大模型推理系统与实践

第 14 讲课程总结与最终汇报

张书豪

华中科技大学计算机学院

2027 · 研究生课程课件

教学目标

核心结论

本讲收束课程主线，并通过最终项目汇报统一概念、机制、代码与实验分析。

- 回顾全课程的核心方法论。
- 总结项目中最常见的成功与失败模式。
- 讨论如何把课程项目延伸为长期研究。

回顾课程主线

核心结论

整门课围绕五条主线展开：工作负载、调度、状态、执行路径和验证方法。

- 工作负载决定问题边界。
- 调度定义系统控制点。
- 状态对象决定容量与复用空间。
- 执行路径决定收益如何兑现。
- 验证方法决定结论是否可信。

回顾课程的统一分析框架

核心结论

这门课反复强调的框架只有一套：状态对象、控制点、收益链路与验证方法。

- 先识别系统里真正长期存在的对象。
- 再识别谁在改变系统行为。
- 再追踪收益如何从局部传递到端到端。
- 最后用实验判断结论是否成立。

课程能力目标

核心结论

课程的最终目标是建立系统研究所需的稳定分析习惯与工作方法。

- 先识别状态对象。
- 再识别控制点。
- 进而分析影响链路。
- 最后用实验验证结论。

项目汇报评价标准

核心结论

最终汇报的评价重点在于问题定义是否清楚、结果是否可信。

- 是否说明了系统对象和控制点。
- 是否报告了主要指标与关键探针。
- 是否说明了失败场景和机制边界。

项目常见成功模式

核心结论

课程项目中表现较好的案例通常具有清晰的问题定义与简洁的验证路径。

- 题目聚焦，有明确最小场景。
- 机制简单，但能解释清楚影响链路。
- 实验设计能明确支持或否定当前假设。

项目常见失败模式

核心结论

项目失败通常来自问题范围过大、验证边界不清，或者把实现工作误当成研究结论。

- 题目太大，无法在一个学期内完成。
- 只做了实现，没有形成可解释结果。
- 只有局部数字，没有端到端结论。

从课程项目到长期研究

核心结论

一个好的课程项目，应该至少能够沿着三条方向继续推进。

- 扩大 workload 与实验边界。
- 进入更真实的代码与系统环境。
- 从原型结果提炼成更明确的问题假设和论文叙事。

讲解：课程总结课真正要收束的是什么

核心结论

最后一讲不只是把前面内容重复一遍，而是要让学生看到整门课其实一直在重复同一套分析框架。

- 工作负载告诉我们问题边界在哪里。
- 控制点告诉我们系统行为由谁改变。
- 状态对象告诉我们容量、复用和稳定性为何会成为主矛盾。
- 验证方法告诉我们哪些结论值得相信。

讲解：讲义、tutorial、实验和代码如何形成真正闭环

核心结论

你前面强调四层材料要互相呼应，最后一讲就应该把这件事明确讲出来。

- slides 负责提出问题和建立概念框架。
- tutorials 负责把概念映射到代码对象和思考路径。
- experiments 负责把问题变成可执行、可复查的证据收集过程。
- code 负责提供真实控制点，让讨论不漂浮在抽象层面。

例子：这门课最后到底希望学生带走什么（1）

核心结论

课程总结不应只回顾目录，而应明确学生已经获得的一套可重复的方法。

不是记住若干论文名字

论文只是样本，真正带走的应是分析系统问题的框架。

例子：这门课最后到底希望学生带走什么（2）

核心结论

课程总结不应只回顾目录，而应明确学生已经获得的一套可重复的方法。

也不是掌握一套固定 runtime

runtime 会变，但对象、控制点、收益链路和验证方法这套框架可以迁移。

例子：这门课最后到底希望学生带走什么（3）

核心结论

课程总结不应只回顾目录，而应明确学生已经获得的一套可重复的方法。

课程结论

这门课真正的产出，是一套进入真实推理系统后还能继续使用的工作方法。

例子：一个优秀最终汇报通常长什么样（1）

核心结论

好的最终汇报通常不追求面面俱到，而是把问题和证据压到很实。

开场

先用一句话讲清系统对象、控制点和目标指标。

例子：一个优秀最终汇报通常长什么样（2）

核心结论

好的最终汇报通常不追求面面俱到，而是把问题和证据压到很实。

主体

用最小实现和最关键实验支撑核心因果链，而不是堆大量细碎结果。

例子：一个优秀最终汇报通常长什么样（3）

核心结论

好的最终汇报通常不追求面面俱到，而是把问题和证据压到很实。

结尾

诚实说明失败条件、边界和后续路线，往往比强行夸大收益更有研究价值。

例子：一个看似很努力但不合格的汇报（1）

核心结论

课程总结课也要明确失败模式，这样学生才知道之后如何避免。

问题一

讲了很多实现细节，却没有说明到底改变了哪个控制点。

例子：一个看似很努力但不合格的汇报（2）

核心结论

课程总结课也要明确失败模式，这样学生才知道之后如何避免。

问题二

图很多，但 workload、对照和失败边界都不清楚。

例子：一个看似很努力但不合格的汇报（3）

核心结论

课程总结课也要明确失败模式，这样学生才知道之后如何避免。

问题三

把工程量直接当成研究贡献，导致整场报告没有中心判断。

代码例子：最终汇报推荐结构（1）

核心结论

课程最后一讲可以给出一份较硬的汇报骨架，帮助学生把材料组织成研究叙事。

结构模板

1. Problem and system object
2. Changed control point / mechanism
3. Minimal implementation path
4. Evidence, boundary, next step

代码例子：最终汇报推荐结构（2）

核心结论

课程最后一讲可以给出一份较硬的汇报骨架，帮助学生把材料组织成研究叙事。

课堂应追问

为什么这个结构里没有单独的“工程量展示”章节？

代码例子：最终汇报推荐结构（3）

核心结论

课程最后一讲可以给出一份较硬的汇报骨架，帮助学生把材料组织成研究叙事。

课程结论

汇报的中心是研究判断，而不是开发过程流水账。

代码例子：课程项目资料如何互相对齐（1）

核心结论

你前面要求的“讲义、tutorial、实验和代码互相呼应”，在最后汇报里必须显式体现。

对齐关系

```
slides -> problem framing
tutorials -> concept and code map
experiments -> evidence and procedure
code -> control point and implementation
```

代码例子：课程项目资料如何互相对齐（2）

核心结论

你前面要求的“讲义、tutorial、实验和代码互相呼应”，在最后汇报里必须显式体现。

课堂应追问

如果你的最终报告无法指出这四层材料各自承担什么角色，说明前期材料组织仍然是断裂的。

代码例子：课程项目资料如何互相对齐（3）

核心结论

你前面要求的“讲义、tutorial、实验和代码互相呼应”，在最后汇报里必须显式体现。

教学意义

课程闭环不是文档美观问题，而是研究训练是否真正闭合的问题。

例子：从课程项目继续做研究的三条路线（1）

核心结论

课程结束后，真正有潜力的项目通常沿着问题深化、系统加深和证据扩展三条路线继续走。

路线一：问题深化

把一个课程级现象收紧成更明确的系统假设。

例子：从课程项目继续做研究的三条路线（2）

核心结论

课程结束后，真正有潜力的项目通常沿着问题深化、系统加深和证据扩展三条路线继续走。

路线二：系统加深

从教学代码进入真实 runtime 或异构平台，验证机制是否仍成立。

例子：从课程项目继续做研究的三条路线（3）

核心结论

课程结束后，真正有潜力的项目通常沿着问题深化、系统加深和证据扩展三条路线继续走。

路线三：证据扩展

把最小实验扩大到更真实 workload、更严格对照和更强失败分析。

课堂练习：给未来的自己留一张研究路线卡（1）

核心结论

课程最后的练习，不是再做一题，而是把当前项目转成可继续推进的路线图。

练习一

写出你项目里最值得继续研究的一条假设。

课堂练习：给未来的自己留一张研究路线卡（2）

核心结论

课程最后的练习，不是再做一题，而是把当前项目转成可继续推进的路线图。

练习二

写出下一步最值得补的一个实验和一个代码入口。

课堂练习：给未来的自己留一张研究路线卡（3）

核心结论

课程最后的练习，不是再做一题，而是把当前项目转成可继续推进的路线图。

练习三

用一句话回答：这门课之后，你进入一个新推理系统时会先看什么？

课程结束语

核心结论

大模型推理基础设施研究的核心，在于使系统在复杂负载、持续状态与异构约束下稳定运行并持续兑现性能收益。

- 这是研究问题，也是工程问题。
- 希望大家离开这门课后，已经具备进入真实系统继续做研究的能力。

本讲参考资料

- **课程完整材料与后续更新**（课程仓库） 原文链接
- **SpecGen**（MLSys 2026） 原文链接
- **GhostServe**（MLSys 2026） 原文链接

课件中的论文数字均指作者在特定模型、硬件和工作负载下报告的结果，不代表无条件性能。

对应 Tutorial

核心结论

本讲对应 Tutorial 13。先独立作答，再使用对应 Tutorial 的教师讨论要点核对。

- 文件：`build/tutorials/Tutorial_13_ 课程总结与汇报.pdf`
- 动手运行、日志记录与提交要求统一查看 `build/experiments/` 和对应 experiment 源目录下的 `assignment_spring-2027/`。