Compiler Lab8 LLVM

201250104 苏致成

January 2023

1 概述

1.1 目标

使用 LLVM 将函数定义、函数调用翻译为中间代码 LLVM IR，允许数组作为函数的参数。

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. 在 visitFuncDef 中，如果形参是数组，则设置为指针类型，并且在当前作用域下记录该指针。
2. 在 visitLvalExp 中，如果当前符号是指针，则修改对应的 GEP 指令，如果是数组，则维持原样。
3. 在 scope 中，新增函数 putPointer、getPointer、getArrays，用以判断是指针类型还是数组类型。
4. 在 visitAssignStmt 中，进行和上述 visitLvalExp 类似处理。

以下举出部分处理指针和数组不一致的核心代码（选自 visitCallFuncExp() 函数）：

```
if (currentScope.getArrays(token)) {
    LLVMValueRef[] tmp = new LLVMValueRef[2];
    tmp[0] = constDigit[0];
    tmp[1] = constDigit[0];
    PointerPointer<LLVMValueRef> pp = new PointerPointer<>(tmp);
    refs[i] = LLVMBuildGEP(builder, currentScope.resolve(token), pp, 2, "arr");
} else if (currentScope.getPointer(token)) {
    LLVMValueRef arrayPointer = currentScope.resolve(token);
    refs[i] = LLVMBuildLoad(builder, arrayPointer, "arrPtr");
} else {
    refs[i] = this.visit(paramCtxs.get(i));
}
```

3 遇到障碍

3.1 返回值问题

问题描述：会出现如下报错：lli-13: lli: out.ir:7:9: error: stored value and pointer type do not match store i32 %0, i32** %i32Array, align 4

解决方式：主要原因是之前的实验中默认函数形参是 i32Type 类型，但是这次是有指针类型，没有在函数的参数声明部分进行重写。

3.2 返回值问题

问题描述：会出现如下报错：error: expected instruction opcode。如果是如下代码，可能部分路径存在没有返回值的情况，导致报错：

```
int main() {
    int a = 1;
    if (a==1) {
        return 1;
    } else {
        return 2;
    }
}
```

解决方式：在 funcDef 的时候给其加上 LLVMBuildRet，而不管其在代码体内是否有显式声明。

3.3 函数传参报错

问题描述：报错如下：error: '@usic_start' defined with type 'i32 (i32*)*' but expected 'i32 ([15 x i32])*' %name = call i32 @usic_start([15 x i32] %usic_forever)

解决方式：处理全局数组，处理如下：

```
@Override
public boolean getArrays(String name) {
    if (arrays.contains(name))
        return true;
    if (enclosingScope != null)
        return enclosingScope.getArrays(name);
    return false;
}
```

3.4 函数传参报错 2

问题描述：函数指针可能继续作为实参继续调用其他函数，如下：

```
int func(int q[]){}
int b(int a[]){
    func(a);
}
```

解决方式：判断当前数组指针是否在当前作用域，注意不需要考虑父作用域。