Stage-2实验报告

吴垒 2020010916

###### 实验内容

修复Stage-1存留的对LogicAND指令的tac码到汇编代码翻译问题。原本的翻译过程改变了存储lhs、rhs原始值的寄存器，在按照实验文档上指令修正后解决。

本部分主要需要修改的是frontend/typecheck/namer.py 及frontend/tacgen/tacgen.py，对应符号表构建及三地址码生成。

Namer.py在Step5 中需要修改：visitDeclaration，注意这里需要访问ident.value而不是name，value对应变量名，name对应对象类型（如identifier），同时需要使用setattr（‘symbol’，symbol）设置symbol属性以便后续翻译过程中访问该节点对应的符号值；visitAssignment，首先判断当前作用域符号表是否存在左值变量名（expr.lhs.value），之后模仿Binary分别访问左右值获得val即可；visitIdentifier同样按照注释补全即可。

Tacgen.py在Step5中需要修改：visitIdentifier，直接将之前设置的symbol属性的Temp设置为其val即可；visitDeclaration，需要为decl.getattr（‘symbol’）分配一个新的寄存器，需要注意的是如果有init\_expr，需要先对其调用accept计算出其值对应的寄存器，再用其赋值到decl.getattr（‘symbol’）新分配的寄存器，最后还需将表达式的右值寄存器设置为整个decl的val；visitAssignment，先计算右值寄存器，再将其赋值到左值寄存器，最后将表达式右值寄存器赋值到整个表达式的val，与有init\_expr的declaration相似。

还需在backend/riscv/riscvasmemitter.py中添加 visitAssign语句，使用Riscv.Move即可，问答墙中已有提示。

Namer.py在Step6中需要修改：visitCondExpr，分别检查cond、then、otherwise即可。tacgen.py在Step6中需要修改：visitCondExpr，完全仿照visitIf完成，但需要注意的是区别在于If then构成的块不需要有整体的val，但条件表达式需要将表达式的整体设置一个val并赋值，可选用then的寄存器存储最终值，因此对else分支还需要将else的内容存入then分支对应的寄存器。

###### 思考题

Step5：1.扩大栈帧空间

addi, sp, sp, -16

Step5：2.支持重复定义后的代码修改

在重复定义后，符号表内不能简单地覆盖原本已存在符号，因为已存在符号的调用与重新定义后的调用不为同一调用；故需要在每次声明一个符号后为符号表内该符号添加一个唯一索引，表示其在当前作用域内的声明次数。

同时对现有程序查询到同名声明报错，应不再报错而是直接向符号表添加一个新的声明。

对于一个函数下第x次声明符号，其实际作用域应变为从其所在函数下的第x次声明开始，到第x+1次声明前结束（不存在x+1次声明时即到函数结束）。

对于存在init\_expr的变量声明，需先访问init\_expr以防循环调用。

Step6：1.悬吊else？

If、else语句的语法分析工具主要存在于ply\_parser.py中，对应处理函数为p\_if\_else及p\_if，对应statement有两种状态：statement\_matched、statement\_unmatched。设置If……else……语句块下嵌跟在If后的语句一定为matched，而Unmatched状态下才会使用If……匹配。这样保证了对于样例If……If……else……的情况下，读到第一个If后状态为unmatched，读到第二个If后也为unmatched，再读入else时若与第一个If匹配，则会将第二个If所在块变为matched，但单独If只能出现在unmatched块内，故else只会与第二个If匹配。

Step6：2.条件表达式不短路的修改？

将then、otherwise的访问过程（accept）直接挪到visitCondBranch之前即可。

###### 借鉴内容

实验思路指导与问答墙