$PL/0\epsilon$ 思考题解答

Jinghui Hu

<2024-03-20 Wed>

1. 语法分析
2. 语义分析
3. 代码优化

1

语法分析

思考题 1： 文法歧义

1. 造成文法歧义的本质是没有对运算的优先级进行限制
2. 参考 $PL/0\epsilon$ 文法关于表达式的定义，可以通过添加括号的方式来避免歧义

$$expr \rightarrow [+|-]\ term\ \{\ addop\ \ term\ \} \quad (1)$$

$$term \rightarrow number \mid \text{‘(’}\ \ expr\ \ \text{‘)’} \quad (2)$$

$$addop \rightarrow +|- \quad (3)$$

$$number \rightarrow \text{Unsigned Number} \quad (4)$$

思考题 2： 条件文法扩充

需要扩展 $PL/0\epsilon$ 关于条件的文法来支持布尔运算

$$bool \rightarrow bool \,||\, join \mid join \tag{5}$$

$$join \rightarrow join \;\&\&\; cond \mid cond \tag{6}$$

$$cond \rightarrow expr \;\; relop \;\; expr \tag{7}$$

$$relop \rightarrow \texttt{<}\mid\texttt{<=}\mid\texttt{>}\mid\texttt{>=}\mid\texttt{=}\mid\texttt{<>} \tag{8}$$

2

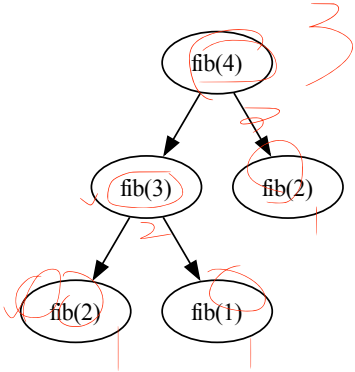
语义分析

思考题 1： 斐波那契数列

```
function fib(n : integer ): integer; begin
   if n <= 2 then
      fib := 1
   else
      fib := fib(n-1) + fib(n-2);
end;

begin
   write(fib(4))
end
.
```

思考题 1：斐波那契数列（续）



符号表是 静态 的，运行栈是 动态 的

思考题 1：斐波那契数列（续）

对应的符号表如下：

	sid	name	cate	type	value	label
tid=1	1	_start	4	1	0	L001
	2	fib	4	1	0	L002
tid=2	3	n	6	1	0	L003

可以通过 `prtir` 验证

```
./bin/prtir ./example/fib.pas
```

思考题 2： 函数重载

```
var ans : integer;
function f(x : integer):integer; begin f := x end;
function f(x, y : integer ):integer; begin f := x + y end;
begin
   ans := f(1);
   write(ans);
   ans := f(3, 4);
   write(ans)
end.
```

思考题 2：函数重载（续）

1. 基于 v1.1.8 新建一个分支 feat/overload 进行开发

   ```
   git checkout -b feat/overload v1.1.8
   ```

2. 查看修改内容

   ```
   git diff v1.1.8 feat/overload
   ```

思考题 2：函数重载（续）

```
# 编译代码， -g 表示添加调试信息
fpc -g overload.pas
# 查看程序符号表
nm overload | grep 'PROGRAM'
```

输出结果中包含：`f()` 重置函数的名称

```
pcc/example » nm overload | grep 'PROGRAM'
0000000000000158 N _$PROGRAM$_Ld1
0000000000000176 N _$PROGRAM$_Ld2
0000000000401190 T DEBUGEND_$P$PROGRAM
0000000000401090 T DEBUGSTART_$P$PROGRAM
0000000000401090 T P$PROGRAM_$$_F$SMALLINT$$SMALLINT
00000000004010b0 T P$PROGRAM_$$_F$SMALLINT$SMALLINT$$SMALLINT
000000000042e840 b U_$P$PROGRAM_$$_ANS
```

思考题 2：函数重载（续）

1. 符号表添加一些操作函数，可以处理传入待操作的符号表 `stab`
2. 定义类型简写映射
   - VOID => V, INT => I, CHAR => C, STR => S
3. 函数（或过程）进入符号表时改名，同名当时参数列表不同的对应不同 `syment_t`
   - 第一个 `f()` 函数修改成 `f$I`
   - 第二个 `f()` 函数修改成 `f$I$I`
4. 调整函数（或过程）头声明是查询符号表的时机
5. 调用函数（或过程）时，对参数列表进行类型推导

3

代码优化

思考题 1：SSA 的 DAG 图

$$
\begin{aligned}
p &\leftarrow a + b \\
q &\leftarrow p - c \\
p &\leftarrow q * d \\
p &\leftarrow e - p \\
q &\leftarrow p + q
\end{aligned}
$$

$$
\begin{aligned}
p_1 &\leftarrow a + b \\
q_1 &\leftarrow p_1 - c \\
p_2 &\leftarrow q_1 * d \\
p_3 &\leftarrow e - p_2 \\
q_2 &\leftarrow p_3 + q_1
\end{aligned}
$$