PA3实验报告

计53 方闽杰 2015011287

为了支持各个新语言特性，对各个文件所作修改如下：

1. **case表达式**

TransPass2.java：重载了visitCaseExpr和visitCaseBlock两个方法。转换为的tac代码结构如下：

Loop:

Cond = (mainExpr == Ci)

If(Cond != 0) Branch Bi

…

Branch Default

Bi:

Expr = Ei

Branch Exit

…

Default:

Expr = DefaultExpr

Exit:

1. **串行循环卫士**

TransPass2.java：重载了visitDoStmt方法。转换为的tac代码结构如下：

Loop：

If(Ci != 0) Branch Bi

…

Branch Exit

Bi:

Statements

Branch Loop

…

Exit:

此外，还需调用loopExits栈，把Exit标签适时压入弹出以支持break语句。

1. **complex**

为了给complex分配64位空间而不涉及已有tac命令的改变，决定将complex以32位指针的形式存储在Temp中。指针指向的64位空间，前32位存储实部，后32位存储虚部。

TransPass2.java：

在各个需要给变量关联上Temp的地方，若变量的类型为complex，则调用库函数Intrinsic.ALLOCATE分配空间，并将指向该段空间的指针存入Temp中关联给对应的变量；

在visitBinary中，add和mul操作处，若操作数为complex类型，则分配空间，分别取出两个操作数的Temp指向的地址中存储的实部和虚部，做加法和乘法。

在visitAssign中，进行若干次访存，将右部表达式的实部和虚部取出，存到左值的实部和虚部当中。

在visitLiteral中，分配空间，将纯虚部常量转化为0+nj形式的complex类型的值。

在visitUnary中，对于#$%这三个运算符，进行相应的访存操作。

在visitPrintComp中，遍历需要打印的表达式，对于每个表达式，访存获取实部虚部，调用库函数将其打印出来。

Translater.java：重写了getNewForClass。原来的实现不考虑各个成员函数的类型，只是计算大小，分配空间，全部赋为0。现在更改为，依据各个成员函数的类型进行初始化。如果是complex类型，则分配空间，将实部虚部赋为0，将空间地址保存在类的对象中。其他类型依旧赋值为0。

1. dcopy和scopy

TransPass2.java：重载visitSCopyExpr和visitDCopyExpr方法，新增DeepCopy方法。对于浅拷贝，新分配一段内存空间，对于待复制对象中的所有成员函数，如果不是complex类型，则直接将Temp值复制到新的内存空间中，否则新分配一个64位空间给complex，并且复制实部虚部。对于深拷贝，对于非class类型，处理与浅拷贝相同。对于class类型，则递归地调用DeepCopy进行复制。

1. super

TransPass2.java：修改visitCallExpr。因为在PA2中，callExpr的symbol已经定位在对应的父类当中，因此直接查找这个symbol的scope的owner的虚表即可。

1. 除0错误检查

RuntimeError.java：加入除零错误的错误信息。

Translater.java：新增genCheckDivByZero方法，接收一个Temp作为参数。生成的tac代码将Temp与0比较，如果相等则报错并且调用Intrinsic.HALT终止进程。

TransPass2.java：在visitBinary中，对于div和mod操作，在生成对应的tac代码前调用上述的genCheckDivByZero方法进行检查。