloop |
    E
term -> factor factorloop
factorloop ->
    "*" factor factorloop |
    "/" factor factorloop |
    E
factor -> ident | number | "(" expression ")"
```

此语法与 pl0.pdf 所给语法的不同之处为，此处 var 句的定义既可以用逗号分割，也可以用分号分割。

带来的问题是下述合法程序编译不过：

```
var a;
a:=1.
```

并且由于这一改动，使得语法变成了非 LL(1) 的，因为 FOLLOW(varopt)^FIRST(varloop)={ident,...} 非空。