IR自动生成-3
实验讲解

徐 伟

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年04月10日

CONTENT

- C++ 知识回顾
- LLVM 简介
- Light IR C++ 库
- IR 自动化生成框架

IR 自动化生成框架总览

- 了解 Light IR Module 类的层次化结构，IRBuilder 类创建指令的流程后
- 接下来介绍框架 CminusfBuilder 类通过访问者模式遍历AST，调用 Light IR C++ 库自动化的生成 IR

访问者模式简介

- **AST (抽象语法树) 简介**
  - 分析树在 Lab1 语法分析的过程中被构造，AST 则是分析树的浓缩表示，使用运算符作为根节点和内部节点，并使用操作数作为子节点

- **Visitor Pattern (访问者模式) 概念**
  - AST 类有一个方法接受访问者，将自身引用传入访问者，而访问者类中集成了对不同 AST 节点的访问规则

访问者模式示例

```
//Visitor.h
class AddExp;
class IntExp;
class Visitor
{
public:
    int result = 0;
    virtual void visit(AddExp*);
    virtual void visit(IntExp*);
};
```

```
//Exp.h
#include  "Visitor.h"
class Exp{
public:
    virtual void accept(Visitor&v) = 0;};

class AddExp : public Exp{
public:
    Exp* rhs;
    Exp* lhs;
    AddExp(Exp* lhs, Exp* rhs) : lhs(lhs), rhs(rhs) {}
    virtual void accept(Visitor&v) override final;};

class IntExp : public Exp{
public:
    int value;
    IntExp(int value) : value(value) {}
    virtual void accept(Visitor&v) override final;};
```

访问者模式示例

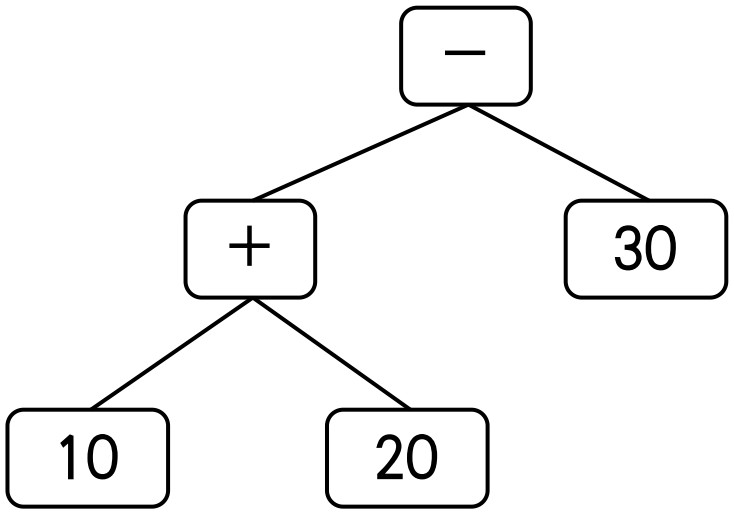
```
//Visitor.cpp
#include "Exp.h"
void Visitor::visit(AddExp* add_exp){
    add_exp->lhs->accept(*this);
    add_exp->rhs->accept(*this);}

void Visitor::visit(IntExp* int_exp){
    result += int_exp->value;}

void AddExp::accept(Visitor & v){
    v.visit(this);}

void IntExp::accept(Visitor & v){
    v.visit(this);}
```

```
//main.cpp
#include<iostream>
#include<string>
#include "Exp.h"
using namespace std;

int main(){
    Exp* exp = new IntExp(1);
    for(int i = 2; i < 4; i++){
        exp = new AddExp(exp, new IntExp(i));    }

    Visitor CalSum;
    exp->accept(CalSum);
    cout << "Result is " << CalSum.result << endl;
    return 0;        }
```

访问者模式简介

- **访问者模式例子：**
  - 10+20−30

AST 结构

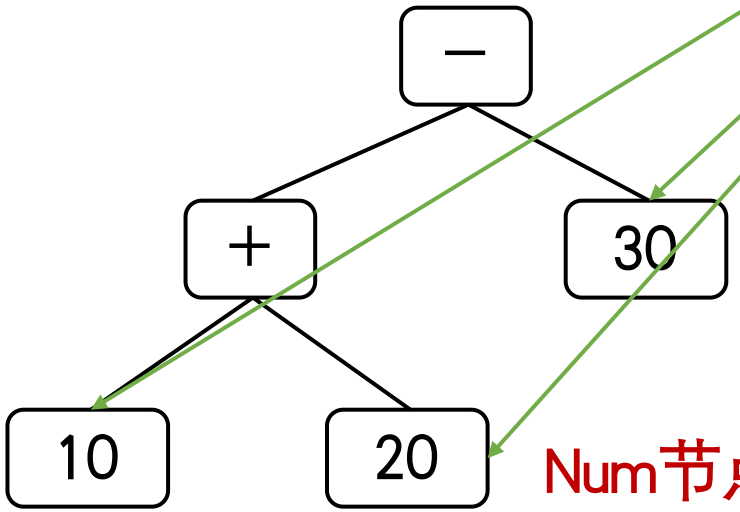


访问者类

```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

- **访问者模式例子：**
  - 10+20−30

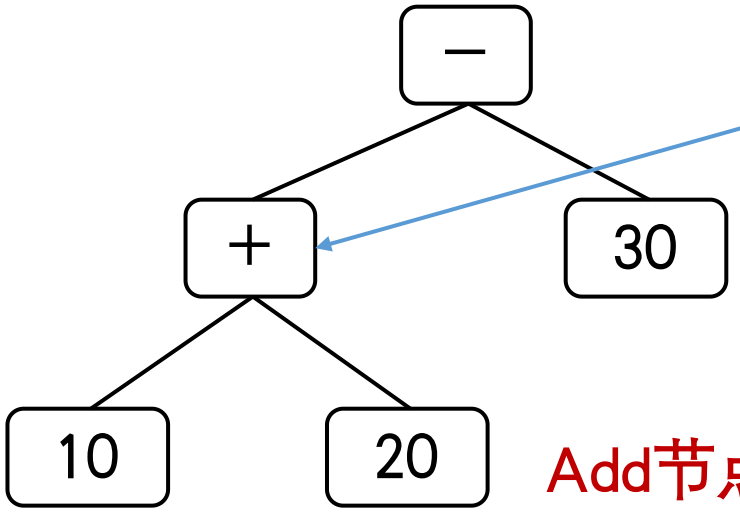


Num节点的访问规则

```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

- **访问者模式例子：**
  - 10+20−30

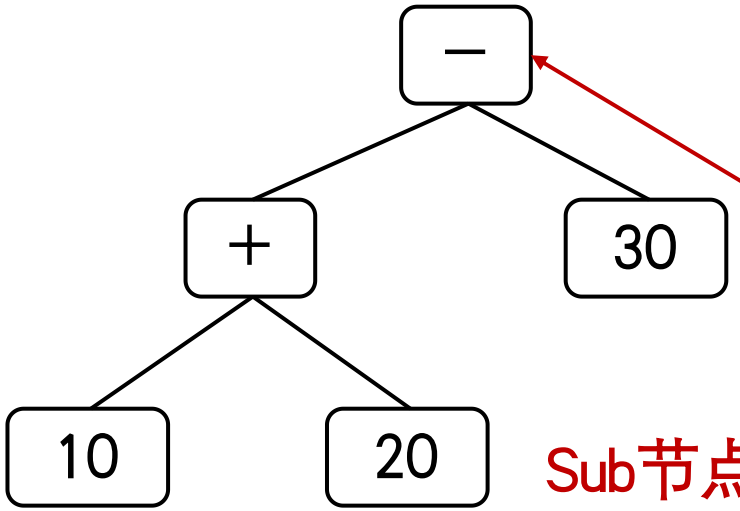


Add节点的访问规则

```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

- **访问者模式例子：**
  - 10+20−30

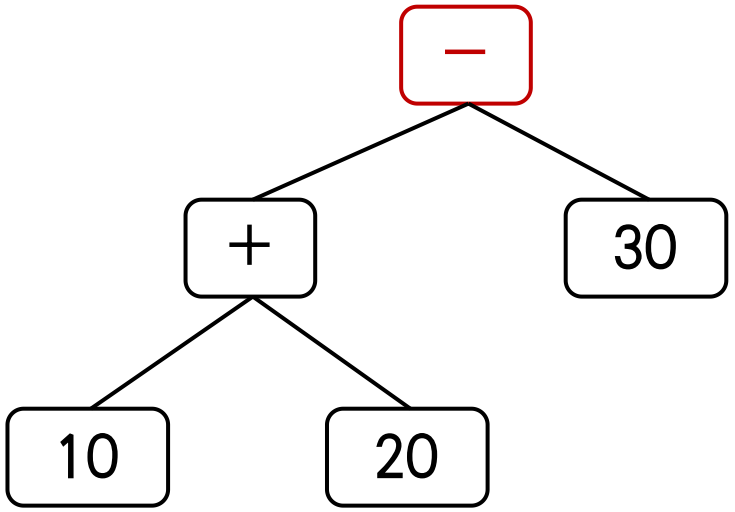


Sub节点的访问规则，且SubNode为根节点

```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

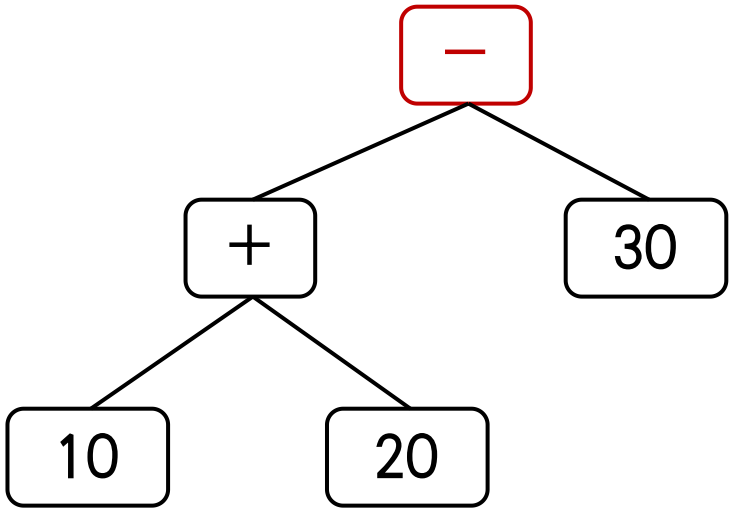
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

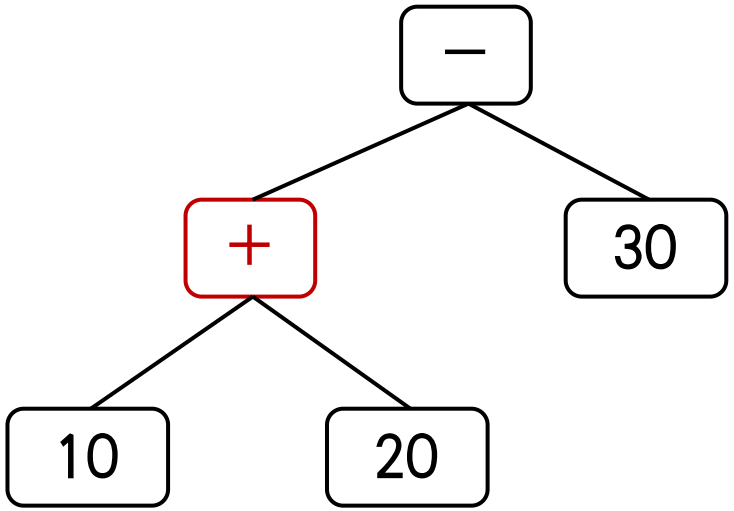
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

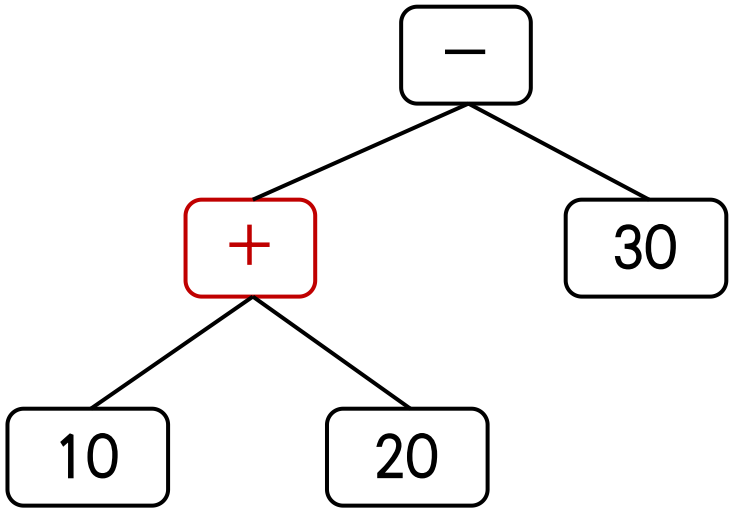
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

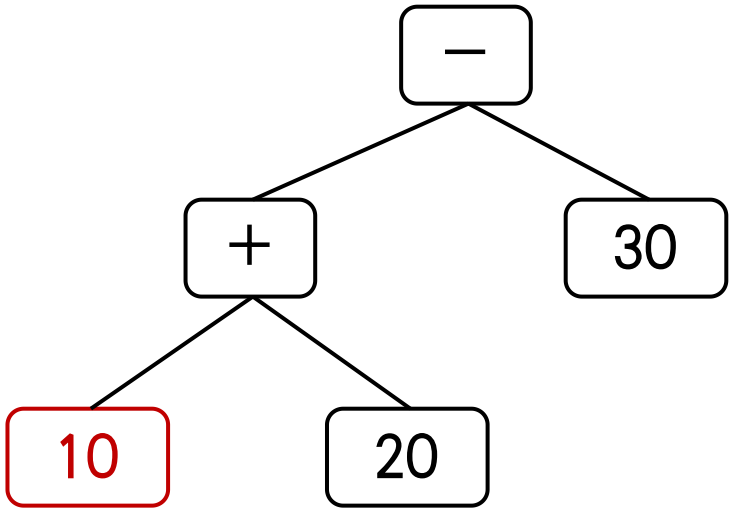
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

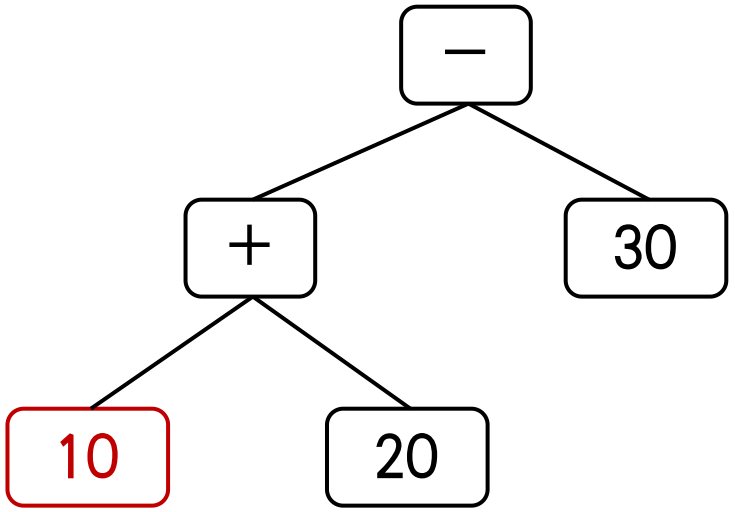
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

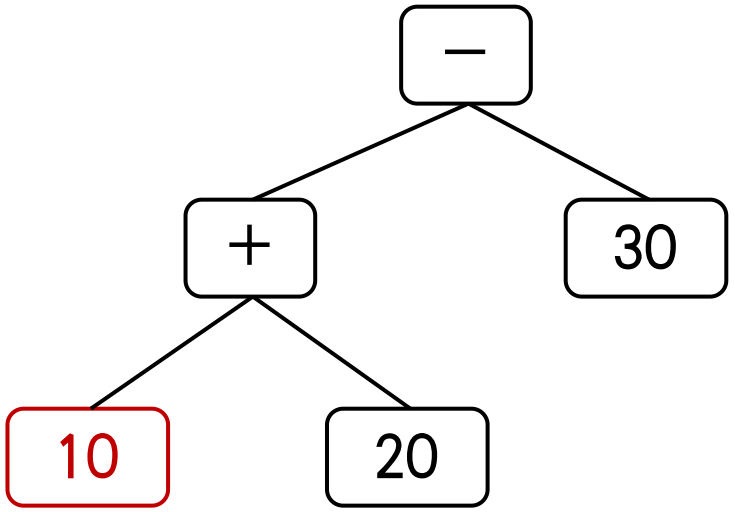
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

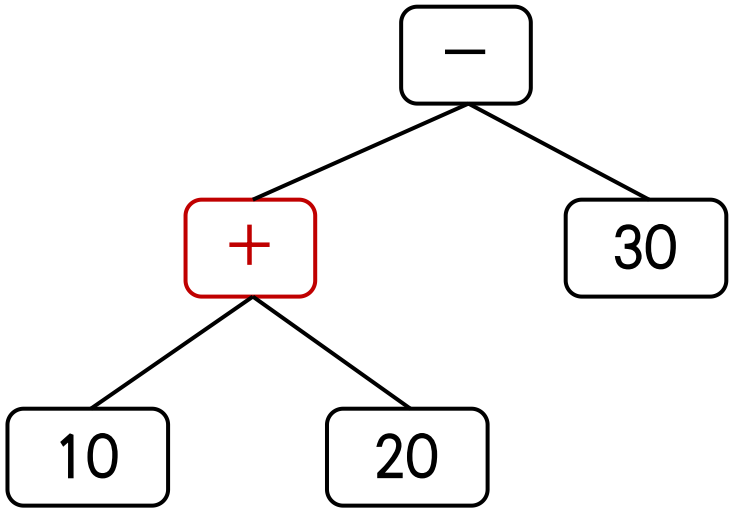
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

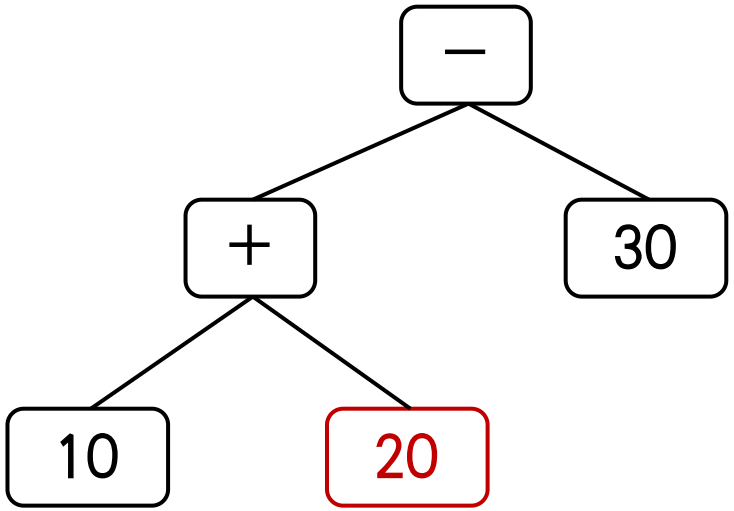
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

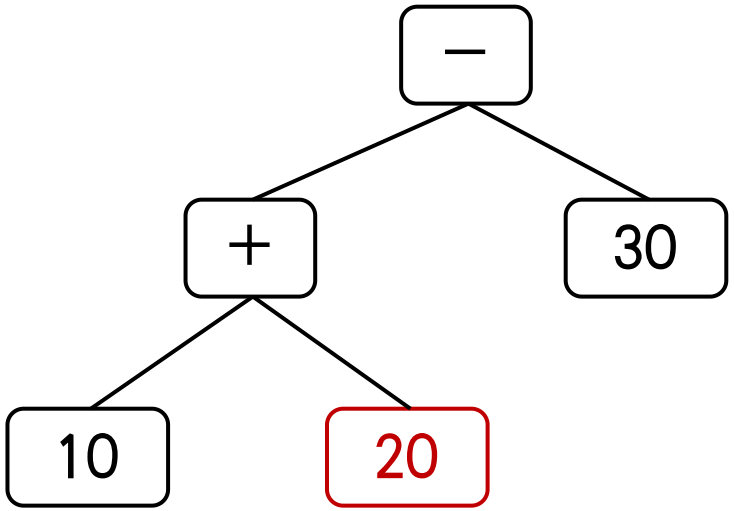
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

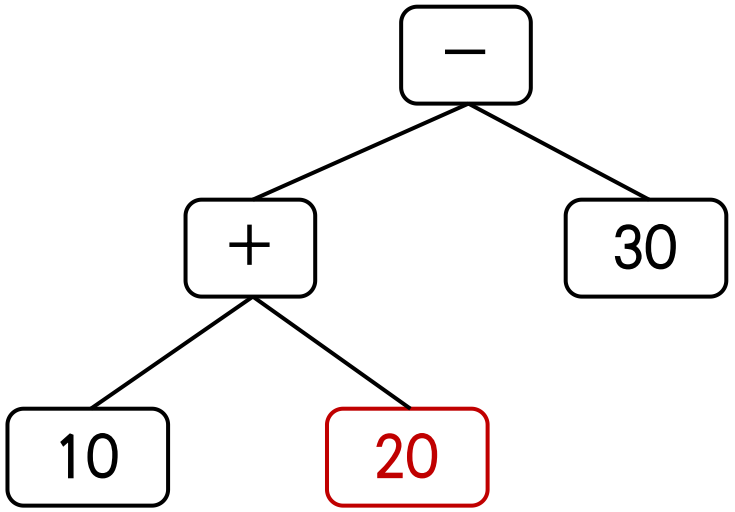
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

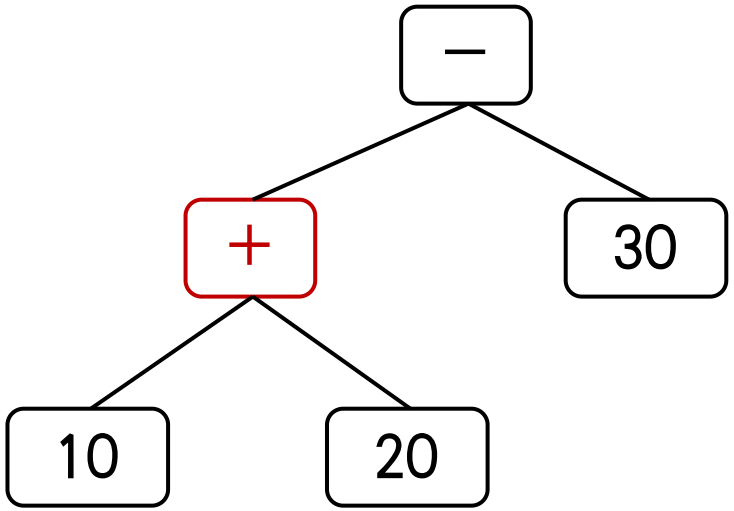
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

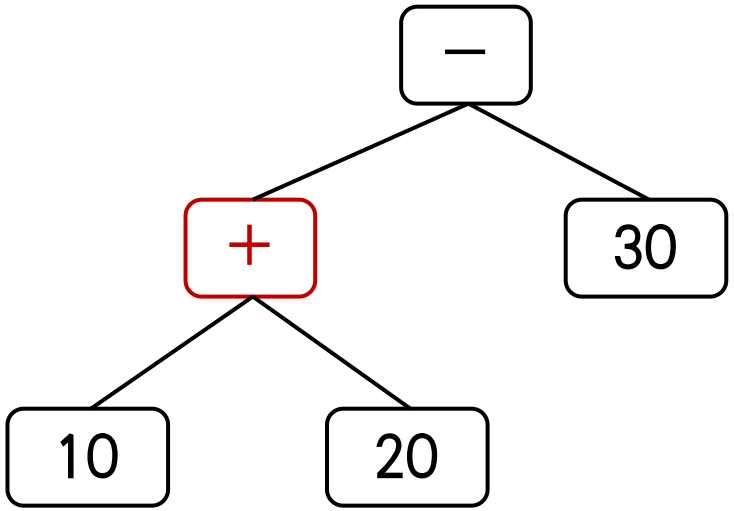
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

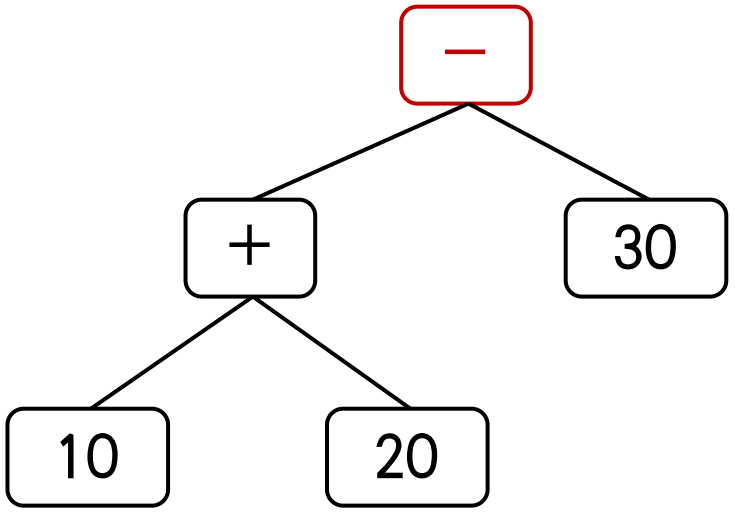
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

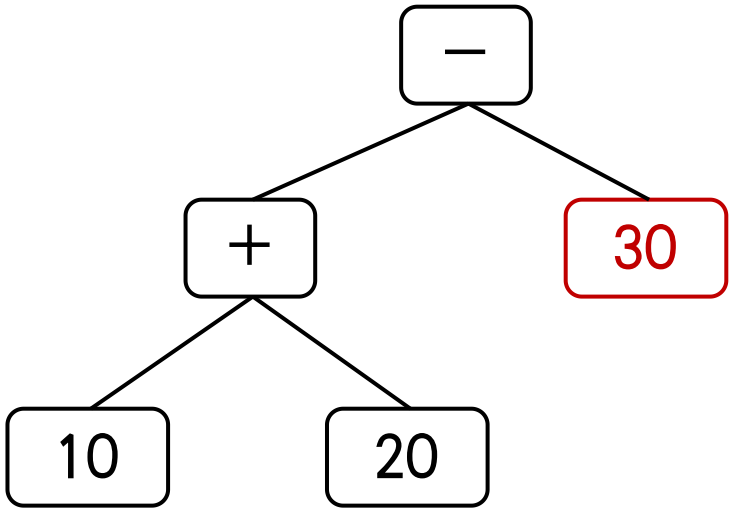
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

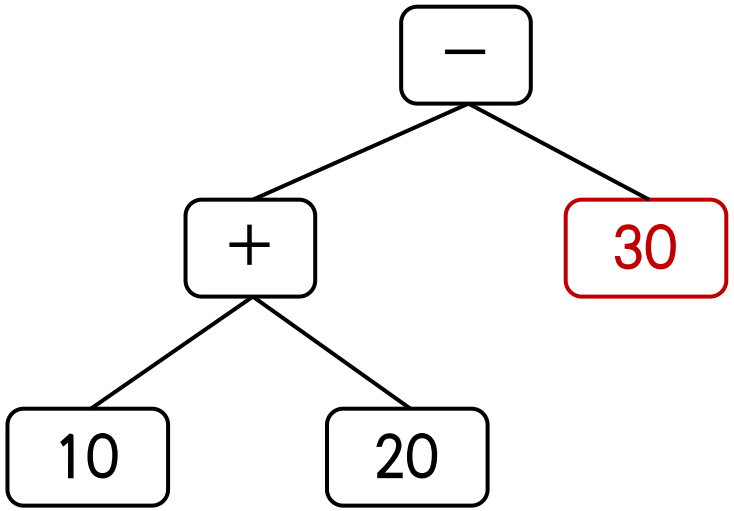
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

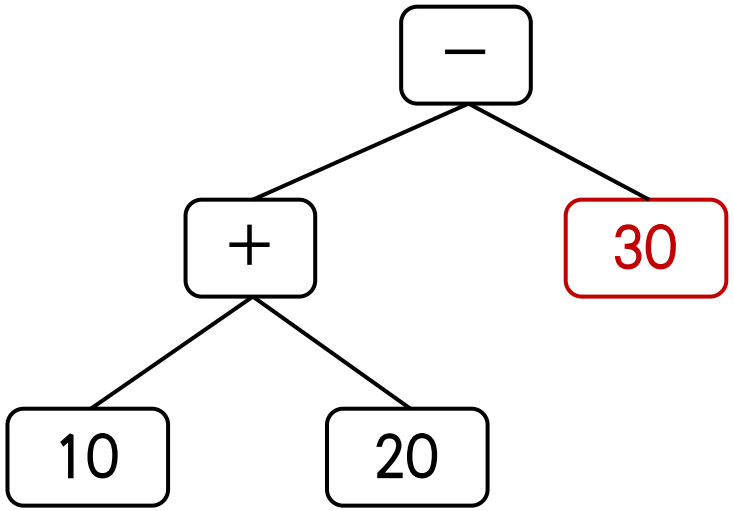
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

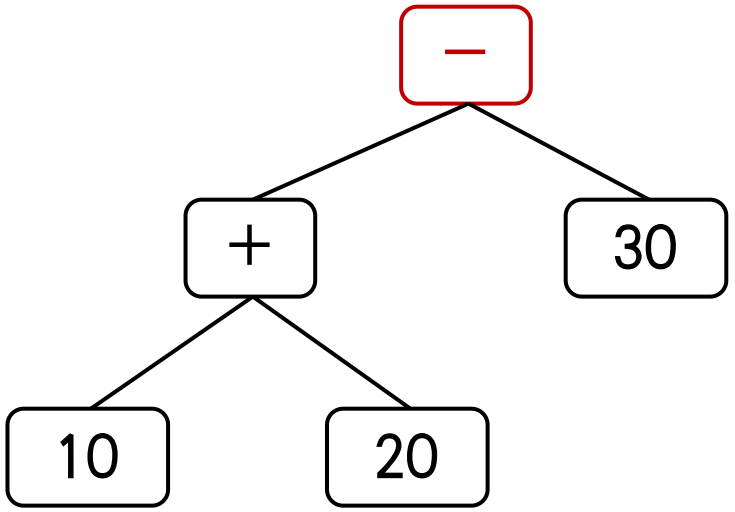
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

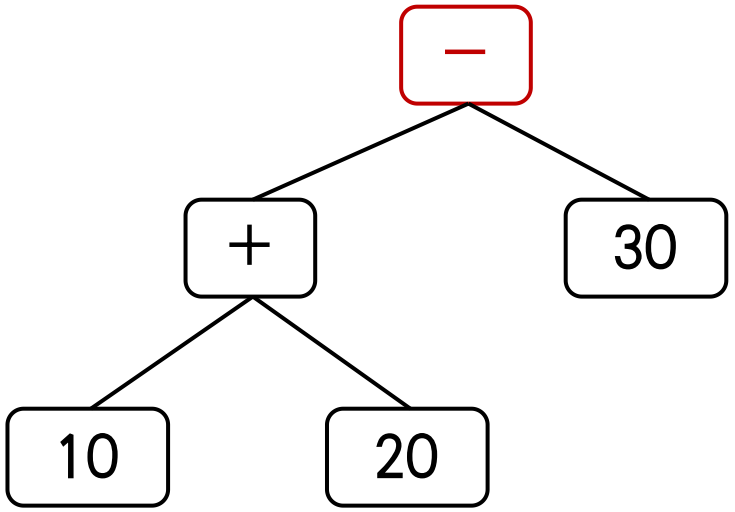
- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

访问者模式简介

- **访问者模式例子：**
  - 10+20−30




```
class Calc : public ASTVisitor {
  private:
    int visit(NumNode &) {
        int ret = NumNode.num;
        return ret;
    }
    int visit(AddNode &) {
        int op1 = AddNode.op1->accept(*this);
        int op2 = AddNode.op2->accept(*this);
        int ret = op1 + op2;
        return ret;
    }
    int visit(SubNode &) {
        int op1 = SubNode.op1->accept(*this);
        int op2 = SubNode.op2->accept(*this);
        int ret = op1 - op2;
        return ret;
    }
};
```

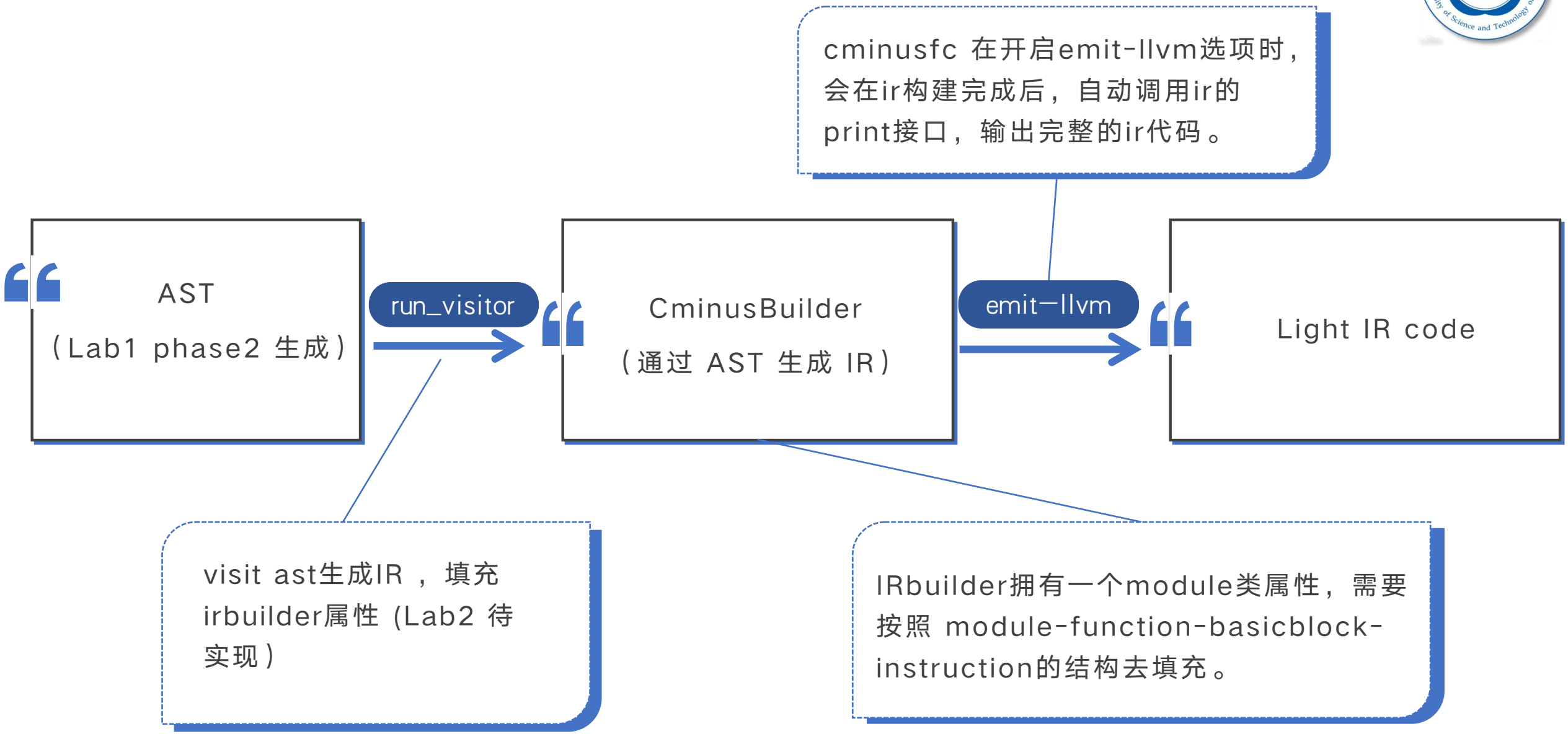
Lab 2 自动化 IR 生成

- 基于已有 AST 通过访问者模式进行 IR 自动化生成

- AST（Lab1 phase2 已实现）

- 自动化生成工具：CminusBuilder 类
  - 参考 include/cminusf/cminusf_builder.hpp
  - TODO: 需添加一些属性信息
- 访问者模式：CminusBuilder 中不同 ASTnode 的 visit 函数
  - src/cminusfc/cminusf_builder.cpp
  - TODO：visit 函数待实现

Lab 2 自动化 IR 生成



CminusfBuilder

```
class CminusfBuilder : public ASTVisitor {
    public:
    CminusfBuilder() {
        module =
std::make_unique<Module>();
        builder
=std::make_unique<IRBuilder>(nullptr,
module.get());
        private:
        virtual Value *visit(ASTProgram &)
override final;
    ...
        std::unique_ptr<IRBuilder> builder;
        Scope scope;
        std::unique_ptr<Module> module;
        struct {} context; // TODO
    };
```

CminusfBuilder类：

module：一个Module类指针，其组织结构包括其内function，function内BB，instr。也是本次是在在构建builder时需要填充的核心属性

visit函数：访问者模式下，帮助访问各类ASTnode的函数接口，这里只给出了对于ASTProgram 的visit函数

context：构建ir中的上下文信息，可以当作全局变量使用，在实现时根据需要补充。

AST->lightIR 转换示例

Cminusf源文件

```
int main(void)
{
    return 0;
}
```

emit-ast

AST实例

```
program
--fun-declaration: main
----compound-stmt
------return-stmt
--------simple-expression
----------additive-expression
------------term
--------------num (int): 0
```

emit-llvm

lightir 源文件

```
define i32 @main() {
label_entry:
  ret i32 0
}
```

一起努力
打造国产基础软硬件体系！

徐　伟

国家高性能计算中心(合肥)、信息与计算机国家级实验教学示范中心

计算机科学与技术学院

2025年04月10日