Compiler Lab5 LLVM

201250104 苏致成

December 2022

1 概述

1.1 目标

使用 LLVM 将函数和局部变量翻译为中间代码 LLVM IR。

2 实现

2.1 使用工具

git、Antlr、Intellij idea、JDK11、Makefile、LLVM。

2.2 实现功能

向 main 方法中传递文件参数与输出路径参数，其余过程详见“目标”。

2.3 实现过程

1. 初始化 LLVM，创建 module，初始化 builder。
2. 重命名基本类型，创建常量。
3. 重写 visitFuncDef，声明函数形参、返回值等，标记当前作用域存放此函数。
4. 重写 visitVarDef，区别单个数与数组的类型，标记当前作用域存放此局部变量。
5. 重写 visitConstDef，与上述基本一致。
6. 重写 visitReturnStmt。
7. 重写 visitLvalExp，此时 lval 处于右值位置。
8. 重写 visitAssignStmt，此时 lval 处于左值位置。
9. 重写 visitCallFuncExp，传入实参，调用函数。

3 遇到障碍

3.1 重新赋值

问题描述：对左值进行重新赋值时，出现类似“左值非指针”的报错。

解决方式：将原先 LLVMBuildAlloca 的返回值放入当前作用域予以记录，并非 LLVMBuildStore 的返回值。

3.2 函数返回值错误

问题描述：根据 clang 语句生成的结果来看，若 int 作为返回值的函数没有 return 指令 LLVM 会生成 return i32type 0; 的类似指令；若 void 作为返回值的函数，LLVM 会生成 return void; 的指令

解决方式：在 visitFuncDef 函数末尾判断上述两种情况，判断是否需要加上该指令。

3.3 callFuncExp

问题描述：如果调用函数并且没有 return 语句，LLVM 会默认将调用语句的返回值构建 return 语句，导致返回值类型不匹配等问题。

解决方式：在调用的时候，给返回值的类型赋予空字符串即可。

3.4 GEP 相关

问题描述：不清楚如何使用 GEP。

解决方式：翻阅手册，如 varDef 时即给数组类型赋予初值的情况处理如下：

```
LLVMValueRef vectorPointer = LLVMBuildAlloca(builder, vectorType, ctx.IDENT().getText())
LLVMValueRef[] refs = new LLVMValueRef[2];
refs[0] = constDigit[0];

for (int j = 0; j < vecSize; j++) {
    refs[1] = LLVMConstInt(i32Type, j, 0);
    PointerPointer<Pointer> pp = new PointerPointer<>(refs);
    LLVMValueRef pointer = LLVMBuildGEP(builder, vectorPointer, pp, 2, "pointer");
    LLVMBuildStore(builder, initVals[j], pointer);
}
```

3.5 无法获取数组定义的大小

问题描述：如 int a[3]=1,2; 时，由于 3 为 exp，返回值为 LLVMValueRef，无法将其强转为整数类型。

解决方式：查看手册，发现助教保证了 3 这个位置为 exp 的情况只可能是 const，因此将其强转为 int 类型。

PS：后经提醒，LLVMConstValueRef 应该可以将其值提取出来，但未予尝试。

3.6 funcDef 时，作用域出错

问题描述：后续函数获取值时，无法获取形参的值。

解决方式：在 visitFuncDef 中，应当先进入定义域中，再将形参加入当前作用域。而进入定义域之前，需要 paramsType，所以相当于即便同样是遍历形参列表，也要分开两次遍历。

3.7 lval 处理

问题描述：lval 作为左值时，需要获取其指针，亦其分配空间的位置，并非其值；lval 作为右值时，需要获取其内容，并非其位置。

解决方式：删除 visitLval ，lval 作为右值时，将其逻辑放入 visitLvalExp 中解决；lval 作为左值时，将其逻辑放入 visitAssignStmt 中解决。同时左值放入作用域的表中时是放入其指针，若其作为右值访问时再 LLVMBuildLoad 即可。