arXiv:2602.16554v1 [cs.LO] 18 Feb 2026

MERLEAN：量子计算自动形式化的智能体框架

Yuanjie Ren*
麻省理工学院，剑桥，MA 02139
yuanjie@mit.edu

Jinzheng Li*东北大学，波士顿，
MA 02115
li.jinzh@northeastern.edu

Yidi Qi*东北大学，波士顿，
MA 02115
y.qi@northeastern.edu



摘要

我们介绍 MerLean，这是一个用于量子计算自动形式化的全自动智能体框架。MerLean 从 LaTeX 源文件中提取数学陈述，将其形式化为基于 Mathlib 构建且经过验证的 Lean 4 代码，并将结果转换回人类可读的 LaTeX，以供语义审查。我们在三篇理论量子计算论文上评估了 MerLean，总计根据 114 条陈述生成了 2,050 条 Lean 声明。MerLean 在这三篇论文上均实现了端到端形式化，将验证负担降至仅需检查新引入的定义和公理。我们的结果表明，智能体自动形式化能够扩展至前沿研究，既可作为机器验证的同行评审实用工具，也可作为可扩展的高质量合成数据挖掘引擎，用于训练未来的推理模型。我们的方法也可以推广到数学和理论物理领域的其他严谨研究。

1 引言

自动形式化（Szegedy，2020年）的目标是将自然语言中的数学陈述自动转换为可由 Isabelle（Paulson，1994年）、Coq（Team，2024年）和 Lean（Moura & Ullrich，2021 年）等系统验证的形式化代码。近期研究表明，大语言模型凭借其语义处理能力，非常适合执行这项任务（Wu 等人，2022年；Jiang 等人，2022年；Tarrach 等人，2024年；Weng 等人，2025年）。

虽然自动形式化领域的大多数研究聚焦于纯数学，但物理学中的应用也备受关注：PhysLean/HepLean（Tooby-Smith，2024年）和 Lean-QuantumInfo（Meiburg 等人，2025年）等 Lean 库目前正在开发中，后者还专门利用自动形式化工具来加快这一进程。理论量子计算是这一工作流程的理想试验场。2025年，arXiv quant-ph 类别的投稿量达到 11,891 篇（arXiv，2025年），造成了验证瓶颈，可能使审核速度赶不上同行评审系统的处理能力。从功能上看，该领域是应用数学的一个分支，与线性代数、图论和代数拓扑有着深刻联系。这些成果严谨的性质使其适合进行神经符号验证，而业界的浓厚兴趣则确保了实际影响。

*贡献相同。

在这项工作中，我们介绍 **MerLean**，一个全自动智能体框架，旨在从 LaTeX 源文件中提取、整理并形式化数学陈述，且在形式化过程中无需人工介入。形式化成功后，我们会运行第二个智能体，将经过验证的代码“自动非形式化”回自然语言，并生成蓝图，供人类专家审查形式化代码与原始意图在语义上是否一致。我们在三篇理论量子计算论文上评估了 MerLean：其中一篇是尚未发表的 手稿（确保不存在数据污染），另外两篇则已发表。MerLean 成功完成了这三篇论文的端到端形式化；对于依赖 Mathlib 中尚不可用的数学工具的结果，还引入了必要的定义和公理。形式化输出的交互式示例可在 https://doxtor6.github.io/MerLean-examples/ 查看。

该框架的适用范围不止于量子计算。任何依赖形式化数学证明的学科，都可以采用这种神经符号方法来扩大验证规模并加速发现。

2 相关工作

基于大语言模型的自动形式化和自动定理证明器在竞赛级问题上展现了令人期待的能力（Wu 等人，2022年；Jiang 等人，2022年）。后续研究在此基础上进一步改进，致力于扩展证明细节（Tarrach 等人，2024年）和验证忠实性（Li 等人，2024年；Liu 等人，2025年；Peng 等人，2025年）。面向特定领域的应用还包括欧几里得几何（Murphy 等人，2024年）、生物医学文本（Zhang 等人，2025年）和物理学（Zhang 等人，2026年）。

近期研究转向了智能体系统，将前沿大语言模型与工具集成相结合，以支持交互式定理证明。Breen 等人（2025年）展示了 Ax-Prover，这是一个基于模型上下文协议的多智能体系统，使用 Claude Sonnet，在域外基准测试中优于专用证明器。Xu 与 Odersky（2026年）通过机械化 System Capless（14,000+ 行 Lean 4 代码），在 189 项带注释的任务中实现了 87% 的任务成功率，展示了“智能体证明自动化”；其工作流程由人工指导，人类提供定义、定理和证明策略，而智能体负责机械性的证明工程。Liu 等人（2026年）提出了 Numina-Lean-Agent，将 Claude Code 与基于模型上下文协议的 Lean 工具（包括用于查询目标和搜索定理的 Lean-LSP-MCP）结合起来；该系统无需模型训练便解决了全部 12 道 2025 年普特南数学竞赛题目，并形式化了 Brascamp–Lieb 定理（8,000+行），不过仍需要大量人工指导。

在架构上，MerLean 与这两个系统都十分相似——采用在生成—检查—优化循环中与 Lean 4 进行智能体交互的范式，以前沿大语言模型作为推理骨干，并致力于对研究级数学进行形式化。其主要特点在于，在我们测试的领域中，MerLean 无需人工指导即可实现整篇论文的全自动端到端形式化。

3 框架

MerLean 是一个双向自动形式化框架，由两条互补的流程组成：自动形式化 将数学研究论文从 LaTeX 翻译为基于 Mathlib 构建的、经过验证的 Lean 4 库，而自动非形式化 则将形式化代码转换回人类可读的 LaTeX，以供专家审阅。这种往返式设计既确保了机器验证的正确性，也确保了人类可验证的语义一致性。该框架采用基于 Claude Code 的前沿大语言模型智能体，通过多轮智能体交互工作，并依据编译器反馈、忠实度检查以及借助工具对 Mathlib 的探索，迭代优化输出。图 1 展示了整体架构。

3.1 自动形式化

陈述提取。 给定一篇 LaTeX 论文，智能体首先提取其中所有数学陈述，包括定义、定理、引理、命题、推论和注记，并将其整理为结构化的 JSON 表示。每条陈述对应论文中的一个数学结果，其中包含唯一标识符（例如 Def 1、Thm 2）、以自然语言表述的数学内容、对其他陈述的明确依赖关系，以及在可用时提供的证明概要。提取过程

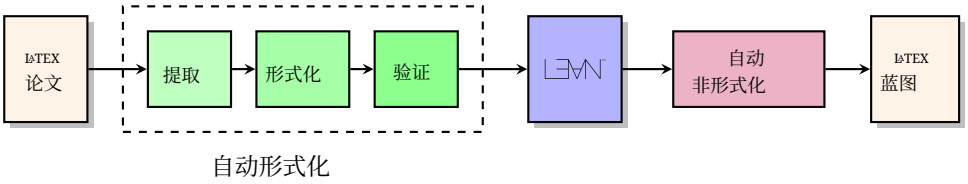


图 1：MerLean 架构。自动形式化 从 LaTeX 论文中提取陈述，将其形式化为 Lean 4 代码，并验证其忠实性。自动非形式化 则将经过验证的代码转换回人类可读的 LaTeX 蓝图或文件。

会进行多轮迭代，每一轮都会检查并完善上一轮的输出：将“通过标准论证”等含糊表述扩展为具体的证明步骤，补充缺失的中间引理，并确保陈述的排序使依赖项位于依赖它们的陈述之前。

迭代形式化。 对于每条提取出的陈述，智能体都会进入编译修复循环。它生成由一个或多个声明组成的 Lean 4 代码，将其写入库结构中的相应文件并进行编译。每条陈述通常会生成多个声明，因为智能体会自主引入辅助定义、辅助引理和类型类实例，以支持目标结果。如果编译失败，系统会解析错误消息（如类型不匹配、未知标识符、策略失败等），并将其反馈给智能体；智能体会分析错误并生成修正后的版本。这个循环会持续进行，直到代码编译时不再出现错误和警告，或达到最大尝试次数。在此过程中，智能体可以使用 lean-lsp-mcp 提供的多种诊断工具。lean-lsp-mcp 是一个模型上下文协议服务器，可将 Lean 语言服务器的功能直接提供给智能体；此外，智能体还可以使用 lean goal 检查特定位置的证明状态，使用 lean hover info 查看类型签名和文档，并使用语义搜索工具（leansearch、loogle）查找相关的 Mathlib 引理。智能体还可以在 Mathlib 源代码中搜索用法模式，并验证所引用的引理确实存在。

忠实性检查。 仅仅编译成功是不够的：大语言模型可能生成能够通过类型检查、却曲解数学含义的代码（例如，证明一个平凡命题，而非预期的定理）（Lin 等人，2025年）。成功构建后，MerLean 会反思结果是否符合原意。如果符合，程序就继续处理下一条陈述；否则就继续尝试。

公理处理。 前沿研究往往依赖尚未纳入 Mathlib 的结果（例如，某些系数环上的 K¨unneth 公式）。MerLean 通过显式的公理声明来处理这种情况，并清楚地区分有意设定的假设与未完成的证明（sorry）。如果经过最多次数的尝试后形式化仍然失败，“公理阶段”会将阻塞性的子目标转换为公理，从而生成带有透明假设的部分形式化；随着 Mathlib 不断扩展，这些假设便可得到补充。

3.2 自动非形式化

解码器会逆转形式化过程，将经过验证的 Lean 4 库转换回人类可读的 LaTeX。尽管 doc-gen4 等现有工具能够根据 Lean 源代码生成可读性尚可的文档，但其输出仍与 Lean 的类型论语法紧密绑定，缺乏形式化方法专业知识的读者难以理解。因此，MerLean 使用执行形式化任务的同一大语言模型来完成非形式化：既然该模型已证明自己能够形式化特定研究领域中的内容，它也应当能够将每项声明转换为自然语言，使不具备 Lean 预备知识的领域专家也能理解。当然，为防止数据泄露，不会向非形式化代理提供任何原始陈述或论文内容。该流程会解析所有 Lean 文件、构建依赖图，并生成两种互补的输出：一种是兼容 leanblueprint、可通过网页探索依赖结构的交互式蓝图；另一种是面向不熟悉形式化方法的读者、独立成篇的教科书式叙述。任何未经验证的假设（公理声明）都会被醒目标出，从而确保形式化边界完全透明。

4 实验

我们在理论量子计算领域的三篇论文上评估 MerLean：其中一篇尚未发表的手稿用于确保不存在数据污染，另外两篇已发表论文则用于评估其在现有文献上的表现。

- **论文 A：平衡乘积码（Breuckmann 和 Eberhardt，2021 年）**。本文研究由链复形的张量积和平衡积构造的量子码。所用的数学工具包括同调代数、复形的张量积、扩展图和谱扩张。
- **论文 B：容错量子计算（Williamson 和 Yoder，2024年）**。本文全面介绍了稳定子码和容错协议。其数学内容包括稳定子形式体系和保利代数、横向门、规范化图、容错态制备和测量。
- **论文 C：量子拓扑**。这是一份未发表手稿，因此其内容从未出现在任何大语言模型训练数据中。手稿证明了量子计算系统上某个映射的若干代数性质和群论性质。详情将于稍后公布。

这三个项目展现了该智能体在前沿严谨研究中进行形式化并弥合逻辑缺口的能力，无论相关内容是否存在于基础大语言模型的训练数据中。Fault-TolerantQC 形式化输出的交互式示例可在 https: //doxtor6.github.io/MerLean-examples/ 查看。

4.1 结果

表 1：按论文汇总的形式化实验。

论文	陈述	Lean 代码行数	声明	Time
平衡积	44	14,997	730	20小时4分钟
容错量子计算	47	18,557	923	11小时41分钟
量子拓扑	23	7,761	397	7小时51分钟

表2：三篇论文中各类陈述的形式化统计。

Type	数量	平均时间	平均编译时间
定义	49	18分0秒	11.7
定理	15	39分41秒	22.4
引理	20	33分22秒	18.3
注记	26	10分34秒	7.1
推论	4	19分23秒	5.5
总计/平均	114	21分54秒	13.0

在这三篇论文中，MerLean 共形式化了 114 条陈述，总计生成 2,050 条 Lean 声明，墙钟时间不足 42 小时。在人工审查的支持下，确保所有新增定义和公理在数学上准确且构造严谨，MerLean 成功形式化了这三篇论文，展示了其处理原创内容和已发表内容的能力。《平衡乘积码》论文中，9.1% 的陈述需要明确公理，这些结果依赖于 Mathlib 尚未具备的工具（例如谱序列、 $\mathbb{F}_2$-链复形的 K¨unneth 同构）。定理最难形式化，平均耗时 39 分 41 秒，并尝试编译 22.4 次；而注记最容易形式化，耗时 10 分 34 秒，平均编译 7.1 次，且无需公理。附录 A.1 展示了一个完整证明的代表性定理。

图 2 展示了按类型和论文细分的每条陈述编译尝试次数分布。在全部三篇论文中，该分布均严重右偏：大多数陈述在

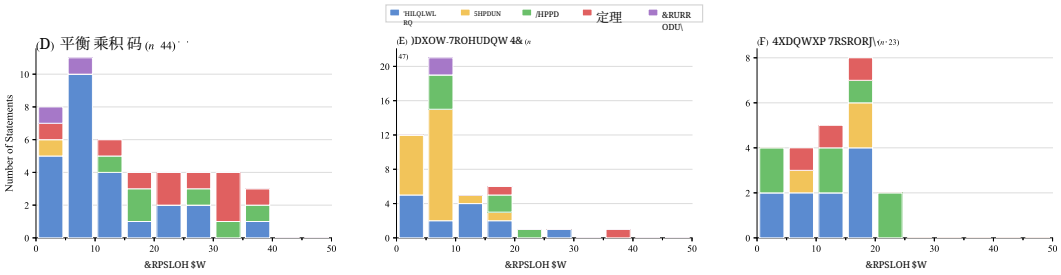


图 2：各论文按陈述类型划分的编译尝试次数分布（进入公理阶段前最多尝试 30 次）。大多数陈述在 1–10 次尝试内即可解决；定理和引理则需要显著更多轮迭代。

1–10 次尝试内即可编译成功，但“困难”陈述构成一条长尾，需要 21+ 次迭代。平衡乘积码的分布范围最宽，这反映出它对尚未纳入 Mathlib 的高级代数工具（谱序列、K¨unneth 公式）的依赖。容错量子计算高度集中于低编译次数区间，47 条陈述中有 23 条在 5–10 次尝试内得到解决；其两条定理是唯一编译次数超过 20 的陈述。量子拓扑的分布更为均匀，这与其规模较小但技术上更为多样的陈述集合相符。在所有论文中，定理和引理主导高编译次数区间，而定义和注记则聚集在较低区间。

4.2 定性观察

编译修复循环行为。 需要 20+ 次编译尝试的难点陈述通常涉及：

1. 依赖类型算术（例如，在类型层面证明 $i - 1 + 1 = i$） 2. 缺失引理，需要创造性地绕开 3. 策略超时，需要重构证明

代理发现的引理。 MerLean 经常引入原论文中未陈述的辅助引理。这些辅助引理是自然的中间步骤，人类数学家通常会将其隐含处理（完整示例见附录 A.2）。例如，该代理引入了：

- edgeBoundary card eq edgeCount（平衡乘积码）：弥合扩展的几何定义（边界大小）与代数邻接计数之间的差距，从而能够形式化推导相对切格尔不等式。
- pauliPair 反对易 ct 已满足（容错量子计算）：从代数上验证了探测器模型中反对易的时空故障会彼此不相交地抵消，这是证明时空综合征提取的全局正确性所必需的中间结果。
- 权重不等式核心（容错量子计算）：通过对切格尔参数 $h(G) \geq 1$ 进行分类讨论，证明了 $w' - |S| + |\partial S| \geq \min(h(G), 1) \cdot d$，从而衔接清除引理与空间距离下界（见附录 A.2）。

公理声明。 平衡乘积码的形式化需要显式声明公理，因为 Mathlib 尚未提供底层数学工具。值得注意的是，原论文直接引用了这些经典定理而未给出证明，因为它们是文献中广为人知的结果；因此，非形式化证明从一开始就未包含在源材料中，这使得公理化成为形式化设置中的自然对应做法。这些公理揭示了与物理学应用相关的具体缺口：

- **K¨unneth 公式**：对于 $\mathbb{F}_2$ 上的链复形，Mathlib 中没有同构 $H_n(C \otimes D) \cong \bigoplus_{p+q=n} H_p(C) \otimes H_q(D)$（kunnethMap injective aux、kunnethMap surjective aux）。

- **张量-同调交换性**：对于平坦模 V，假定同构 $H_q(V \otimes C) \cong V \otimes H_q(C)$ 可通过组合公理 e2PageIsoHomologyTensor ax 得到。从概念上讲，该公理将垂直和水平同调步骤合并，以便在定理3中将谱序列的 E^2 页与底空间和纤维的同调的张量积相对应。单独的公理 verticalHomologyIsoTensor ax 和 horizHomologyIsoTensor ax 虽已定义，但在主要证明中已由这一组合形式取代。
- **谱序列**：通过谱序列计算总体复形同调的工具尚不完善（spectralSequenceIsomorphism 并非平凡结果）。在定理3的纤维丛 Künneth 定理证明中，这一工具用于建立 E^2页与总同调之间的联系。

这些公理在 Lean 代码和生成的蓝图中都被清楚标示，使读者能够立即看出哪些结果依赖于未经验证的假设。附录A.3展示了完整的 Künneth 公式形式化，作为一个具体示例。

5 讨论

5.1 MerLean 面临的挑战

Mathlib 的缺漏与基于公理的形式化。 我们的评估发现，Mathlib 在专门的物理学概念方面存在缺漏，例如，尚未提供 Künneth 公式。MerLean 通过在依赖图中明确声明公理节点来处理这一问题。更广泛地说，许多物理学概念并未用数学语言严格定义（例如量子场论中的概念）。然而，构建涵盖全部既有知识的 Lean 代码超出了本研究（以及类似研究）的范围。因此，基于“公理”开发库是合理的。鉴于上述原因，这些公理的选择既不应过于基础，也不应过于高级，以免使主要结果的推导变得毫无意义。

忠实性检查。 一项关键挑战是确保形式化代码准确反映原文，而不只是能够编译通过。MerLean 的忠实性检查流程用于解决这一问题，而自动非形式化蓝图则揭示了逻辑结构，供人类审查。这些机制相辅相成，尽可能消除幻觉，让人能够立即看出智能体是否证明了一个平凡变体，或是为了满足编译器而捏造了定义。

5.2 MerLean 的未来应用

研究助理。 在我们对一篇未发表的量子拓扑论文进行初步形式化时，有一个引理始终需要依赖公理。查看自动非形式化蓝图后发现，问题源于一个含糊的定义：作者默认的一项约束缺失了。修复 LaTeX 源文件后，MerLean 生成了完整的形式化结果。这表明，双向流程有助于研究者提高其设定、定义和证明的严谨性。

形式化同行评审。 我们对一份未发表手稿的验证证实，MerLean 无需针对特定内容进行预训练即可运行。我们设想，在起草过程中于本地进行自动形式化。正如 LaTeX 成为数学排版的标准，智能体框架也可能使形式化仓库成为静态 PDF 的标准配套，从而改变同行评审的方式：评审者可以依赖机器验证的保证，将注意力完全集中于创新性和科学意义。

合成数据飞轮。 自动形式化在训练专用定理证明器方面也发挥着关键作用，因为大规模合成数据的质量直接取决于形式化的准确性（Xin 等，2024；Kumarappan 等，2025）。MerLean 通过挖掘以相关科学研究为基础的高质量（自然语言，形式化代码）对，促进大语言模型训练形成良性循环。将这些数据反馈给基础模型，将有助于提升下一代智能体的能力。

助力物理学库建设。 Lean 生态系统中形式化物理学的工作日益增多，包括面向一般物理学的 PhysLean/HepLean（Tooby-Smith，2024年）以及面向量子信息理论的 Lean-QuantumInfo（Meiburg 等人，2025年）。MerLean 可通过形式化相关研究论文并提取可复用的定义和引理，加快这些库的建设。研究人员也可以使用 MerLean，通过形式化一批高度相关的论文来创建领域专用库，从而为其研究领域构建连贯的形式化基础，涵盖多项研究中相互关联的定义、引理和定理。

6 结论

我们介绍了 MerLean，这是一个全自动双向框架，可将数学研究论文从 LaTeX 形式化为经过验证的 Lean 4 库。我们在理论量子计算领域的三篇论文上进行了评估，涵盖稳定子码、容错协议、平衡乘积码和同调代数，共生成了 2,000 多条 Lean 声明，包含 41,000+ 行经过验证的代码，证明了前沿研究的全自动形式化是可行的。迭代式编译—修复—验证循环无需人工干预即可有效生成忠实的形式化结果，而自动非形式化流程则使领域专家无需具备形式化方法专业知识，也能审查语义一致性。

展望未来，尽管理论量子计算是一个有效的试验平台，但其数学基础主要是线性代数和泛函分析，因此受益于 Mathlib 的成熟支持。我们计划将评估扩展到其他科学领域以及纯数学的其他分支，这些领域和分支具有更深的依赖链或不同的基础结构，例如代数几何和数论，以全面确立该框架的通用性。

我们还计划在现有的自动形式化基准测试上评估 MerLean，以便与先前的工作进行直接比较。然而，目前的基准测试（如 miniF2F、ProofNet）主要关注孤立的定理陈述，而非包含相互关联的定义、引理和定理的整篇论文形式化。未来，我们计划创建一个专门的基准测试，涵盖多个数学领域中丰富多样的研究论文，从而能够在前沿研究层面系统评估自动形式化系统。

大语言模型使用披露

根据 ICLR 2026 的政策，我们披露本研究中大语言模型的以下用途：

研究。 MerLean 将 Claude（Opus 4.5）用作自动形式化和自动非形式化的核心推理引擎。该大语言模型负责从 LaTeX 中提取陈述、生成 Lean 代码、进行错误诊断与修复，以及将形式化代码翻译为自然语言。本文报告的所有实验结果均由这一基于大语言模型的系统生成。作者已对所有研究贡献进行核验和验证。

写作。 Claude Code 被用作写作助手，用于编辑本稿的部分内容，包括修正拼写错误和改进语法。所有内容均由人类作者审阅、核实和修订，他们对最终投稿的准确性和完整性承担全部责任。

参考文献

arXiv。arXiv 量子物理学投稿统计。https://arxiv.org/year/quant-ph，2025年。

Benjamin Breen、Marco Del Tredici、Jacob McCarran、Javier Aspuru Mijares、Weichen Winston Yin、Kfir Sulimany、Jacob M. Taylor、Frank H. L. Koppens 和 Dirk Englund。《Ax-Prover：用于数学和量子物理学定理证明的深度推理智能体框架》，2025年11月。统一资源定位符 http://arxiv.org/abs/2510.12787。arXiv:2510.12787 [cs]。

Nikolas P Breuckmann 和 Jens Niklas Eberhardt。平衡乘积量子码。*IEEE* 信息论汇刊，67(10):6653–6674，2021 年。

Albert Q. Jiang、Sean Welleck、Jin Peng Zhou、Wenda Li、Jiacheng Liu、Mateja Jamnik、Timoth´ee Lacroix、Yuhuai Wu 与 Guillaume Lample。起草、构思与证明：借助非形式化证明引导形式化定理证明器，2022年11月。统一资源定位符 http://arxiv.org/abs/2210. 12283。arXiv:2210.12283 [计算机科学]。

Adarsh Kumarappan、Mo Tiwari、Peiyang Song、Robert Joseph George、Chaowei Xiao 与 Anima Anandkumar。LeanAgent：面向形式化定理证明的终身学习，2025年3月。统一资源定位符 http://arxiv.org/abs/2410.06209。arXiv:2410.06209 [计算机科学]。

Zenan Li、Yifan Wu、Zhaoyu Li、Xinming Wei、Xian Zhang、Fan Yang 与 Xiaoxing Ma。通过符号等价与语义一致性自动形式化数学陈述。2024年11月。统一资源定位符 https://openreview.net/forum?id=8ihVBYpMV4。

Yong Lin、Shange Tang、Bohan Lyu、Ziran Yang、Jui-Hui Chung、Haoyu Zhao、Lai Jiang、Yihan Geng、Jiawei Ge、Jingruo Sun、Jiayun Wu、Jiri Gesi、Ximing Lu、David Acuna、Kaiyu Yang、Hongzhou Lin、Yejin Choi、Danqi Chen、Sanjeev Arora 和 Chi Jin。Goedel-prover-v2：通过脚手架式数据合成和自我纠正扩展形式化定理证明，2025年。统一资源定位符 https: //arxiv.org/abs/2508.03613。

Junqi Liu、Zihao Zhou、Zekai Zhu、Marco Dos Santos、Weikun He、Jiawei Liu、Ran Wang、Yunzhou Xie、Junqiao Zhao、Qiufeng Wang、Lihong Zhi、Jia Li 和 Wenda Li。Numina-lean-agent：面向形式化数学的开放通用智能体推理系统，2026年。统一资源定位符 https: //arxiv.org/abs/2601.14027。

Qi Liu、Xinhao Zheng、Xudong Lu、Qinxiang Cao 和 Junchi Yan。《重新思考并改进自动形式化：迈向忠实度指标与基于依赖项检索的方法》。2025年。

Alex Meiburg、Leonardo ALessa 和 Rodolfo R Soldati。《广义量子斯坦因引理在 Lean 中的形式化》。《arXiv预印本 arXiv:2510.08672》，2025年。

Leonardo de Moura 和 Sebastian Ullrich。《Lean 4 定理证明器与编程语言》。载于《自动演绎——*CADE 28*：第 *28* 届国际自动演绎会议，线上活动，*2021*年*7*月*12*日至*15*日，论文集》，第 625–635 页，柏林、海德堡，2021年7月。Springer 出版社。国际标准书号 978-3-030-79875-8。数字对象标识符：10.1007/978-3-030-79876-5 37。统一资源定位符：https://doi. org/10.1007/978-3-030-79876-5_37。

Logan Murphy、Kaiyu Yang、Jialiang Sun、Zhaoyu Li、Anima Anandkumar 与 Xujie Si。欧几里得几何的自动形式化，2024年5月。统一资源定位符 http://arxiv.org/abs/2405.17216。arXiv:2405.17216 [计算机科学]。

Lawrence C. Paulson（编）。*Isabelle*，《计算机科学讲义》第828卷。Springer-Verlag，柏林/海德堡，1994年。国际标准书号 978-3-540-58244-1。数字对象标识符：10.1007/BFb0030541。统一资源定位符 http://link.springer.com/10.1007/BFb0030541。

Zhongyuan Peng、Yifan Yao、Kaijing Ma、Shuyue Guo、Yizhe Li、Yichi Zhang、Chenchen Zhang、Yifan Zhang、Zhouliang Yu、Luming Li、Minghao Liu、Yihang Xia、Jiawei Shen、Yuchen Wu、Yixin Cao、Zhaoxiang Zhang、Wenhao Huang、Jiaheng Liu 与 Ge Zhang。CriticLean：面向数学形式化的评论器引导的强化学习，2025年7月。统一资源定位符 http://arxiv. org/abs/2507.06181。arXiv:2507.06181 [计算机科学]。

Christian Szegedy（编）。通向自动形式化与通用人工智能的一条有前景的路径，2020年。

Guillem Tarrach、Albert Q. Jiang、Daniel Raggi、Wenda Li 与 Mateja Jamnik。请提供更多细节：以更详细的证明改进自动形式化。2024年6月。统一资源定位符 https: //openreview.net/forum?id=AkJvzpYMvK。

Coq 开发团队。Coq 证明助手，2024年6月。统一资源定位符 https://zenodo. org/records/11551307。语言：英语。

Joseph Tooby-Smith。Heplean：高能物理数字化，2024年。统一资源定位符 https://arxiv.org/abs/2405.08863。

Ke Weng、Lun Du、Sirui Li、Wangyue Lu、Haozhe Sun、Hengyu Liu 和 Tiancheng Zhang。《大语言模型时代的自动形式化：综述》，2025年5月。统一资源定位符：http: //arxiv.org/abs/2505.23486。arXiv:2505.23486 [cs]。

Dominic J. Williamson 和 Theodore J. Yoder。《通过对逻辑算符进行规范化实现低开销容错量子计算》，2024年。统一资源定位符：https://arxiv.org/abs/2410.02213。

Yuhuai Wu、Albert Q. Jiang、Wenda Li、Markus N. Rabe、Charles Staats、Mateja Jamnik 和 Christian Szegedy。《使用大语言模型进行自动形式化》，2022年5月。统一资源定位符：http: //arxiv.org/abs/2205.12615。arXiv:2205.12615 [cs]。

Huajian Xin、Daya Guo、Zhihong Shao、Zhizhou Ren、Qihao Zhu、Bo Liu、Chong Ruan、Wenda Li 和 Xiaodan Liang。《Deepseek-prover：通过大规模合成数据推进大语言模型中的定理证明》，2024年。统一资源定位符：https://arxiv.org/abs/2405.14333。

Yichen Xu 和 Martin Odersky。《智能体证明自动化：一个案例研究》，2026年1月。统一资源定位符 Lhttp://arxiv.org/abs/2601.03768。arXiv:2601.03768 [计算机科学]。

Hanning Zhang、Ruida Wang、Rui Pan、Wenyuan Wang、Bingxu Meng 和 Tong Zhang。《PhysProver：推进物理学自动定理证明》，2026年1月。统一资源定位符 http://arxiv.org/abs/2601.15737。arXiv:2601.15737 [cs]。

Lan Zhang、Marco Valentino 和 Andre Freitas。《真实场景中的自动形式化：评估大语言模型对现实世界数学定义的处理能力》，2025年9月。统一资源定位符 http://arxiv.org/abs/2502.12065。arXiv:2502.12065 [cs]。

形式化示例

我们提供容错量子计算形式化的典型示例，以展示 MerLean 输出的质量。

A.1 定理：规范测量的容错

Williamson 和 Yoder（2024年）的定理 2（改编）：使用合适的图 G 实现算法 1 的容错方案，其时空故障距离为 $d_{\mathrm{ST}} = d$，其中 d 是原始码的距离。具体而言，若 (1) 切格常数满足 $h(G) \geq 1$，且 (2) 测量轮次满足 $t_o - t_i \geq d$，那么任意时空逻辑故障的权重至少为 d。

```
/-- 主定理（容错）：时空故障距离等于 d。在以下条件下：1. h(G) >= 1 （切格常数至少为 1）2. (t_o - t_i) >= d（测量轮次数量充足）时空故障距离 d_ST 恰好等于 d。 -/ theorem FaultToleranceTheorem[Fintype V] [Fintype E] [Fintype M] [Nonempty M](DC : 检测器集合 V E M) (基础结果 : 结果赋值 M) (逻辑效应 : 时空故障 V E M -> Prop) (cfg : 容错配置) (h_all_decompose : forall F, IsSpacetimeLogicalFaultDC 基础结果 逻辑效应 F ->LowerBoundCase DC 基础结果 逻辑效应 cfgF) (h_exists_d : exists F, IsSpacetimeLogicalFaultDC 基础结果 逻辑效应 F /\ F.weight (intervalRounds cfg) = cfg.d) : spacetimeFaultDistance DC 基础结果 逻辑效应 (intervalRounds cfg) = cfg.d := by
```

```
-- 获取权重为 d 的逻辑故障以证明其存在
获取 <F_d, hF_d_log, hF_d_权重> : = h_存在_d
-- 上界: d_ST <=d
有 h_le : 时空故障距离 DC 基础结果
    逻辑效应（区间轮次 cfg） <= cfg.d: = 通过
  计算 时空故障距离 DC 基础结果
      逻辑效应（intervalRounds cfg）
    <= F_d.权重 (区间轮次 cfg) :=
      时空故障距离_le_权重 DC 基础结果
        逻辑效应（intervalRounds cfg）F_d hF_d_log
    _ = cfg.d:=hF_d_权重
-- 下界: d          _ST >= d
存在 h_ge : 时空故障距离 DC 基础结果
    逻辑效应（区间轮次 cfg） >= cfg.d := 根据
  -- 获取达到最小值的故障
  存在 h_has : hasLogicalFault DC 基础结果
    逻辑效应 := <F_d, hF_d_log>
  获得 <F_最小值，hF_最小值_对数，hF_最小值_权重> : =
    时空故障距离_为_min DC 基础结果
      逻辑效应 (区间轮次 cfg) h_具有
  -- 将下界应用于 F_最小值
  有 h_最小值_大于等于: = 时空故障距离_大于等于_d DC
    基础结果 逻辑效应 cfg F_最小值 hF_最小值_对数
    (h_全部_分解 F_min hF_min_log)
  计算 cfg.d
      <= F_min.权重 (intervalRounds cfg) := h_min_ge
    _ = 时空故障距离 DC 基础结果
      逻辑效应 (intervalRounds cfg) := hF_min_权重
-- 合并
欧米伽
```

A.2 智能体发现的辅助引理

该引理是 Williamson 与 Yoder（2024）形式化工作的一部分，具体讨论空间距离下界。智能体自行引入了这个中间引理，以完成容错量子计算形式化中的空间距离下界证明。给定一个类 Cheeger 扩张参数 $h(G)$、一个大小至多为清理后算子权重的清理集，以及一个满足等周不等式的边界，该引理证明：

$$w' - |S| + |\partial S| \;\geq\; \min(h(G),\,1)\cdot d,$$

其中 w' 是清理后权重，$|S|$ 是清理集大小，$|\partial S|$ 是边界大小，d 是码距。证明根据 $h(G) \geq 1$ 是否成立进行分类，并在每个分支中使用 nlinarith 构造 calc 链。

```
定理 weight_不等式_核心 (hG : R) (hG_非负 : 0 <= hG) (d : N) (_hd_正 : 0 < d)
(cleanedWeight : N) (hCleaned : cleanedWeight >= d) (cleaningSetSize : N)
(hCleaningBound : cleaningSetSize <= cleanedWeight) (boundarySize : N)
(hCheeger : (boundarySize : R) >=hG * cleaningSetSize) : (cleanedWeight : R) -
cleaningSetSize + boundarySize>=minCheegerOne hG * d :=by simp only [
minCheegerOne ]by_分情况讨论 hG_ge_1 : hG >= 1. -- 情况 h(G) >= 1:
boundarySize >= cleaningSetSize，因此结果>= cleanedWeight >= d
```

```
      rw [min_eq_right hG_ge_1]have hBound : (boundarySize : R) >=
cleaningSetSize := by calc (boundarySize : R) >= hG * cleaningSetSize :=
hCheeger_ >= 1 * cleaningSetSize :=by nlinarith_ = cleaningSetSize :=
one_mul _have hCleaned' : (cleanedWeight : R) >= d :=Nat.cast_le.mpr
hCleaned linarith . -- Case h(G) < 1push_neg at hG_ge_1rw [min_eq_left
(le_of_lt hG_ge_1)]have hClean : (cleanedWeight : R) >= d :=Nat.cast_le.
mpr hCleaned have hS' : (cleaningSetSize : R) <= cleanedWeight :=Nat.cast_
le.mpr hCleaningBound calc (cleanedWeight : R) - cleaningSetSize +
boundarySize>= cleanedWeight - cleaningSetSize+ hG * cleaningSetSize :=
by linarith_ = cleanedWeight+ (hG - 1) *cleaningSetSize := by ring_ >=
cleanedWeight+ (hG - 1) *cleanedWeight := by nlinarith_ = hG *
cleanedWeight := by ring_ >= hG *d := by nlinarith
```

A.3 公理示例：K¨unneth 公式

此示例取自 Breuckmann & Eberhardt (2021) 的形式化工作，其中 K¨unneth 公式对于计算平衡乘积码的同调至关重要。K¨unneth 公式是同调代数中的一项基本结果，它将两个链复形的张量积的同调与各自同调的张量积联系起来。给定 $\mathbb{F}_2$ 上的链复形 $C = (C_\bullet, \partial^C)$ 和 $D = (D_\bullet, \partial^D)$，该公式表明：

$$H_n(C \otimes D) \;\cong\; \bigoplus_{p+q=n} H_p(C) \otimes H_q(D).$$

在诸如 $\mathbb{F}_2$ 这样的域上，所有模都是平坦的，因此 Tor 修正项消失，K¨unneth 映射是同构。该同构将一对同调类 $[z] \in H_p(C)$ 和 $[w] \in H_q(D)$ 映射到类 $[z \otimes w] \in H_{p+q}(C \otimes D)$。在 Lean 4 中证明这一点需要：(i) 证明循环的张量积是一个循环；(ii) 证明边界得到保留，因此该映射可下降到同调；(iii) 确立双射性。步骤 (i)–(iii) 涉及展开 Mathlib 全复形的内部余极限结构，目前尚不支持；这些内容已被公理化。完整的形式化如下所示。

下面的 4 个片段对应类型的定义，接着是三个证明步骤 (i)–(iii)。

首先，我们定义源类型（同调的张量积的直和）和目标类型（张量积的同调）：

```
变量 (C D : F₂链复形)

/-- 所有 F₂-模都是平坦的（自由模都是平坦的）。 -/实例 平坦_F2_模 (M :
类型*)  [加法交换群 M][模 F₂ M] : 模.平坦 F₂ M :=模.平坦.由 自由_


/-- 同调空间的张量积。 -/
不可计算的 定义 同调张量积 (p q : Z) : 类型 :=
  (C.同调 p)            (x) [F2]  (D.同调 q)
```

```
...

/-- 指标集：满足 p + q = n 的有序对 (p,q)。 -/
def Künneth 指标 (n : Z) : Type :=
  { pq : Z x Z // pq.1 + pq.2 = n }

/-- 直和 (+)_{p+q=n} H_p(C) (x) H_q(D)。 -/
noncomputable def Künneth 直和 (n : Z) : Type :=
  Pi_0 (i : Künneth 指标 n)          ,
    同调张量积 C D i.val.1 i.val.2

...

/-- 张量积复形在次数 n 处的同调。 -/
不可计算的缩写 TensorHomology (n : Z) : 类型 :=
  (TensorProductComplex C D).Homology n
```

步骤（*i*）：构造循环上的映射。我们证明，循环的张量积仍是一个圈，从而可以在循环上定义交叉积：

```
/-- 从 C_p (x) D_q 到 (C (x) D)_{p+q} 的映射。 -/
不可计算的 定义 tensorInclusion (p q : Z) :
    (C.X p) (x) [F2] (D.X q) ->l [F2]
      (张量积复形 C D).X (p + q) :=
  (TensorProductComplex.i C D p q).hom

/-- 公理：当 z 和 w 都是循环时，z (x) w 是一个圈。
d(z (x) w) = dz (x) w +- z (x) dw = 0。          -/
公理 循环_张量_循环_是_循环'_辅助
    (C D : F₂链复形) (p q : Z)
    (z : C.X p) (hz : z in C.循环 p)
    (w : D.X q) (hw : w 属于 D 的循环 q) :
    tensorInclusion C D p q (z (xt) w) 属于
      (TensorProductComplex C D).循环 (p + q)

/-- 从 C.Cycles p (x) D.Cycles q 到全循环的映射。 -/
不可计算 定义 cyclesCrossProduct (p q : Z) :
    (C.循环 p) (x) [F2] (D 的循环 q) ->l [F2]
      (张量积复形 C D).循环 (p + q) := 通过
  ...  -- 将圈提升的双线性映射_张量_圈_是_圈'
```

步骤 *(ii)*：下降到同调。我们公理化了边界在交叉积下保持不变这一性质，从而确保该映射下降为同调上的良定义映射：

```
/-- 公理：边界 (x) 圈是一个边界。若 z = dz'，则 z (x) w = d(z' (x) w),
因为 dw = 0。 -/公理 边界_张量_圈_是_边界'_辅助 (C D : F₂链复形)
(p q : Z) (z : C.Cycles p) (hz : z.val 属于 C 的边界 p) (w : D.Cycles q) : (循
环交叉积 C Dp q (z (xt) w)).val 属于 (张量积复形 CD).Boundaries (p +
q)/-- 公理：圈 (x) 边界是一个边界。若 w = dw'，则 z (x) w = d(z (x)
w')，因为 dz = 0。 -/公理 圈_张量_边界_是_边界'_辅助 (C D : F₂链复
形) (p q : Z) (z : C.Cycles p)
```

```
        (w : D.Cycles q) (hw : w.val 属于 D 的边界 q) : (循环交叉积 C D p
        q (z (xt) w)).val 属于 (张量积复形 C D).Boundaries (p + q)
```

步骤（三）：双射性与主定理。最后，我们构造 K¨unneth 映射，并将其双射性公理化，以建立同构：

```
...

/-- 交叉积下降到同调：⟦z⟧ (x) ⟦w⟧ ↦ ⟦z (x) w⟧。
    -/
非计算性 定义 Künneth 分量映射 (p q : Z) :
    同调张量积 C D p q →l[F2]
      张量同调 C D (p + q) :=
  张量积提升 (kunnethComponentMapAux2 C D p q)

/-- 从 (+)_{p+q=n} H_p(C) (x) H_q(D) 出发的 Künneth 映射
到 H_n(C (x) D)。    -/
非计算 定义 kunnethMap (n : Z) :
    Künneth 直和 C D n →l[F2]
      张量同调 C D n :=通过
  细化依赖有限支撑函数.lsum N ?_
  引言 ⟨p, q⟩,    hpq⟩
  subst hpq
  exact Künneth 分量映射 C D p q

/-- 公理：Künneth 映射在 F₂ 上是单射。 -/
公理 Künneth 映射_单射_aux
    (C D : F₂链复形)              (n : Z) :
    Function.单射 (Künneth 映射 C D n)

/-- 公理：Künneth 映射在 F₂ 上是满射。 -/
公理 Künneth 映射_满射_aux
    (C D : F₂链复形)              (n : Z) :
    Function.满射 (Künneth 映射 C D n)

/-- Künneth 公式：
(+)_{p+q=n} H_p(C) (x) H_q(D)    ≃ H_n(C (x) D). -/
不可计算的 定义 kunnethEquiv (n : Z) :
    Künneth 直和 C D n          ≃l[F2]
      张量同调 C D n :=
  线性等价.由双射构造 (Künneth 映射 C D n)
    ⟨Künneth 映射_单射 C D n,
     Künneth 映射_满射 C D n⟩
```