第1讲
编译原理和技术导论

李 诚

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

2023年09月04日

汇报提纲

1. 编译器是什么?
2. 为什么要学习编译课程?
3. 编译教学的困境与科大方案
4. 本学期课程设置的变化
5. 选课与退课的恩恩怨怨

什么是/为什么需要编译器?

- **高级语言**
  - 直接面向开发者
  - 与数学公式类似
  - 编程效率高
- **机器语言**
  - 驱动硬件完成具体任务
  - 编程效率低
- **编译器提供程序开发的便捷性**
  - 实现人机交流，将人类易懂的高级语言翻译成硬件可执行的目标机器语言

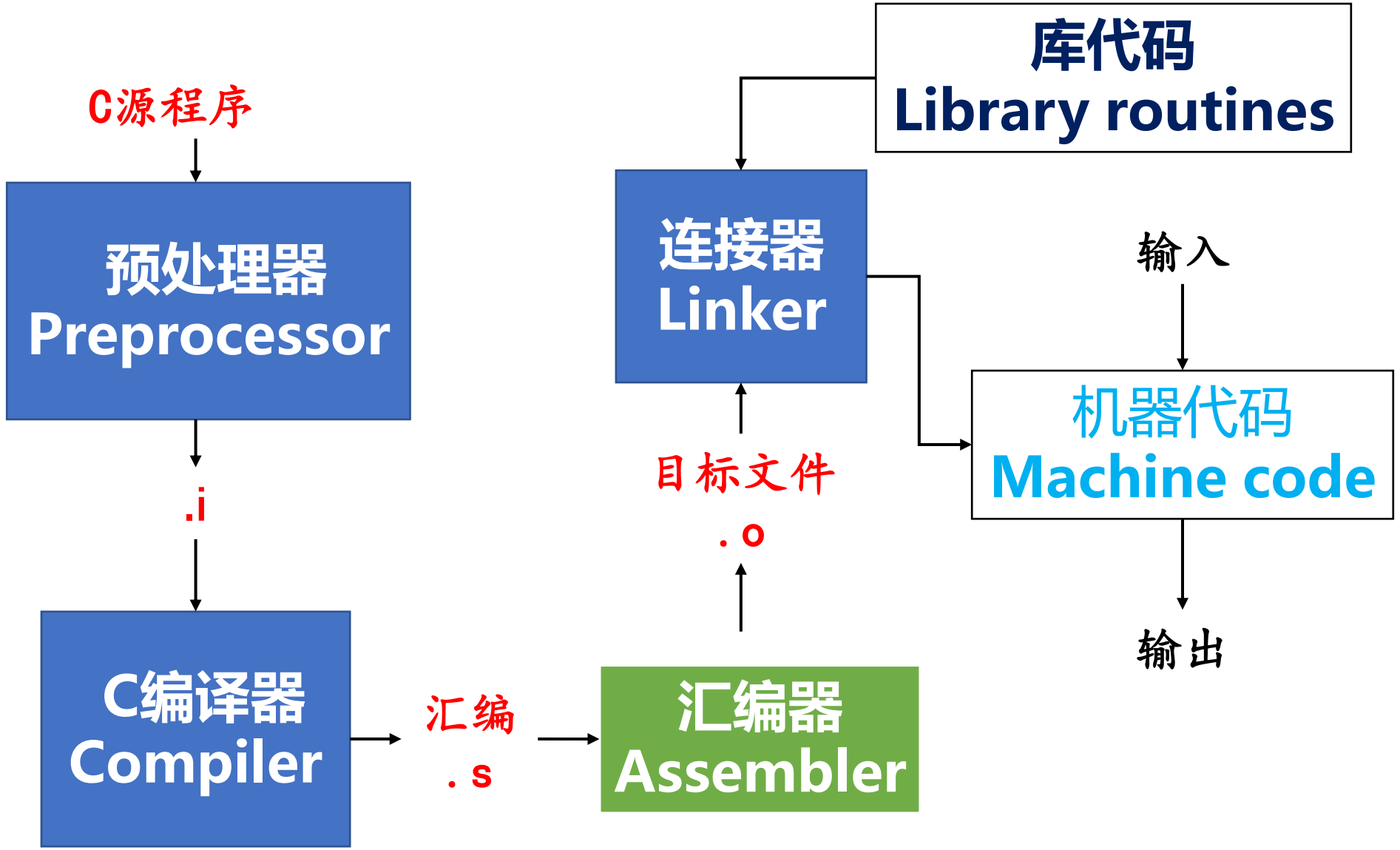
从C程序到可执行文件

```c
#include <stdio.h>
int main()
{
  printf("hello, world!\n");
}

/* helloworld.c */
```

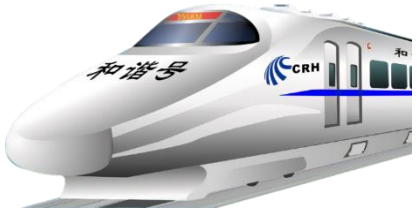
```
[root@host ~]# clang helloworld.c -o helloworld
[root@host ~]# ./helloworld
hello, world!
```

C程序的编译运行过程分解



参考链接：http://www.cs-fundamentals.com/c-programming/how-to-compile-c-program-using-gcc.php

编译器自动优化能力



优化等级	简要说明
-Ofast	在-O3级别的基础上，开启更多**激进优化项**，该优化等级不会严格遵循语言标准
-O3	在-O2级别的基础上，开启了更多的**高级优化项**，以编译时间、代码大小、内存为代价获取更高的性能。
-Os	在-O2级别的基础上，开启**降低生成代码体量**的优化
-O2	开启了大多数**中级优化**，会改善编译时间开销和最终生成代码性能
-O/-O1	优化效果介于-O0和-O2之间
-O0	默认优化等级，即**不开启编译优化**，只尝试减少编译时间

延伸阅读：https://clang.llvm.org/docs/CommandGuide/clang.html#code-generation-options

举例——优化对代码性能的影响

❑1000000000次循环迭代累加

```
#include <stdio.h>
#include <time.h>
int main() {
    int loop = 1000000000;
    long sum = 0;
    int start_time = clock();
    int index = 0;
    for (index = 0; index < loop; index ++)
    {
        sum += index;
    }
    int end_time = clock();
    printf("Sum : %ld, Time Cost : %lf \n", sum, (end_time - start_time) * 1.0 / CLOCKS_PER_SEC);
    return 0;
}
```

循环次数定义

开始计时

循环体

结束计时

代码运行时间输出

举例——优化对代码性能的影响

- gcc –O0 无优化执行

```
gloit@gloit-x1c ~/2022_compiler_demo master gcc -O0 add.c
gloit@gloit-x1c ~/2022_compiler_demo master ./a.out
Sum: 49999999500000000, Time Cost: 3.415244
```

- gcc –O1 中级优化执行

性能提升5倍

```
gloit@gloit-x1c ~/2022_compiler_demo master gcc -O1 add.c
gloit@gloit-x1c ~/2022_compiler_demo master ./a.out
Sum: 49999999500000000, Time Cost: 0.554717
```

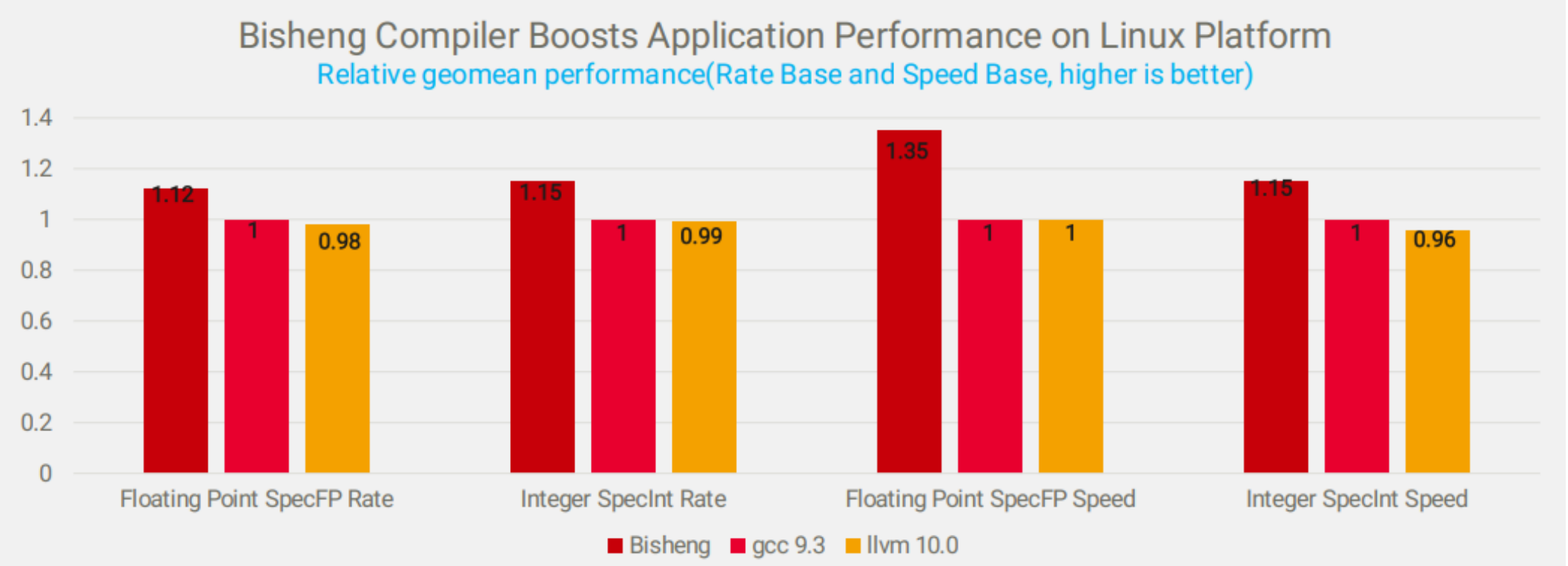
- g

性能提升数十万倍

```
gloit@gloit-x1c ~/2022_compiler_demo master gcc -O2 add.c
gloit@gloit-x1c ~/2022_compiler_demo master ./a.out
Sum: 49999999500000000, Time Cost: 0.000002
```

国产开源编译器——毕昇编译器

- 毕昇编译器通过**编译优化**提升鲲鹏硬件平台上业务的性能体验，SPEC2017性能较业界编译器平均高15%以上。



SPEC作为业界芯片性能评分标准，SPEC的分数可以直观体现出硬件的性能，越高越好

编译器的输入

- 标准的指令式语言(Java, C, C++)
  - 状态
    - 变量
    - 结构
    - 数组
  - 计算
    - 表达式 (arithmetic, logical, etc.)
    - 赋值语句
    - 条件语句 (conditionals, loops)
    - 函数

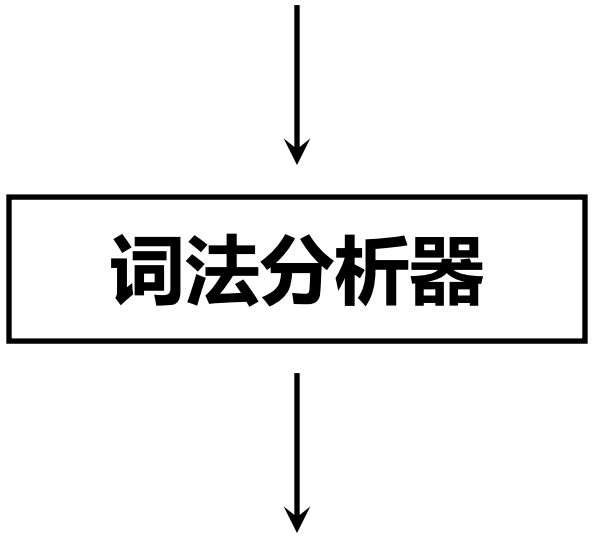
编译器的输出

- 状态
  - 寄存器
  - 内存单元
- 机器码 – load/store architecture
  - Load, store instructions
  - 寄存器操作– Arithmetic, logical operations
  - 分支指令– Branch instructions

词法分析

- 将程序字符流分解为记号（Token）序列
  - 形式：<token_name, attribute_value>

position = initial + rate * 60 ← 字符流



符 号 表

1	position	· · ·
2	initial	· · ·
3	rate	· · ·

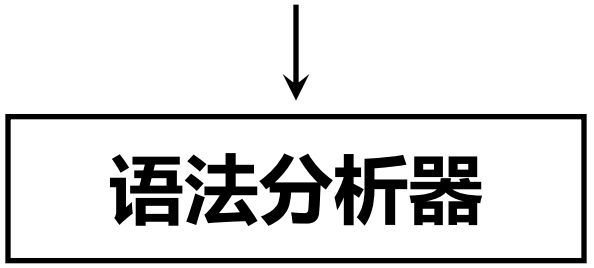
⟨id, 1⟩ ⟨=⟩ ⟨id, 2⟩ ⟨+⟩ ⟨id, 3⟩ ⟨*⟩ ⟨60⟩ ← 记号流

命令行输入：clang -cc1 -dump-tokens xx.c

语法分析

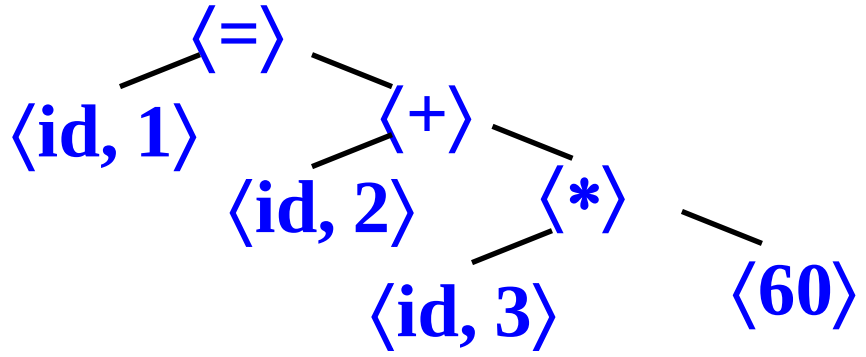
- 也称为解析（Parsing），在词法记号的基础上，创建语法结构

⟨id, 1⟩ ⟨=⟩ ⟨id, 2⟩ ⟨+⟩ ⟨id, 3⟩ ⟨*⟩ ⟨60⟩ ← 记号流



符 号 表

1	position	· · ·
2	initial	· · ·
3	rate	· · ·



← 语法树

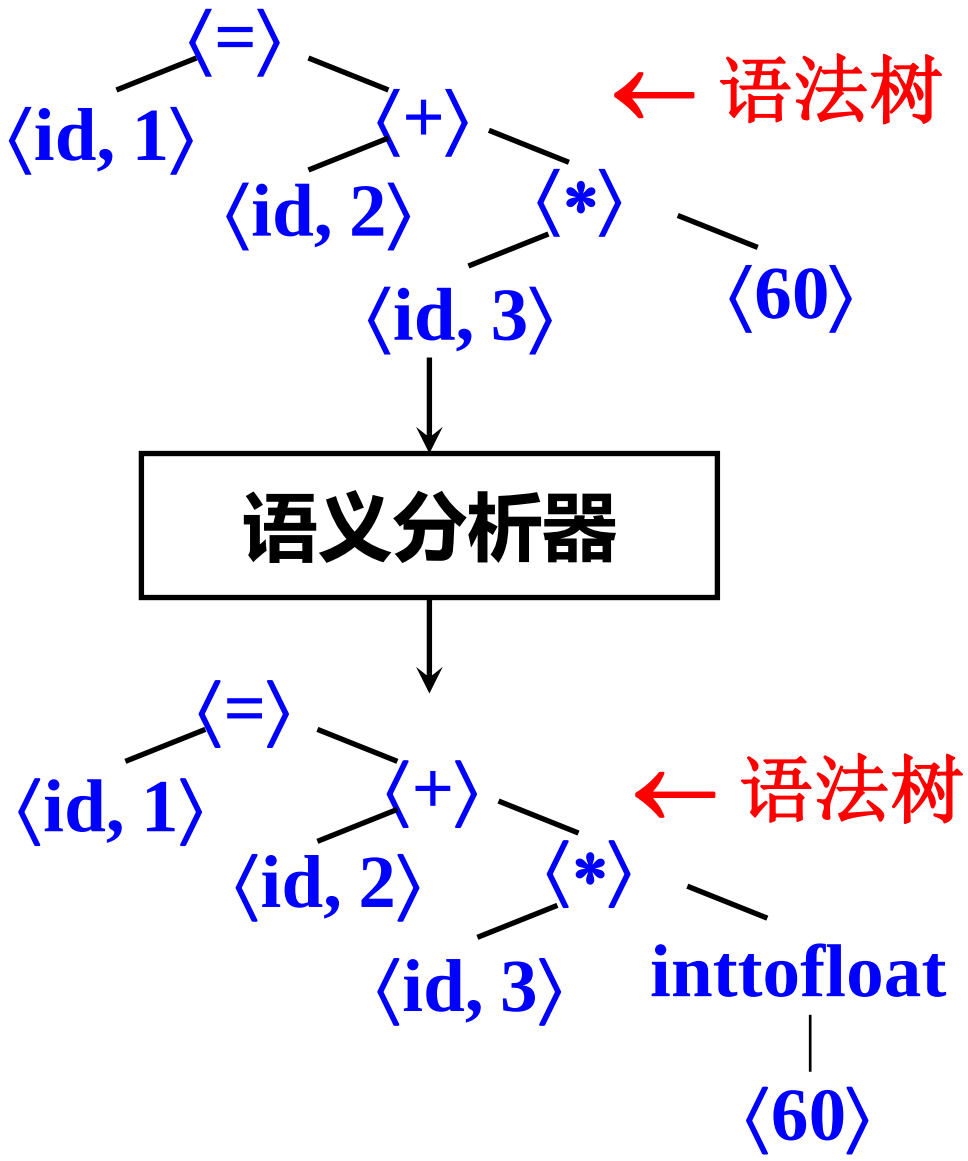
命令行输入：clang -fsyntax-only -Xclang -ast-dump xx.c

语义分析

- 编译器会检查程序中的不一致
  - 如：类型检查（type checking）

符 号 表

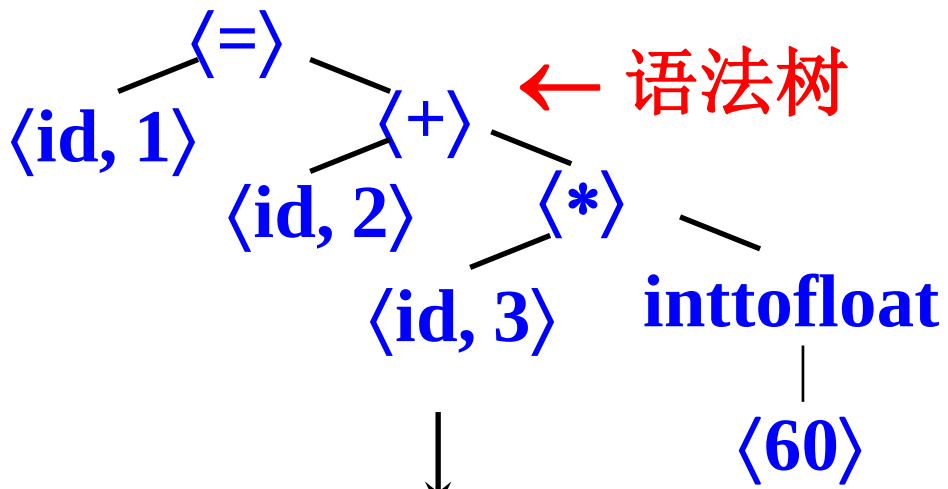
1	position	· · ·
2	initial	· · ·
3	rate	· · ·



注：类型转换在astdump时已经完成

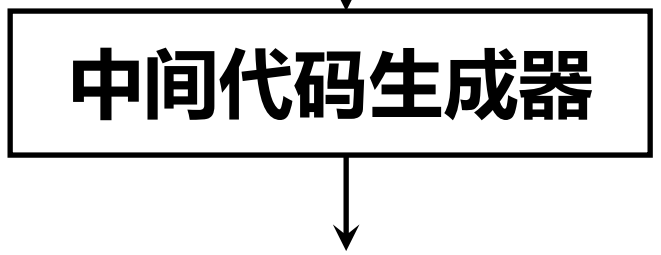
中间代码生成

- 是源语言与目标语言之间的桥梁



符 号 表

1	position	· · ·
2	initial	· · ·
3	rate	· · ·




```
t1 = inttofloat(60)
t2 = id3 * t1
t3 = id2 + t2
id1 = t3
```

← 中间代码

命令行输入：clang -cc1 xx.c -emit-llvm -o xx.ll

机器无关代码优化

- 机器无关的代码优化便于生成执行时间更快、更短或能耗更低的目标代码

```
t1 = inttofloat(60)
t2 = id3 * t1
t3 = id2 + t2
id1 = t3
```

← 中间代码

符 号 表

1	position	· · ·
2	initial	· · ·
3	rate	· · ·

↓

代码优化器

↓

```
t1 = id3 * 60.0
id1 = id2 + t1
```

← 中间代码

命令行输入：clang -S -emit-llvm -O3 test.c -o test-new.ll

目标代码生成

- 如果目标语言是机器代码，必须为变量选择寄存器或内存位置

```
t1 = id3 * 60.0
id1 = id2 + t1
```
← 中间代码

↓

代码生成器

↓

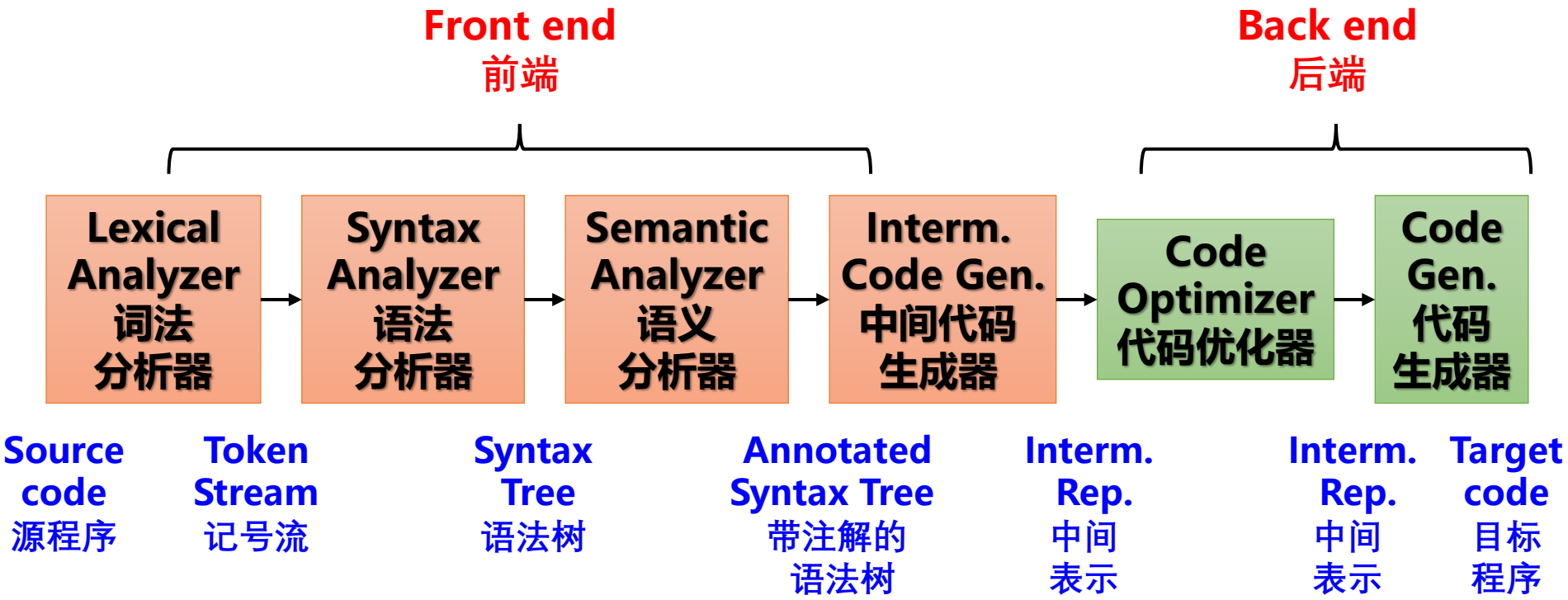
符 号 表

1	position	. . .
2	initial	. . .
3	rate	. . .

```
LDF R2, id3
MULF R2, R2, #60.0
LDF R1, id2
ADDF R1, R1, R2
STF id1, R1
```
← 汇编代码

命令行输入：llc-14 xxx.ll -o xxx.s

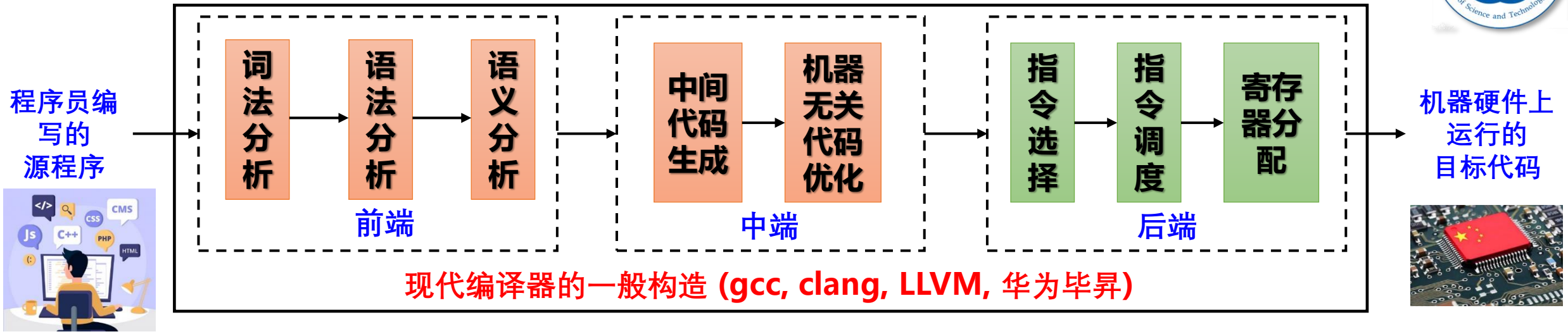
现代编译器的构造/阶段


Front end
前端
Back end
后端
Lexical Analyzer 词法 分析器
Syntax Analyzer 语法 分析器
Semantic Analyzer 语义 分析器
Interm. Code Gen. 中间代码 生成器
Code Optimizer 代码优化器
Code Gen. 代码 生成器
Source code 源程序
Token Stream 记号流
Syntax Tree 语法树
Annotated Syntax Tree 带注解的 语法树
Interm. Rep. 中间 表示
Interm. Rep. 中间 表示
Target code 目标 程序
Symbol Table 符号表
Error Handler 错误处理

汇报提纲

1. 编译器是什么?
2. 为什么要学习编译课程?
3. 编译教学的困境与科大方案
4. 本学期课程设置的变化
5. 选课与退课的恩恩怨怨

编译课程简介



“编写编译器的原理和技术具有普遍的意义，以至于在每个计算机科学家的研究生涯中，该书中的原理和技术都会反复用到。”

——著名计算机专家 *Alfred V. Aho*

“在供应链可控性上，仍存在编译工具依赖国外的情况。”

“我们应重点突破操作系统内核、编译器等关键技术。”

——《中国信息技术产品安全可控年度发展报告》

编译技术历史悠久影响深远

- 图灵奖自1966年颁发以来，共有75名获奖者，其中编译相关的科研人员有21位，占比28%，Alan J. Perlis因编译技术贡献成为第一位获得图灵奖的科学家。

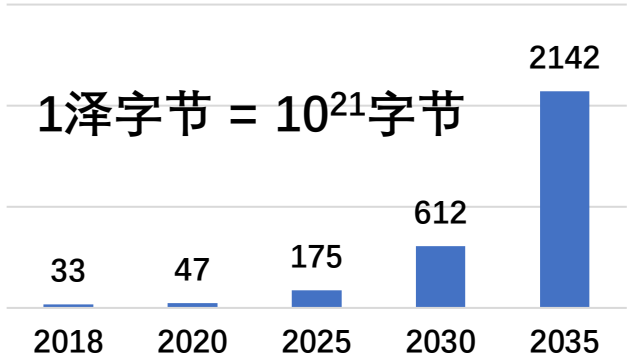
年份	科学家	贡献	年份	科学家	贡献
1966	Alan J. Perlis	高级程序设计技巧，编译器构造	1972	Edsger Dijkstra	程序设计语言的科学与艺术
1974	**Donald E. Knuth**	算法分析、程序设计语言的设计、程序设计	1976	Michael O. Rabin Dana S. Scott	非确定性自动机
1977	**John Backus**	高级编程系统，程序设计语言规范的形式化定义	1979	Kenneth E. Iverson	程设语言和数学符号，互动系统的设计，程设语言的理论与实践
1987	John Cocke	编译理论，大型系统的体系结构，及RISC计算机的开发	2005	Peter Naur	Algol 60语言
2006	**Frances E. Allen**	优化编译器	2020	**Jeffrey David Ullman** **Alfred Vaino Aho**	推进编程语言实现的基础算法和理论、教材撰写

编译技术历史悠久影响深远

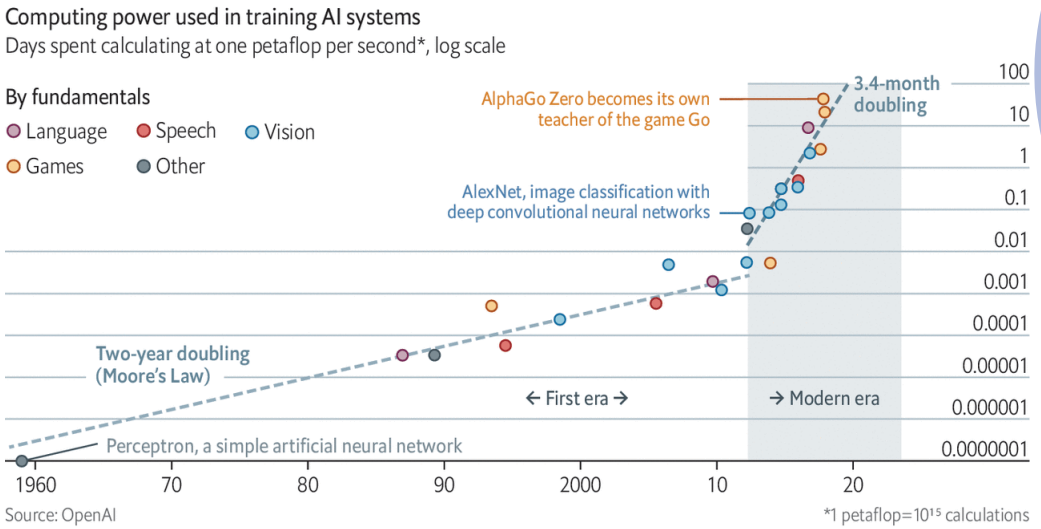
- 图灵奖自1966年颁发以来，共有75名获奖者，其中编译相关的科研人员有21位，占比28%，Alan J. Perlis因编译技术贡献成为第一位获得图灵奖的科学家。

年份	科学家	贡献	年份	科学家	贡献
1980	C. Antony R. Hoare	程序设计语言的定义与设计	1983	Ken Thompson Dennis M. Ritchie	UNIX操作系统和C语言
1984	Niklaus Wirth	程序设计语言设计、程序设计	2001	Ole-Johan Dahl Kristen Nygaard	面向对象编程
2003	Alan Kay	面向对象编程	2008	Barbara Liskov	编程语言和系统设计的实践与理论
2021	Jack J. Dongarra	通过对线性代数运算的高效数值算法、并行计算编程机制和性能评估工具的贡献，引领了高性能计算的世界。			

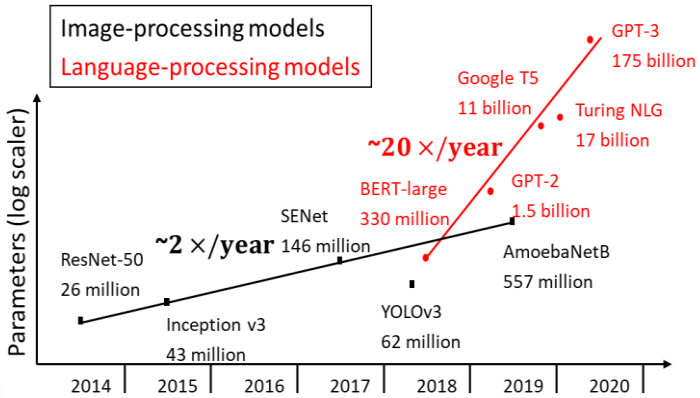
智能时代的开启



数字经济、数字孪生助推
全社会数据的爆炸式增长

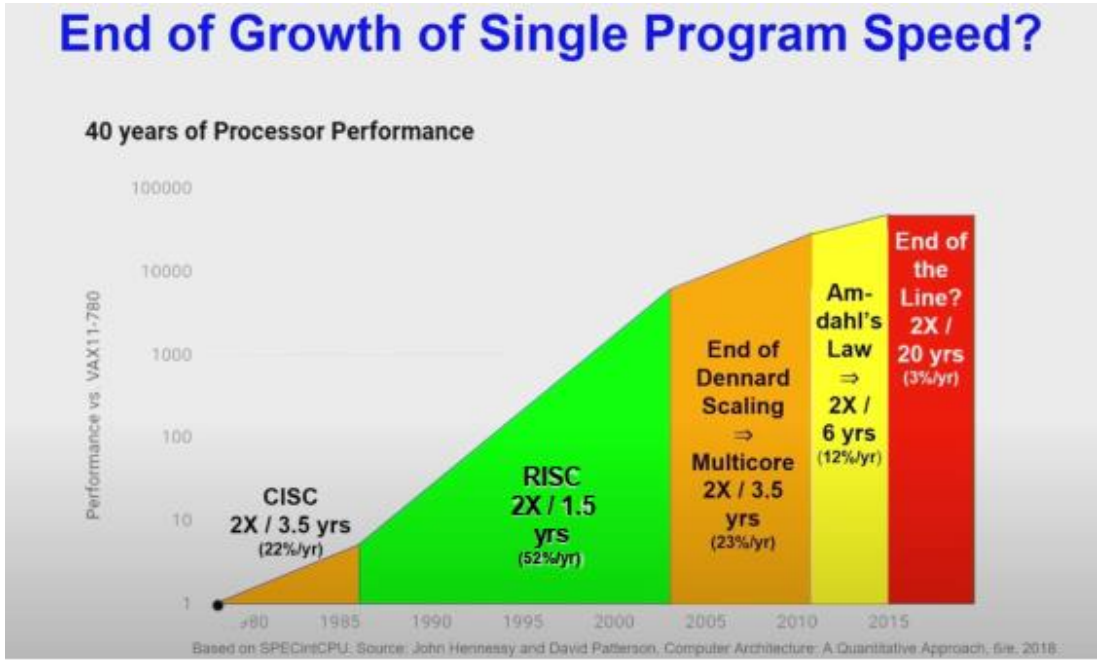


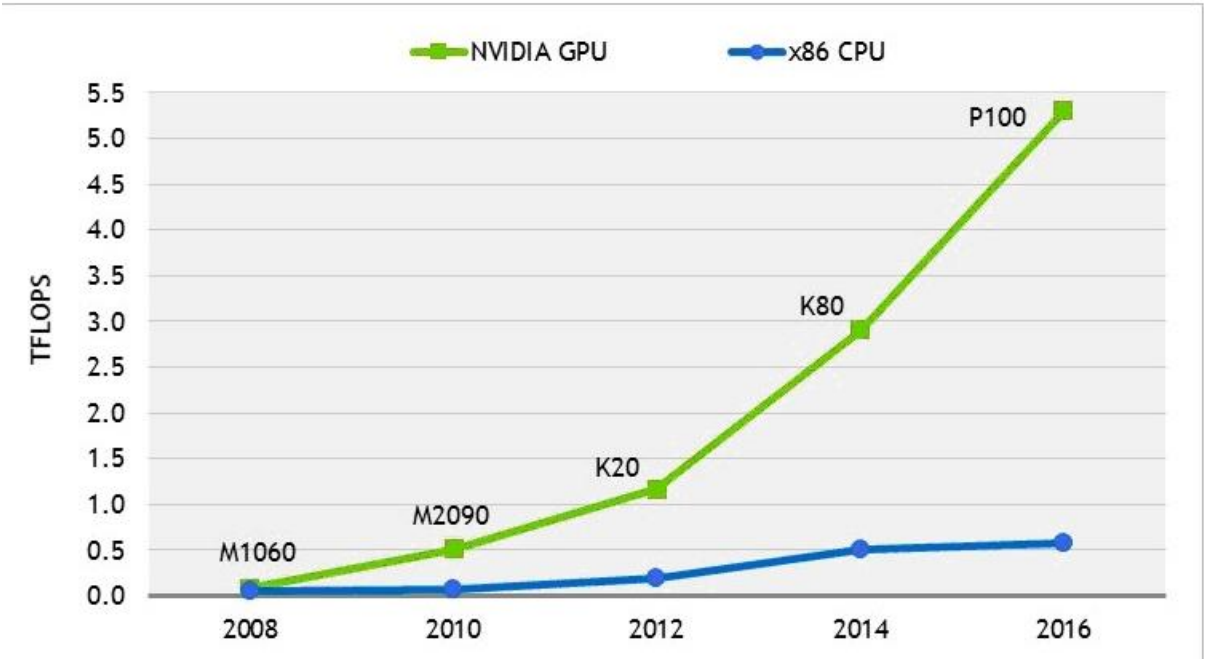
过去十年，AI训练算力需求增
长了30万倍，3.4个月翻一倍



新版摩尔定律：
宇宙中的智能数量每18个
月就会翻一倍

体系结构的黄金时代





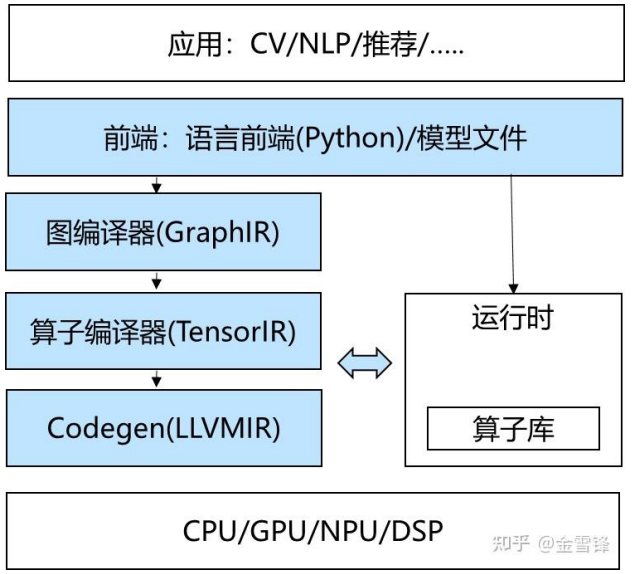
Turing Lecture, Hennessy, Patterson; June 2018 / CACM Feb 2019

https://www.rtinsights.com/gpus-the-key-to-cognitive-computing/

- **领域专用芯片的使用助推了人工智能的发展**

编译是工业界关注的焦点

- A New Golden Age for Computer Architecture
  - By John L. Hennessy, David A. Patterson；Communications of the ACM
- 新硬件推陈出新的时代
  - GPU、DPU、TPU、NPU、xPU
  - 量子计算机
- 新应用不断涌现的时代
  - AI计算、科学计算（AlphaFold）、量子计算
- 编译技术的需求日益旺盛
  - 华为方舟、毕昇编译器、龙芯二进制翻译、AI编译器（XLA、TVM）



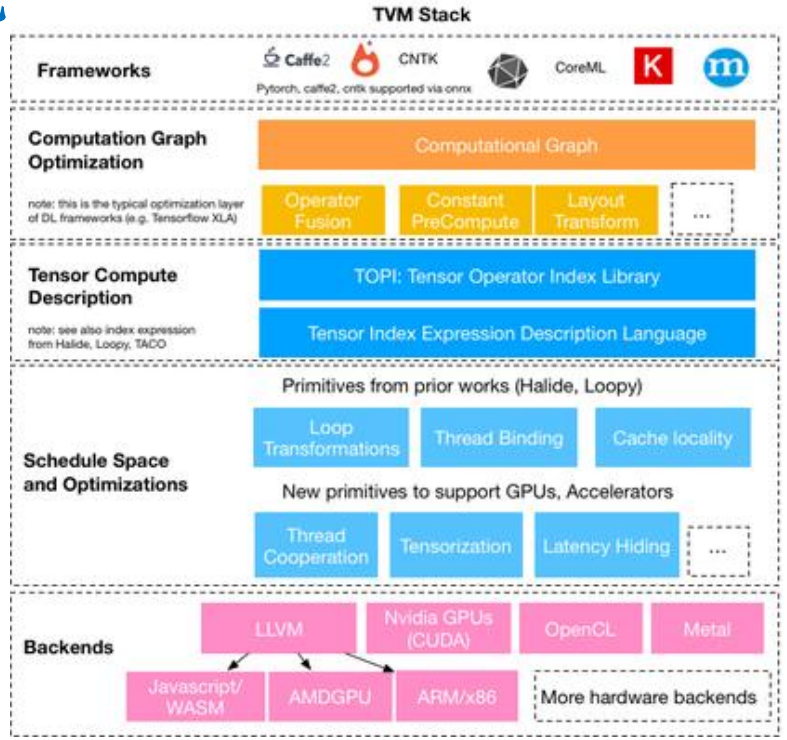
体系结构黄金时代的机会

- 软件为中心
  - 现代编程语言很多是script语言-> 需要解析、动态类型、代码复用
  - 编程高效但是执行不见得高效：Python vs. C
- 硬件为中心
  - 唯一可行的路径是领域专用架构
  - 一颗芯片只做一部分事情，但是性能很好
- 软硬协同的设计思路
  - 专用领域语言+专用领域架构：CUDA + GPU
  - 可以最大化挖掘硬件的并行度

新一代编译技术呼之欲出

❑新一代编译器需要具备以下基础能力

- 为异构芯片提供统一的硬件抽象
- 支持为异构芯片生成对应的目标代码
- 能理解专用领域的语言和编程范式
- 编译框架自身的质量、可靠性和性能



产业界编译人才紧缺

编译器 [杭州·滨江区·长河]

25-50K 经验不限 本科 刘女士 | 编译器与芯片...

华为 计算机软件 不需要融资 10000人以上

Go | 后端开发

加班补助，补充医疗保险，定期体检，五险一金，年终奖，...

编译器开发 [西安·雁塔区·高新软件园]

20-40K·16薪 经验不限 硕士 陈先生 | 研发工程师

龙芯中科 电子/半导体/集成电路 已上市 500-999人

编译器 | Loongarch | 龙芯

定期体检，住房补贴，生日福利，员工旅游，包吃，补充医...

编译器工程师 [上海·浦东新区·金桥]

18-35K 1-3年 本科 王先生 | 软件研发工程师

华为技术有限公司 计算机软件 不需要融资 10000人以上

数据结构 | 架构师 | ARM开发

住房补贴，定期体检，离职补偿，零食下午茶，补充医疗保...

编译器工程师 [上海·徐汇区·徐家汇]

20-30K·15薪 经验不限 本科 韩女士 | 技术经理

麒麟软件 计算机软件 未融资 1000-9999人

算法基础 | Linux | 编译工具链 | 汇编 | 反汇编

年终奖，免费班车，员工旅游，定期体检，五险一金，员工...

职位信息

职位描述：

1、负责基于编译技术的字节跳动核心业务的性能分析及优化；

2、负责各种编程语言(C++/Java/Go/Rust/WebAssembly)及周边基础库在业务中优化、技术推广与生态建设；

3、负责AutoFDO相关优化落地的研发维护；

4、负责前沿编译优化技术在字节跳动核心业务的落地；

5、负责编译技术针对X86_64, ARM64, RISC-V等体系结构的功能开发与性能优化；

6、负责异构编译技术针对AI领域与异构硬件的功能开发与性能优化；

7、参与研发业界领先的性能分析及优化平台。

职位要求：

1、熟悉编译原理以及相关编译优化技术；

2、熟悉LLVM/GCC/GraalVM/OpenJDK/Go中某个编译器框架或者Runtime；

3、熟悉C++/Java/Go/Rust/WebAssembly中某个语言的设计与实现原理；

4、熟悉Intel/AMD x86_64或ARM64、RISC-V体系结构，精通x86_64、ARM64、RISC-V微架构；

5、精通C/C++，熟悉C++底层实现原理；

6、有以下经验者优先：

a. 熟悉LLVM/GCC等C/C++编译器，并参与相关社区者优先；

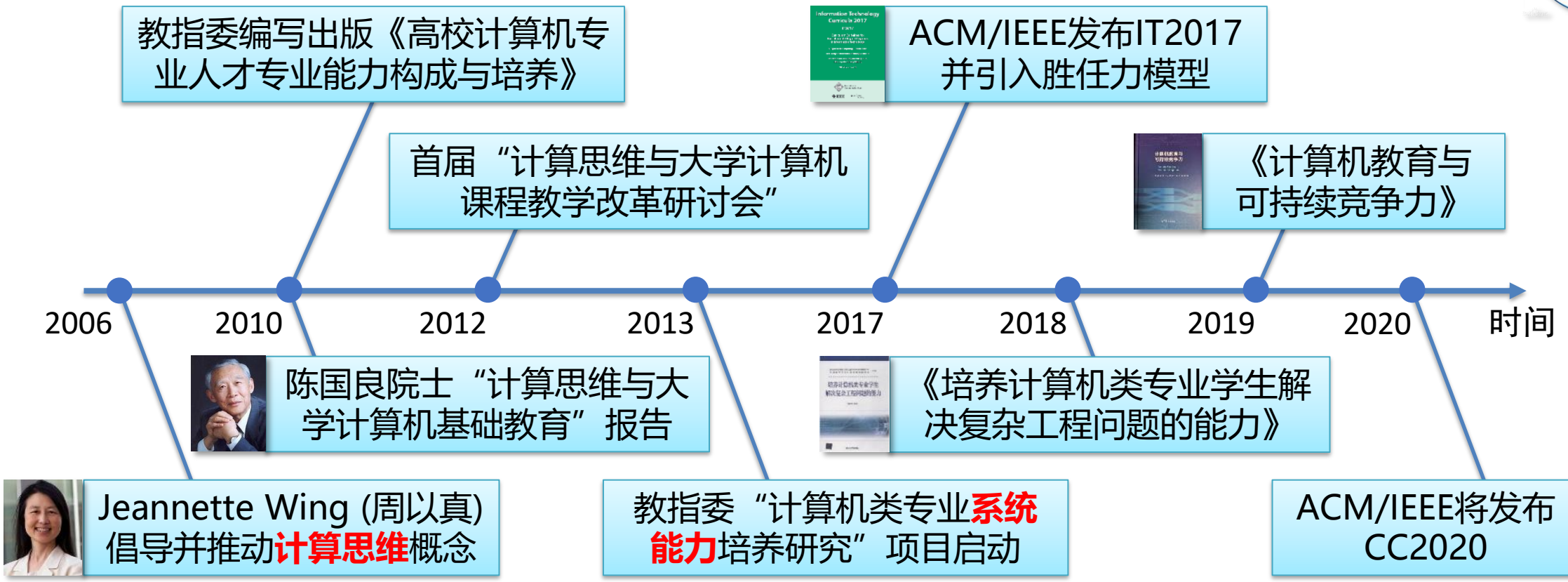
b. 有HPC/AI编译器以及高性能库开发经验者优先。

职能类别：系统架构设计师

汇报提纲

1. 编译器是什么?
2. 为什么要学习编译课程?
3. 编译教学的困境与科大方案
4. 本学期课程设置的变化
5. 选课与退课的恩恩怨怨

教育理念：计算机专业的学生需要对软件和硬件融汇贯通



“在异构计算的时代程序员必须对于算法和硬件模型融汇贯通，才能写出高质量的代码。因此，未来的程序员也必须懂硬件！”

——图灵奖得主David Patterson

编译教学组

张昱
老师

教指委计算机系统专家委员会委员；2019年度高校计算机专业优秀教师奖励计划；2018年获安徽省教学成果二等奖；主讲编译原理H班课程

郑启龙
老师

参与编写《并行算法实践》，译著《编译器构造－C语言描述》；国家级教学成果二等奖(09)；省教学成果特等奖(08)；主讲编译普通班课程

李诚
老师

安徽省青教赛一等奖(2021)；发表SOSP/OSDI等系统顶会论文；主讲编译普通班课程；主编《面向自主指令集的编译器设计与实现》。

李卫海
老师

信息安全专业副教授。研究方向包括多媒体内容安全、遥感信息处理等。主持国自然、国家863等项目。主讲网安编译课程。

徐伟
老师

实验教学中心高级实验师；指导学生获2020CCF大学生计算机系统与程序设计竞赛全国铜奖两项；参与编译课程实践体系建设

- **中科大编译原理课程为安徽省省级精品资源共享课程，第二批国家级线下一流本科课程，是全校本科毕业生最喜欢的十门课程之一**

编译教学及人才培养目标

计算机专业核心课

编译原理和技术

编译器

- ★ 一般构造原理
- ★ 基本设计方法
- ★ 主要实现技术

有助于培养学生的：

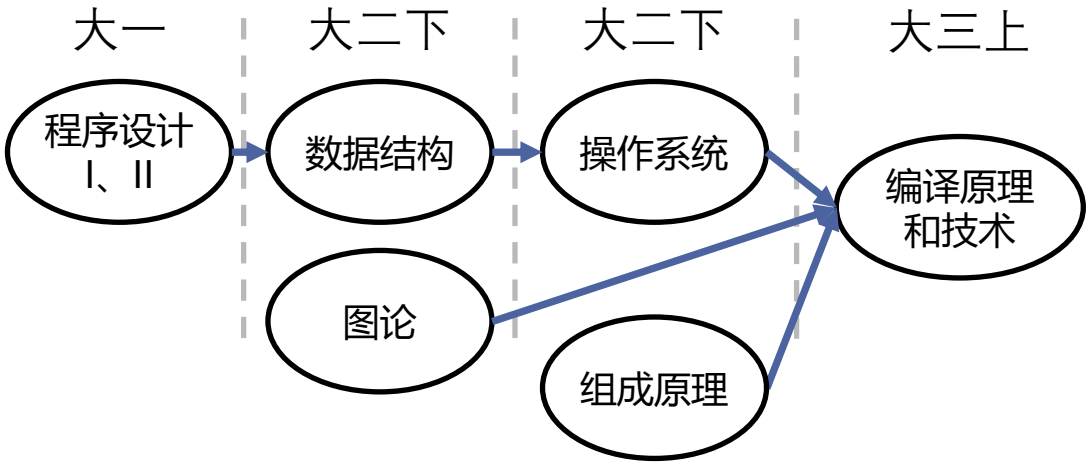
- **计算机系统能力**
- **综合、创新能力**
- **软件工程能力**

目标1：从事编译相关产业的领军人才

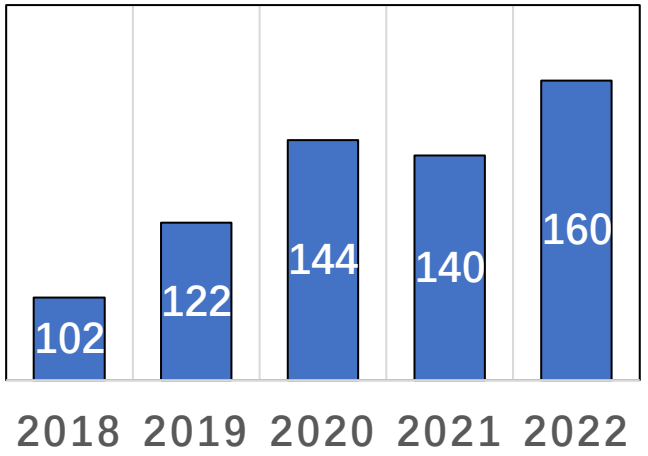
目标2：从事编译相关产业的技术骨干

目标3：使用编译技术解决其他关键问题的人才

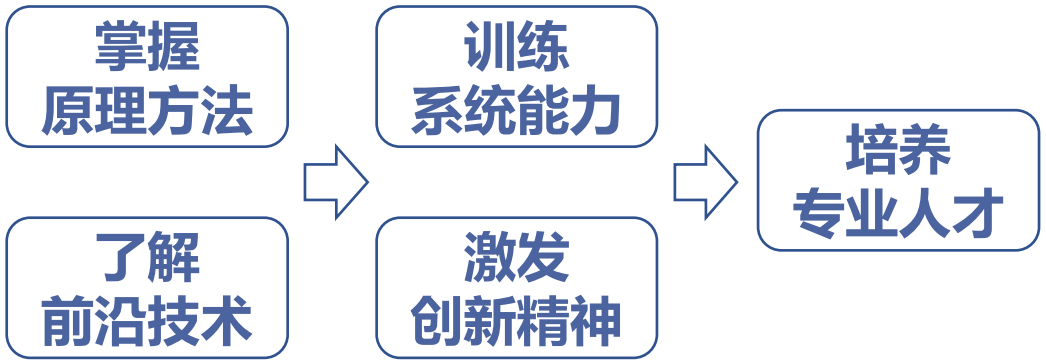
学情分析



专业核心课及其诸多先导课程



选课人数逐年增多



课程教学目标艰巨

知识交叉性强

受众广泛层次分明

学以致用要求高

技术革新速度快

评价指标单一

GPA至上固定思维

教学过程中面临很多挑战

编译教学改革需要解决的重要问题

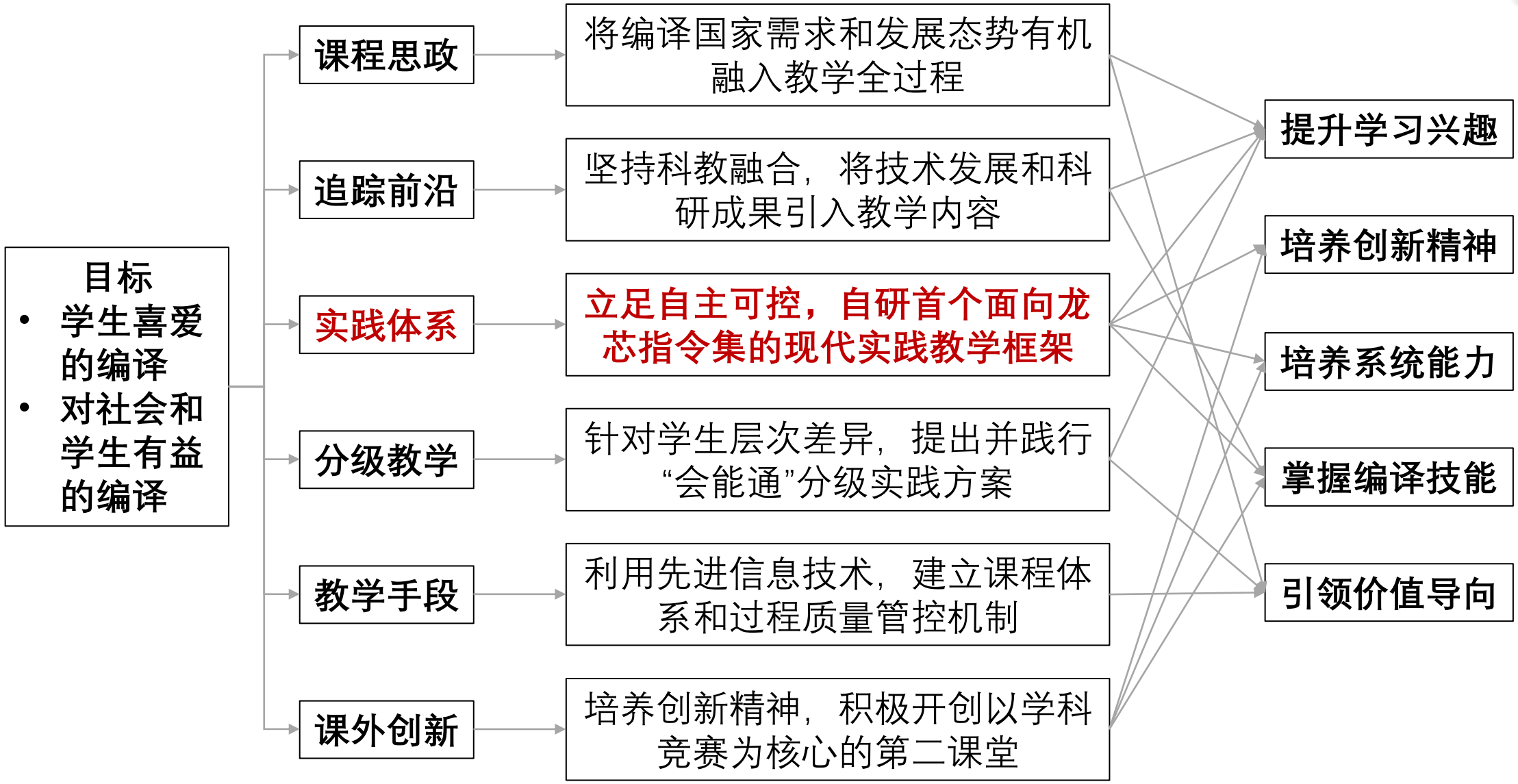
目标
- **学生喜爱的编译**
- **对社会和学生有益的编译**

编译技术不断发展，教学内容跟不上
- 智能计算时代需要更多懂编程语言原理和实现的人才，以构建可靠、高效系统，而中美科技竞争更加剧了对这类人才的需求。
- 传统编译教学体系落后于编译技术的发展，难以满足国家对编译人才的需求。

编译原理抽象难学，学生学习跟不上
- 编译原理课程涉及许多有难度的理论知识和算法，学生学习不易跟上。
- 避免过于注重细枝末节而忽略对编译器的宏观把控，探索实践工作量学习时间与高阶性、挑战度、学生获得感等的最佳平衡。

学生差异化需求大，课程评价跟不上
- 评价体系影响学生正确价值观的形成。
- 只看考试成绩和最终结果不能全面评价学习成效，也难以满足学生在能力和发展上的差异化需求，不利于因材施教、及时发现学习洼地和精准帮扶。

中科大编译课程的改革举措



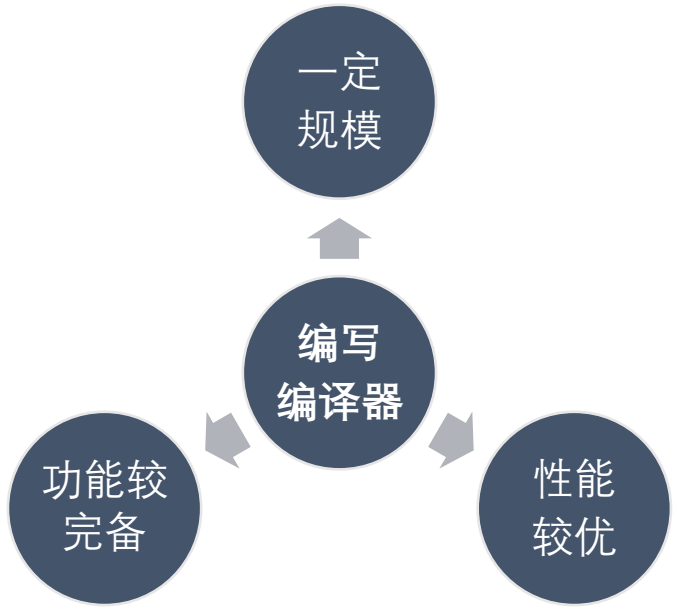
国内外实践体系调研

		前端语言类型	词法器	语法器	中间代码生成	后端代码生成	中间代码优化	寄存器分配
国外高校	ETH		√	√	LLVM	x86	√	√
	剑桥	基于Ocaml	√	√	Ocaml IL	ARM、RISC-V	√	√
国内高校	清华大学(交叉院)	C/C++/Python	√	√	√	RISC-V	×	√
	哈工大		√	√	√	×	×	×
	东北大学	PASCAL	√	√	四地址码	x86	×	×
	东北师大	×	×	×	×	×	×	×
	中科大	C/C++	√	√	LLVM	LoongArch	√	√

- **部分高校未开展编译实践；部分学校未贯通编译实践的全部环节；LLVM重要但未被广泛采用；x86和RISC-V仍是主流；国产自主体系未体现**

编译实验体系总览

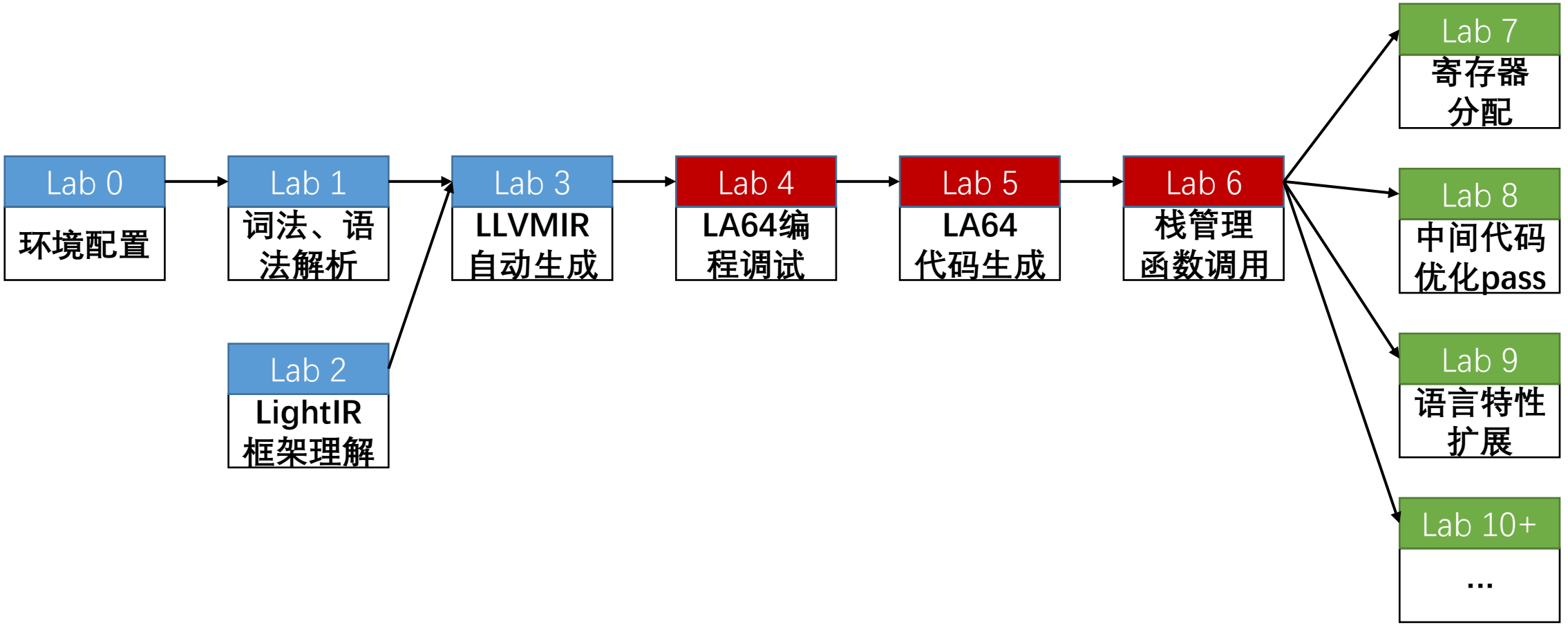
- 目标：面向**国家战略需求**，以**系统和创新能力**的培养为导向，根据计算机学科发展的趋势和新时代大学生的成长需要构建**现代编译实验体系**，体现**先进性、挑战度，增强学生获得感**



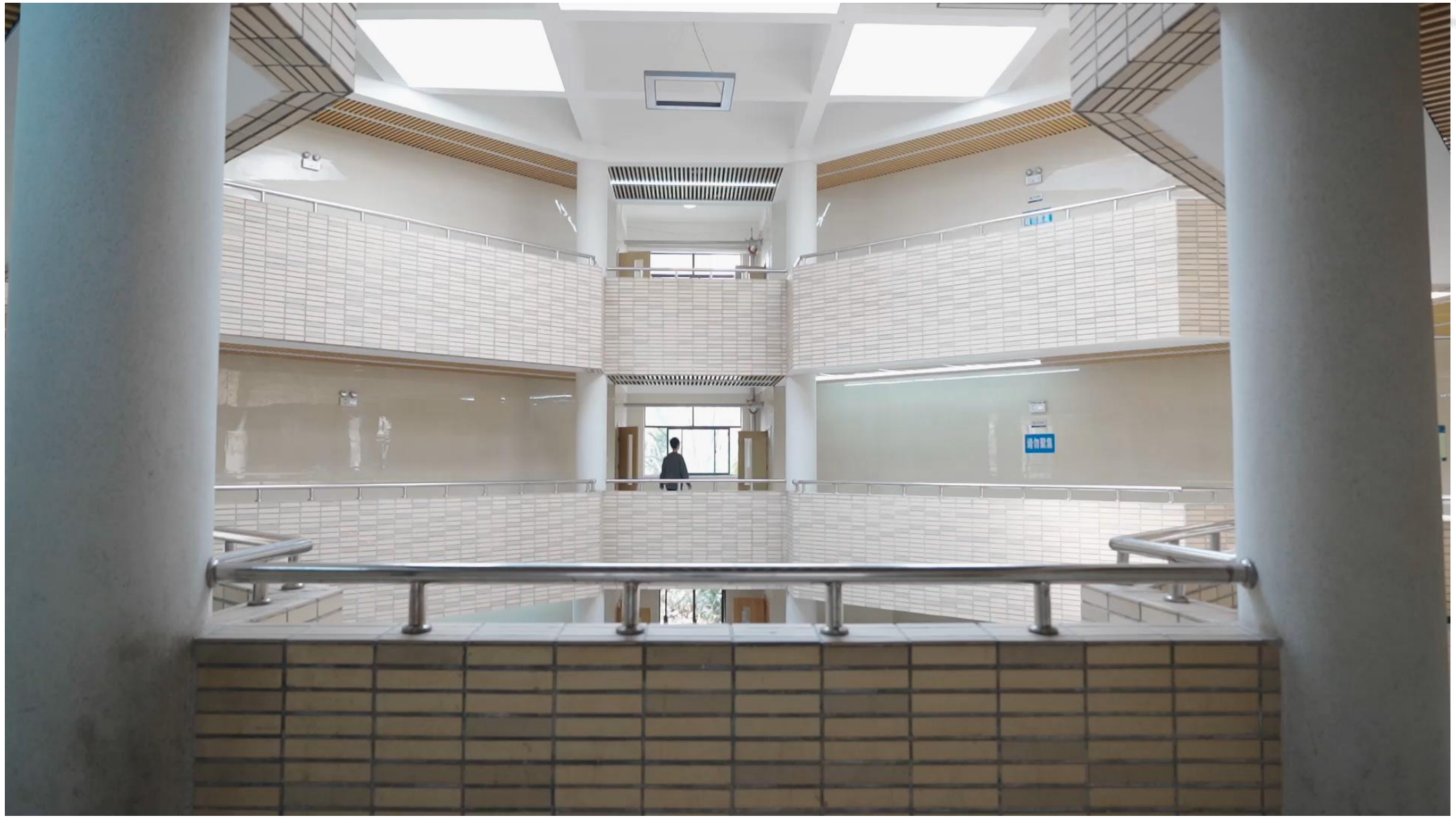
源语言特性	基于Cminus语言，增加对浮点数、数组等复杂特性的支持
词法语法分析器	使用Flex+Bison**工具构造编译器前端**，重点考察对Cminus语言正则表达式和语法产生式的抽象能力
中间代码生成器	使用简化版LLVM中间表达子模块和基于访问者模式的编程框架(**自研**)，自动生成 Cminus程序的中间代码
中间代码优化器	使用封装好的基本块、调用者使用者链表等常用数据结构及其接口(**自研**)，实现简单/复杂代码优化算法
目标代码生成器	支持龙芯(**与龙芯合作研发**)、RISC-V等多指令集后端，鼓励对寄存器分配算法、指令选择算法等进行探索验证

编译实验架构图

❑基础实验+龙芯后端实验+高阶创新实验有机结合



鼓励、引导、指导学生课外创新



教学获奖

- 本团队教师获得了各类奖励30+项，其中全国性奖励20项，省部级3项。

等级	名称	主办单位
全国	第二批国家级一流本科线下课程	教育部
省部级	安徽省第五届普通高校青年教师教学竞赛工科组一等奖	安徽省教育厅
省部级	第二届全国高校教师教学创新大赛（安徽赛区）选拔赛暨安徽省赛正高组二等奖	安徽省教育厅
全国学会	第四届中国计算机教育大会计算机类教学资源建设优秀课程配套资源特等奖	教育部计算机类专业教指委
全国学会	第四届中国计算机教育大会计算机类教学资源建设优秀教学案例特等奖	教育部计算机类专业教指委
全国学会 2020-2022	全国大学生计算机系统能力大赛编译系统设计比赛（华为毕昇杯）优秀指导教师奖	全国大学生计算机系统能力大赛组委会

学生获奖

- 自2020年起，指导学生参加各类学科竞赛取得了40+项奖励，其中26项全国性奖励，7项冠军或一等奖。

竞赛	获奖等级
全国大学生计算机系统能力大赛编译系统设计比赛（华为毕昇杯）	唯一特等奖（1）、二等奖（5）、三等奖（3）
中国高校计算机大赛团队程序设计天梯赛全国总决赛	团体一等奖（1）、三等奖（1）、安徽省高校冠军（1）、团队冠军（1）
CCF大学生计算机系统与程序设计竞赛	全国金奖（1）、全国铜奖（7）
中国计算机学会大数据与智能计算大赛	赛道冠军
华为开发者大赛MindSpore网络模型挑战赛	金奖（2）

专家评价

陈文光 清华大学计算机系教授，兼任青海大学计算机系主任。主要研究领域为操作系统、程序设计语言与并行计算。获国家科技进步二等奖一次，部级科技一等奖两次。现为中国计算机学会杰出会员和杰出讲者，副秘书长，青年科技论坛荣誉委员；ACM中国理事会共同主席。

荣誉证书

中国科学技术大学：

在2020年全国大学生计算机系统能力大赛编译系统设计赛（华为毕昇杯）全国总决赛中，荣获特等奖。

特发此证。

参赛队员：[illegible]

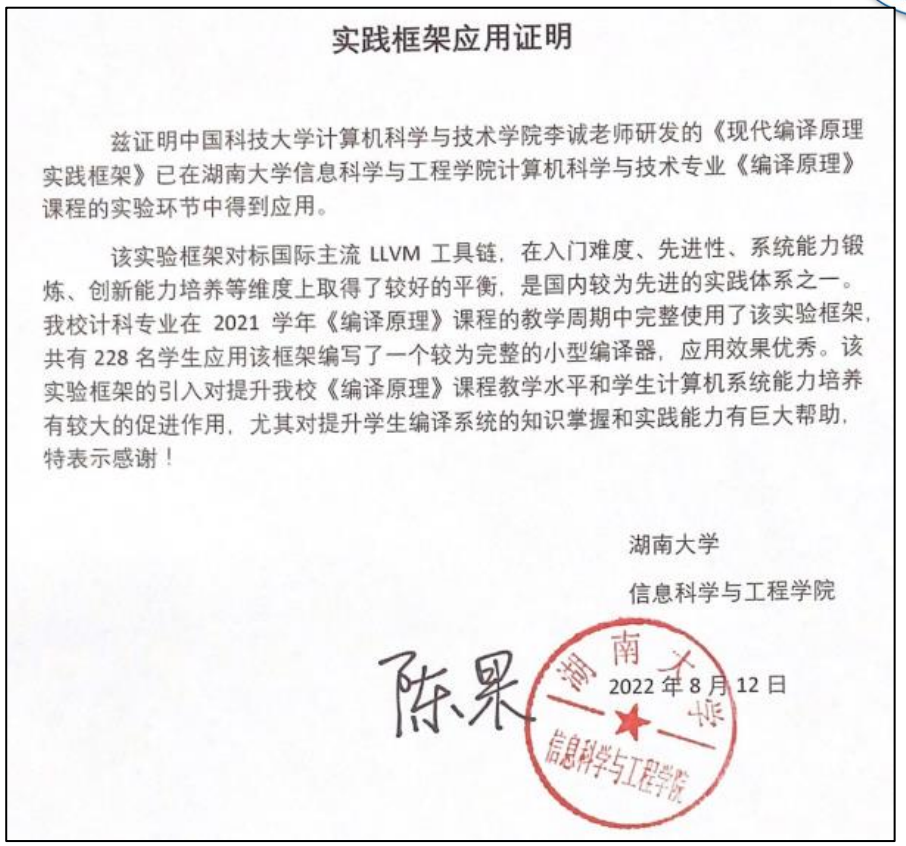
全国高等学校计算机教育研究会
全国大学生计算机系统能力大赛组委会
二〇二〇年八月十六日

2020年编译大赛陈文光评委点评：

“自动向量化和自动多线程优化是编译优化中非常复杂的部分，**中科大队伍**开发的这些优化展现出了他们在编译基础知识上的掌握深度和积极主动的钻研态度，这也是**中科大**在编译和程序设计语言方面**高水平教学的直接体现**。”

推广效果和价值

- 来自中国科大、四川大学、兰州大学、西北工业大学、合肥工大、安徽大学、河南理工大学等高校的教师参加了师资培训



实践框架应用证明

兹证明中国科技大学计算机科学与技术学院李诚老师研发的《现代编译原理实践框架》已在湖南大学信息科学与工程学院计算机科学与技术专业《编译原理》课程的实验环节中得到应用。

该实验框架对标国际主流 LLVM 工具链，在入门难度、先进性、系统能力锻炼、创新能力培养等维度上取得了较好的平衡，是国内较为先进的实践体系之一。我校计科专业在 2021 学年《编译原理》课程的教学周期中完整使用了该实验框架，共有 228 名学生应用该框架编写了一个较为完整的小型编译器，应用效果优秀。该实验框架的引入对提升我校《编译原理》课程教学水平和学生计算机系统能力培养有较大的促进作用，尤其对提升学生编译系统的知识掌握和实践能力有巨大帮助，特表示感谢！

湖南大学

信息科学与工程学院

陈果

2022 年 8 月 12 日

湖南大学 信息科学与工程学院

- 实践框架已在中国科大/湖南大学等计算机专业应用，辐射学生1000+

汇报提纲

1. 编译器是什么?
2. 为什么要学习编译课程?
3. 编译教学的困境与科大方案
4. 本学期课程设置的变化
5. 选课与退课的恩恩怨怨

课程设置

- 时间：每周一(6,7)、三(3,4)
- 地点：高新区GT-B212教室
- 课程信息化平台（3个网站1个群）：
  - 课程主页：
    - https://ustc-compiler-principles.github.io/2023/
    - 发布讲义+实验信息
  - 希冀平台：
    - http://cscourse.ustc.edu.cn/
    - 作业提交、代码提交、自动化评测
  - Git平台：
    - 代码仓库
  - qq群（805431263）
    - 发布通知、非业务交流

参考资料

❑教材和参考书

- 陈意云、张昱，编译原理（第3版），高等教育出版社，2014
- 李诚、徐伟，面向自主指令集的编译器设计与实现，高等教育出版社，2023
- A. V. Aho, M. S. Lam, R. Sethi, and J. D. Ullman 著，赵建华等译，编译原理，机械工业出版社，2017

❑其他资料

- Stanford课程主页

  http://web.stanford.edu/class/cs143/
- MIT课程主页：

  http://6.035.scripts.mit.edu/fa18/

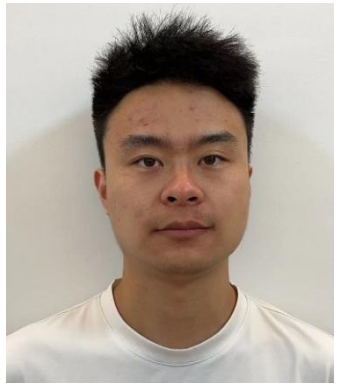
考核要求

- 考核内容包括理论学习、工程实践
- 成绩组成：
  - 平时考核（15%）
    - 按时上课（特殊情况不能来需要书面请假）
    - 按时完成课后作业
  - 考试（30-40%）
  - 工程实践（45-55%）

助教天团

徐伟 博士
高级实验师

金泽文
博士在读



陈清源
博士在读

龚平
博士在读

贺嘉
研究生在读

许坤钊
大四保研

傅申
大四保研

杨柏森
大四保研

李晓奇
大四保研

刘良宇
微软实习

本学期的实验要求（后续可能有变动）

阶段	实验名称	难度系数	时间安排	是否计入成绩
基础实验	Lab 0 系统配置	-	1周	是
	Lab 1 词法语法器	★	1.5周	是
	Lab 2 中间代码生成器	★★	2.5周	是
高阶实验	Lab 3 龙芯后端代码生成器	★★★	2.5周	是
拔高实验	Lab 4 代码优化	★★★★	3周	是
竞赛实验	Lab 5 功能完善+性能打榜	★★★★★★	不计入教学周	否

- **一等奖：3000；二等奖：1500；三等奖：800**
- **外加其他一些礼物**

汇报提纲

1. 编译器是什么?
2. 为什么要学习编译课程?
3. 编译教学的困境与科大方案
4. 本学期课程设置的变化
5. 选课与退课的恩恩怨怨

本课程适合什么样的学生?

- 想深入理解编译技术，并付诸实践的学生！
- 认准目标不轻易放弃的学生！
- 乐于沟通，懂得寻找方法解决问题的学生！
- 喜欢高分，但能正确认识分数与付出之间的关系的学生！
- 不是总抱怨，而是提出建设性意见的学生！

慎重，可退课，if, but not limited to

- 不想花太多时间在课程实践上
- 不喜欢合作，不愿意交流，不懂得沟通
- 不喜欢提前规划学习，总是临时抱佛脚
- 心理承受能力比较低
- 做事情虎头蛇尾
- 希望考高分，但不能正确理解分数与付出间的正确关系
- 严重依赖往届开源实现，寄希望于CSDN、chatGPT

请参考评课社区分数0分或者1分的评论！

一起努力
打造国产基础软硬件体系！

李 诚

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

2023年09月04日