AI 时代程序分析与验证类课程的前沿化、实践化与过程化重构*

文成 秦胜潮**

西安电子科技大学广州研究院，广州 510555

刘杜钢 许智武

深圳大学计算机与软件学院，深圳 518060

摘 要 面向 AI 时代软件开发方式变化，本文针对传统程序分析与验证类课程前沿内容融入不足、实践训练薄弱、评价方式偏重结果等问题，以《程序分析与验证》课程为例，围绕前沿化、实践化、过程化开展课程重构。课程在经典知识基础上融入 AI 赋能程序分析、AI 赋能形式化验证、AI 生成代码质量分析等前沿技术，设计基础实验、专题综述、原型工具开发、前沿论文研读等实践任务，并构建由课堂练习、基础实验、综合大作业、课堂报告与反馈组成的过程化评价机制，为 AI 时代程序分析与验证类课程的内容更新、实践组织和评价优化提供参考。

关键字 程序分析与验证，人工智能，课程重构，可信软件

Frontier-Oriented, Practice-Oriented, and Process-Oriented Redesign of Program Analysis and Verification Courses in the AI Era

Cheng Wen Shengchao Qin

Guangzhou Institute of Technology
Xidian University,
Guangzhou 510555, China;
wencheng@xidian.edu.cn qinshengchao@xidian.edu.cn

Dugang Liu Zhiwu Xu

College of Computer Science and Software Engineering
Shenzhen University
Shenzhen 518060, China
liudugang@szu.edu.cn xuzhiwu@szu.edu.cn

Abstract—In the context of changing software development practices in the AI era, traditional program analysis and verification courses face several problems, such as insufficient integration of frontier topics, limited practical training, and assessment methods that overly emphasize final results. Taking the course Program Analysis and Verification as an example, this paper explores course redesign from three dimensions: frontier orientation, practice orientation, and process orientation. Based on classical knowledge, the course incorporates frontier topics such as AI-empowered program analysis, AI-empowered formal verification, and quality analysis of AI-generated code. It designs practical tasks including foundational experiments, topic-based surveys, prototype tool development, and in-depth analysis of frontier research papers. It also establishes a process-oriented assessment mechanism consisting of classroom exercises, foundational experiments, comprehensive course projects, and classroom presentations with feedback. This work provides a reference for content updating, practice organization, and assessment optimization of program analysis and verification courses in the AI era.

Keywords—Program analysis and verification, artificial intelligence, course redesign, trustworthy software

1 引 言

人工智能技术正在深刻改变软件开发方式。以大语言模型为代表的智能编程工具已逐渐应用于代码生成、程序理解、测试用例生成、缺陷定位和代码修复等环节[1]，在提升开发效率的同时，也带来了代码正确性、安全性和可信性方面的新问题。AI 生成代码可能存在语义错误、边界条件遗漏、安全漏洞和可解释性不足等风险，这要求学生不仅能够借助智能化工具开展软件分析，也能够运用程序分析与验证方法评估和改进代码质量[2]。

程序分析与验证类课程主要围绕语法语义、控制流分析、数据流分析、动态分析、模糊测试、类型系统和霍尔逻辑等内容展开，旨在培养学生理解程序行为、发现软件缺陷、分析程序性质和保障软件可信性的能力。传统教学通常偏重概念讲授、算法推导和书面考核，学生在真实代码分析、工具原型开发、实验复现和实际问题理解等方面训练不足。同时，AI 赋能程序分析、AI 辅助形式化验证、智能缺陷检测、

*基金资助：本文得到国家自然科学基金（62302375, 62472339, 62372304）资助

**通讯作者：秦胜潮 qinshengchao@xidian.edu.cn。

自动规约生成和不变量推导等方向快速发展[3-4]，使课程内容亟需与前沿进展相衔接。

基于此，本文以《程序分析与验证》课程为例，围绕前沿化、实践化、过程化开展课程重构探索。在内容上，课程在保留经典知识体系的基础上融入 AI 时代前沿专题；在实践上，设计基础实验、专题综述、原型工具开发和前沿论文研读等任务；在评价上，构建由课堂练习、基础实验、综合大作业、课堂报告与反馈组成的过程化评价机制。通过上述设计，推动学生从知识理解走向实践应用和前沿思考，为 AI 时代程序分析与验证类课程的建设提供参考。

2 课程重构总体思路

2．1 课程定位与重构目标

程序分析与验证课程面向软件工程、计算机科学与技术、网络与信息安全等相关专业，服务于可信软件人才培养需求。课程内容既包括语法语义、控制流分析、数据流分析、类型系统、霍尔逻辑等经典基础知识，也涉及动态分析、模糊测试、软件缺陷检测和程序正确性验证等应用场景[5]。其教学目标不仅是使学生理解程序分析与验证的基本概念和方法，更重要的是培养学生分析代码质量问题、设计检测策略、理解工具原理和评价分析结果的能力。

AI 时代的软件开发方式对该课程提出了新的要求。一方面，AI 技术正在被用于程序理解、缺陷检测、测试生成和验证辅助等任务，推动程序分析与验证技术向智能化、人机协同方向发展；另一方面，AI 生成代码本身也需要接受质量分析、安全检测和可信性评估。因此，课程重构需要在经典知识体系基础上，进一步强化学生对 AI 前沿技术的理解、对真实代码问题的分析能力以及对分析结果可靠性的判断能力。

基于上述定位，本文将课程重构目标概括为三个方面：一是通过前沿化拓展学生对 AI 赋能程序分析与验证的认知；二是通过实践化提升学生将课程知识应用于代码分析和工具开发的能力；三是通过过程化评价促进学生在学习、实践、表达和反思中的持续改进。

2．2 前沿化、实践化、过程化重构框架

课程重构围绕前沿化、实践化和过程化三条主线展开。三者分别对应课程建设中的“教什么”“怎么练”和“如何评”，共同构成课程知识学习、能力训练与评价反馈的闭环。

前沿化强调课程内容从经典方法讲授扩展到 AI 时代前沿技术。课程在保留程序语法语义、控制流分析、数据流分析、模糊测试和霍尔逻辑等基础内容的同时，引入 AI 赋能程序分析、AI 赋能形式化验证、AI 生成代码质量分析等内容，使学生理解传统程序分析与验证方法在智能化软件开发中的价值与边界。

实践化强调课程学习从书面知识掌握转向面向真实问题的实践训练。课程通过基础实验、专题综述、原型工具开发和前沿论文研读等任务，引导学生将抽象概念转化为具体分析过程，将算法原理转化为可运行的工具或实验方案，并在实践中理解方法的适用条件和局限性。

过程化强调课程评价从偏重最终结果转向关注学习全过程。课程通过课堂练习、基础实验、综合大作业、课堂报告与相互点评等方式，评价学生在问题理解、技术调研、工具实现、结果分析、表达交流和反思改进等方面的表现，形成覆盖任务完成、展示交流和反馈改进的评价闭环。围绕上述三条主线，课程重构的总体框架如表 1 所示。

表 1 程序分析与验证课程重构总体框架

重构维度	主要做法	能力目标
前沿化	融入AI赋能程序分析、AI赋能形式化验证、AI生成代码质量分析等专题	拓展前沿视野，理解技术发展趋势
实践化	设计基础实验、专题综述、原型工具开发、前沿论文研读等任务	提升程序理解、缺陷分析、工具实现和科研阅读能力
过程化	构建课堂练习、基础实验、综合大作业、课堂报告与反馈相结合的评价机制	强化过程评价，提升表达、协作和反思能力

总体来看，该重构框架力图改变传统课程中“重理论、轻实践”“重经典、轻前沿”“重结果、轻过程”的局限，使学生在掌握程序分析与验证基础知识的同时，能够理解 AI 时代软件质量保障的新需求，并逐步形成利用程序分析与验证思想分析、解释和改进代码质量的能力。

3 前沿化课程内容重构

3．1 经典知识体系的保留与强化

程序分析与验证课程具有较强的基础性和体系性。无论软件开发方式如何变化，语法语义、控制流分析、数据流分析、类型系统、模糊测试和霍尔逻辑等内容仍然是理解程序行为、发现软件缺陷和保障软件可信性的核心基础[5]。因此，课程重构并不是削弱经典知识，而是在保留主干内容的基础上，进一步突出其在 AI 时代软件质量保障中的作用。

在课程内容组织上，经典知识体系主要分为三个层次。第一层是程序表示与结构分析，包括语法语义、抽象语法树、控制流图和数据流分析等内容，帮助学生理解程序结构、执行路径和数据依赖关系。第二层是程序行为与缺陷分析，包括动态分析、模糊测试等内容，帮助学生掌握程序运行行为观察、测试输入生成和缺陷发现的基本方法。第三层是程序安全与正确性保障，包括类型系统、霍尔逻辑等内容，帮助学生理解如何从类型安全和形式化推理角度描述与验证程序性质。

通过上述内容，学生能够建立对程序分析与验证基本问题的整体认识，为后续理解 AI 赋能程序分析、AI 辅助验证和 AI 生成代码质量分析等前沿内容奠定基础。

3．2 AI 前沿专题的融入

AI 时代的软件开发不仅改变了编程方式，也推动了程序分析与验证技术的发展。课程在经典知识基础上引入 AI 前沿专题，重点关注 AI 技术如何辅助程序理解、缺陷检测、测试生成、规约生成和形式化验证。

具体而言，课程融入的前沿专题主要包括以下四类：一是 AI 赋能程序分析，介绍大语言模型在代码理解、缺陷定位、静态分析结果解释等方面的应用[6]；二是 AI 赋能形式化验证，介绍大语言模型辅助规约生成、循环不变量推导、定理证明和验证条件构造等方向[3-4]；三是 AI 生成代码质量分析，引导学生关注 AI 生成代码中的语义错误、安全漏洞、边界条件遗漏和可维护性问题[7]；四是智能化测试与缺陷检测，介绍 AI 辅助测试用例生成、模糊测试种子优化和漏洞挖掘等内容[8]。

这些前沿专题并不作为独立课程单元孤立讲授，而是结合经典知识模块进行融入。具体而言，AI 赋能程序分析可与语法语义、控制流分析和数据流分析等内容结合，用于讨论程序理解、路径分析和缺陷解释；智能化测试可与动态分析和模糊测试结合，用于讨论测试用例生成、种子优化和漏洞挖掘；AI 生成代码质量分析可与类型系统和代码安全讨论结合；AI 赋能形式化验证则可与霍尔逻辑、规约生成和不变量推导等内容结合。通过这种模块化融入方式，课程既能体现前沿性，又能避免前沿专题与经典知识脱节。

3．3 课程知识结构重构

为适应 AI 时代程序分析与验证课程的教学需求，本文将课程内容组织为“经典知识基础层+AI 前沿拓展层”的知识结构，如图 1 所示。经典知识基础层强调程序分析与验证的基本概念、方法和理论；AI 前沿拓展层强调智能化软件分析、AI 辅助验证和 AI 生成代码质量分析；后续的实践评价环节还将通过基础实验、专题综述、原型工具开发、前沿论文研读和过程化评价，将知识学习转化为能力训练。

在上述知识结构基础上，表 2 从课程模块角度进一步给出了经典知识与 AI 前沿专题的衔接关系。该设计并非要求在每个知识点中单独增加 AI 内容，而是将前沿专题嵌入若干核心模块中，以降低教学负担并增强课程内容的整体性。通过这种方式，AI 前沿内容既能拓展学生对程序分析与验证技术发展的认识，又能反向激活其对经典知识价值的理解。

图 1 AI 时代程序分析与验证课程知识结构重构示意图

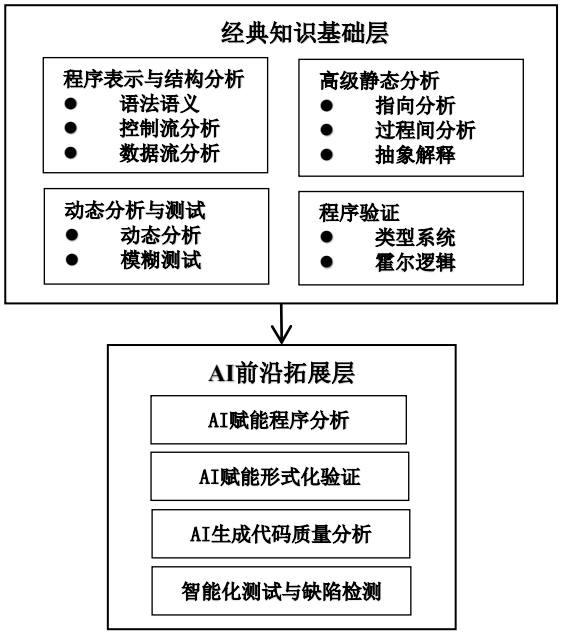


表 2 经典知识模块与 AI 前沿专题的衔接设计

课程模块	主要做法	能力培养目标
程序表示与结构分析	结合语法语义、控制流分析、数据流分析，引入AI辅助程序理解、路径分析和缺陷解释案例	培养程序理解、结构分析和缺陷识别能力
动态分析与模糊测试	引入AI辅助测试用例生成、种子优化和漏洞挖掘等案例	培养测试设计、实验复现和漏洞分析能力
类型系统与代码质量分析	结合AI生成代码中的类型错误、接口误用和安全隐患开展讨论	培养代码质量分析和安全意识
霍尔逻辑与形式化验证	引入AI辅助规约生成、循环不变量推导和验证条件构造等前沿工作	培养形式化思维和可信性分析能力

通过上述重构，课程内容从单一的经典理论讲授，转向“经典基础支撑前沿理解、前沿问题牵引实践训练”的组织方式。学生在掌握基本原理的同时，能够进一步理解AI时代程序分析与验证技术在程序理解、缺陷检测、代码质量分析和形式化验证等方面的新问题和新应用。

4 实践化课程作业设计

前沿化内容需要通过实践任务转化为学生能力。为避免课程学习停留在概念理解和书面推导层面，课程作业围绕基础理解、前沿认知和综合应用三个层次展开，形成由基础实验、专题综述、前沿论文研读、原型工具开发和课堂展示组成的实践化作业体系。其中，基础实验主要作为平时训练，缺陷检测策略设计、专题综述、前沿论文研读和原型工具开发共同构成期末综合大作业的主要内容。

4．1 基础实验与缺陷检测策略设计

基础实验主要面向程序分析与验证的经典内容展开，要求学生围绕语法分析、控制流分析、数据流分析和类型系统等模块完成小型实验任务。通过实现简单分析器或检测器，学生能够理解程序表示、控制流程、数据流值计算和类型安全检测等基本过程。

在基础实验之外，课程进一步设置缺陷检测策略设计任务。学生需要构造一个包含典型 Bug 的小程序，分析其触发条件和根因，并设计相应的检测策略。该任务不要求学生完成完整工具实现，而是强调学生能够从静态分析、动态测试或形式化验证等角度说明如何发现该缺陷。例如，针对数组越界问题，可以分析边界条件和路径约束；针对内存泄漏问题，可以分析资源分配与释放关系；针对数据竞争或死锁问题，可以分析线程交互和同步约束。该环节使学生从“实现基础算法”进一步过渡到“面向缺陷设计分析方法”。

4．2 专题综述与前沿论文研读

为引导学生理解程序分析与验证领域的发展趋势，课程设置专题综述与前沿论文研读任务。学生以小组形式选择一个细分方向开展调研，选题可包括抽象解释、符号执行、模糊测试、程序修复、并发缺陷检测、AI 赋能程序分析、AI 赋能形式化验证等。学生需要整理代表性论文，归纳方法类别和技术路线，并分析不同方法的优势、局限和适用场景。

在专题综述基础上，学生进一步选择近年高水平论文进行深入研读，重点分析论文的研究动机、现有方法不足、核心技术、实验设计、应用意义和局限性。该任务要求学生从读懂论文进一步走向评价论文，从而培养文献调研、知识组织和批判性分析能力。实际实施中，较多学生倾向选择 AI 赋能程序分析与验证相关主题，说明前沿内容能够有效激发学生对课程知识的进一步探索。

4．3 原型工具开发与课堂展示

原型工具开发是综合应用型作业，要求学生围绕缺陷检测、代码分析、测试生成或验证辅助等主题实现小型工具或实验原型。该任务不追求工业级完整度，而是强调问题定义清晰、技术路线合理、实验结果可解释。学生可以实现简单的数据流分析器、缺陷检测脚本、测试用例生成工具，也可以复现前沿论文中的关键流程。

课堂展示是原型工具开发的重要组成部分。学生需要围绕选题背景、方法设计、实现过程、实验结果和局限反思进行汇报，并接受教师和同学提问。随着 AI 编程工具的发展，学生在脚本编写、实验复现和结果可视化方面获得了更多辅助[9]，但课程评价重点并不在于代码规模，而在于学生是否理解核心方法、能否解释实验结果以及是否能够反思方法局限。

表 3 概括了实践化作业设计。三类作业由基础到综合逐步递进，分别支撑程序理解、前沿认知和综合应用能力培养。

表 3 实践化课程作业设计

作业模块	主要内容	组织方式	能力目标
基础实验与缺陷检测策略设计	完成语法分析、控制流分析、数据流分析、类型系统等基础实验；构造典型Bug并设计检测策略	个人完成	巩固基础算法，培养程序理解、缺陷建模和方法选择能力
专题综述与前沿论文研读	围绕程序分析与验证子方向开展综述，整理代表性论文，研读近年高水平论文	小组合作	培养前沿视野、文献调研、知识组织和批判性分析能力
原型工具开发与课堂展示	围绕缺陷检测、代码分析、测试生成或验证辅助等主题开发原型工具，并进行课堂报告	小组合作	提升工具实现、实验复现、结果解释和团队协作能力

通过上述设计，课程作业不再是对课堂知识的简单复述，而是将基础算法实现、真实缺陷分析、前沿文献理解和工具原型开发结合起来，推动学生在实践中形成程序分析与验证能力。

5 过程化课程评价机制

过程化评价是课程重构的重要支撑。程序分析与验证课程不仅要求学生掌握基本概念和算法原理，还要求学生能够完成实验实现、开展文献调研、解释分析结果并进行技术表达。因此，课程评价不能仅依赖期末书面考核，而应覆盖课堂学习、平时实验、期末大作业和课堂展示等多个环节[10]。基于课程目标和作业设计，本文将评价机制划分为课堂练习、基础实验、综合大作业、课堂报告与反馈四个环节，如表4所示。

表 4 过程化评价机制设计

评价环节	评价主体	评价内容	评价作用
课堂练习	教师	基础概念、课堂测试、阶段性练习	促进课堂参与，及时检查知识理解情况
基础实验	教师	算法实现、实验过程、实验报告	作为平时作业，检查理论理解与基础实践能力
综合大作业	教师	缺陷检测策略、专题综述、论文研读、原型工具	作为期末提交成果，评价综合应用、前沿理解和问题分析能力
课堂报告与反馈	教师+学生	成果展示、生生互评、师生点评、反思改进	作为期末展示环节，促进技术表达、横向比较、团队协作与持续改进

5．1 评价机制总体设计

课程评价围绕知识掌握、实践能力、前沿理解、表达交流四类能力展开。课堂练习主要用于考查学生对语法语义、控制流分析、数据流分析、类型系统、霍尔逻辑等基础内容的理解；基础实验承担平时作业功能，用于评价学生的算法实现和实验分析能力；综合大作业作为期末提交成果，重点考查学生对缺陷检测策略、专题综述、前沿论文和原型工具的综合完成情况；课堂报告与反馈则作为期末展示环节，通过成果展示、生生互评和师生点评，促进学生对实践成果的表达、比较和反思。

与传统以期末考试为主的评价方式相比，该机制更加关注学习过程和实践产出。课堂练习用于及时反馈知识理解情况，基础实验用于持续训练实践能力，综合大作业用于考查课程知识与前沿问题的综合应用，课堂报告与反馈则进一步促进学生对实践成果的表达、交流和改进。

5．2 综合大作业评价

综合大作业是期末评价的核心环节，主要对应第四章中的实践化作业设计。评价内容包括缺陷构造与检测策略、专题综述与论文研读、原型工具开发等方面。

在缺陷构造与检测策略部分，重点评价学生能否准确描述 Bug 的触发条件和根因，能否选择合适的静态分析、动态测试或验证方法，并说明该策略的优点、局限和可靠性。在专题综述与论文研读部分，重点评价学生是否能够系统整理代表性工作，归纳技术路线，理解论文的研究动机、核心方法、实验设计和局限性。在原型工具开发部分，重点评价工具目标是否清晰、实现过程是否可复现、实验结果是否可解释，以及学生是否能够对方法适用条件进行反思。

通过综合大作业评价，课程将知识理解、实践实现和前沿分析结合起来，避免评价仅停留在概念记忆或书面推导层面。

5．3 课堂报告与反馈评价

课堂报告是期末综合大作业的重要配套环节。学生以小组形式汇报大作业成果，内容包括选题背景、问题定义、技术路线、实现过程、实验结果和反思改进。汇报过程中，教师和学生围绕选题意义、技术分析、实验设计和结果解释进行提问，帮助汇报小组进一步完善工作。

课堂报告环节引入生生互评与师生点评。生生互评引导学生从评价者角度理解不同小组的工作，评价维度包括选题与问题理解深度、技术分析与整体调研质量、汇报结构与表达逻辑、展示效果与时间控制、工作量与团队配合等。师生点评则重点关注问题抽象、方法选择、实验设计、结果解释和局限分析，引导学生形成反思和改进意识。

通过上述过程化评价机制，课程评价从单一结果考核转向多环节、多主体、多维度评价，更加关注学生在课堂学习、实验实践、综合应用和展示反思过程中的能力形成。

6 课程实践与反思

6．1 课程实践情况

本文以《程序分析与验证》课程为实践对象开展课程重构。课程实施过程中，首先通过课堂讲授和课堂练习帮助学生掌握程序分析与验证的基本概念和核心方法；随后通过基础实验训练学生对语法分析、控制流分析、数据流分析和类型系统等内容的理解；在综合大作业环节，学生围绕缺陷检测策略、专题综述、前沿论文研读和原型工具开发等任务开展实践；最后通过课堂报告、工具展示、提问交流、生生互评和师生点评进行反馈与评价。

从综合大作业选题看，学生对 AI 赋能程序分析与验证方向表现出较高关注。本轮课程共有 25 个小组完成综合大作业和课堂报告，其中 13 个小组选择 AI 相关选题，占比 52%。选题主要覆盖 AI 辅助程序验证、AI 辅助静态分析、智能缺陷检测、规约生成、不变量推导和智能化测试等方向，如表 5 所示。

。

表 5　综合大作业选题类型统计

选题类型	小组数量	占比	典型主题
AI相关选题	13	52%	AI辅助程序验证、AI辅助静态分析、智能缺陷检测、规约生成、不变量推导、智能化测试
非AI相关选题	12	48%	模糊测试、程序修复、并发缺陷检测、内存错误检测、符号执行等
合计	25	100%	—

6．2　实践观察与初步成效

课程实践表明，前沿化内容能够增强学生对课程价值的认识。超过一半的小组选择 AI 相关主题，说明学生能够主动关注 AI 赋能程序分析与验证的发展趋势，并尝试将数据流分析、模糊测试、类型系统、霍尔逻辑等经典知识与代码分析、规约生成、不变量推导等前沿问题联系起来。

实践化作业也提升了学生开展综合任务的积极性。特别是在原型工具开发和课堂展示环节，学生借助 AI 工具完成代码编写、文献梳理、报告组织和 PPT 制作，在一定程度上提高了实践任务完成效率和成果展示质量。AI 辅助开发降低了部分工程实现门槛，使学生能够将更多精力投入到问题定义、技术路线、实验结果解释和方法局限分析中。

同时，AI 工具的参与也使课程评价面临新的挑战[11]。由于学生可以借助 AI 生成部分代码、综述和展示材料，仅依赖最终提交成果难以判断其真实掌握程度。因此，课堂报告、工具展示、提问交流和师生点评成为重要评价环节。通过现场说明方法原理、实现过程、实验结果和局限性，并回答教师和同学的问题，可以更有效甄别学生是否真正理解相关知识，以及掌握程度如何。

6．3　问题与改进方向

课程实施中仍存在一些需要改进的问题。首先，AI 相关选题比例较高，说明前沿内容具有吸引力，但也可能导致学生过度关注热点应用，而忽视程序分析与验证的经典基础。后续需要在选题指导中强调“经典基础支撑前沿理解”，引导学生从控制流分析、数据流分析、模糊测试、类型系统和霍尔逻辑等基础内容出发理解 AI 赋能方法。

其次，AI 工具使用需要进一步规范。AI 工具能够提高代码实现、综述整理和展示制作效率，但也可能带来底层算法理解不足、文献辨析不充分、展示材料重形式轻内容等问题。后续可要求学生在报告中说明 AI 工具使用范围，并通过课堂报告、工具演示和提问答辩重点考查其对分析策略、实现过程和方法局限的理解。

最后，课程仍需在前沿内容更新、任务难度控制和评价细则完善之间保持平衡。后续将进一步建设 AI 赋能程序分析与验证案例库，完善原型工具任务模板、论文阅读清单、综合大作业评分标准和课堂互评表，以持续优化课程实施效果。

7　结束语

本文面向 AI 时代程序分析与验证类课程建设需求，围绕前沿化内容重构、实践化作业设计和过程化评价机制开展探索，尝试在经典程序分析与验证知识体系基础上，引入 AI 赋能程序分析与验证前沿内容，并通过综合大作业、原型工具开发、课堂报告与反馈等方式强化学生的实践应用和反思能力。课程实践表明，该重构思路能够为学生理解 AI 时代软件质量保障问题提供新的教学路径。后续将进一步完善前沿案例库、作业任务模板、AI 工具使用规范和评价标准，持续优化课程实施效果。

参 考 文 献

[1] 毛新军,董威,尹良泽,等.大语言模型辅助软件开发实践的探索[J].高等工程教育研究,2025,(06):33-38

[2] 冯月春,王佳林.基于AI技术融合的软件工程实践教学改革[J].信息与电脑,2026,38(08):248-250

[3] 胡俊杰,陈宇杰,胡义坤,等.大模型驱动的形式化定理证明：综述与展望[J].计算机科学,2026,53(04):1-23

[4] Wen C, Cao J, Su J, et al. Enchanting program specification synthesis by large language models using static analysis and program verification[C]//International Conference on Computer Aided Verification. Cham: Springer Nature Switzerland, 2024: 302-328

[5] Flemming Nielson, Hanne R. Nielson, Chris Hankin. 程序分析原理[M]. 詹博华, 冀振燕, 孙文辉, 译. 北京: 机械工业出版社, 2022

[6] 汪海沛,孙福特,张磊.一种利用大语言模型增强静态分析代码摘要能力的方法[J].小型微型计算机系统,2025,46(05):1177-1183

[7] 王东煜,莫然,詹文静,等.代码自动生成工具 GitHub Copilot 生成代码质量的分析[J].计算机科学,2025,52(07):37-49

[8] 杨宸,高见.基于大语言模型的模糊测试综述[J].计算机应用,2025,45(S2):238-246

[9] 张雨昕,白倩,周进.促进还是抑制：生成式 AI 对学生编程能力的影响——基于 42 项实验和准实验的元分析[J].现代教育论丛,2026,(02):102-112

[10] 姚丽丽,曹阳,李晓明,等.基于 OBE 理念的计算机类专业课程过程化考核评价模型[J].计算机教育,2025,(08):230-235

[11] 乔文豹,于欣宜.突破 AI 代写：计算机项目实践课程教学改革与思政建设[C]. 计算机技术与教育学报,2025,13(6):511-519