QuMeld：用于评测量子比特映射算法的模块化框架

Gabrielius Keibas计算机科学研究所 维尔纽斯大学维尔纽斯，立陶宛
gabrielius.keibas@mif.stud.vu.lt

Linas Petkevi˘cius计算机科学研究所 维尔纽斯大学维尔纽斯，立陶宛
linas.petkevicius@mif.vu.lt

arXiv:2603.01578v1 [quant-ph] 2 Mar 2026

摘要—**量子比特映射问题是量子计算中的一项挑战，涉及将逻辑量子比特映射到量子计算机上的物理量子比特。由于量子计算机拓扑和电路种类繁多，已有许多解决这一问题的方法。针对特定拓扑与电路组合寻找最佳方案仍然十分困难，目前也尚无统一框架，能够系统地评估和比较不同场景下的量子比特映射算法。本文介绍 QuMeld，这是一个旨在解决这一问题的开源框架。该框架目前支持六种量子比特映射算法、十六种量子计算机拓扑以及多种评估指标。框架采用模块化设计，可集成新的映射算法、量子电路、硬件拓扑和评估指标，从而确保其可扩展性，并能适应未来的发展。**

索引词—**量子计算、量子比特映射、量子比特映射问题、量子电路优化、基准测试框架，**

I. 引言

量子计算是一项快速发展的技术，应用于化学 [3], 优化 [7], 机器学习[2]。当前一代量子计算机被称为含噪声中等规模量子（NISQ）时代 [17]。这些设备拥有 50–300 个量子比特，但会受到显著噪声和量子比特间连接性有限的影响。这些物理限制带来了一个挑战：量子算法通常假设量子比特拓扑理想且完全连接。然而，真实量子计算机的拓扑结构施加了严格的连通性约束，每个量子比特只能与其邻近的部分量子比特直接交互。

A. 量子比特映射问题

在物理量子计算机上执行多量子比特量子门时，会出现 量子比特映射问题。当量子算法需要在硬件拓扑中未物理连接的量子比特之间执行双量子比特门（例如受控非门）时，必须通过插入额外的量子门，在相邻量子比特之间移动量子态，从而转换电路。这种转换必须在保证算法正确性的同时，尽量降低门数量、电路深度和执行时间方面的开销。这些因素都会直接影响 NISQ 时代量子计算的保真度。

形式化地，给定一个包含 n 个逻辑量子比特的量子电路 C，以及一个耦合图为 $G = (V, E)$的量子计算机其中 $|V| \geq n$，量子比特映射问题旨在求得一个映射 π：$\{0, \ldots, n-1\} \to V$，以及一系列 SWAP门，使得 C 中的所有双量子比特门都能在 E 的边上执行，同时优化如下指标：

- SWAP次数
- 双量子比特门数量
- 电路深度
- 执行时间

B. 量子比特映射算法的涌现

量子比特映射问题的复杂性和重要性促使人们提出了大量解决方案，各有不同的优势与权衡：

- 启发式搜索 - SABRE [14] 及其变体（LightSABRE [22]）采用带前瞻的双向启发式搜索，以贪心方式选择 SWAP 放置。
- 综合优先 - Rustiq [5] 采用针对基于泡利的量子算法进行优化的 Clifford 电路综合。
- 时间感知路由 -Doustra [4] 使用双源最短路径算法，同时考虑门操作时序和量子比特占用。
- 泡利字符串综合 - PauliForest [21] 利用特征矩阵，根据泡利字符串之间的相互作用映射量子比特。
- 机器学习 - Qiskit AI [6], [13]应用强化学习来预测最优转译策略。

尽管方法多种多样，但针对给定的量子电路、硬件拓扑结构和优化目标组合，选择合适的算法仍然颇具挑战。要选出最合适的算法，需要分析所选量子计算机的拓扑结构和量子电路。

C. QuMeld 的研究动机

在典型用例中，用户通常依赖量子框架提供的默认转译方法。然而，这可能导致意料之外的结果，因为转译后的电路可能比原始电路包含多得多的量子门。因此，研究人员需要自行找出最佳算法。由于评估标准不一致、实现新的映射算法颇具难度，而且需要深入了解量子框架本身，这项任务可能颇具挑战。为解决这些问题，我们创建了 QuMeld 框架，旨在提供系统化的

量子比特映射算法评估与比较。该框架使从业者能够在尝试实际计算之前，研究并选择更合适的拓扑[12]。

QuMeld 提供：

- 模块化架构——清晰划分算法、电路、拓扑和评估指标
- 易于扩展——为添加新组件提供定义明确的接口
- 全面覆盖——支持 6 种先进算法、16 种硬件拓扑和 6 个基准电路
- 自动化评估——在所有实验中统一收集指标
- 开源——可在 https://github.com/gabkeib/ qumeld 获取

本文其余部分安排如下：第二节讨论相关工作；第三节介绍 QuMeld 的架构与设计；第四节描述该框架的功能；第五节总结全文并展望未来方向。

II. 相关工作

A. 量子编译框架

若干全面的量子计算框架都提供电路优化功能：

Qiskit [18] 是 IBM 的开源量子计算框架。它包含一个先进的转译器，提供多个优化级别（0–3），并支持多种路由算法，包括 SABRE 和人工智能转译器。该转译器与 IBM Quantum 硬件紧密集成，能够出色地支持在真实设备上执行电路。然而，Qiskit 可选的算法可能不够丰富，也不会针对特定配置自动推荐最佳算法。

Cirq 和 tket 提供与 Qiskit 类似的功能。Cirq 专注于 Google 的量子硬件和基于网格的拓扑结构，而 tket 则强调硬件无关优化，并支持多个后端。这两个框架都具备充分的功能来准备待执行的电路，但缺乏用于系统化算法比较的专用基础设施。

MQTQMAP [20] 使用 SAT/SMT 求解器为小型电路（$\leq$ 8 个量子比特）寻找可证明的最优映射。虽然这能提供有价值的最优解作为基准测试，但 NP 难复杂度限制了其在拥有 100+ 个量子比特的现代量子计算机上的可扩展性。

B. 专用映射工具

有几种工具专门用于解决量子比特映射问题：

QMAP 和相关的最优映射工具 [16]将电路划分为子架构，以寻找最优解。这些方法有助于理解理论界限

但在大规模量子计算机上面临可扩展性挑战。

针对特定量子算法的专用优化器已相继出现。AOQMAP [9] 面向 QAOA电路

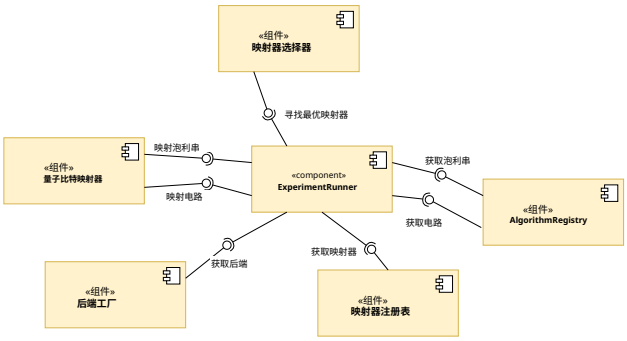

图 1. QuMeld 高层组件图，展示映射器、电路、拓扑与结果之间的交互

采用铃木-特罗特分解，在大幅减少门数量的同时，也需要特定硬件模板。类似地，各种经变分量子本征求解器优化的映射器 [10] 利用了变分算法的结构。

C. 机器学习方法

近期工作已将机器学习应用于量子比特映射：

GNAQC [19]使用图神经网络预测高质量初始映射，相较随机初始化将保真度提高了 12.7%。SA-DQN [15] 将深度 Q 网络与模拟退火相结合。尽管前景可期，这些方法耗时更长，需要训练及用于训练的数据，并需恰当选择超参数。

D. QuMeld 的定位

QuMeld 被设计为面向量子比特映射算法的基准测试框架。虽然 Qiskit、Cirq 等平台主要面向电路编译和硬件部署，QuMeld 则侧重于对量子比特映射算法进行系统评估与比较分析。各类算法可通过统一接口访问，并提供标准化基准。借助该工具，不仅可以进行基准测试，还可以选出最佳算法。凭借其模块化架构，该框架支持集成新的算法和硬件拓扑。

III. QUMELD 架构与设计

A. 高层架构

QuMeld 高层组件图如图 1 所示。ExperimentRunner 是核心协调器。该设计最大限度地降低了组件间的耦合，并为实验提供了统一的协调点。

B. 核心组件

1) QubitMapper 接口：QubitMapper 抽象基类定义了两个关键方法：

- run_circuit(circuit, backend)：将通用的 QuantumCircuit 映射到 backend 定义的拓扑结构。

- run_hamiltonian(pauli_strings, backend)：直接映射泡利字符串列表（可用于变分量子本征求解器或量子近似优化算法）。

这种双重接口既适用于基于电路的算法，也适用于支持Pauli字符串综合的算法。算法根据自身能力实现其中一种或两种方法。

每个映射器都会返回一个电路优化结果，其中包含：

- 映射后的电路
- SWAP次数
- CNOT门数量
- 电路深度
- 优化时间
- 失败原因（如适用）

2）映射器注册表：映射器注册表会自动发现并加载 mappers/ 包中的可用映射算法。这种注册表模式支持动态加载算法，无需在核心框架中硬编码算法名称。

3）映射器选择器：映射器选择器通过以下方式，为给定的电路和拓扑结构确定最优算法：

1）从映射器注册表中检索所有可用的映射器 2）在输入电路/拓扑结构上运行每个映射器 3）根据用户指定的标准（SWAP次数、深度等）比较结果 4）返回性能最佳的映射器

4）算法提供器和哈密顿量提供器：这些提供器组件提供用于测试的量子电路和泡利字符串：

- 算法提供器：从框架的电路库中加载预定义电路（变分量子本征求解器、量子近似优化算法变体）
- 哈密顿量提供器：提供从 JSON 文件加载或以编程方式生成的泡利字符串表示（例如，QAOA 最大割哈密顿量）

提供器模式将数据源与算法实现解耦，因此可以轻松添加新的基准测试电路。

5) 后端工厂：后端工厂创建代表量子计算机拓扑的 Qiskit BackendV2 实例。每个后端都指定：

- 耦合图（量子比特连通性）
- 支持的门集（目前所有后端都设置为相同的门集）
- 模拟噪声特性（T1/T2 时间、门错误）

工厂模式将拓扑结构的创建过程抽象化，用户只需按名称指定拓扑结构（例如“google willow”“ibm eagle”），无需关心具体实现细节。

6) ExperimentRunner：ExperimentRunner 负责协调端到端实验：

1) 加载实验配置（算法、拓扑、电路）2) 对每个（算法、拓扑结构、电路）组合：a) 从算法提供器检索电路 b) 从后端工厂创建后端 c) 从映射器注册表加载映射器 d) 执行映射 e) 收集指标 3) 将结果存储到带时间戳的目录中

C. 实现细节

语言和依赖项：QuMeld 使用 Python 3.13+ 实现，并以 Qiskit 2.1.1 作为主要量子计算库。uv 包管理器负责处理依赖项，ruff 代码检查工具则确保类型安全。

算法集成策略：不同算法需要采用不同的集成方法：

- 原生 Python——LightSABRE（衰减和前瞻启发式）、Rustiq 和 Qiskit AI 通过 Qiskit 的通道管理器系统直接集成。
- 子进程（C++）——Doustra 作为 Qsyn 框架的一部分以 C++ 实现。QuMeld 通过 Python 的子进程模块调用 Doustra，并将电路转换为 OpenQASM 格式进行通信。
- 隔离环境——Qiskit AI 需要 networkx==2.8.5，与 QuMeld 的依赖项冲突。它在独立的虚拟环境中运行，通过序列化电路文件进行通信。

IV. 框架功能

A. 支持的量子比特映射算法

QuMeld 目前集成了六种先进的量子比特映射算法：

1）*LightSABRE*（衰减启发式和前瞻启发式）：SABRE [14] 是一种双向启发式搜索算法，被广泛视为业界先进算法。LightSABRE [22]通过优化启发式方法并采用 Rust 实现，对 SABRE 进行了改进。QuMeld 包含两种 LightSABRE 配置：

1）前瞻启发式：选择交换操作时评估未来的门依赖关系。2）衰减启发式：根据交换概率动态调整量子比特的权重

两者均使用自定义的 Qiskit 通道管理器，并配置 SabreLayout 和 SabreSwap 通道，以获得高质量结果（200 次布局试验、200 次交换试验、4 次前向-后向迭代）。

2）*Rustiq*：Rustiq [5] 专注于面向基于泡利算符的算法的综合优先编译。它使用贪心启发式和最大权匹配构造经优化的 Clifford 电路，以优化门数量或电路深度。QuMeld 通过 Qiskit 的高级综合框架集成 Rustiq，并将其配置为进行深度优化，执行 400 次洗牌。

3) Doustra：Doustra [4] 将用于初始映射的深度优先搜索与同时考虑距离和量子比特占用时间的双源 Dijkstra 路由相结合。这种时间感知方法可减少门碰撞并提高并行性。QuMeld 通过子进程调用 Doustra 的 C++ 实现，并自动处理电路转换和结果解析。

4) PauliForest：PauliForest [21] 是一种专为泡利字符串算符设计的静态映射算法。它根据泡利字符串构建特征矩阵，识别高交互量子比特，并将其映射到中心拓扑位置，以最小化平均距离。QuMeld 集成了 PauliGo 实现，将泡利字符串转换为所需的块格式，并应用综合后 Qiskit 优化。

5) Qiskit AI：Qiskit AI 驱动的转译器 [6], [13]利用强化学习来预测最优转译策略。它将电路划分为多个块，使用基于大型数据集训练的人工智能模型优化每个块，然后重新组合结果。QuMeld 将 AIRouting pass 与优化级别 3 和优化后的布局方法相结合。

B. 支持的量子计算机拓扑

QuMeld 支持 16 种不同的量子计算机拓扑，涵盖不同的连接性模式和量子比特数量，包括多种 IBM 拓扑、Google Willow、IonQ、Rigetti Novera 和 Riken Fujitsu。

- IBM 重六边形（IBM 标准）：Almaden（20 个量子比特）、Cambridge（28 个量子比特）、Falcon（27 个量子比特）、Montreal（27 个量子比特）、Paughkeepsie（20 个量子比特）、Reuschlikon（16 个量子比特）、Tokyo（20 个量子比特）、Manhattan（65 个量子比特）、Rochester（53 个量子比特）、Eagle（127 个量子比特）、Heron（133 个量子比特）
- Google Willow（105 个量子比特）：Google 最近推出的量子处理器，具有经过优化的连接性和纠错能力
- IonQ Harmony（9 个量子比特）：全连接的囚禁离子拓扑结构（边密度 = 1.0）
- Rigetti Novera（9 个量子比特）：方格晶格拓扑（平均度数 ≈ 3.6）
- Riken Fujitsu（256 个量子比特）：大规模超导量子计算机
- 六边形晶格（54 个量子比特）：合成生成的六边形网格

所有拓扑均以 Qiskit BackendV2 实例实现，并具有随机化的噪声特性：T1 弛豫时间 $\sim \mathcal{U}(50,100)$ μs、T2 退相干时间 $\sim \mathcal{U}(20,80)$ μs、量子比特频率 $\sim \mathcal{U}(4.5,5.5)$ GHz，以及符合实际情况的量子门错误率。

C. 基准量子电路

QuMeld 包含六个基准电路，代表常见的量子计算工作负载：

1）变分量子本征求解器电路：

- 氢$_2$ 分子（4 个量子比特，30 个双量子比特门）：小型分子哈密顿量
- 氢化锂分子（12 个量子比特，6023 个双量子比特门）：来自 PauliForest 基准测试的大规模分子哈密顿量
- 随机示例（9 个量子比特，60 个双量子比特门）：随机生成的泡利字符串
- EfficientSU2 试探态（5个量子比特，4个双量子比特量子门）：具有线性纠缠的参数化变分试探态*2*）*QAOA*电路：
- 最大割哈密顿量（5 个量子比特，6 个双量子比特门）：图划分问题哈密顿量
- QAOA 试探态（5 个量子比特，6 个双量子比特门）：混合算子 +问题哈密顿量

所有基于泡利算符的电路都以包含系数的 JSON 文件形式存储，以确保不同算法之间的评估保持一致。

D. 结果存储与分析

结果存储在具有分层结构的带时间戳的目录中：

r esults/<时间戳>/<拓扑结构_名称>/<电路_名称>/<映射器_名称>.json

每个 JSON 文件都包含所有指标，便于进行系统化后处理。QuMeld 提供 Jupyter 笔记本，用于结果加载、表格生成和 CSV 导出。

E. 可扩展性

向 QuMeld 添加新组件非常简单：

- 新映射器——实现 QubitMapper 接口，并将其放入 mappers/ 目录。MapperRegistry 会自动发现它。
- 新拓扑结构——在 BackendFactory 中定义耦合图函数，并在工厂中注册。
- 新电路——将电路生成函数或泡利字符串 JSON 文件放入相应的提供程序目录。
- 新指标——扩展 CircuitOptimisationResult，并在映射器基类中添加计算逻辑。

V. 结论与未来工作

我们介绍了 QuMeld，这是一个用于对量子比特映射算法进行基准测试的开源模块化框架。QuMeld 通过提供一个统一平台，在不同拓扑和电路上系统地评估多种映射算法，弥补了量子计算研究中的一个关键空白。该框架的星形架构、定义明确的接口和设计模式，能够轻松集成新的算法、拓扑和评估指标。

QuMeld 目前支持六种先进算法（LightSABRE 变体、Rustiq、Doustra、PauliForest、Qiskit AI）、十六种量子计算机拓扑（9—256 个量子比特）以及六种代表 VQE 和量子近似优化算法工作负载的基准电路。该框架的模块化设计支持多种实现语言（Python、C++、隔离环境）和评估方法。

A. 可用性与用例

QuMeld 以开源许可证免费提供，网址为 https://github.com/gabkeib/qumeld。研究人员可以使用 QuMeld 来：

- 算法选择 - 为特定电路/拓扑组合确定最佳映射器
- 算法开发——将新的映射算法与最先进的基准方法进行对比评估

- 拓扑结构分析——了解硬件连接模式如何影响映射质量
- 可复现研究——在不同研究中生成一致且可比较的结果

B. 未来工作

未来发展的几个方向包括：

1) 集成新的优化算法，包括 EffectiveQM [1], S Win [8], 和禁忌搜索 [11]。2) 支持纠错并将其纳入比较指标。3) 改进自动算法选择，通过使用先进的决策树或机器学习方法来预测最优算法。4) 为待测试算法添加规模更大的电路，以获得更加多样化的结果。5) 性能优化。通过并行化实验执行来优化框架开销

参考文献

[1] C. Luo、S. Cao、S. Guo 和 C. Hu。《面向量子比特映射的有效局部搜索》。*IEEE Transactions on Computers*，2025年。[2] Jianshu Cao、Richard J Cogdell、David F Coker、Hong-Guang Duan 等。《量子生物学再探》。*Science Advances*，6 4888，2020年 14( ):eaaz 。[3] Yudong Cao、Jonathan Romero、Jonathan P Olson、Matthias Degroote 等。《量子计算时代的量子化学》。*Chemical reviews*，119(19):10856–10915，2019年。[4] CY Cheng、CY Yang、YH Kuo、RC Wang、HC Cheng 和 CY Huang。《大型量子电路上通过双源最优路由实现的鲁棒量子比特映射算法》。*ACM TRANSACTIONS ON QUANTUM COMPUTING*，5(3)，2024年9月。[5] Timoth´ee Goubault de Brugi`ere 和 Simon Martiel。《更快速、更精简的哈密顿量模拟电路综合》，2024年。[6] Ayushi Dubal、David Kremer、Simon Martiel、Victor Villar、Derek Wang 和 Juan Cruz-Benito。《利用强化学习进行泡利网络电路综合》，2025年。[7] Edward Farhi、Jeffrey Goldstone 和 Sam Gutmann。《量子近似优化算法》，2014年。[8] H Fu 等。《一种有效的量子比特映射滑动窗口方法》。载于*IEEE International Conference on Computer Design*，2023年。[9] YJ Ji、KF Koenig 和 I Polian。《通过面向算法的量子比特映射提升数字化反绝热量子优化的性能》。*PHYSICAL REVIEW A*，110(3)，2024年9月。[10] Yuhang Ji 等。《面向变分量子算法的面向算法的量子比特映射》。*arXiv*预印本，2024年。[11] H Jiang、YX Deng 和 M Xu。《基于禁忌搜索的量子比特映射》。*JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY*，39(2)，2024年3月。[12] Gabrielius Keibas 和 Linas Petkevi˘cius。《根据量子计算机拓扑优化量子算法方案》。载于*2025 IEEE*第*12*届信息、电子与电气工程进展研讨会 *(AIEEE)*，第1–6页。IEEE，2025年。[13] David Kremer、Victor Villar、Hanhee Paik、Ivan Duran、Ismael Faro 和 Juan Cruz-Benito。《利用强化学习实现实用且高效的量子电路综合与转译》，2025年。[14] Gushu Li、Yufei Ding 和 Yuan Xie。《解决NISQ时代量子设备的量子比特映射问题》。载于第二十四届编程语言与操作系统架构支持国际会议论文集，第1001–1014页，2019年。[15] TF Li、WB Liu 和 F Wang。《基于深度Q网络的量子比特映射模型优化》。载于*2024*年第五届计算机工程与应用国际会议，2024年。[16] Tom Peham 等。《量子电路映射的最优子架构》。*ACM Transactions on Quantum Computing*，2023年。[17] John Preskill。《NISQ时代及之后的量子计算》。*Quantum*，2:79，2018年8月。

[18] Qiskit 贡献者。《Qiskit：用于量子计算的开源框架》。https://github.com/Qiskit/qiskit，2021年。[19] T. LeCompte、F. Qi、X. Yuan、N. -F. Tzeng、M. H. Najafi 和 L. Peng。《用于降低量子电路误差的基于机器学习的量子比特分配》。*IEEE*量子工程汇刊，4，2023年。[20] Robert Wille 和 Lukas Burgholzer。《MQT QMAP：高效的量子电路映射》。国际物理设计研讨会，2023年。[21] Y. Li、Y. Zhang、H. Deng、M. Chen 和 Z. Li。《PauliForest：面向近期量子模拟的连接性感知综合与面向泡利的量子比特映射》。*IEEE*集成电路与系统计算机辅助设计汇刊，2024年。[22] Henry Zou、Matthew Treinish、Kevin Hartman、Alexander Ivrii 和 Jake Lishman。《LightSABRE：轻量且增强的SABRE算法》，2024年。