



群聊：ICS2026春-柴谢班



该二维码7天内(3月8日前)有效，重新进入将更新

计算机系统基础 11

谢旻晖

2026.3

ICS2 - 柴谢班

班级容量500人

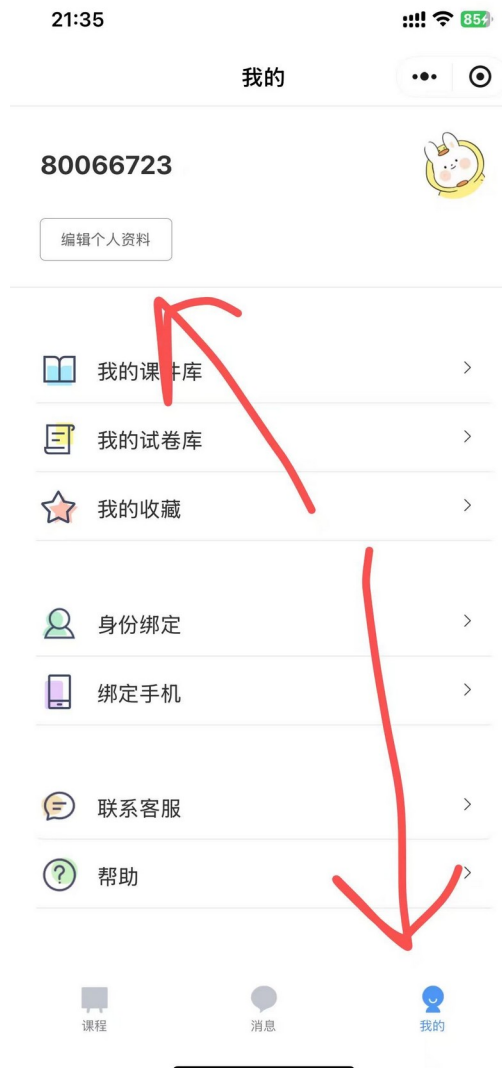


二维码有效期截至 2026-03-30

邀请码:8GXOXC

雨课堂 公众号中输入，长期有效

雨课堂编辑个人资料，修改姓名学号



目录

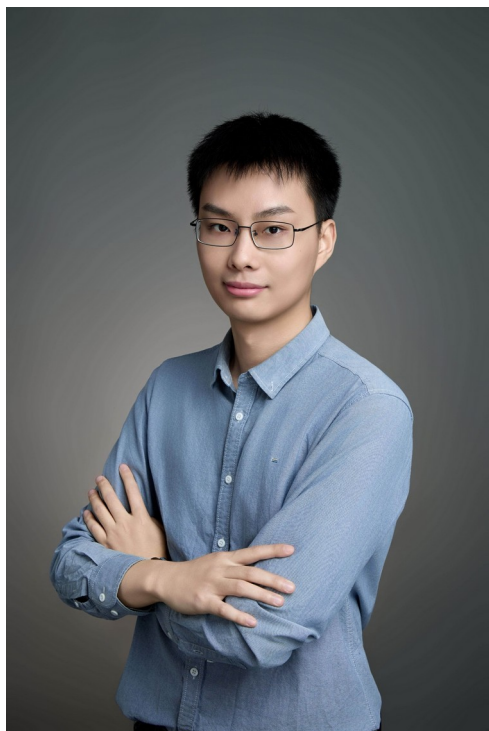
- 教师、助教信息
- 教材和成绩构成
- 学习动机

授课教师：柴云鹏

- 中国人民大学信息学院
 - 教授、博士生导师
- 云计算与大数据系统实验室 CDSLab
 - 信息楼 239、505、124、125
 - Email : ypchai@ruc.edu.cn
 - WeChat: chaiyunpeng1983
- 研究方向
 - 云计算、分布式系统、大数据系统、新硬件、AI4Sys



授课教师



谢旻晖

Email: xieminhui@ruc.edu.cn

电话 / 微信 : 18801119418

办公室 : 信息楼 505

研究方向 : 机器学习系统、存储系统

博士 : 清华大学计算机系 (2019-2024)

本科 : 南京大学计算机系 (2015-2019)

课程介绍

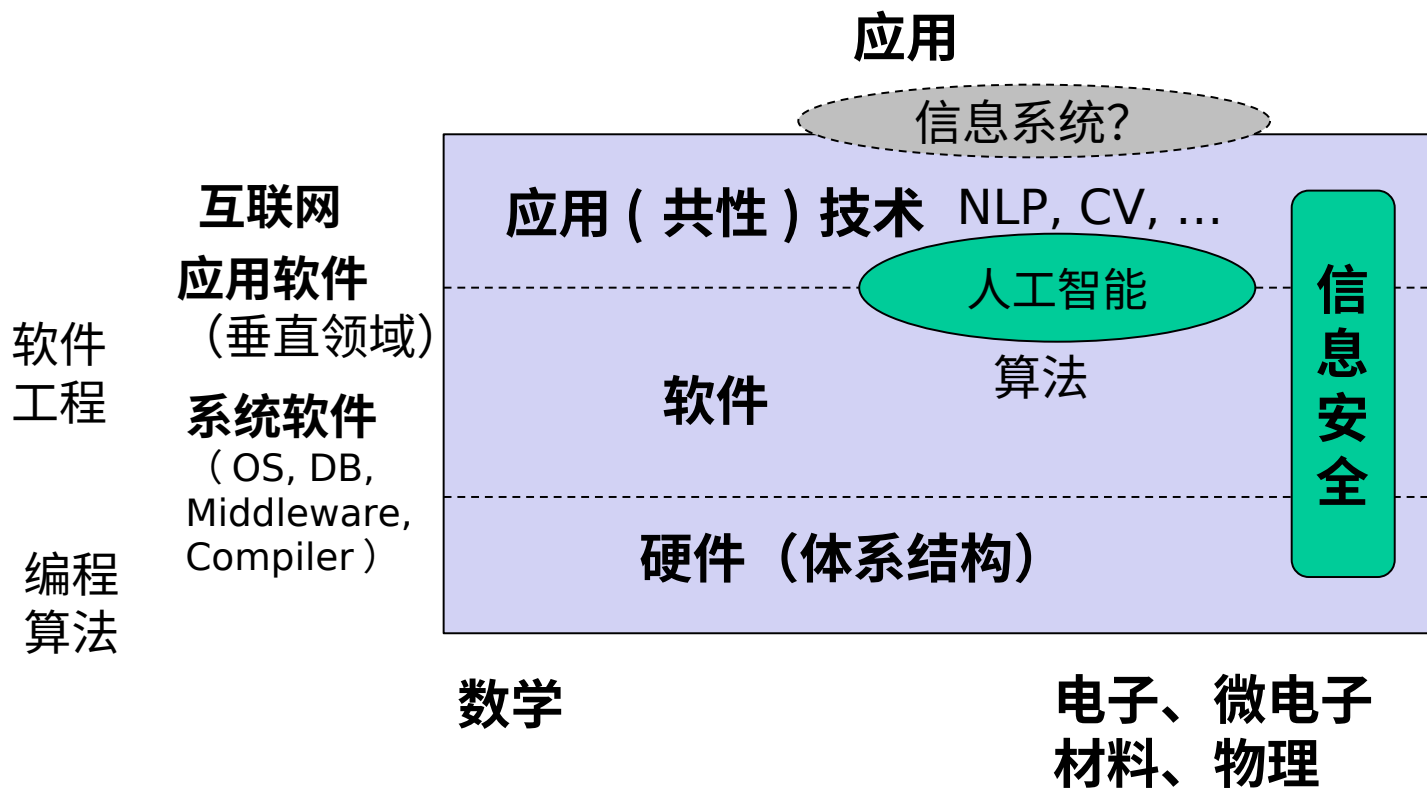
- 计算机系统主线
 - **大四：毕业设计**（复杂开源系统分析 / 测评 / 扩充 / 优化）
 - LevelDB, RocksDB
 - TiKV
 - Cgroup, Storage Provisioner
 - Ceph
 -
 - **大三：构造复杂系统**（CPU*、操作系统*、编译器、分布式系统#、数据库*）
 - * 图灵班课程
 - # 大数据专业课程
 - **大二：《计算机系统基础》 I 和 II**（平台课）
 - **大一：（C 语言程序设计）**

Motivation

- 为什么要学习这么课?
 - 计算机大类（计算机 / 软工 / 大数据 / 信安 / 人工智能）专业三大基础课
 - 程序设计 I 和 II (*)
 - 数据结构与算法 I 和 II (-> 应用, 含 AI 各类应用)
 - 计算机系统基础 I 和 II (-> 复杂工程 / 系统)
 - 本科课程前半场: 以上 + 数学 (-> 理论)
 - 本科课程后半场: 更复杂 / 更细分的专业课
 - 外场: 竞赛、科研 (拔尖人才)
 - 计算机底层 / 硬件相关课程已经大幅减少
 - 数电、汇编、组成、体系结构、微机原理 (嵌入式)

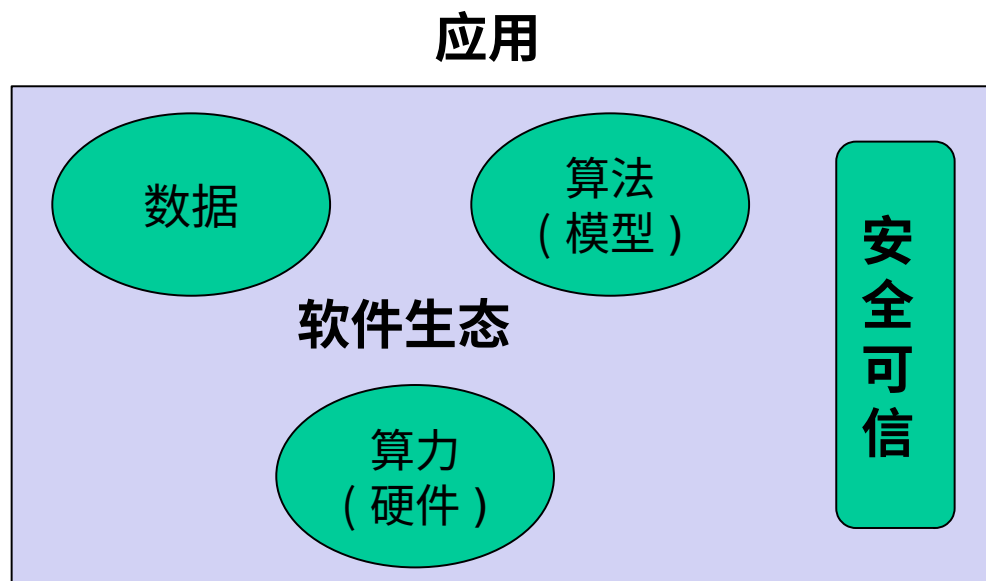
Motivation

- 怎么理解计算机科学与技术学科（!= 专业）？
 - 分层架构、不断扩展



Motivation

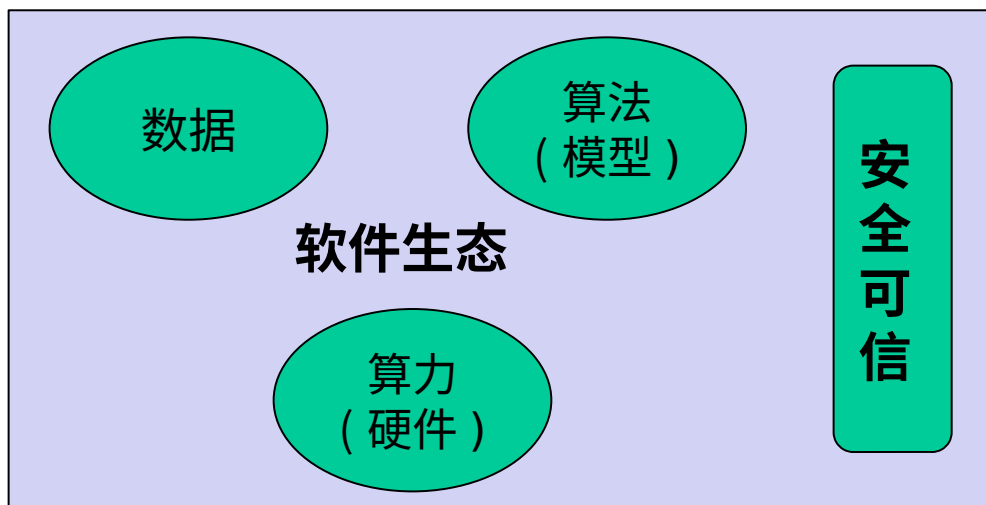
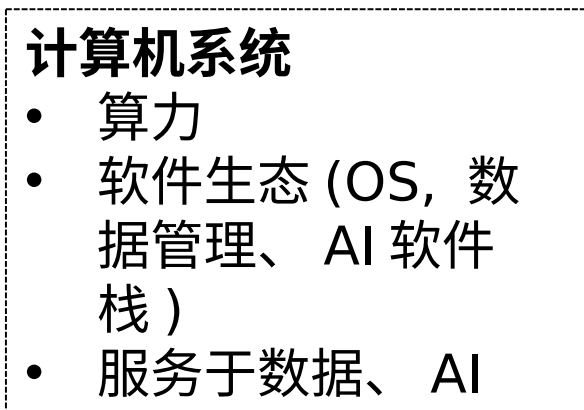
- 怎么理解计算机科学与技术学科 (≠ 专业) ?
 - 时代: 计算 -> 数据 -> 智能
 - (另一种视角) 三要素: 算法、算力、数据



Motivation

- 怎么理解计算机科学与技术学科 (≠ 专业) ?
 - 时代: 计算 -> 数据 -> 智能
 - (另一种视角) 三要素: 算法、算力、数据

应用



系统能力课程徽章



系统能力培养 · 荣誉勋章



- 往年，助教常常被大量重复的问题所困扰。
- 因此，他们不得不在群里反复发送消息，以广播某个常见问题——或者索性草率地回复。

为了避免该情况，助教们为大家提供了 **答疑吐槽灌水论坛**
匿名提问的方式既避免尴尬，也分享公共问题的回答

forum.rucics.tech 校园网访问 ruc 邮箱注册

开始正式课程之前，

- 迎接 AI 时代的变化
- 为什么在 AI 时代需要学习 ICS

开始正式课程之前

- 迎接 AI 时代的变化

State of the world



World ▾ Business ▾ Markets ▾ Sustainability ▾ Legal ▾ Commentary ▾ Technology ▾ Investigations

AI startups revolutionize coding industry, leading to sky-high valuations

By Anna Tong and Krystal Hu

June 3, 2025 6:46 PM PDT · Updated June 3, 2025



Meta's Zuckerberg pledges hundreds of billions for AI data centers in superintelligence push

14 February 2025

EDUCATION

Dancing with digital partners: The creative revolution of generative AI

FINANCIAL TIMES

JS COMPANIES TECH MARKETS CLIMATE OPINION LEX WORK & CAREERS LIFE & ARTS HTSI

Opinion **Artificial intelligence**

At work, a quiet AI revolution is under way

State of the world



Andrej Karpathy

@karpathy

It is hard to communicate how much programming has changed due to AI in the last 2 months: not gradually and over time in the "progress as usual" way, but specifically this last December. There are a number of asterisks but imo coding agents basically didn't work before December and basically work since - the models have significantly higher quality, long-term coherence and tenacity and they can power through large and long tasks, well past enough that it is extremely disruptive to the default programming workflow.

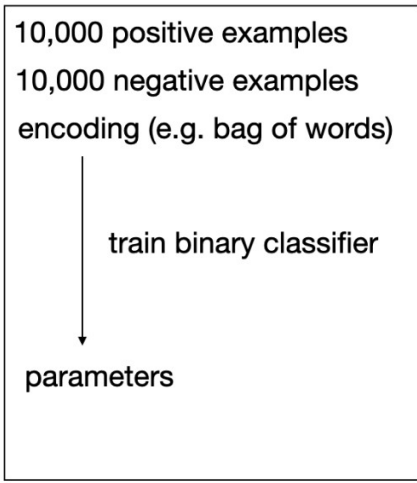
很难传达在过去两个月里由于 AI 而发生的编程变化有多大：这不是以“常规进展”的方式逐步发生的，而是具体在去年十二月发生的突变。虽然有若干需要注意的地方，但我认为在十二月之前，编码代理基本上不起作用，而自那以后基本上就能工作了——模型的质量显著提升，具备更强的长期一致性和韧性，能够攻克大型和长期任务，远超以往，足以对默认的编程 workflow 造成极大扰动。

State of the world

Software 1.0

```
python Copy  
  
def simple_sentiment(review: str) -> str:  
    """Return 'positive' or 'negative' based on a tiny keyword lexicon."""  
    positive = {  
        "good", "great", "excellent", "amazing", "wonderful", "fantastic",  
        "awesome", "loved", "love", "like", "enjoyed", "superb", "delightful"  
    }  
    negative = {  
        "bad", "terrible", "awful", "poor", "boring", "hate", "hated",  
        "dislike", "worst", "dull", "disappointing", "mediocre"  
    }  
  
    score = 0  
    for word in review.lower().split():  
        w = word.strip(",.!?;:") # crude token clean-up  
        if w in positive:  
            score += 1  
        elif w in negative:  
            score -= 1  
  
    return "positive" if score >= 0 else "negative"
```

Software 2.0



Software 3.0

You are a sentiment classifier. For every review that appears between the tags
<REVIEW> ... </REVIEW>, respond with **exactly one word**, either
POSITIVE or NEGATIVE (all-caps, no punctuation, no extra text).

Example 1
<REVIEW>I absolutely loved this film—the characters were engaging and
the ending was perfect.</REVIEW>
POSITIVE

Example 2
<REVIEW>The plot was incoherent and the acting felt forced; I regret
watching it.</REVIEW>
NEGATIVE

Example 3
<REVIEW>An energetic soundtrack and solid visuals almost save it, but
the story drags and the jokes fall flat.</REVIEW>
NEGATIVE

Now classify the next review.

State of the world

职位描述

微信扫码分享

Golang

golang

服务端开发

计算机相关专业

后端开发

容器技术

Kubernetes

1. 2 年左右基础设施/云计算后台开发经验

2. 熟练掌握vibe coding及AI使用，熟悉Golang或其他任意主流编程语言；

3. 有容器、k8s或其他云平台开发

4. 有存储、大数据、AI、通用计算

5. 有任意云计算/云原生相关开源

I messed up my Google PM Vibe Coding Interview

I had my 2nd round of Google PM interview yesterday after doing well in my first round and was totally blindsided by the format. I had no clue there would be a vibe coding interview—I was expecting a traditional PM interview, but that didn't happen.

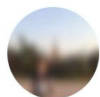
During the interview, I ended up being very unstructured. Having built AI applications at my own startup, I know the pitfalls of every tool out there, and in these situations, the main bottleneck is the backend. So, I focused all my energy on the backend during the interview.

What I *should* have done was approach it like a product design interview: prepare a small PRD, use that to create a prompt, and then maybe run it in something like Firebase Studio, Lovable, or V0. Instead, I jumped into building my Langchain backend agent first and completely messed up the code for tool calling in Langchain. I was stressed and just blanked on the exact implementation.

My biggest regret is that, if I'd even had the slightest clue about the format, I would have come in with a structure prepared. I just didn't manage my 45 minutes well. The knowledge was there, but the execution was honestly pathetic. Not sure what else to say—I guess I just needed to share.

Deep down, what's making me feel worse is that I don't have too many opportunities—if I mess up the ones that come, it feels like I'm done for, especially since there are limited opportunities in India for AI jobs.

Edit 1: Another reason for posting here is anyone who is planning on interviewing for a similar role knows there could be vibe coding interviews. In all my research I didn't find any mentions of a vibe coding interview, so hopefully this knowledge will help someone else in the future



李先生

今日

腾讯云 · 专家工程师

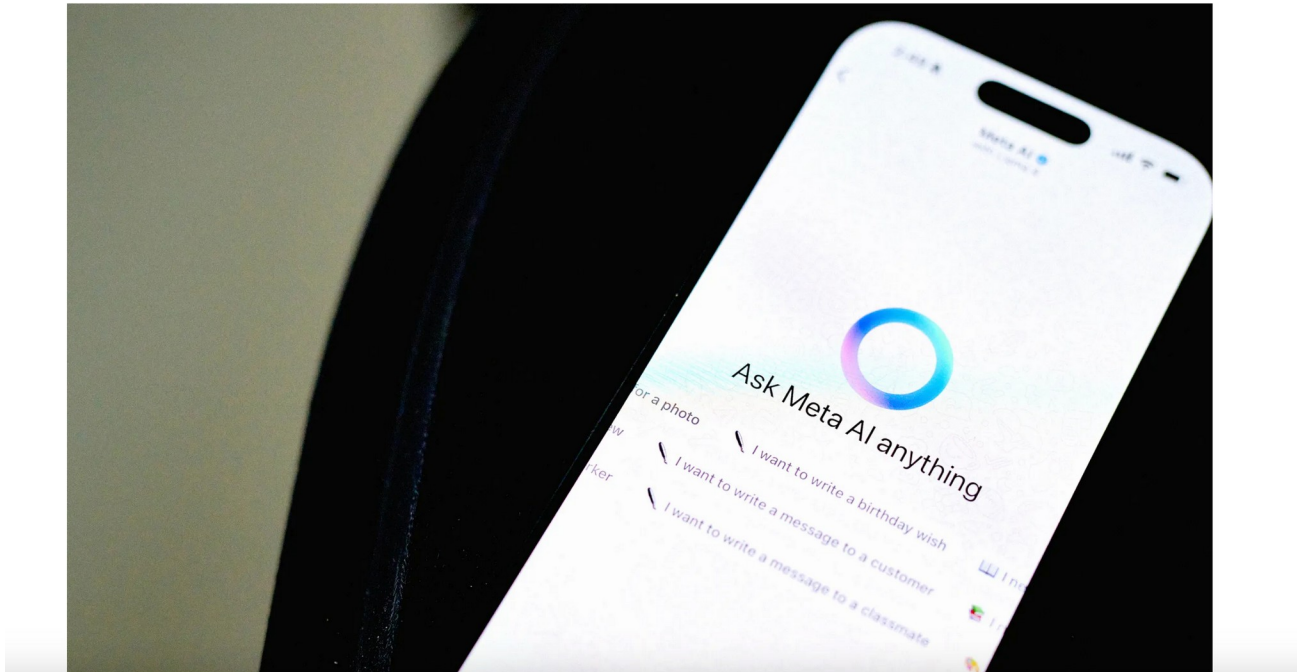
State of the world

JASON KOEBLER

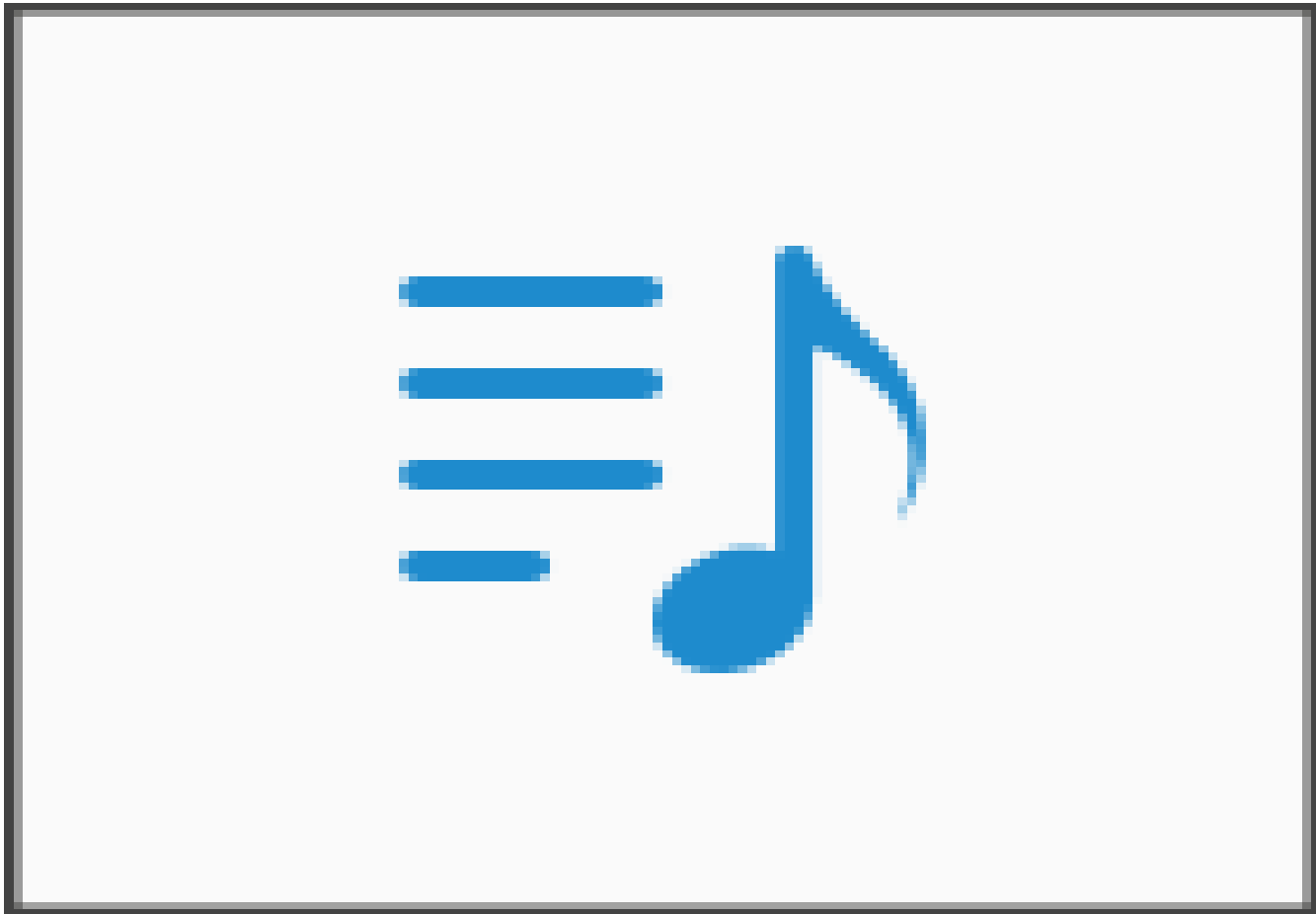
BUSINESS JUL 29, 2025 9:00 AM

Meta Is Going to Let Job Candidates Use AI During Coding Tests

Mark Zuckerberg has said vibecoding will be a major part of Meta's engineering work in the near future.



State of me



Question

The screenshot shows the Zhihu website interface. At the top, there is a navigation bar with the Zhihu logo and links for '关注', '推荐', '热榜', '专栏', '圈子', '付费咨询', and '知学堂'. A search bar contains the text 'AI 程序员' and a search icon. Below the search bar, there are tabs for '综合', '用户', '论文', 'AI 搜索', '专栏', '盐选内容', '电子书', '圈子', '话题', '视频', and a '筛选' button. The main content area displays three search results for the query 'AI 程序员':





- 人工智能真的会让程序员在5年内失业吗?**
杨天: 这个也是我说的有架构能力的人会苟一阵的原因。AI 搞不定大框架, 普通中低阶的程序员也搞不定。c AI 会有幻觉。没错, 但是AI不会累, 改... [阅读全文](#)
赞同 355 | 225 条评论 | 02-27
- ai 已经能编出很完美的程序, 程序员这个行业以后是不是会消失?**
pansz: 如果真按照成本定价, 程序员那点工资根本不够AI消耗的。所以程序员显然没法消失, 因为AI行业最终发现算力不是无限增长的... [阅读全文](#)
赞同 137 | 56 条评论 | 02-09
- 你觉得现在的 ai (例如 cursor) 可以替代程序员吗?**
吕小布: 资本也不是傻子, ai 真能取代程序员, 现在各大厂早就开始招聘ai编程人员了。未来的某天也许ai真的能取代程序员 [阅读全文](#)
赞同 185 | 126 条评论 | 2025-09-29

Below the search results, there is a section for '大家都在搜' (What everyone is searching for) with a list of trending topics:

- 美以袭击伊朗 445 万
- 哈梅内伊遇袭身亡 444 万
- 中国男篮险胜中国台湾男篮 444 万
- 2025 年度个税汇算 444 万
- 伊朗局势 444 万
- 以色列宣布袭击伊朗 442 万
- 中国手机行业将迎来全面... 440 万
- 伊朗: 即将投入一系列「...」 348 万
- 反诈老陈账号被封禁 328 万
- 美以刺杀伊朗最高领袖和... 309 万

At the bottom of the page, there are links to '帮助中心', '举报中心', and '关于知乎', along with the website's ICP license information: '京 ICP 证 110745 号 · 京 ICP 备 13052560 号'.

Bad News (2026.2.27) ——Block

10:59    

← 发帖

今天我们要把@blocks缩小一些。这是我给公司的说明。

####

今天，我们要做出公司历史上最艰难的决定之一：我们将裁员近一半，从一万多人缩减到不足六千人。这意味着超过四千人将被要求离职或进入咨询阶段。我会坦率地告诉大家发生了什么，为什么会这样，以及这对每个人意味着什么。

首先，如果您是受影响的员工之一，您将获得20周的工资，外加每工作一年1周的工资，截至5月底的股权，6个月的医疗保险，公司提供的办公设备，以及5000美元的过渡补助，用于帮助您顺利过渡（如果您身处美国境外，您将获得类似的帮助，但具体细节将根据当地规定而有所不同）。我希望您首先了解这一点。今天，所有人都会收到通知，无论您是被要求离职、进入协商阶段还是被邀请留任。

我们做出这个决定并非因为公司陷入困境。我们的业务发展强劲，毛利润持续增长，客户数量不断增加，盈利能力也在不断提升。但情况已经发生了变化。我们已经看到，我们正在开发和使用的智能工具，结合规模更小、组织结构更扁平的团队，正在催生一种全新的工作方式，从根本上改变了公司建设和运营的模式。而且，这种趋势正在迅速发展。

我当时有两个选择：要么随着这种转变逐步推进，在几个月甚至几年内慢慢裁员；要么坦诚面对现状，立即采取行动。我选择了后者。反复裁员会打击士气，分散注意力，也会损害客户和股东对我们领导能力的信任。我宁愿现在就采取果断而明确的行动，从我们坚信的立场出发，继续发展，也不愿缓慢地裁员却最终得到相同的结果。规模更小的公司也让我们有空间按照自己的方式发展业务，而不是不断地应对市场压力。

如此大规模的决策必然存在风险，但原地踏步同样充满风险。我们已进行全面评估，确定了未来业务稳定增长所需的人员和岗位，并从多个角度对这些决策进行了压力测试。我承认我们可能在某些方面存在失误，因此我们预留了灵活性来应对这些失误，并确保为客户做出正确的选择。

我们不会突然把人从Slack和邮件里删除，假装他们从未存在过。沟通渠道会一直开放到周四晚上（太平洋时间），这样大家就可以好好道别，分享任何想说的话。我也会在太平洋时间下午3:35主持一个视频直播，感谢大家。我知道这样做可能会让人觉得尴尬。但我宁愿这种略显尴尬但充满人情味的方式，也不愿显得冷冰冰、效率低下。

致即将离开的各位.....我感谢你们，也为让你们经历这一切感到抱歉。是你们缔造了公司今天的辉煌，这一点我将永远铭记于心。这个决定并非否定你们的贡献，你们未来在任何组织都会是不可或缺的人才。

致留下的各位.....这是我do.的决定，我会承担责任。我希望你们能与我携手共建。我们将以智能为核心打造这家公司，贯穿我们工作的方方面面，包括我们的创造方式、工作方式以及服务客户的方式。我们的客户也会感受到这种转变，我们将帮助他们顺利过渡到未来：他们可以利用我们的技术，通过我们的界面，直接构建自己的功能。这就是我目前的工作重点。明天我会发邮件通知大家。

杰克

26年2月27日, 5:12 · 2418万 查看



Bad News (2026.2.27) ——Block



11:17

← 发帖

TKL The Kobeissi Letter @KobeissiLetter 订阅

BREAKING: Block stock, \$XYZ, surges over +20% after announcing plans to cut over 40% of their workforce.

翻译帖子

Block, Inc. USD

054.61 H66.23 L54.25 C65.50 -10.94 (-20.05%)

@KOBESSILETTER

26年2月27日, 5:13 · 179万 查看

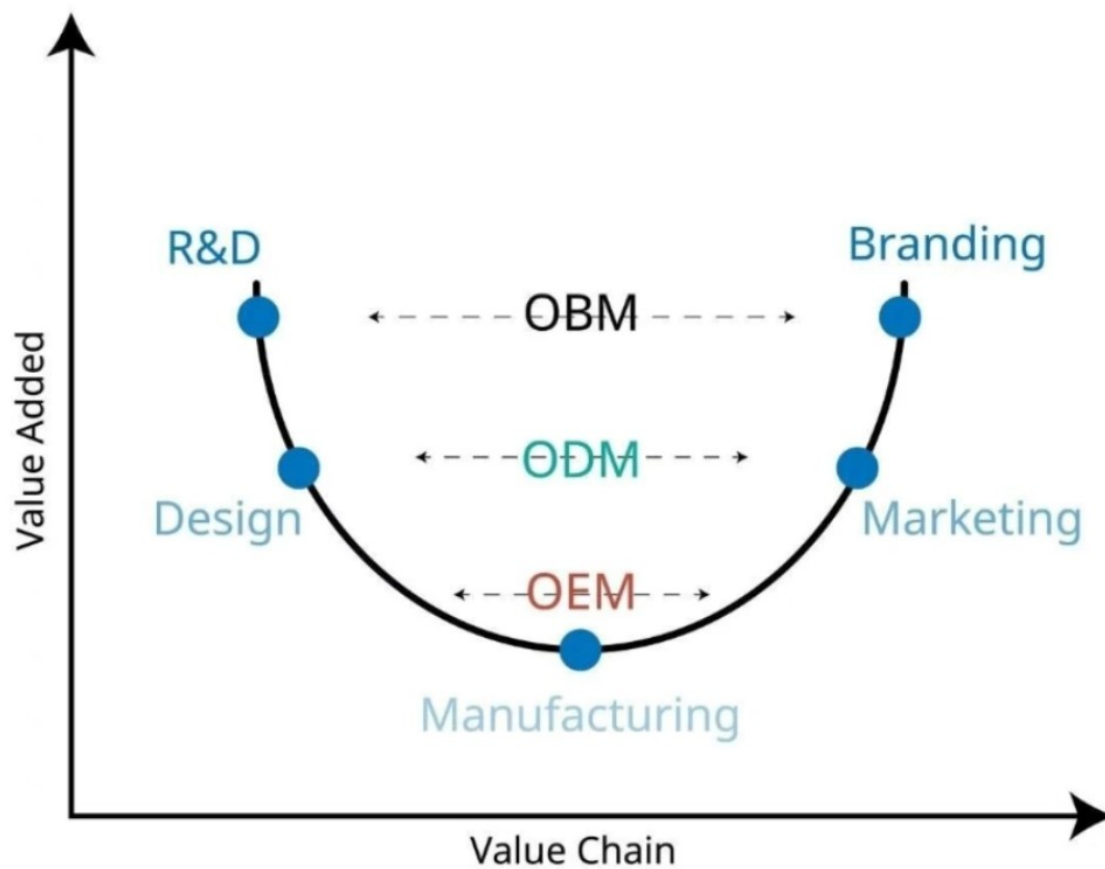
404 转发 400 评论 2,670 喜欢 444 点赞

发布你的回复



AI 时代下的 CS 教育

- 迎接变化



AI 时代下的 CS 教育

CS146S: The Modern Software Developer

Stanford University • Fall 2025



[Overview](#)

[Syllabus](#)

[FAQ](#)

AI coding strategies that work delivered to your inbox.

Get updates

Course Description

In the last few years, large language models have introduced a revolutionary new paradigm in software development. The traditional software development lifecycle is being transformed by AI automation at every stage, raising the question: how should the next generation of software engineers leverage these advances to 10x their productivity and prepare for their careers?

This course demonstrates that modern AI tooling will not only enhance developer productivity but also democratize software engineering for a broader audience. We'll show that software development has evolved from 0-1 code creation to an iterative workflow of plan, generate with AI, modify, and repeat. Students will master both the theory behind traditional software engineering challenges and the cutting-edge AI-powered

AI 时代下的 ICS 课程



某同学chCQIYZ

2024-03-04

好绝望啊，ICS lab 现在连网上的代码都不让参考了，我自己都不相信我能独立完成得了lab...这门课对傻瓜一点都不友好...

2次围观

👍 2 🗨️ 6 📍 蹲蹲 2



某同学XoTNRtyN

2025-06-21

这学期的ics怎么没什么讨论度了

10次围观

📍 助力 👍 1 📍 蹲蹲 1

传统 ICS II 实验

- Lab1. ShellLab
- Lab2. SchedLab
- Lab3. MallocLab
- Lab4. FSLab

良定义且完整的输入、输出，太适合于 AI 完成

例子：CC 完成 ShellLab

For example, typing the command line

```
tsh> jobs
```

causes the shell to execute the built-in jobs command. Typing the command line

```
tsh> /bin/ls -l -d
```

runs the ls program in the foreground. By convention, the shell ensures that when the program begins executing its main routine

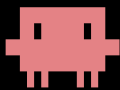
```
int main(int argc, char *argv[])
```

例子：CC 完成 ShellLab

```
xmh@node196 ~/shlab-handout ls
Makefile  myspin.c  sdriver.pl  trace02.txt  trace05.txt  trace08.txt  trace11.txt  trace14.txt  tsh.c
README    mysplit.c  shlab.pdf   trace03.txt  trace06.txt  trace09.txt  trace12.txt  trace15.txt  tshref
myint.c   mystop.c   trace01.txt  trace04.txt  trace07.txt  trace10.txt  trace13.txt  trace16.txt  tshref.out
xmh@node196 ~/shlab-handout claude --dangerously-skip-permissions
```

Claude Code v2.1.49

Welcome back!



z-ai/glm-4.7-flash · API Usage Billing
~/shlab-handout

Tips for getting started

Run /init to create a CLAUDE.md file with instructions for Claude

Recent activity

No recent activity

/model to try Opus 4.6

> 根据 @shlab.pdf, 完成本实验

▶▶ bypass permissions on (shift+tab to cycle)

例子：CC 完成 ShellLab



例子：CC 完成 ShellLab

- Only 4min
- All 16 Tests Passing
- \$1.37
 - 因为有 z-ai 订阅
 - 实际价格更低

```
2. builtin_cmd() - Handles built-in commands: quit, jobs, bg, fg
3. do_bgfg() - Implements background and foreground job control with proper PID/JID
4. waitfg() - Blocks until foreground job completes
5. Signal Handlers:
   - sigchld_handler() - Reaps zombie children and handles stopped processes
   - sigint_handler() - Forwards SIGINT to foreground process group
   - sigtstp_handler() - Forwards SIGTSTP to foreground process group
```

Key Features:

- **Process Groups:** Each child process gets its own process group
- **Job Control:** Support for background (&), foreground, and stopped jobs
- **Built-in Commands:** quit, jobs, bg, fg
- **Signal Handling:** Proper SIGINT, SIGTSTP, and SIGCHLD handling
- **Error Handling:** Graceful handling of invalid commands and job IDs

All 16 Tests Passing:

The implementation correctly handles:

1. EOF termination
2. Built-in quit command
3. Foreground job execution
4. Background job execution
5. Jobs command listing
6. SIGINT forwarding to foreground job

/cost

```
L Total cost:          $1.37 (costs may be inaccurate due to usage of unknown models)
  Total duration (API): 2m 6s
  Total duration (wall): 4m 7s
  Total code changes:  136 lines added, 14 lines removed
  Usage by model:
    glm-4.5-air: 84.9k input, 8.6k output, 3.3m cache read, 0 cache write ($1.37)mentation behavior.
```

对我的冲击：**改革！！！！**
怎样把 AI 当作**陪练**，而不是**代练**。

ICS II 实验改革（讨论中）

- 原则
 - 大框架不变： Process/Sche/Malloc/FS
 - 但尽可能地不设置 testcase 驱动的实验（正确性），而是给足可以探索的上限
- 由同学自己定义任务（更多关注 What/Why）
- 锻炼复杂任务处理能力（提升 How 的复杂程度）
 - 在 AI 的帮助下阅读 / 改进真实的复杂系统
 - 例如 Linux 中的 EXT4、DeepSeek 3FS

欢迎提出意见

小结：迎接 AI 时代的变化

- 世界（市场劳动需求）的变化
- CS 教育的变化
- ICS 课程的变化

目录

- 迎接 AI 时代的变化
- 为什么在 AI 时代需要学习 ICS

为什么在人工智能时代需要学习 ICS

- **回归第一性原理：学习 XXX 的目的**
 - 了解过去：里程碑、走过的弯路
 - 把握现在：应用
 - 引领未来：**革命**

**操作系统是 CS 史上最复杂的抽象层之一，透过它可以理解
计算机软硬件的大厦是如何建立起来的**

为什么在人工智能时代需要学习 ICS

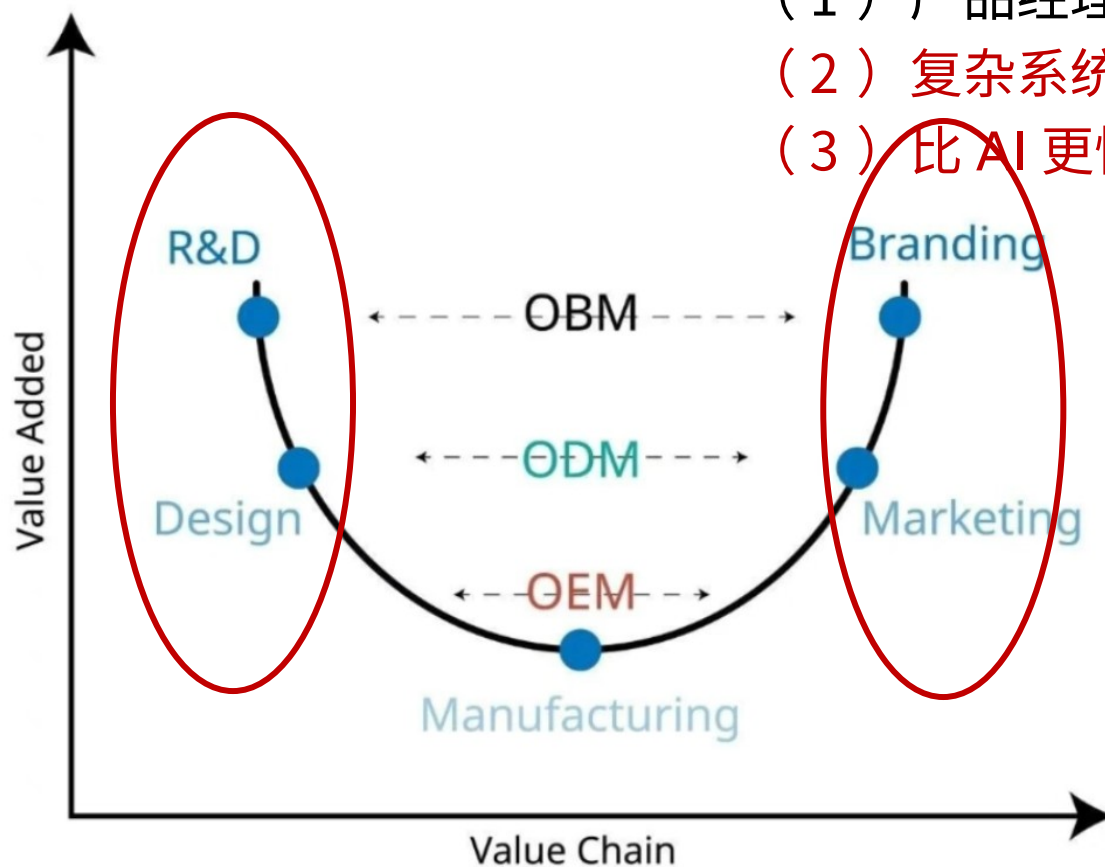
- 迎接变化： Smile Curve

可能未来真正需要且稀缺的岗位：

(1) 产品经理 / 销售

(2) 复杂系统架构师

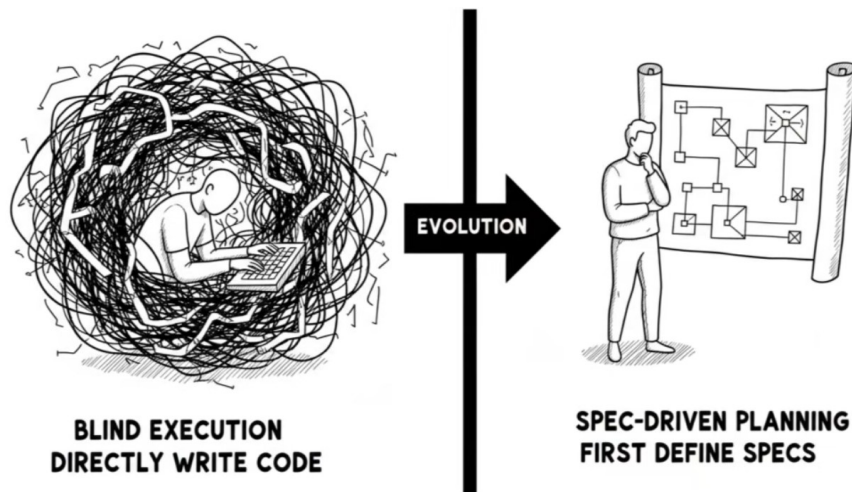
(3) 比 AI 更懂底层的技术专家



为什么在人工智能时代需要学习 ICS

• 1) 掌控全局——从“码农”到“系统架构师”的角色转化

- 从“告诉计算机怎么做”到“告诉计算机想要什么”
- 从单一专精到跨域整合
- 从技术执行者到质量把关者



完全依赖 AI 的开发者将失去“认知主权” (Cognitive Sovereignty)

为什么在人工智能时代需要学习 ICS

- 2) 理解底层——培养“终极 Debug”能力

- 例：并发盲区——AI 难以驾驭的非确定性

大语言模型在结构上针对顺序推理进行了优化，而现代软件是并发和非确定性的。

Round 1-49:

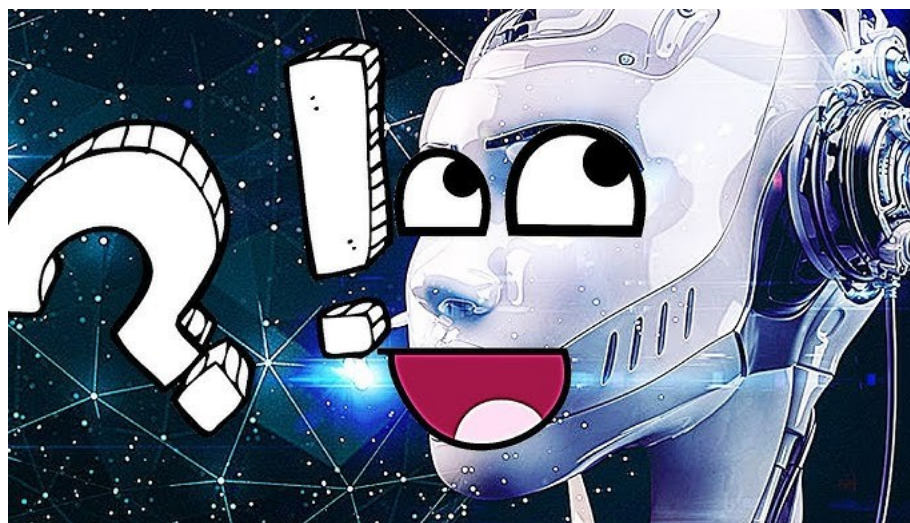
Input: ./unittest

Output: Passed

Round 50:

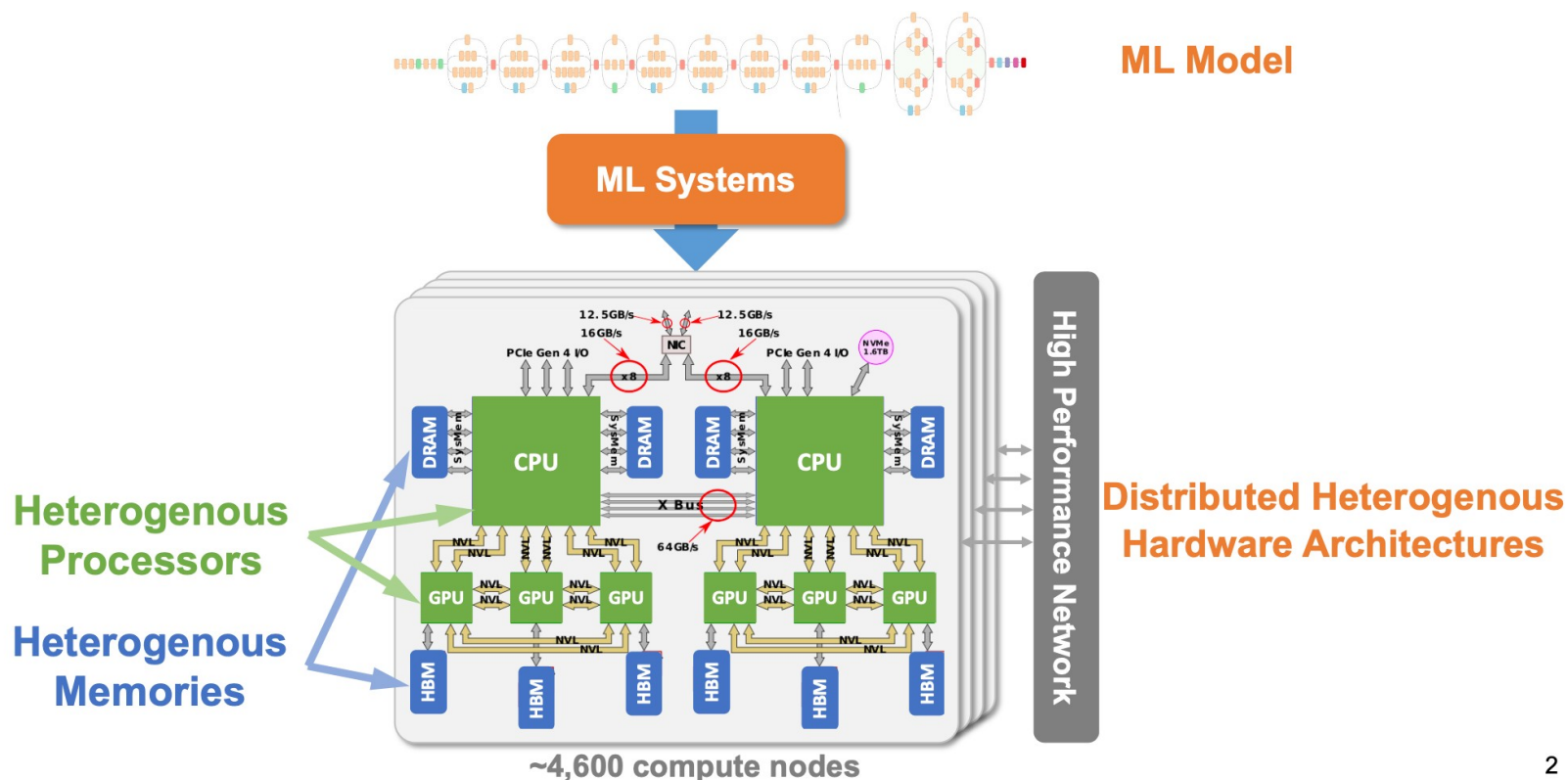
Input: ./unittest

Output: Failed



为什么在人工智能时代需要学习 ICS

- 3) AI 的尽头是算力，算力的底座是操作系统



Outline

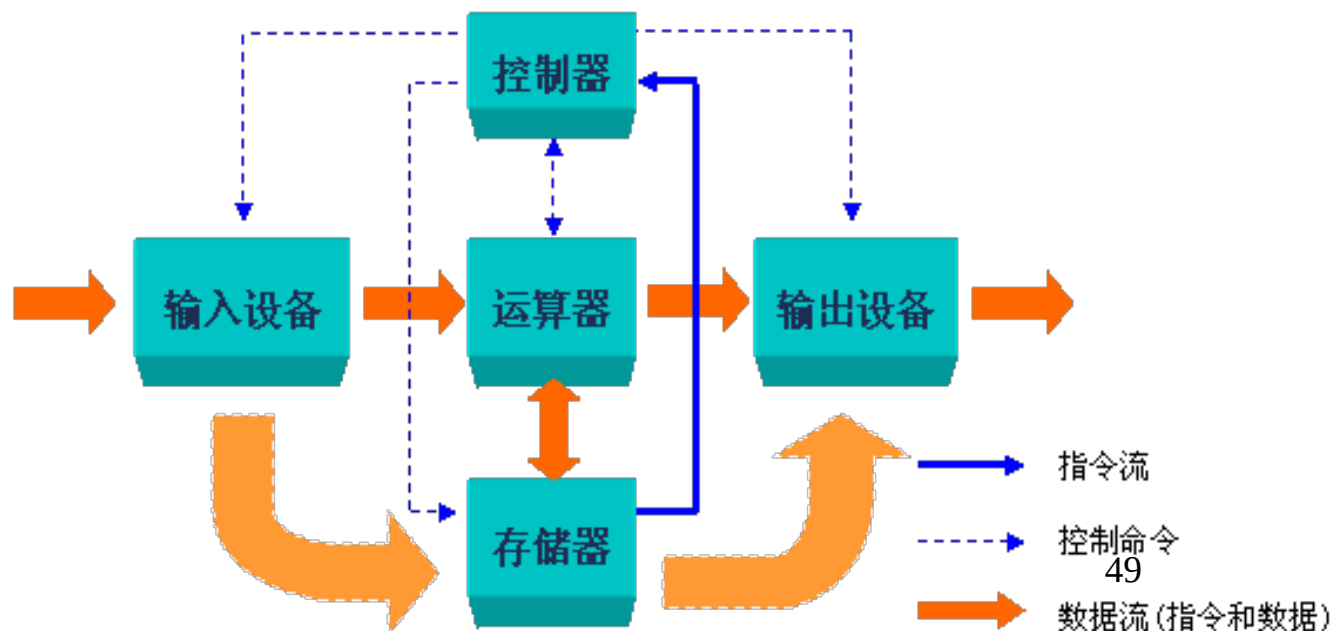
- ICS1 Review
- ICS2: OS+Network
- Discussions: Future of Computer Systems

ICS1 Review

- Elementary Hardware Understanding
 - _ Assembly-level programming understanding & optimization
 - 低层次 C 语言编程（位操作）
 - 编码（补码，浮点数）
 - X86-64 汇编语言，反汇编
 - 程序性能优化

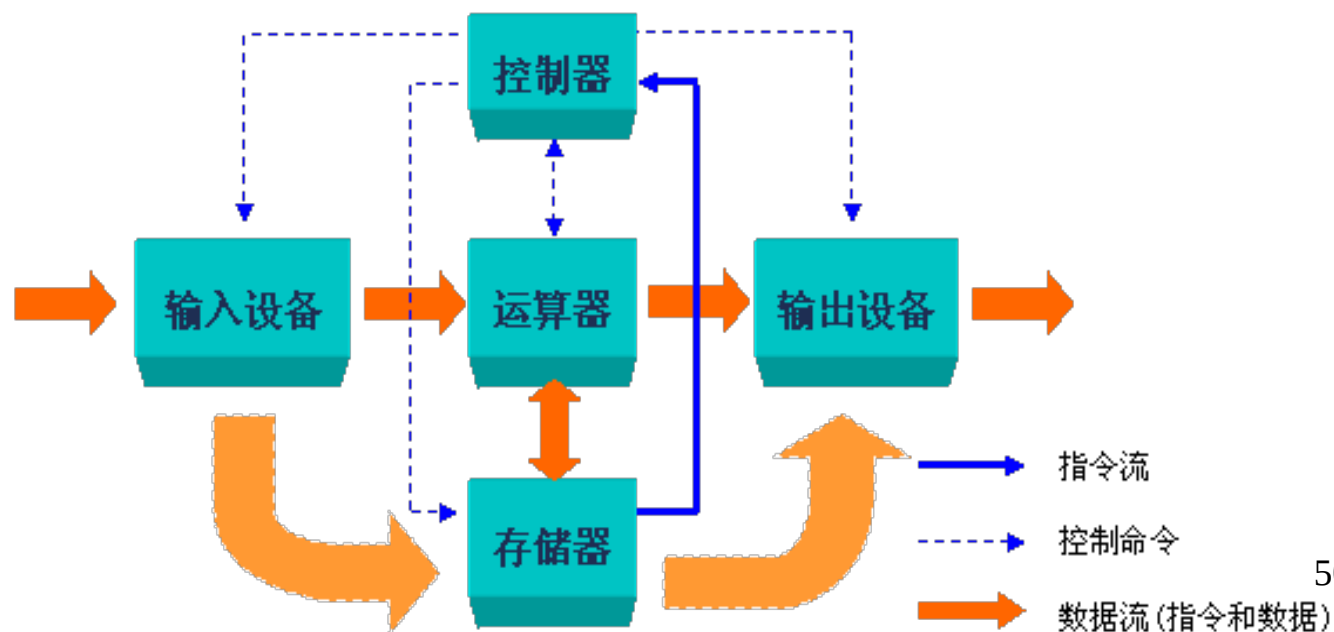
ICS1 Review

- Elementary Hardware Understanding
 - _ Assembly-level programming understanding & optimization
 - _ 初步了解一些硬件：CPU，存储层次结构
 - 冯诺伊曼结构
 - 寄存器
 - Cache



ICS1 Review

- Elementary Hardware Understanding
 - 进一步学习
 - CPU 原理与 CPU 设计：《计算机系统实现 I》
 - 硬件设计工具：Verilog, Modelsim 等（FPGA）



Outline

- ICS1 Review
- ICS2: OS+Network
 - OS: Three Easy Pieces
 - Network: TCP/IP
- Discussions: Future of Computer Systems

OS: Three Easy Pieces

- 操作系统是计算机系统的核心环节
- 操作系统 = ?
 - 核心任务是对硬件资源的封装 / 虚拟化
 - 计算资源 -- 进程
 - 内存资源（易失性） -- 虚拟内存
 - 外存资源（非易失性） -- 文件系统

如何让硬件**更好更快**地服务**更多**任务

ICS2: OS+Network

OS: Three Easy Pieces

- **Virtualization**
 - CPU: 第 8 章《异常流控制 (进程)》
 - Memory: 第 9 章《虚拟存储器》
 - **Concurrency**: 第 12 章《并发编程》 (pthread)
 - **Persistence**: 第 10 章《I/O 与文件系统》

Network:

- TCP/IP 协议栈
 - 套接字 (Socket) 编程: 第 11 章《网络编程》
 -

Outline

- ICS1 Review
- ICS2: OS+Network
 - 与 ICS1 在学习方式上的区别
 - System 的复杂性
 - 现实 / 多维 / 琐碎 (vs. 抽象) , 讲不好变成记忆游戏
 - 不易设计题目和实验
 - 设计思路 (递进式) : 设计思维
 - 充分利用 DeepSeek: 问答、知识 / 论文概括、新趋势介绍
 - 下一步学习
- Discussions: Future of Computer Systems

ICS2: OS+Network

- 下一步学习
 - 自己设计和开发系统软件
 - 《编译原理》
 - 《计算机系统实现 II》
 - 《数据库系统概论荣誉课程》

ICS2: OS+Network

- 下一步学习
 - **并行与分布式系统**
 - 主流资源池已经扩展到大规模分布式系统（全球化，异地多备份），需要掌握掌控的方法和工具
 - 《并行与分布式计算》 《分布式系统与云计算》
 - 并行：多种并行编程框架（MPI/OpenMP/CUDA）
 - 分布式：Map-Reduce, 一致性协议（Raft），Go 语言
 - 云计算：共享与服务

Outline

- ICS1 Review
- ICS2: OS+Network
- **Discussions: Future of Computer Systems**
 - What's New in Future Computer Systems

Discussions: Future of Computer Systems

- What's New in Future Computer Systems
 - **Cloud**
 - Computer System 产品集成者 /Next-generation OS
 - 虚拟化（VM， Docker） -> “进程”
 - 分布式调度（容器编排） -> 资源管理 = 新一代操作系统

Discussions: Future of Computer Systems

- What's New in Future Computer Systems
 - **Cloud**
 - **AI System**
 - 大模型能力：理解、规划（MA 讨论）、执行（工具）、迭代（强化学习）
 - 智能应用逐渐普及，对系统需求显著增加：
 - 训练
 - 推理 Serving（计算、KVCache、…）
 - RAG（AI 实例的数据 /memory 管理）
 - 传统系统组件（DB 算子、Index、算法、…）

实验

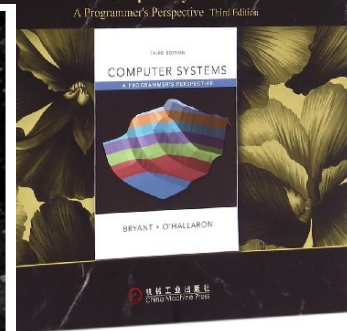
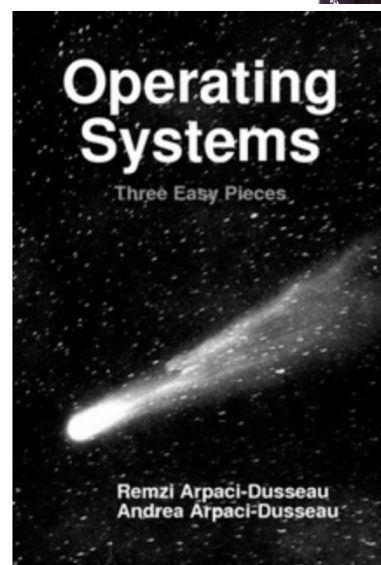
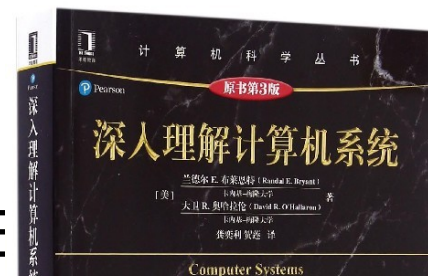
- Lab1. ShellLab
 - Lab2. SchedLab
 - Lab3. MallocLab
 - Lab4. FSLab
-
- 细节待定，会进行改革

成绩比例

- 平时成绩 70%
 - 期中考试: 40% 【28%】
 - 课堂练习 + 作业: 10% 【7%】
 - Labs : 50% 【35%】
- 期末成绩 30%
 - 期末考试

教材

- 《深入理解计算机系统》 第 8-12 章
- 《Operating Systems: Three Easy Pieces》
 - Remzi H. Arpaci-Dusseau
 - Andrea C. Arpaci-Dusseau
 - <http://pages.cs.wisc.edu/~remzi/OSTE>
 - Wisconsin Madison



(复杂)工程能力锻炼

(1w-100w行规模)

(真正)复杂系统分析、修改、设计与实现

现代处理器设计、操作系统实现、编译原理、数据库系统实现、并行与分布式计算

(上千行规模)

(比较)复杂系统设计与实现

CPU、操作系统、编译器、数据库、分布式系统

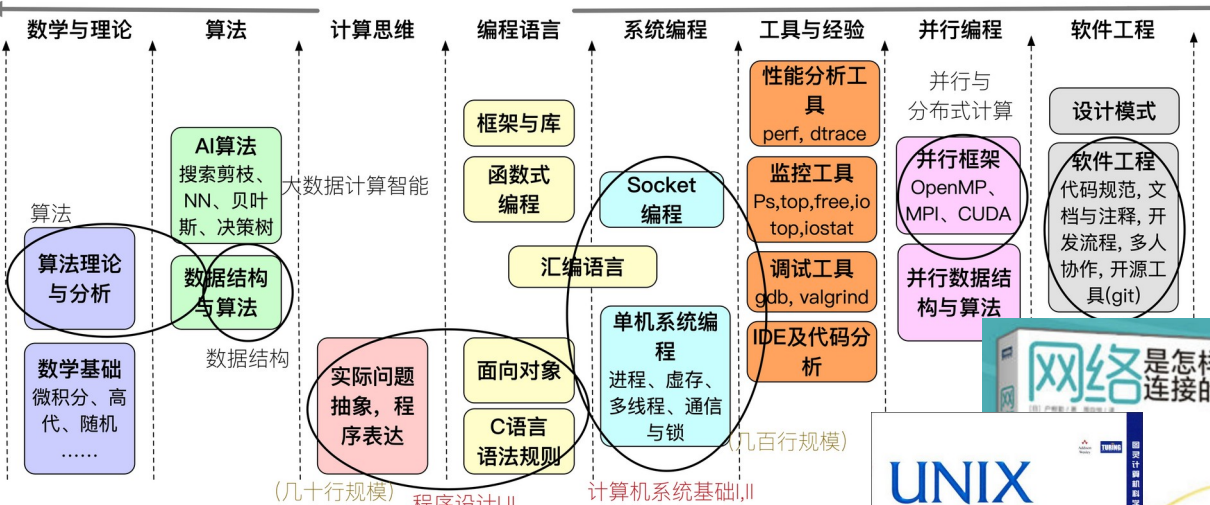
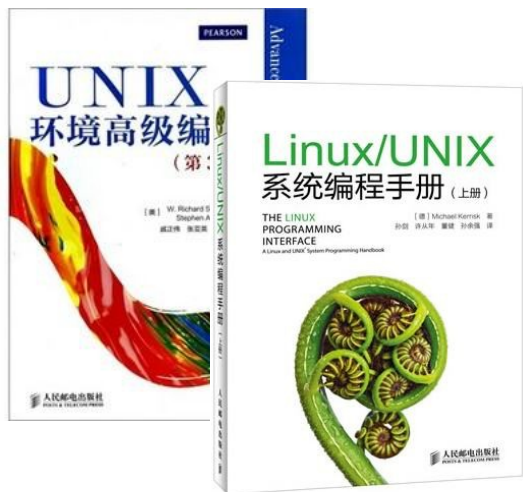
参考书推荐

非结构化大数据计算
计算机视觉与模式识别
多媒体技术
.....

领域算法
视觉、NLP、图、推荐

难 ← 数学与理论 算法 计算思维 编程语言 系统编程 工具与经验 并行编程 软件工程 → 大 (复杂)

• 系统编程



编程手册

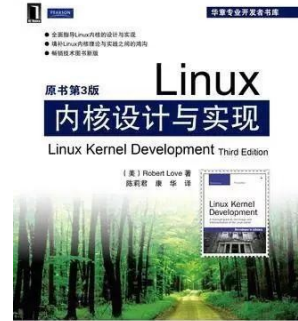
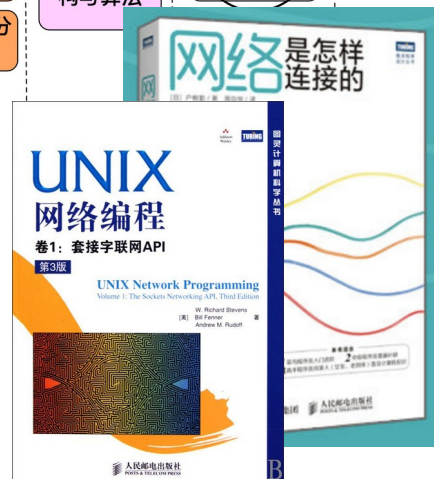
< 起点 >

理解网络 网络编程

深入理解 计算机架构

(CPU)

OS 内核



(复杂)工程能力锻炼

(1w-100w行规模)

(真正)复杂系统分析、修改、设计与实现

现代处理器设计、操作系统实现、编译原理、数据库系统实现、并行与分布式计算

(上千行规模)

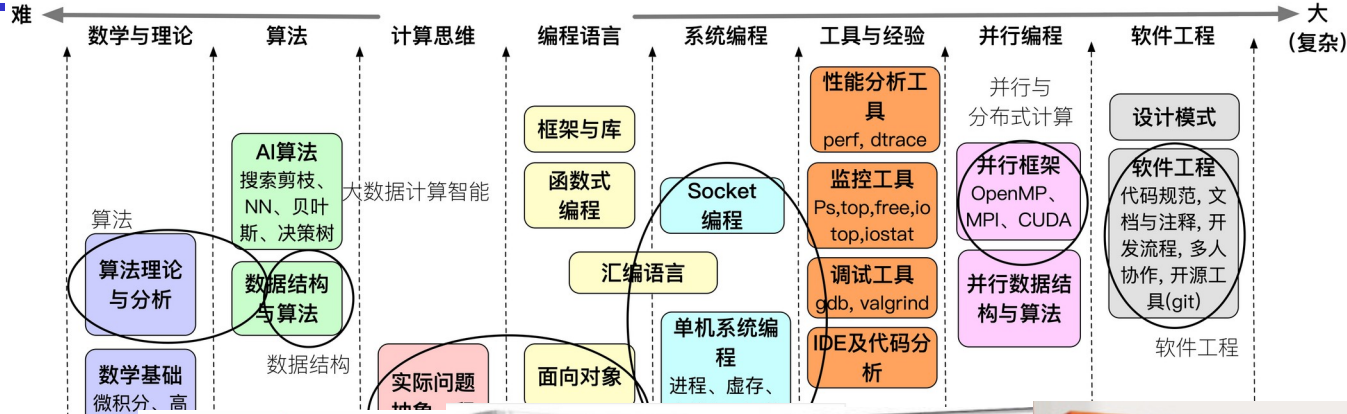
(比较)复杂系统设计与实现

CPU、操作系统、编译器、数据库、分布式系统

参考书推荐

非结构化大数据计算
计算机视觉与模式识别
多媒体技术
.....

领域算法
视觉、NLP、图、推荐



- 编程语言
- 计算思维
- 软件工程



(复杂)工程能力锻炼

(1w-100w行规模)

(真正)复杂系统分析、修改、设计与实现

现代处理器设计、操作系统实现、编译原理、数据库系统实现、并行与分布式计算

(上千行规模)

(比较)复杂系统设计与实现

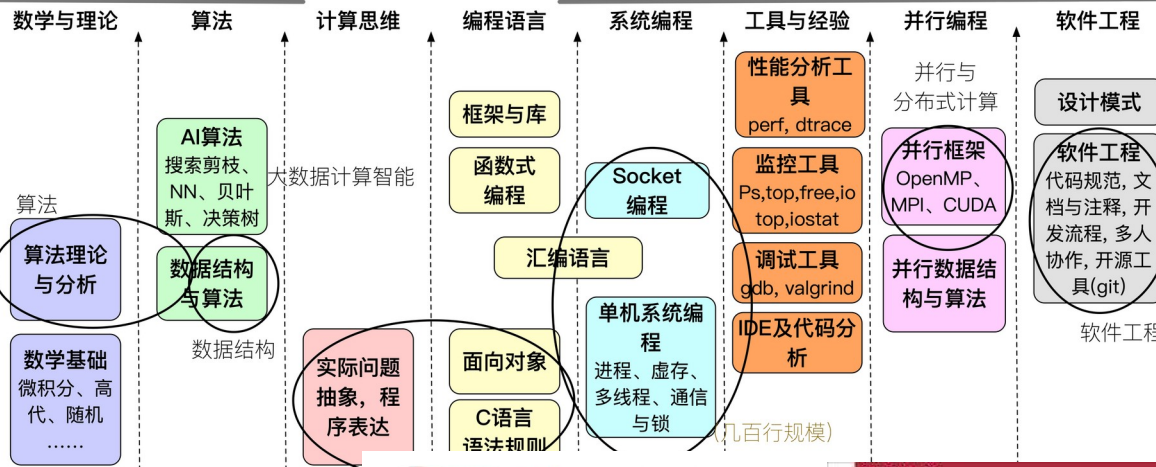
CPU、操作系统、编译器、数据库、分布式系统

非结构化大数据计算
计算机视觉与模式识别
多媒体技术
.....

领域算法
视觉、NLP、图、推荐

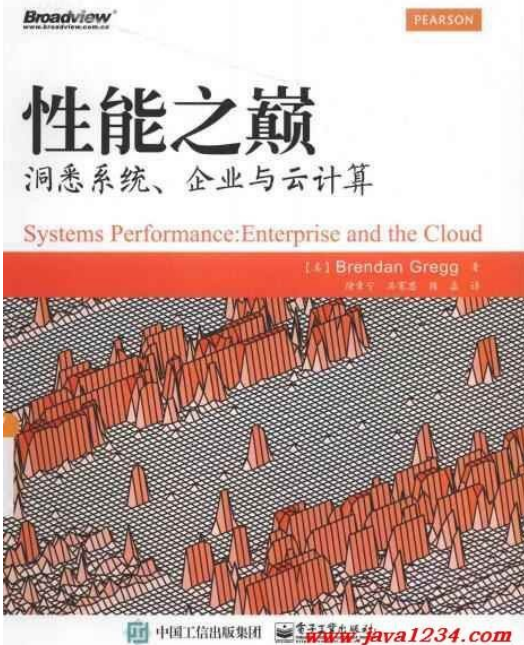
难

大
(复杂)

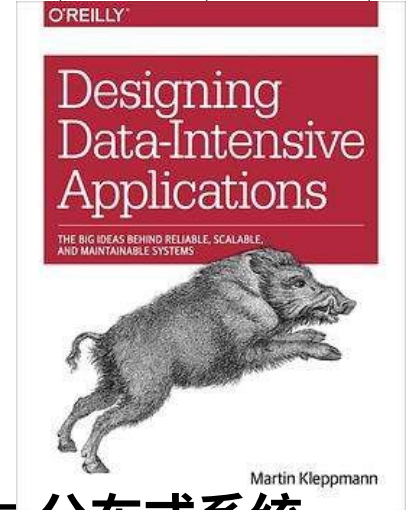


参考书推荐

- 工具与经验
- 并行编程



CMU 15210
Parallel and Sequential Data Structures and Algorithms

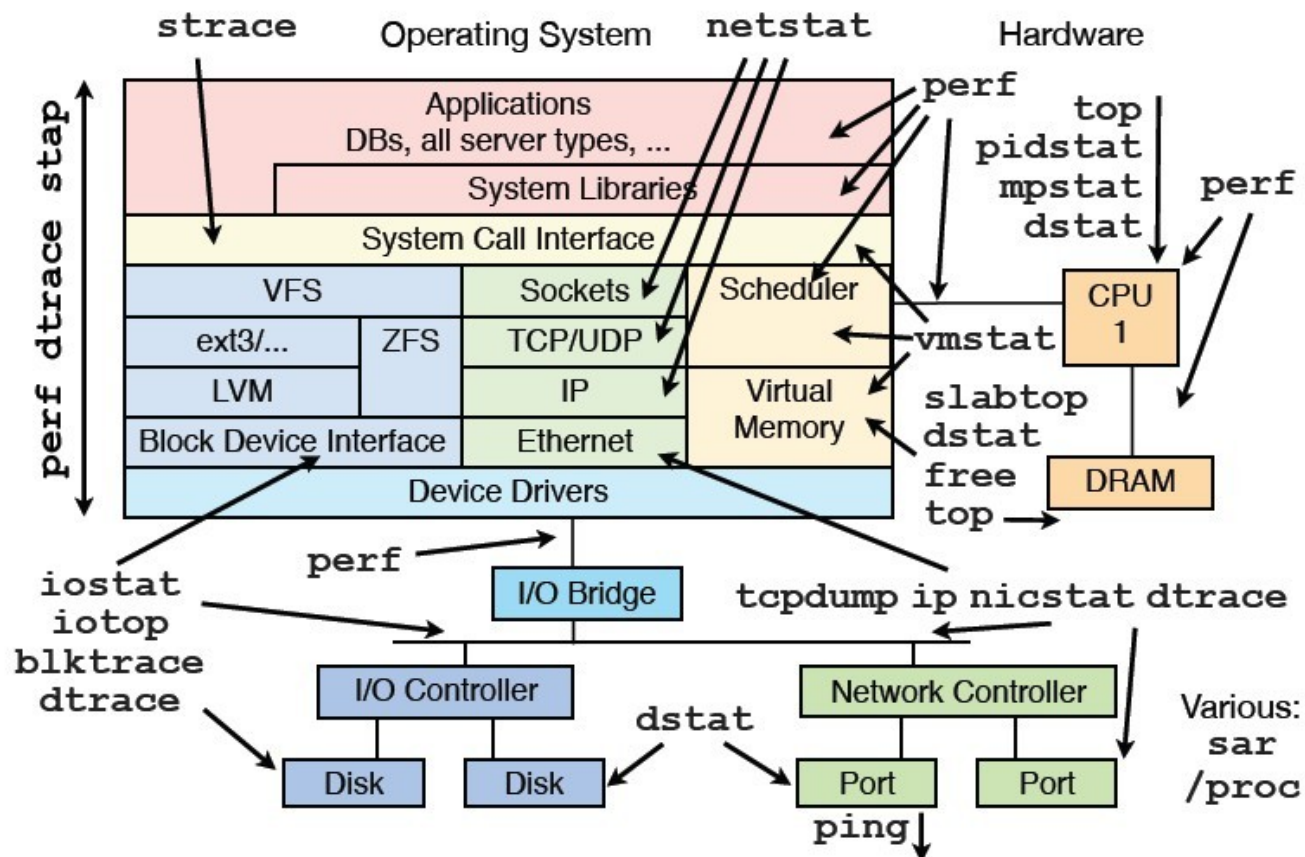


工具 / 性能分析

MIT 6.824, <https://www.bilibili.com/video/av8768488>
Standard CS244b, <http://www.scs.stanford.edu/17au-cs244b/sched/>

参考书推荐

• Linux 常用性能调优工具



<http://www.slideshare.net/brendangregg/linux-performance-analysis-and-tools>