

DCS292: 编译器构造实验(Compiler Construction)

Lab0: 开启“智能编译”

Yat Compiler Construction with AI

yatcc-ai.com



张献伟、YatCC团队

中山大学 计算机学院

国家超级计算广州中心

2025.2.27

www.nscg-gz.cn

OUTLINE

目 录

中山大学计算机学院

School of Computer Science & Engineering

一、YatCC-AI介绍

二、整体实验设计

三、总结与展望

YatCC-AI? YatCC?

□ **YatCC-AI: Yat Compiler Construction with AI** [AI赋能]

□ **Yat Compilation Course: From University** [来自中大]

□ **Yet Another Tiny C Compiler: For Practical Training** [面向实践]

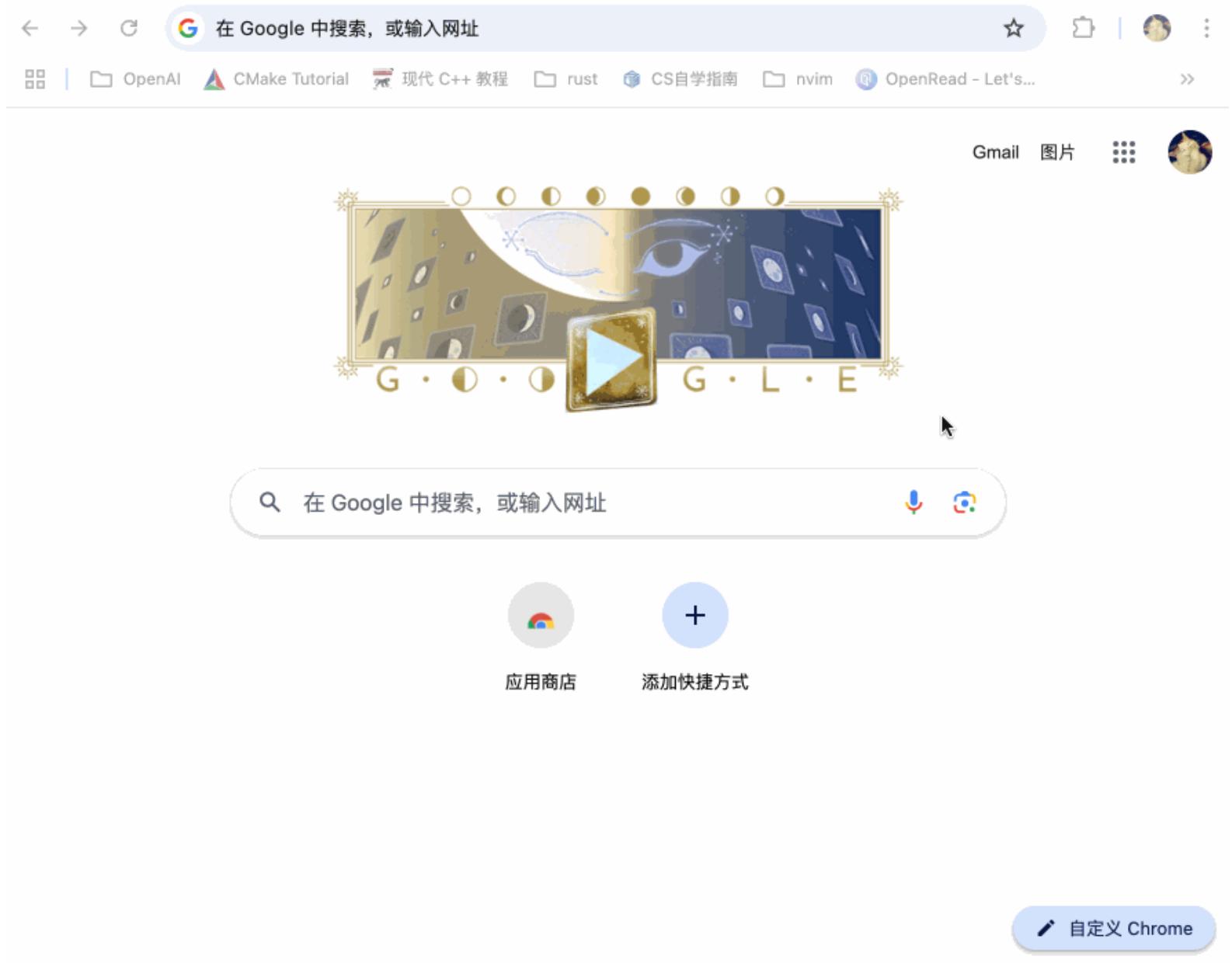
□ **Your AI Time Cool Compiler: Follow Technical Trend** [紧跟前沿]

□ **Young Architect Training, Coaching, Cultivating: Further Goal** [不止编译]

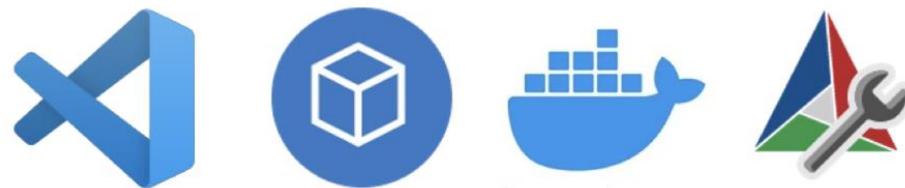
□ **Young, Active Team with Close Collaboration: Fantastic Team** [卓越团队]



演示 Demo



网页即开即用、DeepSeek全程赋能



YAT COMPILER

Your AI Time Cool Compiler

Just open browser — no setup, no hassle!

Chat, Code, Create — rock your LLM agent!

- **课程网站:** yatcc-ai.com, [yatcc.github.io](https://github.com/yatcc)
- **实验代码:** <https://github.com/arcsysu/YatCC>
- **实验文档:** <https://arcsysu.github.io/YatCC>
- **视频教程:** <https://space.bilibili.com/3546650047941291>
- **自动评测:** <https://arcsysu.terryinc.top>



Resource Hub

Course Archive

Catalog of past course information and teaching materials of 2025, 2024, 2023, 2022, 2021.

Reference Manual

Documents that cover the complete process of experimentation and offer a wealth of practical examples.

Source Code

Detailed source code framework with 300+ commits, 200+ stars.

Auto Grader

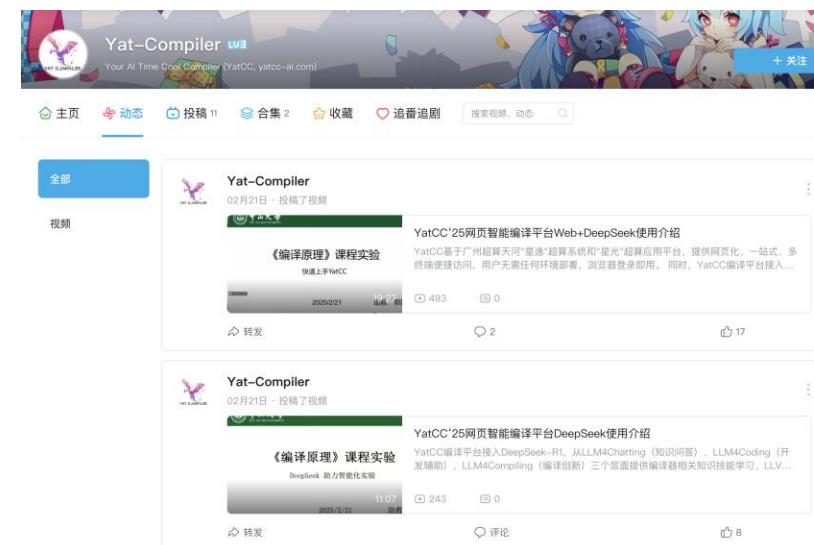
Real-time evaluation Autograder that automatically runs testing cases to provide instant feedbacks.

Video Tutorial

A pile of video clips providing step-by-step operational guidance.



Autograder



OUTLINE

目 录

中山大学计算机学院
School of Computer Science & Engineering

一、YatCC-AI介绍

二、整体实验设计

三、总结与展望

□ 编译器是持续演进的核心基础软件

- 衔接上层应用程序和底层硬件架构
- 计算机生态一直在改变，带动编译器持续发展

PyTorch



编译器



□ LLVM是学习编译与开展编译工作的理想平台

- 发展逾20年，设计良好、易于上手、社区活跃
- 工业界和学术界需求广泛，但面临严峻人才缺口

人民日报 | 有品质的新闻

打开

推动国产基础软件加快发展（
创新谈）

人民日报 海生 2024年7月1日 00:00

拥有强大的基础软件，信息产业、数字经济
的“大厦”才能建得高。反之，如果缺少
强大的基础软件，其他软件和信息服务的发
展也将受到制约



方舟编译器

多端多语言，轻量低开销



□ 既有教学与业界存在一定偏差

- 强经典编译技术阶段，弱实用编译器全局实践
- 闭环小型教学项目 vs. 工业级编译基础设施

基于LLVM的编译实践教学体系

mini-compiler



LLVM



<https://clang.llvm.org/>

https://clang.llvm.org/get_started.html

基于友好开发体验的LLVM编译实践教学

□ 打通课程教学和业界需求，促进产教融合

- 基于LLVM：平衡实验难度，理论结合实际
- 学习LLVM：了解复杂软件，衔接业界实践
- 对比LLVM：参考标准实现，向业界前沿看齐

□ 定义一套现代化的标准实验环境

- 开箱即用，**全自动**环境配置
- 提供完整的开发、测试、打包等**全流程**支持
- 降低上手难度，提高实验完成率

□ 向领域专家成长，or像领域专家一样探索

- 前沿研究或工程实践

S Y s Y 类C 编程语言



分阶段构建小型编译器

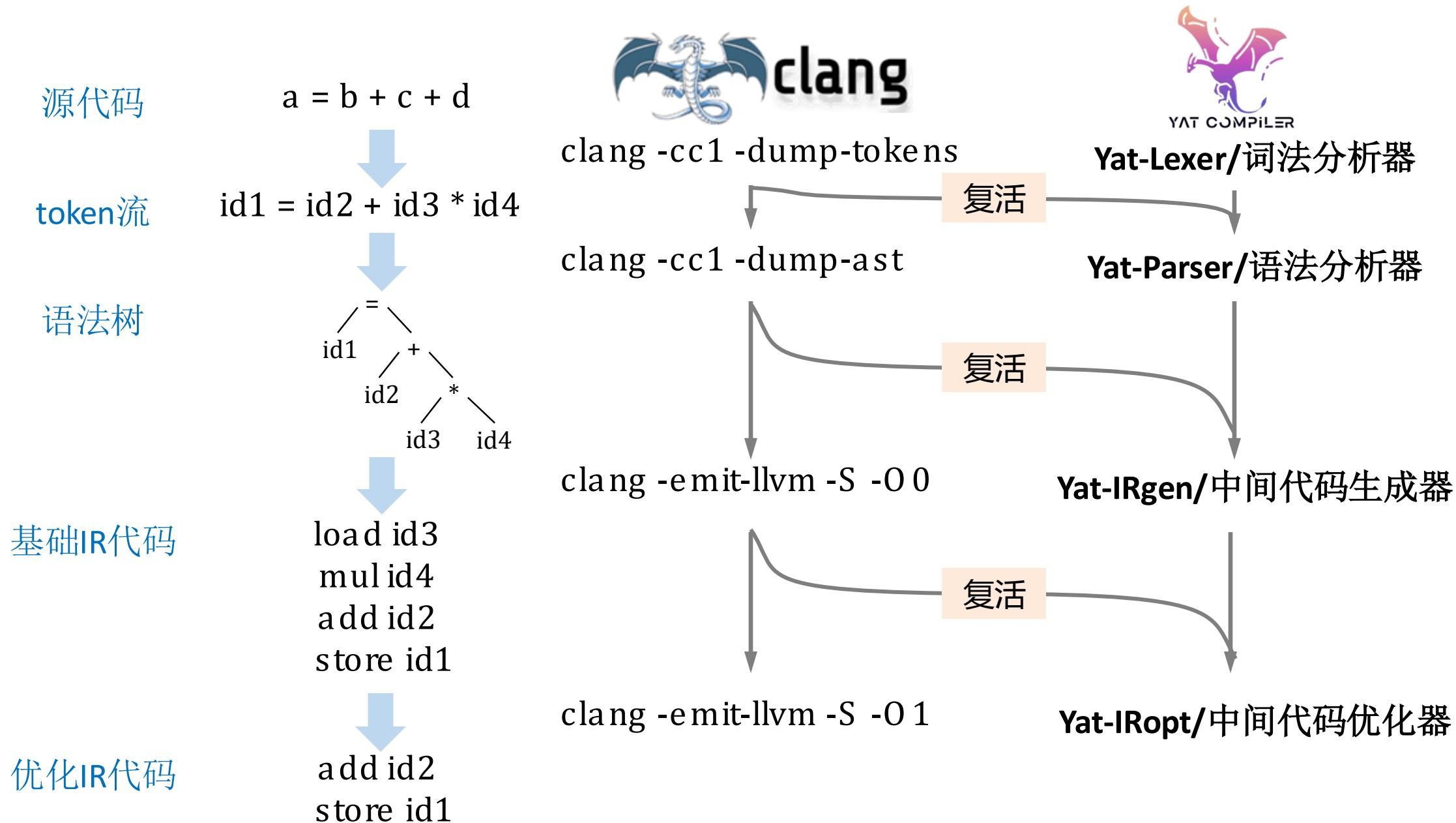


LLVM 编译基础设施



<https://compiler.educg.net/#/>

面向实践：基于LLVM开展实验



□ 一站式标准环境设置，开箱即用

- 基于开发容器提供**标准化的实验环境**
- 完整设计项目框架，**全流程自动化，零命令行操作**
- 预置**良架构的基础示例代码**，实验难度可调节



□ 分阶段推进，实验模块之间解耦

- 对应四个编译阶段，“**复活**”以隔绝前一阶段的影响



YAT COMPILER

Your AI Time Cool Compiler

□ 集成单元测试，本地+在线自动评测系统

- 以LLVM为参考答案，**自动运行测例、生成成绩单**：
本地实时反馈、在线成绩排名

Just open browser — **no setup, no hassle!**

Chat, Code, Create — **rock your LLM agent!**

LLM4Chatting

面向所有课

- 知识问答推理

LLM4Coding

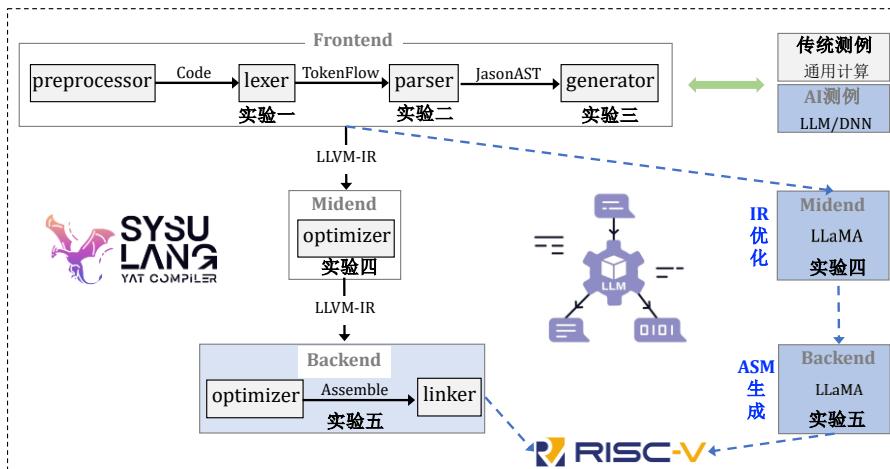
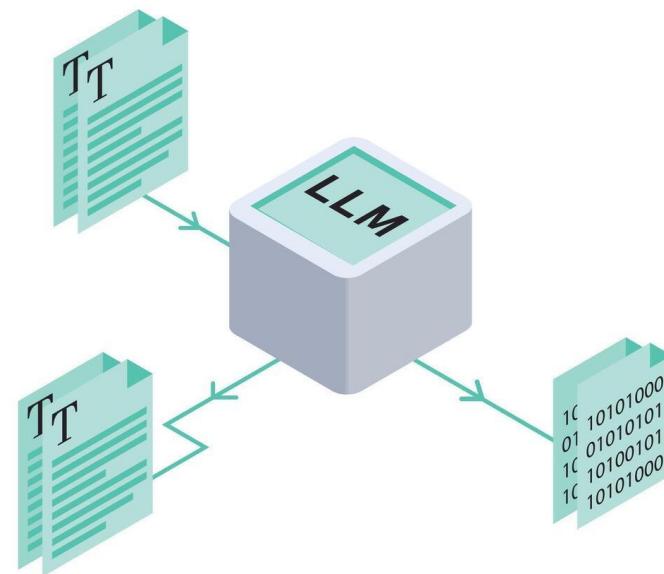
面向有编程实践的课

- 代码开发调试

LLM4Compiling

面向编译课

- 编译优化、汇编生成

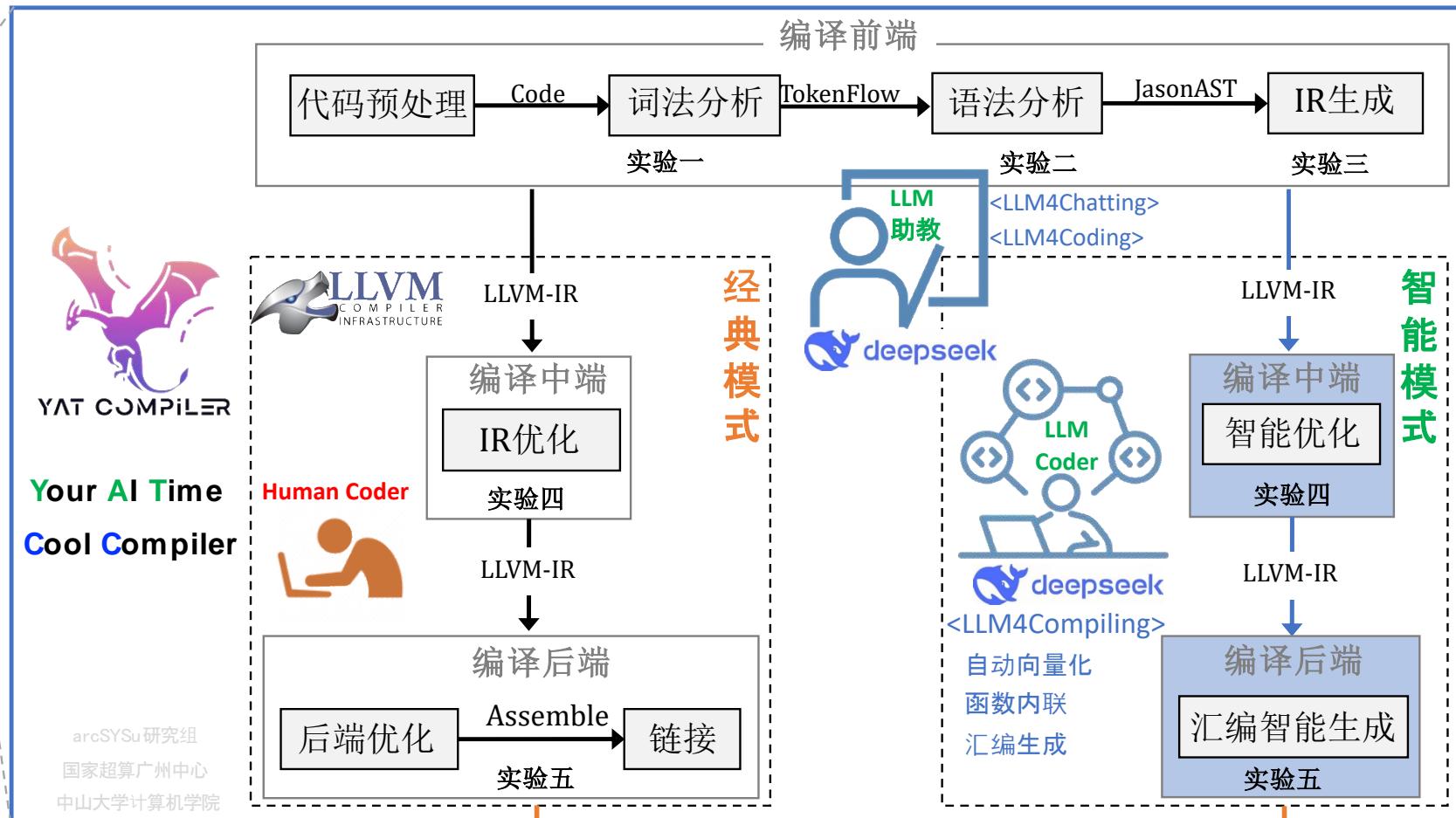


LLM Compiler

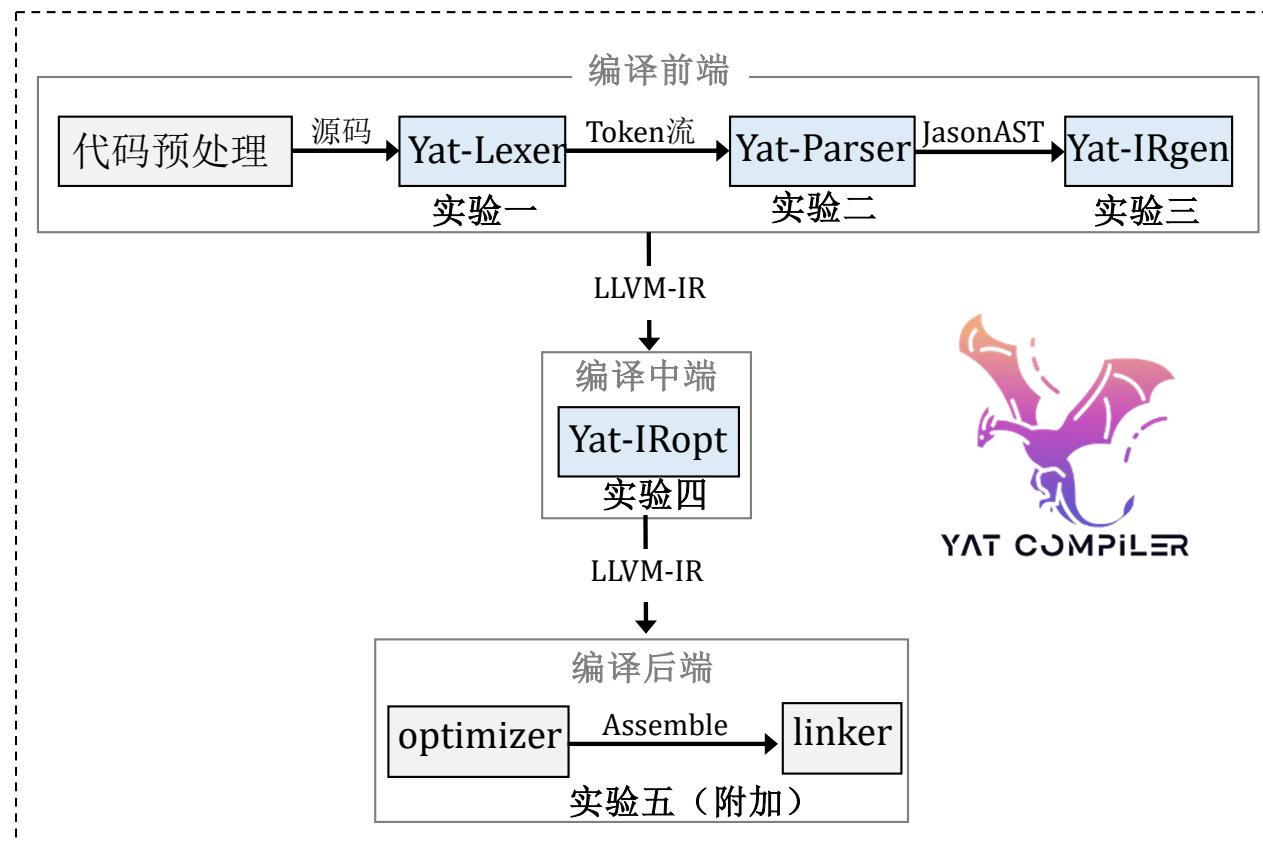
Meta LLM Compiler is a state-of-the-art LLM that builds upon Code Llama with improved performance for code optimization and compiler reasoning.

 huggingface.co

整体实验架构



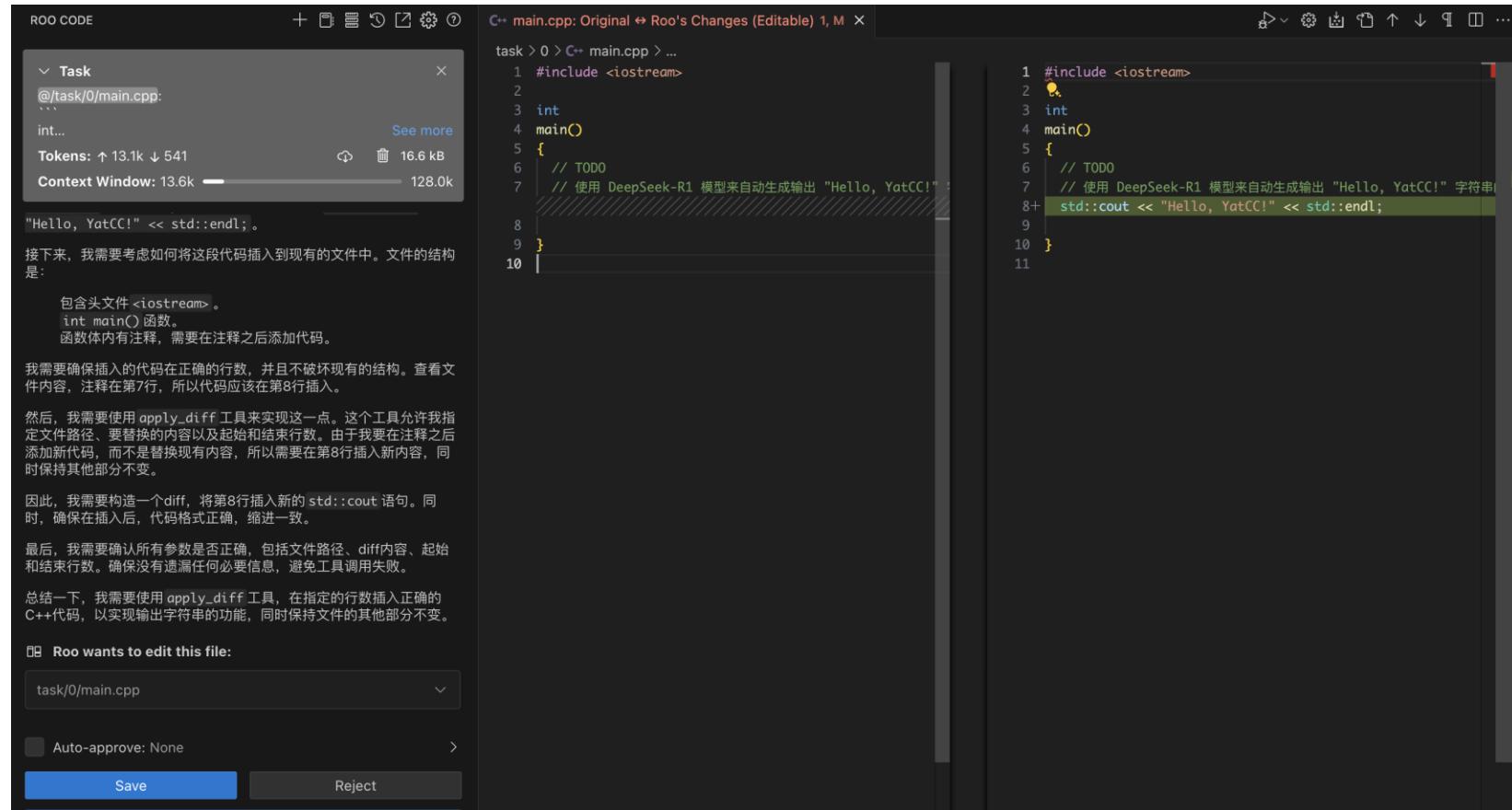
整体任务安排



实验	〇	一	二	三	四	五
内容	Env	Lexer	Parser	IRgen	IRopt	ASMgen
代码量/LOC	0	250	1000	1500	~500	<500
用时/小时	1~2	2~6	24~72	36~108	24~108	<72
预留时间/天	7	14	30	30	30	14

目标要求：熟悉环境，开启“智能编译”之旅

- 环境设置
- 熟悉开发工具
- 熟悉任务构建、评分流程
- 熟悉DeepSeek大模型工具
- . . .

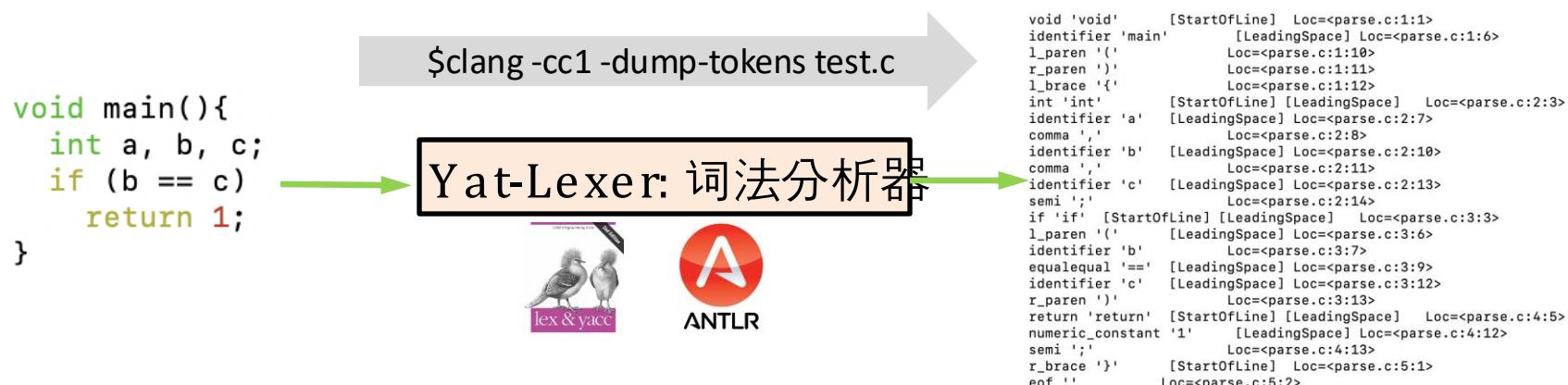


□ 目标要求：实现词法分析器，识别解析源代码token

- 采用Flex或ANTLR作为开发工具，输出结果与Clang输出匹配
- 需识别记录词号，对应文本内容、文件路径和词的行列号

□ 输入输出

- 输入：clang -E预处理过的源代码
- 输出：与Clang词法分析器一致的token流

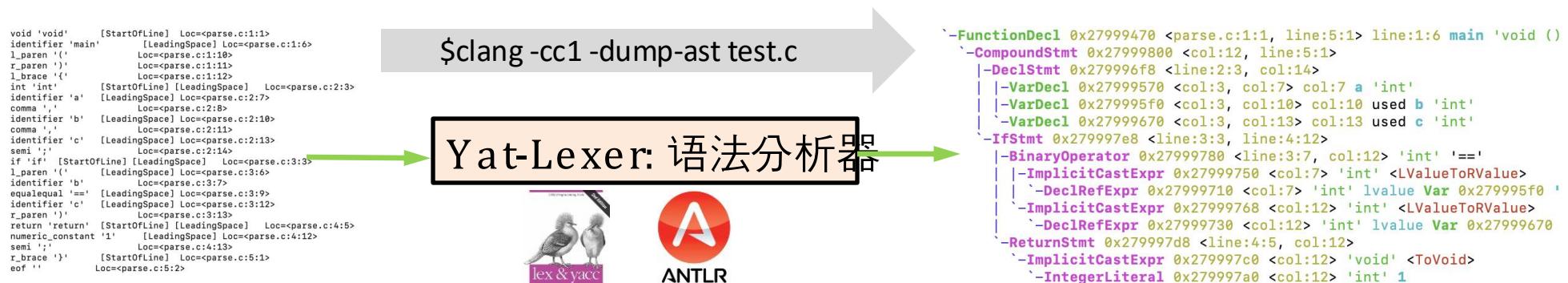


目标要求：实现语法分析器，识别源程序语法结构

- 基于ANTLR或Flex+Bison自动解析工具，写出相应语法规则
- 识别语言的所有可能语法结构，构建解析树 (parse tree)

输入输出

- 输入：task1或Clang词法分析器生成的token流
- 输出：json格式抽象语法树，与标准Clang AST一致



目标要求：实现中间代码生成器，生成源语言IR表示

- 基于LLVM设施和所提供实验框架实现，调用相关API和库
- 翻译出源程序对应的LLVM格式可读IR

输入输出

- 输入：task2或Clang语法分析器输出的json格式AST
- 输出：符合LLVM IR规范的中间表示文件



目标要求：实现中间代码优化器，对IR进行迭代转换

- 基于LLVM设施开发编写优化遍Pass，实现IR代码分析和转换
- 实现常量传播、死代码消除、循环无关变量移动等优化

输入输出

- 输入：task3或Clang -O0提供的LLVM IR
- 输出：经过优化器处理的LLVM IR

```
; ModuleID = 'in_test.c'
source_filename = "in_test.c"
target datalayout = "e-m:o-i64:64-i128:128-m32:64-8128"
target triple = "armv8-neon-vfpv4-macos11.0"

@a = global i32 0, align 4

; Function Attrs: noinline nounwind optnone ssp uwtable(sync)
define i32 @main() #0 {
entry:
  %retval = alloca i32, align 4
  store i32 0, ptr %retval, align 4
  store i32 1, ptr @a, align 4
  %0 = load i32, ptr @a, align 4
  %cmp = icmp eqe i32 %0, 0
  br i1 %cmp, label %if.then, label %if.else

if.then:                                ; preds = %entry
  store i32 1, ptr %retval, align 4
  br label %return

if.else:                                  ; preds = %entry
  store i32 0, ptr %retval, align 4
  br label %return

return:                                    ; preds = %if.else, %if.then
  %1 = load i32, ptr %retval, align 4
  ret i32 %1
}
```

\$clang -emit-llvm -S O1 test.c

Yat-IRgen: IR 优化器



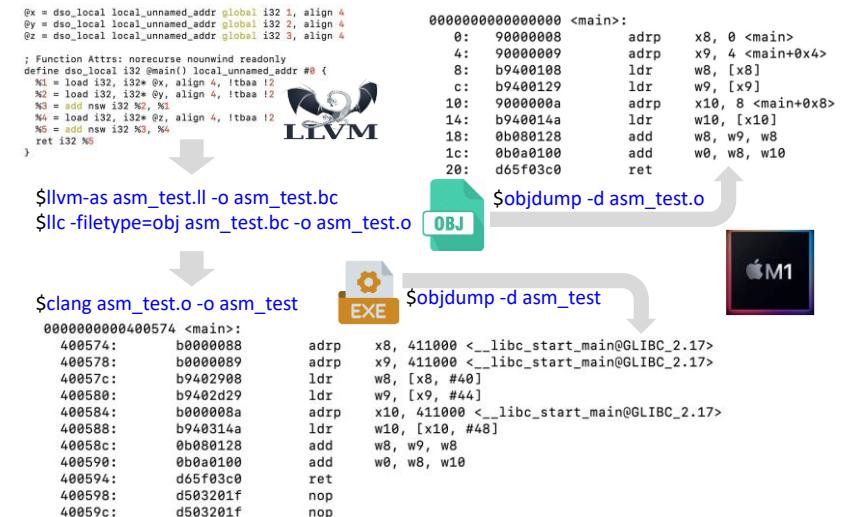
```
define dso_local i32 @main() local_unnamed_addr #0 {
  ret i32 1
}
```

目标要求：将IR转换为目标机器汇编指令

- 使用LLVM库，将LLVM IR转换为汇编指令
- 生成的汇编指令能正确编译运行

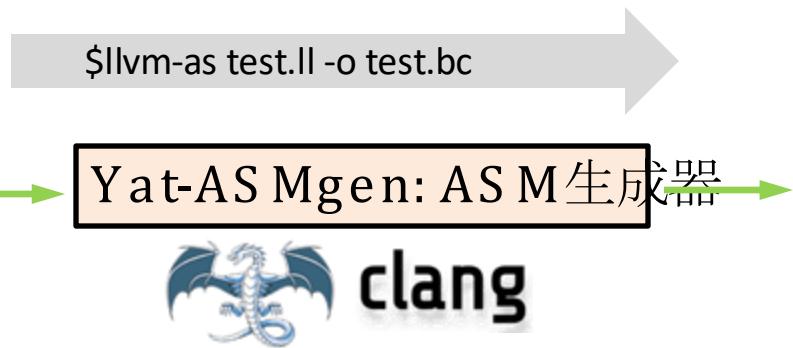
输入输出

- 输入：task4或Clang提供的LLVM IR
- 输出：汇编指令



```

16 ; Function Attrs: nounwind
17 @_Znwm@plt
18 entry:
19   %call = tail call @__sys_getint() #3
20   store i32 %call, ptr @loopCount, align 4, !tbaa !2
21   tail call void @__sys_starttime(i32 @noundef 37) #3
22   %3 = load i32, ptr @loopCount, align 4, !tbaa !2
23   %tmp17 = icmp slt i32 %3, 0
24   br i1 %tmp17, label @while.cond1.preheader, label @while.end7
25
26 while.cond1.preheader:
27   %kindvars.iv = phi i32 [ %kindvars.iv.next, @while.cond1.preheader ], [ 0, @entry ]
28   %xi.010 = phi i32 [ %add6, @while.cond1.preheader ], [ 0, @entry ]
29   %sum.018 = phi i32 [ %rem, @while.cond1.preheader ], [ 0, @entry ]
30   %add5 = add nsw i32 %kindvars.iv, 0
31   %add6 = add nsw i32 %xi.010, %sum.018
32   %rem = srem i32 %add6, 134209537
33   %add8 = add nsw i32 %xi.010, 1
34   %kindvars.iv.next = add i32 %kindvars.iv, 6
35   %nextcond.not = icmp eq i32 %add6, 0
36   br i1 %nextcond.not, label @while.cond.while.end7.crit_edge, label @while.cond1.preheader, !llvm.loop !6
37
38 while.cond.while.end7.crit_edge:
39   %xi = add nsw i32 %0, -1
40   store i32 %xi, ptr @global, align 4, !tbaa !2
41   br label @while.end7
42
43 while.end7:
44   %sum.018.issac = phi i32 [ %rem, @while.cond.while.end7.crit_edge ], [ 0, @entry ]
45   tail call void @__sys_stoptime(i32 @noundef 37) #3
46   tail call void @__sys_putint(i32 @noundef %sum.018.issac) #3
47   tail call void @__sys_putc(i32 @noundef 10) #3
48   ret i32 0
49
    
```



```

31 main:
32   pushq   %rbp
33   .seh_pushreg   %rbp
34   movq   %rsp, %rbp
35   .seh_setframe %rbp, 0
36   subq   $48, %rsp
37   .seh_stackalloc 48
38   .seh_endprologue
39   call   __main
40   movl   $0, -4(%rbp)
41   movl   $0, -8(%rbp)
42   call   __sys_getint
43   movl   %eax, loopCount(%rip)
44   movl   $15, %ecx
45   call   __sys_starttime
46   jmp   .L4
47 .L4:
48   movl   $0, -12(%rbp)
49   movl   $0, -16(%rbp)
50   jmp   .L5
51 .L5:
52   movl   -8(%rbp), %eax
53   movl   %eax, %ecx
54   call   func
55   movl   global(%rip), %eax
    
```

OUTLINE

目 录

中山大学计算机学院

School of Computer Science & Engineering

一、YatCC-AI介绍

二、整体实验设计

三、总结与展望

□ 教学奖项：2024年中国计算机教育大会教学案例**全国一等奖**

- “基于友好开发体验的LLVM编译实践教学”

□ 系统竞赛：2023年华为毕昇杯编译系统设计赛**全国一等奖**

- 全国大学生计算机系统能力大赛编译系统设计赛

□ 教学论文：2022中国计算机教育大会优秀论文**一等奖**

- “基于Clang/LLVM构建编译实践全局观”，《计算机教育》



□ 定位**核心基础**专必课程本质，落实**产教融合**目标

- 动手构建编译器：基于理论教学，互补理论教学

□ 面向**实践**，强化**过程**，突出**特色**

- 以**LLVM**为蓝本，现代一站式开发体验，**分阶段解耦**推进
- 平衡**基础性**、**实践性**和**综合性**，构建编译**系统全局观**
- 提供代码框架、文档、视频教程等多样资源，**全程引导**

□ **超算**+ DeepSeek

- 浏览器即开即用、大模型全程赋能



Join US



TOGETHER WE

GO FURTHER

COME AND
JOIN US!

谢谢!

