浅析理论计算机科学与形式化方法课程中重要思想与意识的教育*

刘万伟 李暾 华强胜 毛晓光 沈立

国防科技大学计算机学院 长沙 410073
华中科技大学计算机科学与技术学院 武汉 430074
国防科技大学计算机学院 长沙 410073

摘 要 理论计算机科学及形式化方法系列课程包含了计算机与数学的诸多交叉领域的重要知识内容。同时，与之相关的诸多重要思想、概念贯穿于该系列课程中。从某种角度来说，掌握这些方法并形成相应意识的重要性丝毫不亚于知识本身的学习。本文从事相关多门课程的教学经验，归纳总结出了结合理论计算机与形式化方法系列课程中 5 种方法、4 种重要思想以及 4 个重要意识。讨论了这些方法、思想、意识的具体内涵，点明了其在形式化方法教育中的作用，并给出了相应的示例及实施建议。

关键字 形式化方法，理论计算机科学，方法概念，思想意识

A Preliminary Analysis of the Education of Important Ideas and Awareness in the Courses of Theoretical Computer Science and Formal Methods

Wanwei LIU Tun LI Qiangsheng HUA Xiaoguang MAO Li SHEN

College of Computer Science and Technology
National University of Defense Technology,
Changsha 410073, China;
{wwliu, tunli }@nudt.edu.cn

School of Computer Science&Technology
Huahong University of Science&Technology
Wuhan 430074, China
qshua@hust.edu.cn

College of Computer Science and Technology
National University of Defense Technology
Changsha 410073, China
{xgmao, lishen}@nudt.edu.cn

Abstract—The series of courses on theoretical computer science and formal methods encompasses important knowledge in many interdisciplinary fields between computer science and mathematics. Meanwhile, numerous significant ideas and concepts related to this area are interwoven throughout the series of courses. In a sense, mastering these methods and developing the corresponding awareness is no less important than learning the knowledge itself. Based on the teaching experience of multiple relevant courses, this paper summarizes five methods, four important ideas, and four crucial awareness aspects from the series of courses on theoretical computer science and formal methods. It discusses the specific connotations of these methods, ideas, and awareness, highlights their roles in formal methods education, and provides corresponding examples and implementation suggestions.

Keywords—formal methods, theoretical computer science, method-related concept, methodological awareness

1 引 言

理论计算机科学及形式化方法[1][2]是计算机学科重要的分支，它是计算机科学产生的源头，并为该学科的发展提供坚实的理论支撑。该系列课程的教与学，对于教师和学生而言，具有高阶性、挑战度；同时，这类课程最终的落脚点在于创新性。在计算机科学与软件工程学科的本科与研究生课程体系中，这些课程位于计算机科学与数学的交叉领域，具体包括：离散数学、抽象代数、形式语言与自动机、计算理论（包括可计算理论、计算复杂性理论）、数理逻辑、计算机逻辑、形式化方法、系统分析与验证等。这些课程一方面具有宽广的知识内容谱系，与计算机科学的其他分支有着千丝万缕的联系；另一方面又包含非常多的思想、方法—— 它们有时体现为具体的定义、概念；有些体现为特定的方法、技巧；还有一些则体现为某些特定意识的迁移与具体化。

就某一具体的课程而言，必然需要特定的教学方法、手段，对其课程内容进行系统的讲授。与此同时，从另一个维度，计算机理论课程中的重要思想、方法、意识的培养则显得更为重要 —— 某些思想、方法可能会贯穿、应用于多门课程。从某种意义来讲，学习-内化-运用这些思想、方法，并产生一定程度的创新与发展，是形式化方法教育更为重要的目标，对于研究生教育更是如此。

* **基金资助**：20222169：本文得到教育部“基础学科拔尖学生培养计划 2.0 研究课题”资助。

本文着眼于理论计算机科学与形式化方法系列课程的教学，归纳出若干重要的思想、方法以及若干需要着重培养的意识。对这些思想、方法的产生、意义、及其运用进行分析与讨论；点明了这些方法、思想、意识的具体内涵，以及其在形式化方法教育中的作用；同时，就具体方法及思想与课程内容的结合进行了探讨。

2 若干方法与概念

2．1 归纳方法

实际上，对于归纳（Induction）的概念，学生最早从中学阶段便开始接触，如学习自然数集上的归纳法时。而后，在离散数学课程中正式系统学习，并将其扩展至一般的良序上的归纳法。在某些后继课程中（如数理逻辑、集合论），会进一步将其扩展至更一般良基上的归纳法，如“超限归纳法”、“公式结构归纳法”。一般而言，归纳用于证明某个集合上的性质，或者用以定义/生成具有某些具体的集合或类。归纳法的精髓在于给出归纳步骤与归纳规则，使得性质成立的范围得以扩展，或者能够构造新的元素。归纳过程一定需要完全覆盖目标集合，因此寻找特定的良序/良基是归纳过程的另一个关键。有时，需要人为的构造某些“秩函数”，以元素的函数值确定某个良基（或者良序）。典型的证明如“欧拉图的充要条件”（离散数学）、“消解方法的完全性”（数理逻辑）等。此外，在某些相关课程中，还需要掌握类似于“联立归纳法”的技巧，用一组相关的归纳命题共同进行归纳。总之，归纳方法是理论计算机科学、形式化方法甚至一般数学学科中普遍运用的方法，在此类课程学习中具有重要的地位。

2．2 递归方法

递归（Recursion）是与归纳有着密切关联的一个数学概念，它是解决复杂问题的有力工具。它具体指在计算/定义/证明某些结果/概念/结论时，嵌套的使用相同或相似的计算/定义/证明过程的现象。本质上来说，前面提及的“归纳定义” 即为递归定义，只是该递归的出口对应良基的极小元。对于计算机专业的学生而言，从程序语言相关课程便开始接触到基于递归的计算过程，而证明中涉及到的递归手法大致合同于归纳证明，但其重点在于如何将当前问题转化至具有更低“秩”的相同问题的证明。一个与递归具有密切联系的概念是“不动点”，它是对递归计算的整体角度上的刻画。不动点是许多问题中的核心概念，如：计算理论中的“递归定理”、时序/模态逻辑中的“极大/极小”-不动点算子、形式语义学中关于循环的语义定义（完全偏序集上的极小不动点）、进程代数中的不动点算子等。事实上，不动点的概念与应用几乎遍布各个数学领域—— 如拓扑学、泛函分析、数值计算中。该概念深入的影响与推动了形式化方法与理论计算机科学的发展。

2．3 对角线方法

对角线方法（Diagonalization）是指将某命题 A 否定后，得到“命题 B 成立当且仅当 B 不成立”这样的矛盾，从而间接证明 A 必然成立。虽然它整体上从属于反证法，然而使用这类手法给出的证明简洁、优雅、构造性强。该方法得名于康托尔关于“有理数严格少于实数”的证明。之所以单独强调“对角线”方法，是因为很多学科分支的奠基性定理都是基于这正方法给出的证明。例如：集合论中罗素悖论、“一个集合不可能与其幂集等势”的证明；递归论中“图灵机停机问题不可判定”的证明；计算复杂性理论中“层次定理”的证明。因此，掌握与熟悉该种证明方法对于理解理论计算机科学的核心定理将会起到非常重要的作用。同时，有意识的锻炼学生对角线方法的运用，可以使其进一步体会到计算机科学的逻辑严谨性，并可强化其构造性证明的技巧运用。

2．4 归约方法

笼统的说，归约（Reduction）是指将问题的求解转化为另外一个问题的求解的方法。主要包括映射归约与图灵归约 —— 前者通过一个可计算的映射 f，将问题 A 的输入转化为问题 B 的输入，使得 $w\in A$ 当且仅当 $f(w)\in B$ 对所有的输入 w 皆成立；后者则更具有一般性，它将 B 的求解作为可调用的子过程，设计 A 的求解过程。因此，映射归约可以视为将对 B 的过程调用限制在最后一步的图灵归约，而其余步骤的为输入转化。如果对映射的计算过程加上多项式时间或者对数空间的限制，则映射归约和图灵归约分别特化为卡普[3]（Karp）归约和库克（Cook）归约。归约是递归论与计算复杂性理论中最为核心的方法 ——在递归论中，映射归约与 s-m-n 定理配合使用，可以有效的证明某些集合的递归性/递归可枚举性/非递归可枚举性；图灵归约是确定问题相对可解/不可解的有利工具；此外，二者通过跃迁定理建立有机的联系。在计算复杂性理论中，问题的完全性的证明一般都是通过归约方法证明。此外，这些归约方法本身也包含高超的技巧：它能够将一个领域（如图论、数论、逻辑等）的问题通过映射巧妙地转换另一个领域的问题，并且保持充要性。卡普给出若干 **NP** 完全问题的归约，也标志着计算复杂性理论的建立。归约方法的学习，对于培养学生的问题转化能力起到至关重要的作用。

2．5　转化方法

转化（Transformation）是指将某种数学表示等价的转化为另一种数学表示方法。事实上，数学表示的形式对于问题求解的难易程度 —— 一般，不同的数学表示能够从不同的侧重面揭示问题的特点。比如，就“上下文无关语言关于取逆封闭”这个性质而言，基于文法的证明相较于基于自动机的证明更加直观、简洁。再如，经典的模型检验主干流程是把时序逻辑公式转换为自动机之后执行图上的搜索算法，这样更容易对算法本身进行直观的理解。计算机科学中最典型的转化首推丘奇-图灵论题的证明：它通过基于编码的转化，展示了λ-演算、图灵机、部分递归函数之间的相互等价性。另一方面，同类数学表示的“范式化”（Normalization）也是一种重要的转化。如：自动机的确定化、将上下文无关语言写为乔姆斯基范式、将一阶逻辑公式转化为前束范式/斯科伦范式等。通过该种转化，通常可以获得性质简洁的证明。实现转化的技术手段多种多样，最典型的方法包括编码（Encoding）与模拟（Simulation）两种 —— 前者如丘奇-图灵论题的证明；后者如可计算理论中图灵机格局的模拟。转化是理论计算机科学与形式化方法学习中较为核心的方法，是必须要熟练掌握的核心技能。

3　几种主要思想

3．1　分类思想

分类（Classification）是将研究的对象按照某种标准分为若干相关的类，可以有效压缩研究对象的规模，并同时揭示研究对象内在固有本质的思想。比如，在离散数学和抽象代数中研究的等价关系、同余关系、商代数，就是这种思想的体现。在计算复杂性理论中，根据问题之间的“可归约性”，将其分为若干个复杂度空间，如大家熟知的 **P** 类、**NP** 类、**PSPACE** 类、**EXP** 类等。每个空间中可能包含若干个典型的“完全问题” —— 如 **NP** 类中的 SAT 问题，该问题可被此类中其他问题在多项式时间内归约到。

再如范畴论中“某些数学对象在同构意义下是唯一的”等描述，正是分类思想的体现。事实上，数学对象的分类，正是某些学科的核心研究内容 —— 比如，在拓扑学中，人们注重研究各个空间之间的同胚/同伦/同调关系。一方面，完成分类后，只需研究同一分类中某个对象，使得问题空间得以化简；另一方面，分类的标准的制定也体现了人们对于数学对象之间本质差异的认知。在学习过程中，应较早的使学生了解分类这一思想，并强化相关的训练。

3．2　反驳思想

反驳（Refutation）也是反证法一种，其目标在于阐明“某个对象不属于某个集合/分类”、“某性质不能被某种范式刻画”、“某目标无法基于某种手段达成”、“某结论无法在某公理系统中被证明”等。具体示例如：递归论中阿克曼函数非原始递归性的证明；数理逻辑中某公理/规则独立性的证明；抽象代数中三大尺规作图问题的不可行性及五次以上代数方程无根式通解的证明；以及哥德尔不完全定理的证明[4]。不同于一般的反证法，反驳的主要过程在于阐述目标集合/分类/范式所应具有的必要条件。比如：可用尺规作图得到的点对应之复数，其（在初始域上）极小多项式次数必为 2 的某个方幂，而“三等分角”、“倍立方体”、“化圆为方”问题中所需的点不满足该性质，因此无法做出。再如：可用根式求解之代数方程，其分裂域的伽罗华群必为可解群，而某些方程（如 $x^5-5x+2=0$）的伽罗华群不是可解群，因而不可解。对于反驳过程而言，给出其必要条件是最为核心的部分 —— 它从本质上刻画了目标集合/分类/范式等的能力的一种界限。给出此必要条件时，一般需要结合其他的方法进行。如：在证明阿克曼函数非原始递归性时，需要给出原始递归函数某个增长速度的界限，该界限一般基于结构归纳法给出[5]。

3．3　近似思想

很多情况下，当某些问题难以求解甚至无法求解时，应转而寻求其满足工程需求的近似解，或者给出在某些限制下的特殊解，这就是近似（Approximation）的思想。

事实上，该思想在如数值分析等诸多计算机相关的学科中被广泛采用。典型的场景如：超越方程、大规模线性方程组的迭代求解 —— 在方程无法给出解析解时，可用在一定误差范围内的数值解进行替代。在理论计算机科学中，近似的思想也被广泛采用 —— 例如，在静态分析中使用的抽象解释方法；在模型检验中使用的限界模型检验方法、以及迭代-精化方法；计算机逻辑学中针对某些时序逻辑的限界可满足性求解；在计算理论中针对近似算法的研究[6]，等等。

这些都是针对原问题难以直接求解情况时给出的“增量式”近似求解方法。实际上，强化学生对近似思想的认同，在科学研究中也有着非常重要的意义，它往往是对困难的理论问题着手研究时不可或缺的方法论手段。

3．4　近似思想

所谓对偶（Dualization），就是指借助数学对象间存在的对称或对应关系，获取其相应性质，或将问题求解进行转化的思想。在代数学以及范畴论中，存在诸多对偶的概念。如：常态射/余常态射、回拉/外推、

核/余核、极限/余极限、协变/反变函子等。这些概念之间偶性的存在，使得其相应性质的对偶版本可以“不加证明”的直接导出，其依据是“对偶原理”这一元性质。在逻辑学中，人们为了定义和书写的简洁性，一般会定义出许多派生算子。

这些算子一般在语法上满足某种对偶关联。对偶性更为重要的应用体现在最优化理论[7]中 —— 如：通过引入拉格朗日乘子，转为求解一个与原问题关联的对偶问题，它有时能将某些非凸问题转化为凸问题，从而降低优化求解的复杂性。此外，在可计算与计算复杂性理论中，同样也采用对偶思想对复杂性空间进行分类，如 **NP** 的对偶空间被命名为 **coNP**，将 **RP** 的对偶空间命名为 **coRP** 等。对于这类空间及其对偶，人们尤其关心它们应用集合操作后获得新空间的性质，如 **NP** ∩ **coNP**、 **RP** ∩ **coRP** 等 —— 后者实际上等于 **ZPP** 空间。再如，在泛函、拓扑等学科中，对偶性也是非常重要的研究手段 —— 原空间/代数结构到某个数域上的同态构成了该空间的对偶空间/对偶代数，如：对偶线性空间、上同调群等概念。

4 若干重要的意识

4．1 推广意识

从某种意义上讲，推广（或泛化，Generalization）是科学研究本身的主要目标之一。推广的目的在于：将某些概念扩充至限制更少的条件下，使得原概念中的主要特征得以保持，但将某些定理扩展至更加一般的情形中，使得原结论得以增强；或者将某些方法的使用限制得以放松，使该方法可应用的情形更加宽泛。比如：对于“极限”这个概念而言，最初在分析学中基于 ε-δ 语言对其给出定义；在拓扑学中，由于“距离”不一定继续存在，故转而基于有向集和开集网对其重新定义；在范畴论中，由于拓扑结构不再存在，又转而给出基于锥及锥态射的定义。

上述定义的变化过程经历了从特殊向一般的推广，但始终围绕满足特定性质元素的“存在唯一性”这一基本原则进行。再如，将正规语言（RL）的泵引理推广至上下文无关语言（CFL）的泵引理，不仅能够进一步拓宽结论的使用范围，还可从结论的形式上进一步确认“RL 是特殊的 CFL”这一事实。同时可以启发思考：若将针对 CFL 的证明针对 RL 这一特殊子集进行，便可洞悉线性文法相对于上下文无关文法的特殊之处。

最后，方法的推广是科研中最重要的意识，是研究生教育的重要环节：它促使学生将研究对象进一步拓展，从而产生新的研究成果。比如，思考如何将针对有限状态系统的模型检验算法推广至特定的无限状态系统的模型检验中？能否将某种时序逻辑的可满足性判定拓展至表达能力更强的逻辑中？

4．2 对比/类比意识

对比或者类比的意义在于：通过研究相似/相近/对偶概念之间的关联与区别，从细节发掘导致差异的本质，同时也启发将某问题的解决方案迁移至相关问题的求解思路。比如，在讲授计算复杂性时，可将“时间层次定理”与“时间间隙定理”这两个看上去相矛盾的定理放在一起讲授，使学生意识到“时间可构造”这个前提条件在第一个定理中起到的作用，并促使其回顾证明的细节并进一步深化对这个条件作用的理解。再如，在讲授抽象代数中的“伽罗华对应”相关内容时，会涉及两个关键的“伴随”公式 $G=\mathrm{Gal}(E/\mathrm{Inv}G)$ 与 $F=\mathrm{Inv}\,\mathrm{Gal}\,E/F$，其中 E 为 F 关于某个多项式 $f(x)$ 的分裂域。此时应当促使学生思考：后者的成立需要 “可分”这个先决条件，即 $f(x)$ 在 E 上没有重根。

因此，从本质上说，这两个伴随公式成立的条件是不同的。由此，培养对细节的注重，可以使学生真正体会到科学上“差之毫厘、谬以千里”的道理。于此同时，对比意识的培养在激发学生创新性探究中起着非常重要的作用。比如，若对比“RL 关于求交封闭”以及“CFL 关于求交不封闭”这两个结论，便很快意识到“具有两个栈的下推机与具有一个栈的下推机的表达能力是不同的”。接下来，学生自然会探究前者究竟对应那种形式语言，从而进一步明晰“图灵机就是具有两个以上堆栈的下推机”这一实质性原理。

4．3 折衷-渐进意识

在针对具体的科学问题进行研究时，难免碰到一些挑战性极高的问题。对于这些问题，往往难以找到较为直接的突破口，因此可以暂时考虑问题在特殊条件下或者具有特殊约束时的解决方案。比如，在计算机逻辑学领域，PCTL 的可满足性问题是一个十分重要的开放问题，长久以来人们难以给出此问题的解答，甚至提供较为明晰的解决思路（该问题自 1995 年提出，直至 2025 年才给出完全的解答[8]）。但在此之前，人们曾经从多个角度对该问题的几种特殊形式进行了考察：比如考虑量化 PCTL（即公式中仅出现概率 0 和 1 的 PCTL 片段）的 SAT 问题 ——该逻辑是可判定的，同时发现了该逻辑某些非同寻常的性质，比如某些可满足的公式不存在有限模型。

此外，人们还研究了在对模型大小加以限制时 PCTL 的 SAT 问题，同样也取得了一定的进展。折衷-渐进的思想与之前提及的“近似”思想具有非常多的相通之处：前者强调将问题的条件进行特殊化，后者

则侧重于放松问题的精度要求，二者都是有效的推进问题解决的重要手段和思维方式。

4．4　大胆假设的意识

历史上，很多重大的科学突破来源于对某些前提的创造性假设，它使得研究者能够突破“惯性思维”的限制，获得进的结论与方法，甚至建立新的研究分支和方向。比如，罗巴切夫斯基通过将欧式几何的第五公设（过一点最多能做出一条直线平行于给定直线）进行修改，创立了“罗氏几何”。

这一在当时看来明显违反直觉的假设，从根本上促进了几何学的飞速发展，它为其他类型的非欧几何（如黎曼几何）的创立奠定了思维基础。在数理逻辑学科的发展过程中，这一思想催生了“力迫法”（Forcing）的产生 —— 当时，在哥德尔已经给出康托尔连续统假说（GCH）的正模型后，科恩又给出其反模型的构造方法，从而彻底说明了 GCH 与集合论 ZF 公理系统的独立性。在计算复杂性理论中，**P** 与 **NP** 之间的关系是长久以来悬而未决的问题。

事实上，在不同的假设下人们能够获得不同的结论：一方面，如果认为 **P**＝**NP**，那么许多复杂度空间（典型的是 **PH**）便会塌缩，它能够极大的化简复杂度的分层；另一方面，在 **P**≠**NP** 的假设下，人们能够得到许多深刻的结论，比如在 **P** 与 **NP** 之间存在一个无穷的严格分层（即，拉德纳定理）。再如，在 **NP**⊆**BPP** 这个假设下，进一步会有 **NP**⊆**RP** 的结论。因此，即使在某些较强的前提未知的情况下，也不妨碍人们暂时绕过这些障碍，对其进行进一步的探索，这正是对问题进行大胆假设的价值所在。

5　实施效果

课程组近年来在理论计算机科学以及形式化方法系列课程的教学中，逐步强化了对课程中重要概念、方法、意识的培养，强调了重要概念、方法在不同课程中的具体体现，并强化了各个相关课程中知识点的衔接。以下选取具有典型性的三门课程的近三年的学生成绩作为说明 —— 统计数据涵盖了低年级本科生、高年级本科生、研究生，以及不同学校的必修、选修课程。统计要素包括课程的修课人数、考核获优秀的人数以及相应的优秀率、考核的及格人数以及相应的及格率、该次考核的平均成绩等。所有成绩均由形成性成绩与终结性成绩按照教学大纲规定加权计算获得。

华中科技大学（卓班）开设的《离散数学（一）》，是一门面向计算机专业低年级本科生的必修课，其内容涵盖集合论、关系、图论等，该课程于每年春季学期开课。表 1 给出了该课程 2023-2025 学年的成绩对比。

国防科技大学面向计算机及软件工程等专业高年级本科生开设的《计算理论》是一门选修课程。该课程内容包括形式语言、可计算理论、计算复杂性理论等三大部分，开课学期为秋季。表 2 给出了该课程 2021-2023学年的成绩对比（该课程在2024年未开课）。

《数理逻辑》是国防科技大学研究生课程，主要涵盖了证明论与部分模型论的相关内容，该课程于每年的秋季学期开课。表 3 给出了该课程 2022-2024 学年的成绩对比。

表 1　华中科技大学《离散数学（一）》课程成绩对比

学年	学生人数	优秀人数	优秀率	及格人数	及格率	平均分
2023	37	24	64.86%	37	100%	89.68
2024	51	19	37.25%	51	100%	85.56
2025	66	43	65.15%	66	100%	89.20

表 2 国防科技大学《计算理论》课程成绩对比

学年	学生人数	优秀人数	优秀率	及格人数	及格率	平均分
2021	17	1	5.9%	16	94.1%	77
2022	9	5	55.6%	9	100%	90
2023	19	6	31.5%	18	94.7%	81.2

表 3 国防科技大学《数理逻辑》课程成绩对比

学年	学生人数	优秀人数	优秀率	及格人数	及格率	平均分
2022	76	0	0%	71	93.4%	71
2023	72	6	8.3%	65	90.2%	74.4
2024	43	8	31.5%	39	90.7%	76..7

从上述对比数据可以看出：对重要概念、方法和意识的强调能够有效的提升学生的学习效果。此外，还可以看出：随着知识体系的加深，该教学方法的效果逐渐凸显 —— 对与低年级本科生课程而言，由于涉及的方法、概念较少，其成绩主要受试题难度因素的制约；而对于研究生课程而言，其学习需要对概念、方法更加融会贯通，强化该方面的培养能够有效提升学习的效果。

6 结束语

本文总结了理论计算机科学及形式化方法中若干较为典型的几种方法概念、思想、意识。我们认为：知识是思想方法的载体，而思想方法是知识的蒸馏，二者相辅相成，共同贯穿于本学科学生教育培养的各个环节中。

对于本文讨论的思想、方法、意识，可根据具体课程的内容特点在恰当的时机进行着重强化。比如，对于归约方法的训练，一般集中于计算理论或计算复杂性等课程中实施；但若没有开设此类课程，也可在离散数学或算法类课程中对其进行介绍 —— 如通过讲授基于图论算法的命题逻辑公式可满足性的判定。再如，转化方法虽然大量运用于计算机逻辑学、形式语言、自动机理论等学科中，但实际上这种方法在离散数学与编译原理中也有所涉及，可以对其进行适时的强调。

本文所列内容不可能涵盖理论计算机科学与形式化方法的全部方法概念、思想、意识。同时它们也并不是完全孤立、泾渭分明的 —— 相反，它们之间往往具有一定的交叉与融合。此外，可以预见：随着学科的发展，相应的内容也会动态调整变化。比如，鉴于人工智能技术对计算机相关学科带来的影响，与之相关的方法也会逐步引入。典型的例子如“嵌入方法”（Embedding），它能够将离散的符号映射为连续的张量，从而使用相应的数学工具。这些都是在本学科的教育教学中值得关注的问题。

参 考 文 献

[1] 王戟，詹乃军，冯新宇，刘志明. 形式化方法概貌，软件学报[J]，2019 (1)：33-61.

[2] 张广泉. 《形式化方法》课程建设探索与实践，计算机技术与教育学报 [J]，2021 vol 9 (2) :59-64.

[3] Karp.R.M. Reducibility among combinatorial problems. R. E. Muller et al eds. *Complexity of Computer Computation* [C]. pp.85-103 Plenum Press, New York. 1972.

[4] Gödel, K. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I. *Monatshefte für Mathematik und Physik* [J]. vol. 38, no. 1, 1931, pp. 173–198.

[5] Ackermann, W. Zum Hilbertschen Aufbau der reellen Zahlen. *Mathematische Annalen* [J], *99*(1), 118–133. 1928.

[6] Sipser.M. *Introduction to the Theory of Computation* (*3rd ed*) [M]. CENGAGE Learning. 2013.

[7] Boyd S., Vandenberghe L. *Convex optimization* [M]. Cambridge Press, 2013.

[8] Chodil M, Kucera A. The satisfiability and validity problems for probabilistic computational tree logic are highly undecidable. In Proc. *ICALP'25* [C], vol 334 of LIPIcs, pp:151:1-151:20, 2025.