lab2 实验报告

PB20111630 张艺耀

问题1: getelementptr

请给出 `IR.md` 中提到的两种 getelementptr 用法的区别,并稍加解释:

- `%2 = getelementptr [10 x i32], [10 x i32]* %1, i32 0, i32 %0`

  %2 获取一个int [10]类型数组的第%0个元素地址（指向int [10]类型数组第%0个元素的指针）

- `%2 = getelementptr i32, i32* %1 i32 %0`

  %2 获取一个%1地址加上偏移量为%0的地址

问题2: cpp 与 .ll 的对应

请说明你的 cpp 代码片段和 .ll 的每个 BasicBlock 的对应关系。

cpp代码片段中`builder->set_insert_point(bbName)`语句作为划分每个基本块的起始语句，两个此语句之间的片段对应.ll文件中的每个BasicBlock。

```
auto mainFun = Function::create(FunctionType::get(Int32Type, {}),
                                "main", module);
auto bb = BasicBlock::create(module, "entry", mainFun);
builder->set_insert_point(bb);
```

- 以上cpp语句生成main函数入口（其他函数类似），相当于.ll文件中的每个函数的首个BasicBlock，此BasicBlock以下一个BasicBlock的开头的前一条语句或者整个代码的最后一条语句作为最后一条语句。

```
// true
   builder->set_insert_point(trueBB);
   builder->create_store(CONST_INT(233), retAlloca);
   builder->create_br(retBB);
```

- 对于含有条件或循环语句的代码，cpp代码中每个branch操作（`create_cond_br`或`create_br`)的开始对应于一个.ll文件中BasicBlock的开始，此BasicBlock以下一个BasicBlock的开头的前一条语句或者整个代码的最后一条语句作为最后一条语句。

问题3: Visitor Pattern

分析 `calc` 程序在输入为 `4 * (8 + 4 - 1) / 2` 时的行为：

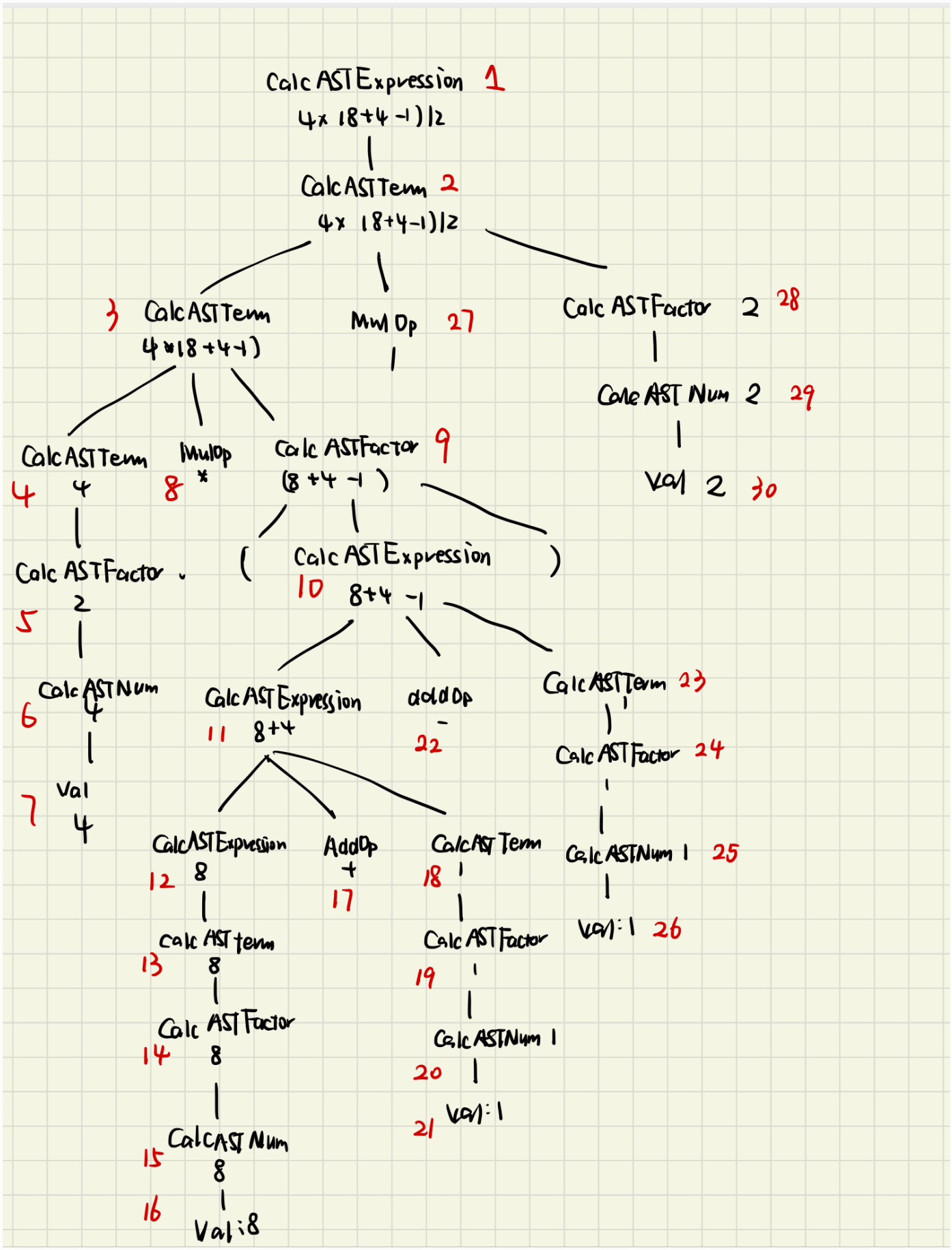
1. 请画出该表达式对应的抽象语法树（使用 `calc_ast.hpp` 中的 `CalcAST*` 类型和在该类型中存储的值来表示），并给节点使用数字编号。
2. 请指出示例代码在用访问者模式遍历该语法树时的遍历顺序。

序列请按如下格式指明（序号为问题 2.1 中的编号）：

3->2->5->1

如图 访问顺序由红色数字标出（由于LP和RP在calc_ast.cpp文件中直接被略过，故不计入）

```
else if (_STR_EQ(n->name, "factor")) {
        if (n->children_num == 3) {
            return transform_node_iter(n->children[1]); //直接略过了LP和RP
        } else {
            ...
        }
```


CalcASTExpression 1
4x 18+4-1)12
CalcASTTerm 2
4x 18+4-1)12
3 CalcASTTerm
4*18+4-1)
MulOp 27
CalcASTFactor 2 28
CalcASTNum 2 29
Val 2 30
CalcASTTerm 4 4
4
MulOp 8 *
CalcASTFactor 9
(8+4-1)
CalcASTFactor 2
5
(CalcASTExpression)
10 8+4-1
CalcASTNum 4
6
Val 4
7
CalcASTExpression 11 8+4
AddOp 22 -
CalcASTTerm 23
CalcASTFactor 24
CalcASTNum 1 25
Val:1 26
CalcASTExpression 12 8
AddOp 17 +
CalcASTTerm 18
CalcASTTerm 13 8
CalcASTFactor 19
CalcASTFactor 14 8
CalcASTNum 1 20
Val:1 21
CalcASTNum 15 8
16
Val:8

实验难点

描述在实验中遇到的问题、分析和解决方案。

在编写`.ll`文件时，遇到了一些问题。首先是标号问题，应按照数字的出现顺序对中间变量进行命名，否则会报错。

对于br指令的目的地址也有相似的问题，当使用数字作为label时，label必须要在之前先在br中出现，否则会识别不到此label。

在编写`.cpp`文件时，需要注意开辟存储空间和变量的存取问题，使用变量之前要先从存储空间中Load变量，变量的标识符最好服从实例代码的标准防止弄混。

问题3需弄清每个产生式以免出现无中生有的情况。

实验反馈

本次实验文档较为详细，看一遍例子基本上可以明白怎么编写`.ll`和`.cpp`文件。

文档中并没有对`align`有详细的解释，但猜测对齐应该对程序生成结果影响不大。

可以介绍一下共享指针。