DC-MBQC：基于测量的量子计算分布式编译框架

Yecheng Xue[1]，Rui Yang[1],Zhiding Liang[2],和 Tongyang Li[1]*北京大学计算机科学学院，中国北京[2]香港中文大学计算机科学与工程学系，中国香港 电子邮件： {mycts, ypyangrui, tongyangli}@pku.edu.cn, zliang@cse.cuhk.edu.hk

arXiv:2601.00214v1 [quant-ph] 1 Jan 2026

摘要——分布式量子计算（分布式量子计算）是一种扩展量子系统规模的有前景技术。尽管量子电路模型的分布式量子计算已取得显著进展，但针对基于测量的量子计算（测量基量子计算）的研究仍少得多。基于测量的量子计算是一种与电路模型本质不同、尤其适用于光子量子平台的通用量子计算模型。本文提出 DC-MBQC，首个专为测量基量子计算设计的分布式量子编译框架。我们发现并解决了实现基于测量的量子计算分布式量子计算的两项关键挑战。首先，在量子处理单元（量子处理单元）之间分配任务时，我们开发了一种自适应图划分算法，在平衡各量子处理单元工作负载的同时保留图态结构。其次，针对量子处理单元间通信，我们提出层调度问题，并设计了一种求解算法。考虑到现实的硬件要求，我们优化了运行量子程序的执行时间及相应的所需光子寿命，以避免光子损失导致的灾难性故障。实验结果表明，所需光子寿命提升了 $7.46\times$ ，使用 8 个全连接量子处理器可实现 $6.82\times$ 的加速，这进一步证实了分布式量子计算在光子系统中的优势。

源代码公开发布于 https://github.com/ qfcwj/DC-MBQC。

I. 引言

近年来，量子计算因有望加速解决因数分解 [51], 无结构数据库搜索 [26],以及许多其他具有实际意义的问题 [19]而受到广泛关注。随着领域不断发展，从含噪声中等规模量子时代迈向早期容错阶段 [46], ，多种技术路线上的量子硬件不断进步，错误率也有所降低，包括超导 [2], [23], 囚禁离子系统 [35], [38], 中性原子系统 [10], 和光子芯片 [1], [47]。现阶段，尚不清楚哪种物理平台最终能够胜任容错量子计算（FTQC）：在这一阶段，数千个逻辑量子比特将在数百万个物理量子比特之上，同时运行算法逻辑和量子纠错协议 [25]。上述所有平台最终都需要采用分布式量子计算（DQC） [15] ，才能扩展至容错运行阶段；届时，模块化量子处理单元（QPU）将通过光纤互连 [36]。

*通讯作者。

表 I 分布式纠缠生成综述（未经蒸馏）。 *表示保真度是根据后选择结果估算的，可能会被高估。

平台	保真度	时钟速度	Exp.
超导 [33]	79.3%	~ MHz	✓
量子点 [54]	61。6%	7.3 kHz	✓
囚禁离子 [36]	86.1%	9.7 Hz	✓
囚禁离子 [53]	94。0%	182 赫兹	✓
中性原子 [50]	98。7%*	30 赫兹	✓
中性原子 [34]	~ 99.9%	~ 100 千赫兹	✗
光子	99.72%* [47]	~ 兆赫兹 [1]	✓

分布式量子计算的可扩展性取决于两个重要因素：远程纠缠生成的保真度和相应的时钟速度。远程贝尔对与局部幺正操作相结合，是在分布式量子计算中实现任意远程双量子比特门（包括受控非门和交换门）的关键机制。然而，这一过程本质上容易出错，通常会引入比局部操作更高的有效错误率。据 [52], 估计，为了维持分布式量子计算中量子纠错协议的有效性，远程纠缠保真度需超过 90 %，时钟速度需达到 MHz 级。达到这两个阈值后，随着量子纠错码距离增大，含噪远程纠缠门产生的错误可以得到抑制；在快速量子纠错周期中，背景退相干误差仍可忽略不计，相较于其他噪声源微不足道。

我们在表 I 中总结了近期展示在两个量子处理单元之间生成远程纠缠这一能力的进展。与原子系统相比，超导量子比特和量子点等固态量子硬件通常具有更高的系统时钟频率，但远程门保真度较低。迄今为止，囚禁离子系统展示了最高的远程纠缠生成保真度，达到 94%，且无需后选择 [53]。文献 [36] 近期展示了一种远程受控Z门，保真度为 86.1%，但由于需要额外冷却，时钟速度仅为 9.7 Hz。中性原子系统正变得越来越受欢迎，因为

它们有望在单个量子处理单元上运行量子纠错实验。此外，还有若干旨在提升可重构原子阵列分布式量子计算能力的前景方案 [17],[34], [58] ，尤其是通过原子复用，将远程纠缠生成率从 10 Hz [50] 提升至 10^5 Hz 量级。最后，光子系统似乎是分布式量子计算最合适的候选方案，因为其远程纠缠保真度高（99.72%）、时钟速度快（∼ MHz）；PsiQuantum [47]的 Omega 芯片组和 Xanadu [1]的 Aurora 系统已展示了这一点。

尽管实现分布式量子计算具有潜在优势，但在计算机体系结构领域，针对光子系统的相关编译器设计仍未得到充分探索，即使是在单个量子处理单元的场景下也是如此。OneQ [61] 是首个面向光子硬件上的测量基量子计算（MBQC）的可扩展编译框架 [12], [49],，作者描述了将电路模型中的量子程序映射为基于光子资源态的图计算，并优化易出错的融合操作数量的多个步骤 [7]。传统上，量子程序采用电路模型设计，因为它简单且与经典逻辑电路相似。

在测量基量子计算中，更具体地说，在单向量子计算（1WQC）中 [48], ，计算通常分三个步骤实现。首先，我们制备一种称为资源态的大型纠缠态，它通常定义在图上。实际上，实验上很难制备并维持足够大规模的图态 [28] ，同时达到足够高的保真度以执行有意义的计算。更具可扩展性的方法是制备一组小型资源态，并通过融合操作动态地将它们组合成更大的资源态 [7]。其次，我们执行一系列单量子比特投影测量，称为测量模式，这些测量可能取决于先前的测量结果。最后，需要进行副产物校正 ，以处理量子测量结果固有的随机性，因为这些随机结果会导致未测量态的演化具有非确定性，并可能产生额外的不期望演化。我们将在第 II-A 节和第 II-B 节介绍测量基量子计算和融合的细节。

基于 OneQ 的结果，参考文献 [37]进一步采用电路切割技术，将大型电路分解为若干小型子电路，并减少融合操作的数量每个子电路中的融合操作。然而，这种方法在对各子电路的测量统计数据进行后处理时，可能会产生指数级开销 [30]。最近，OnePerc [59] 发现，融合操作的概率性可能会在融合失败时导致测量基量子计算中的计算图发生剧烈变化。这些研究工作共同推动了面向光子系统的测量基量子计算编译器设计，使其更加高效、实用。

然而，这些先前的研究对光子硬件提出了很强的假设，尤其是以吉赫兹级别（1 纳秒/周期）生成资源态的时钟周期，以及光纤延迟线 [11], [59]中的光子存储时间。由于高估了这些硬件要求，在执行长距离操作时丢失光子的概率可能与易出错融合操作的失败率相当，并会导致在光子系统上执行大规模量子程序时发生灾难性故障。包括 OneQ 在内的先前文献 [11], [61]估计，在光子损失概率为 ∼ 5% 的情况下，延迟线中存储的光子最长寿命约为 5000 个时钟周期。根据图1，如果相应的资源态生成速率变为 10 纳秒/周期和 100 纳秒/周期，该概率可能会骤升至 36.9% 和 99.9%，甚至高于当前 29% 的实验融合失败率 [27]。尽管在某些实验设置下 [20], [41],有可能实现 1 纳秒/周期的时钟速率，但目前尚无明确证据表明商业光子量子硬件 [1], [47] 能够以同样的速度生成资源态。这对光子能够在延迟线中存储的生命周期施加了根本性限制。

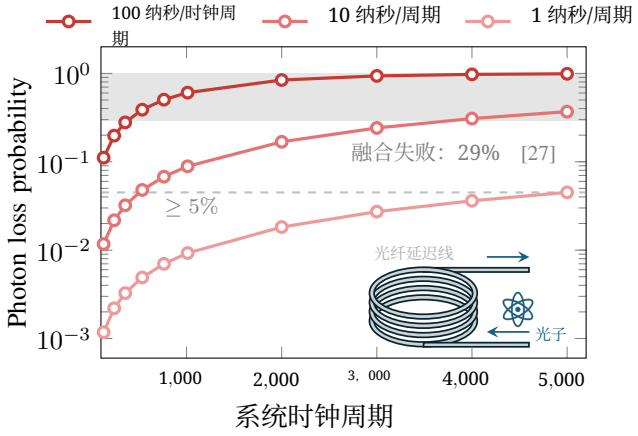

图 1. 不同资源态时钟速率对光子损失概率的影响示意。该概率估算为 $1-e^{-\alpha L}$，其中 $\alpha = 0.2$ dB/km 是当前最先进光纤中的衰减率，而 L 是光子以 2/3 光速在延迟线中行进的距离。

近期的研究已开始着手解决这一关键的物理限制。例如，OneAdapt [60] 指出，光子存储时间较长的一个主要原因，是需要对不同计算步骤之间的连接进行路由。在其前身 OnePerc [59], 的编译模型中，这可能要求将光子存储在延迟线中长达不受限制的时钟周期数。为缓解这一问题，OneAdapt 提出了一种“动态刷新”机制。该技术主动管理每个光子的存储时长，并通过在计算中重新映射即将超过预设寿命限制的光子来刷新它们。这限制了此类远距离连接所需的存储时间。不过，OneAdapt 主要关注路由导致的存储时间；我们仍需要一个更全面的框架，以系统地计算所需光子寿命的所有来源。

为应对上述挑战并推动可扩展测量基量子计算的发展，我们提出了一套全面的解决方案，系统地对光子

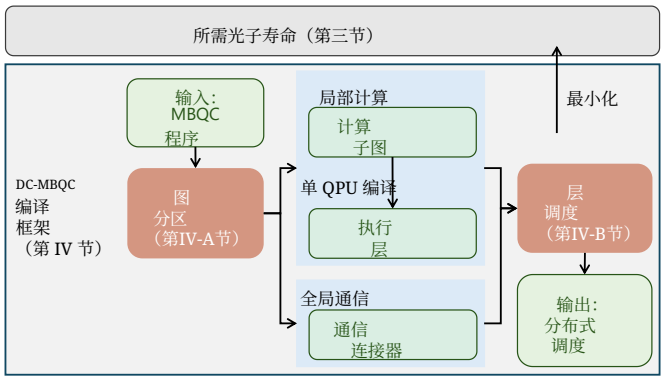

图 2. 我们用于最小化分布式 MBQC 中所需光子寿命的方法概览。

存储资源进行形式化建模，并通过分布式编译（图 2）减轻其负担。我们在此框架中的主要贡献如下。

- 我们对**所需光子寿命**进行了形式化定义（第三节）。该指标首次统一了延迟线中光子存储时长的所有来源，为优化 MBQC 编译器提供了一个基础目标。
- 我们提出了**DC-MBQC**（第 IV 节），这是首个支持分布式 MBQC 的编译框架，重点是最小化所需光子寿命。我们的框架将编译分解为两个关键子问题：自适应图划分（第IV-A节）和层调度（第IV-B节），并针对这两个问题开发了高效的算法。该框架的模块化设计确保了与各种单量子处理单元编译器的兼容性，为未来集成量子纠错铺平了道路。
- 我们的实验结果充分验证了分布式方法的有效性。在使用 8 个全连接量子处理单元时，DC-MBQC 将所需光子寿命最多降低**7.46**× ，并实现了**6.82**×的加速，展示了通向可扩展、高效光子 MBQC 系统的可行路径。

II. 背景

A. 光子 *MBQC* 程序

MBQC 程序在概念上由两个部分定义：一个图态以及相应的测量模式。计算的基础是图态，它是在无向图 $G = (V, E)$ 上构造的大规模多量子比特纠缠态。在此图中，每个顶点 $i \in V$代表一个量子比特，每条边 $(i, j) \in E$ 代表相应一对量子比特之间的纠缠。该态由一组对易稳定子算符的 $+1$本征态唯一确定：

$$K_i = X_i \prod_{j \in N(i)} Z_j, \quad \forall i \in V,$$

其中， X_i 和 Z_j 分别是作用于量子比特 i 和 j 上的泡利 X 和泡利 Z 算符，而 $N(i)$ 是图 G中顶点 i 的邻居顶点集合。

测量基量子计算中的计算通过对图态中的量子比特执行测量模式来进行。测量模式是由一系列自适应单量子比特测量M_i^α组成的，其中后续测量的参数 α 可能取决于先前测量的结果。 M_i^α 表示对量子比特 i进行破坏性测量，将其投影到基 $\{|\pm_\alpha\rangle_i = (|0\rangle \pm e^{i\alpha}|1\rangle)/\sqrt{2}\}$ 上，并产生概率性结果 $s_i \in \{0, 1\}$。

这个经典测量结果 s_i 会立即向前传递，以调整后续测量。这种调整是在后续相邻量子比特上施加泡利副产物校正（X 或 Z）。校正类型由 MBQC 程序的设计预先确定，而实时测量结果 s_i 则充当经典触发信号，用于判断是否应用预定义的校正（例如，当 $s_i = 1$时），以确保计算保持确定性。这些校正在单次运行中完成，无需多次采样，因为修正逻辑依赖于这个单比特 s_i，而非概率分布。实际上，这些校正并非作为新门施加，而是吸收到测量参数 α中。通过 M_i^α进行测量、并施加校正 X^s和 Z^t(s, $t \in \{0, 1\}$) 的量子比特，等价于测量一个

$$M_i^\alpha X_i^s Z_i^t = M_i^{(-1)^s \alpha + t\pi}.$$

这些依赖关系由一个有向无环图正式表示，即依赖图 $G' = (V, E')$。在该图中，一条边（i, j）表示量子比特 j 的测量基取决于量子比特 i的测量结果。这些依赖关系分为 X 依赖或 Z 依赖，分别对应所需的泡利校正类型。Z 依赖可以传播到计算末尾，并通过一种称为信号移位的技术 [13], 在经典层面处理，从而将其从实时依赖约束中移除。

该模型在抽象的逻辑层面描述了测量基量子计算。从实际角度来看，大多数测量基量子计算程序都需要一个庞大的图态，而在计算开始时就完整地构建该图态几乎是不可能的。这一根本性难题促成了一种替代范式：动态生成图态并逐步消耗其中的量子比特 [14], [40], ，这构成了第 II-B 节和第 II-C 节中所讨论的光子实现的基础。

B. 光子测量基量子计算架构

由于事先创建并保存大型图态极其困难，光子架构通过在计算过程中动态生成图态来克服这一挑战。核心策略是从小型、标准化且易于制备的资源态开始，例如图 4 (a) 所示的环形或星形态。随后通过融合逐步将这些态组合成更大的纠缠结构。如图 4 (b) 所示，融合操作会分别从两个资源态中消耗一个光子，使原光子的邻居顶点彼此纠缠。构造出的图中任何剩余且不需要的光子，都可以通过简单的 Z 基

测量将其从计算中解纠缠出来；由于信号移位 [13], ，这种测量不会引入实时依赖。通过这种方法，可以逐步创建任意图态。

这一动态生成过程由图 3 所示的硬件架构进行协调。主要组件包括一个由资源态生成器（RSG）构成的二维排列网格、一组融合装置以及测量设备。在每个系统时钟周期中，每个 RSG 都会生成一个资源态。在一个周期内同时生成的所有资源态构成一个资源态层，为计算建立了离散时间维度。

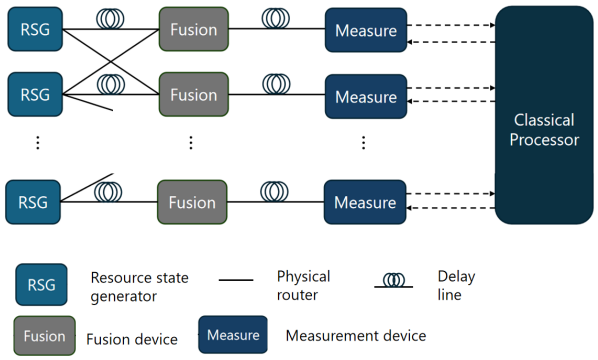

图 3. 采用一维排列资源态生成器（RSG）的测量基量子计算架构。

由于硬件约束，许多架构假设只允许进行局部融合。通常，同一层内相邻 RSG 之间允许层内融合，而同一 RSG 的相邻层之间允许层间融合 [59], [61]。为实现这些局部融合无法直接支持的任意远距离逻辑连接，系统会采用一种称为路由的基本硬件能力。如图 4 (c) 所示，路由通过融合链，有效地在两个相距较远的量子比特之间建立纠缠。

第 II-A 节概述的这些架构约束和测量依赖关系带来了一个关键的系统级挑战：光子存储。光子通常需要等待融合过程中的同步，或等待用于确定测量基的经典信号；在物理上，这通过延迟线（校准至特定时钟周期的光纤）实现。然而，存储时间延长会直接增加光子损失的风险，而这种故障模式会严重损害程序保真度，在实际时钟速率下尤为如此。尽管这一影响至关重要，编译选择与存储导致的保真度下降之间的关系，却一直缺乏正式的高层级度量。为弥补这一空缺，我们在第 III 节提出所需光子寿命，将其作为评估和优化编译器性能的一项基础指标。

为实现分布式计算，可通过预告纠缠建立量子处理单元之间的连接 [6], [29]。通过这种方法，通信资源的生成在物理和逻辑上都与主计算流程解耦。借助这一机制，辅助光子由

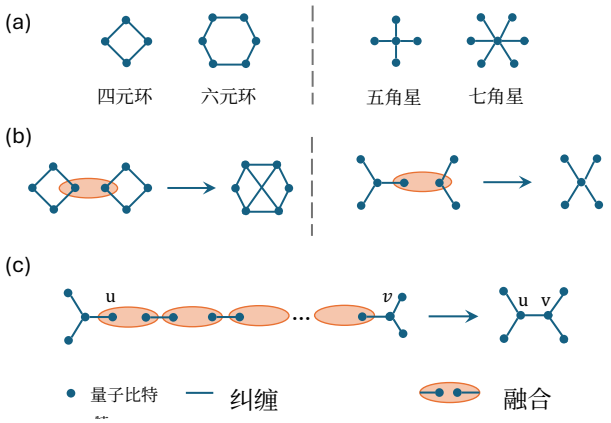

图 4. （a）不同的资源态图；（b）融合对资源态的影响；（c）路由示意图。融合过程完成后，光子 u 和 v 相连接，形成一个纠缠对。

专用通信端口生成，这些端口通常位于量子处理单元芯片的边缘。收到确认远程连接成功的经典预示信号后，辅助光子便成为可用的通信资源。随后，可将该辅助光子与计算光子融合，以连接两个量子处理单元。

C. 光子测量型量子计算编译器

光子测量型量子计算编译器的核心任务，是将由图态和测量模式定义的抽象逻辑 MBQC 程序映射到受限的物理硬件上。该过程可分为两个主要阶段：问题抽象和时空映射。

在第一阶段，编译器会建立计算模型，不同编译器采用不同的抽象策略。一种方法是将输入逻辑程序的需求抽象为适合直接映射到硬件的形式。例如，OneQ 编译器 [61] 会构建一个融合图，其中每个节点对应一个物理资源态，每条边表示一次融合操作。这种表示直接描述了硬件需要执行的物理操作。另一种方法是直接抽象物理硬件，包括 OnePerc [59] 和 OneAdapt [60]，它们会创建逻辑中间表示（IR）。该 IR 可视为离散的三维网格（二维空间、D 维时间），由按时间排序的一系列二维逻辑层组成。在此网格中，节点表示量子比特，边表示相应量子比特之间允许的纠缠。这样，输入图态便可直接映射到该逻辑 IR，而概率性硬件上的执行任务则由名为 2D-重整化 [59] 的独立机制处理。这种逻辑抽象之所以可行，源于一项关键发现：生成一个可靠逻辑层所需的物理层数量（PL 比率）会稳定在一个常数值附近 [59]。因此，高层编译器可以在逻辑层上进行规划和优化，将映射问题与硬件固有的随机性解耦。

因此，第一阶段会将编译问题转化为一个定义明确的问题：将输入的计算图映射到受约束的三维资源网格上。

在第二阶段，编译器将求解这一映射问题。它将计算图中的每个节点分配到三维时空网格中的一个节点，确保计算图中每对相连的节点在三维网格中仍然相连，可以是直接相连，也可以通过其他节点相连。最终输出是一系列执行层，精确规定了每个时钟周期所需执行的物理或逻辑操作。执行这一系列层即可完成 MBQC 程序。

III. 所需光子寿命

在本节中，我们提出所需光子寿命 τ_{photon} 这一概念，将其作为衡量编译器在光子 MBQC 架构上性能的关键指标。该指标不再仅使用程序执行时间，还能反映有限光子存储时间这一关键硬件约束。

根据光子在 MBQC 程序中的作用，光子测量基量子计算架构中的光子可分为以下三类；如图 5 所示，每一类所需的光子寿命都受到不同的时间约束。

- 融合光子：发送至融合装置并与另一个融合体融合的光子。一对融合体必须在融合装置处同步。因此，较早执行层中的融合光子必须存储在延迟线中，等待其融合伙伴到达。
- 测量光子：发送至测量装置并以确定的基进行测量的光子。该基可能取决于其他测量光子的测量结果。因此，在所有依赖关系都得到解决且其测量基确定之前，必须将它存储在延迟线中。
- 移除光子：程序中不需要、通过在泡利 Z 基下测量而移除的光子。由于 Z 依赖可以传播至计算末尾，并通过信号移位以经典方式处理，[13], 移除光子不会影响所需光子寿命。

我们将已编译 MBQC 程序的所需光子寿命 定义为光子必须存储在延迟线中的最大时钟周期数。算法 1 展示了详细的计算过程。在该算法中，G 是 MBQC 程序的依赖图，而执行层是编译器的输出。LayerIndex(u) 是光子 u 所属执行层的索引。在计算 τ_{measuree} 时，TOPOSORT(G) 对所有节点进行拓扑排序

G按拓扑顺序排列。通过此方法，MTime$[u]$ 计算光子 u最早可测量的时刻，并将 τ_u 作为所需寿命。我们假设光子在生成一个时钟周期后到达测量装置，且计算测量基需要一个时钟周期。

分布式编译引入了另一种光子——连接器，其定义见第 IV 节，相应的生命周期计算见定义 IV.1。

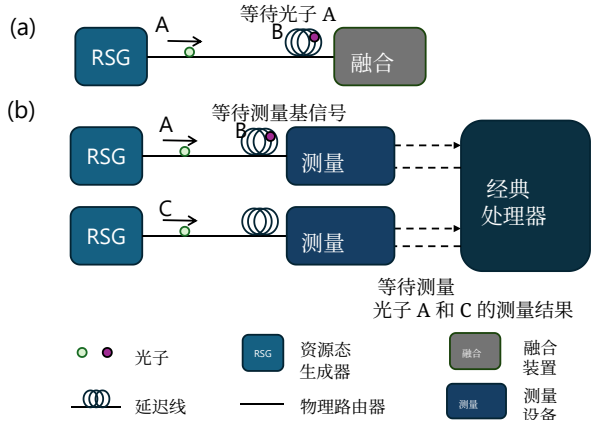

图 5. 所需光子寿命的主要来源示意图。(a) 当融合光子（光子 B）与其融合伙伴（光子 A）在不同执行层中生成时，必须等待光子 A。(b) 测量光子（光子 B）在等待经典信息时必须被存储。之所以需要等待，是因为其测量基取决于其他光子（A 和 C）的测量结果。

算法 1所需光子寿命计算

要求： 执行层；依赖图 G

确保： 所需光子寿命 τ_{photon}

函数计算寿命(G,执行层)
  // 第 1 部分：计算融合体所需寿命 τ_{fusee}
  $\tau_{\text{fusee}} \leftarrow 0$
  对于每一对融合体 (u, v) **执行**
    $\tau_{(u,v)} \leftarrow$ abs(层索引(u) − 层索引(v))
    $\tau_{\text{fusee}} \leftarrow \max(\tau_{\text{fusee}}, \tau_{(u,v)})$
  结束循环
  // 第 2 部分：计算测量所需生存期 τ_{measuree}
  $\tau_{\text{measuree}} \leftarrow 0$
  对于每个节点 u（位于 List(TOPOSORT(G)) 中）**执行**
    MTime$[u] \leftarrow$ 层索引(u) $+1$
    对于每个父节点 v （位于 Parent(u) 中）**执行**
      MTime$[u] \leftarrow$ 最大值(MTime$[v]+1$, MTime$[u]$)
    结束循环
    $\tau_u \leftarrow$ MTime$[u]$ −层索引(u)
    $\tau_{\text{measuree}} \leftarrow \max(\tau_{\text{measuree}}, \tau_u)$
  结束循环
  // 第 3 部分：返回所需光子寿命
  $\tau_{\text{photon}} \leftarrow \max(\tau_{\text{fusee}}, \tau_{\text{measuree}})$ **返回** τ_{photon}
结束函数

IV. DC-MBQC框架

为克服单个量子处理单元的可扩展性限制，出现了各种分布式量子计算方法，包括电路分布和电路切割 [5]。

电路分布旨在将超出单个量子处理单元容量的大型量子电路映射到多个通过量子网络连接的量子处理单元上执行。计算通过对长寿命、有状态的量子比特进行酉演化来完成，而量子处理单元间通信则通过远程数据传输和远程门操作等远程操作实现。编译器面临的核心挑战在于分配逻辑量子比特

至各个量子处理单元，并在资源约束 [15], [22] 下实现远程双量子比特门。相比之下，测量基量子计算程序通过对大型纠缠图态进行单量子比特测量来执行，而量子处理单元之间的互连则可以通过光子融合实现。计算模型的根本差异意味着需要专门的编译器框架，现有的电路分布技术无法直接适用。

另一种主要方法是电路切割，旨在将大型量子电路分解为可在资源有限的设备上执行的较小子电路，然后通过经典后处理重建全局可观测结果 [43], [55]。然而，这种重建通常会引入指数级的采样开销 [30]。测量基量子计算可通过融合建立量子处理单元间通信，从而避免这一瓶颈，因此其核心关注点与电路切割有所不同。

据我们所知，本文首次提出了一个专为测量基量子计算设计的物理分布式计算框架。分布式编译框架必须应对以下两项挑战：

- **工作负载分配**。由于编译器将输入的测量基量子计算程序建模为计算图，工作负载分配需要将计算图分区并分配到多个量子处理单元上，同时平衡工作负载并尽量减少代价高昂的量子处理单元间通信。
- **通信与同步**。计算图的分区不可避免地会切断逻辑纠缠，从而产生若干对连接器，即位于新形成子图边界处的量子比特。这带来了一个关键挑战：如何高效地在各量子处理单元之间重新建立这些连接。

虽然可以通过将连接器与辅助通信资源融合来在物理上实现这一点（见第 II-B 节），但将连接器路由至通信资源仍是一项艰巨的挑战。一种简单的方法可能是在每个量子处理单元上专门划定固定的物理“边缘区域”来实现这一目的，如图6(a)所示。然而，这会施加严格的空间约束，严重限制 OneQ [61],OnePerc [59], 和 OneAdapt [60] 等位置无关编译器的优化能力。为克服这一问题，我们利用时间维度，在三维资源网格中引入连接层。连接器通过层间融合路由至这些连接层，然后再路由至

通过这些特殊层上计算资源的层内融合，获得相应的通信资源，如图6(b)所示。这一策略实现了至关重要的时空解耦，使单QPU编译器能够优化计算布局，而不受通信端口固定物理位置的限制。

然而，在固定的连接层上，可同时路由至通信资源的连接器数量，本质上是网格上的边不相交路径（EDP）问题。该问题旨在寻找指定终端之间数量最多且互不重叠的路径，已知属于 NP难问题 [24]。此外，可用通信资

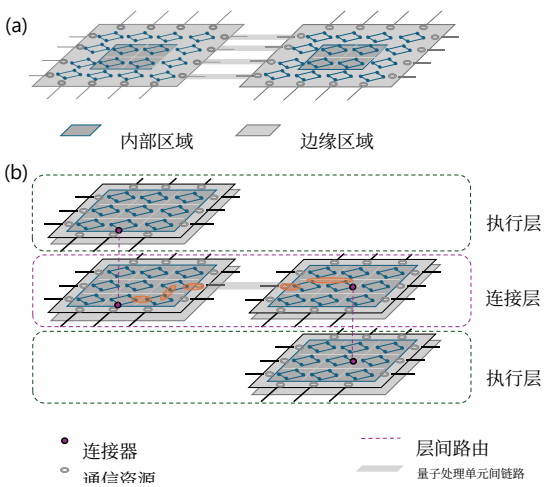

图6. (a) 固定边缘区域示意图。(b) 通过连接层进行量子处理单元间通信的示意图。

源的数量和位置取决于具体硬件。由于我们的框架侧重于高层设计和调度，因此我们将层内并发路由的复杂性抽象为一个实用的硬件参数——连接容量$K_{\max}$，它表示单个层所能支持的最大并发连接数。

尽管存在 NP难性，我们仍可以为此 $K_{\max}$ 给出上界和下界。根据门格尔定理，互不相交路径的最大数量（即 $K_{\max}$ 的上界）受网格上连接器集合与通信资源集合的最小割限制。这意味着，对于大小为 $L \times L$ 的层，$K_{\max} = O(L)$。一个直接的下界是 $K_{\max} \geq 4$，因为存在四条边。我们在第V-E节中进行了敏感性分析实验，结果表明，当 $K_{\max} = 4$ 时，我们的 DC-MBQC 编译器具有高性能。

基于上述抽象，我们的 DC-MBQC框架将分布式编译分解为两个子问题：

- **图划分**。在第IV-A节中，我们提出了一种自适应图划分算法，用于划分计算图，在负载均衡与通信最小化之间权衡，同时保留子图的局部结构，以便高效地进行单量子处理单元编译。

- **层调度**。 在第IV-B节中，我们将层间调度的挑战形式化为层调度问题，证明其NP难性，并提供高效的启发式解决方案。

A. 图划分

我们分布式编译器的首要核心任务，是将整体计算划分到多个量子处理单元上。尽管从概念上看，这与主流电路模型中的图划分类似 [3], [4], ，但我们的测量基量子计算（MBQC）背景在根本上有所不同。基于电路的划分器针对持久量子比特的交互图进行操作，

其目标是尽可能减少量子隐形传态等非局部操作的数量。相比之下，我们的 MBQC 模型基于由瞬态光子构成的图态，主要优化目标是最终阶段所需的光子寿命。这使得关注重点不再仅限于量子处理单元间的通信需求，还必须考虑后续的单QPU编译和层调度。因此，现有的基于量子门的方法并不适用于分布式 MBQC。

在我们的框架中，我们将计算图进行分区，其中每个节点代表一个基本资源单元；因此，实现均衡的分区可直接促进工作负载均衡。在测量基量子计算的背景下，有效的分区算法还必须协同优化两个通常相互竞争的目标：

- 最小化通信：应尽量减少分区之间被切断的边数量，因为每条被切断的边都对应一次代价高昂的跨 QPU 通信。
- 保留局部结构：分区应保持其内部结构的高质量。内部连接性强的子图可由单 QPU 编译器更高效地进行编译 [61]。

这两个目标之间存在权衡。标准的 k 路分区算法 [32]擅长在严格执行负载均衡的同时最小化切边，但会忽略子图结构。相反， [9], [56] 等社区检测算法旨在最大化模块度 [39],；模块度是一种广泛认可的指标，通过衡量子图（社区）内部连接的密度相对于社区之间连接的密度，来量化分区的质量。然而，这些算法无法保证固定数量的分区或工作负载均衡。

为权衡这种取舍，我们提出一种自适应图划分算法（算法 2），融合两种方法的优势，并在不平衡度—模块度权衡空间中进行启发式搜索。算法从 METIS 库生成的完全均衡分区（$\alpha = 1$）开始；该库实现了标准的多层 k 路分区方案 [32]。随后，算法通过逐步放宽均衡约束（α），并采用乘法步长因子（γ），迭代探索更优的分区。只有当新分区带来的模块度增益大于 ϵ_Q 时，搜索才会接受这个均衡性较差的分区；当模块度增益停滞或达到最大不平衡度（$\alpha_{\max}$）时，搜索终止。该过程能够高效地找到一个结构良好（模块度高Q）且相对均衡（低于 $\alpha_{\max}$）的分区。

算法 2自适应图划分

输入： 计算图 G，模块度改进阈值 ϵ_Q，最大失衡因子 $\alpha_{\max}$，学习速率 γ
输出： 最优划分 P_{best}
$\alpha \leftarrow 1$
$Q_{\text{best}} \leftarrow -1$
当True **执行**
  $P \leftarrow$ 划分(G, α)
  $Q \leftarrow$ 模块度(P)
  如果 $Q > Q_{\text{best}}$ **，则**
    $Q_{\text{best}} \leftarrow Q$, $P_{\text{best}} \leftarrow P$
  结束如果
  $\Delta Q \leftarrow Q$ –先前的模块度
  如果 $\Delta Q > \epsilon_Q$ 且 $\alpha < \alpha_{\max}$ **，则**
    $\alpha \leftarrow \alpha \cdot \gamma$
  否则如果 $\Delta Q < -\epsilon_Q$ **则**
    $\alpha \leftarrow \alpha / \gamma$
  else
    跳出
  结束如果
结束循环

B. 层调度

我们首先将层调度问题形式化如下。在这一形式化框架中，我们将编译输出建模为两种不同类型的任务：主任务（$J_{i,\ j}$），表示为特定量子处理单元编译的执行层；同步任务（S_k），表示量子处理单元间的通信事件。

定义 IV.1（层调度问题）。 该问题旨在离散时间范围 $\mathcal{T} = \{1, \ldots, T\}$ 内，对一组 n 量子处理单元 $\mathcal{Q} = \{Q_1, \ldots, Q_n\}$上的两类任务——主任务和同步任务——进行优化调度。给定一个相同的连接参数 $K_{\max}$ 。

a) 任务：

- 主任务。每个量子处理单元 Q_i 都静态分配了一组主任务，$\mathcal{J}_i = \{J_{i,1}, \ldots, J_{i,m_i}\}$。
- 同步任务。定义一个独立的同步任务集合 $\mathcal{S} = \{S_1, \ldots, S_K\}$，其中每个 S_k 都与不同处理器 ($i \neq i'$) 上的一对主任务 ($J_{i,\ j}$, $J_{i',\ j'}$) 相关联。*b)* 决策变量： 各任务的开始时间为决策变量：

- $j_{ij} \in \mathcal{T}$：主任务 $J_{i,\ j}$ 的开始时间。
- $s_k \in \mathcal{T}$：同步任务 S_k 的开始时间。

c) 约束： 调度必须遵循以下约束：

- **机器互斥：** 处理器 Q_i 在任意时刻最多只能执行一个主任务，或最多执行 $K_{\max}$ 个同步任务。

$$\forall i \in \{1, \ldots, n\},\ \forall t \in \mathcal{T},$$
$$\sum_{j=1}^{m_i} \mathbb{I}(j_{ij} = t) + \left\lceil \frac{1}{K_{\max}} \sum_{S_k \text{ involves } Q_i} \mathbb{I}(s_k = t) \right\rceil \leq 1$$

其中 $\mathbb{I}(\cdot)$ 是指示函数。

- **同步：** 与处理器 M_i 和 $M_{i'}$ 相关联的同步任务 S_k 必须同时执行。通过为其开始时间分配单个决策变量 s_k，即可从根本上确保这一点。

d) 目标函数：目标是找到一种调度方案，使第三节中定义的所需光子寿命最小化。如上所述，主任务是局部计算中的执行层，而同步任务则处理远程通信。局部计算 τ_{local} 所需的寿命通过算法 1计算，其中层索引由相应主任务的开始时间 j_{ij} 替代。远程通信 τ_{remote} 所需的寿命主要由相应执行层与连接层之间的层间融合造成。因此，

$$\tau_{\text{remote}} = \max_{S_k \text{ associated with } J_{i,j}} \text{abs}(s_k - j_{i,j}).$$

最终目标是 min max($\tau_{\text{local}}, \tau_{\text{remote}}$)。

定理 IV.2 (层调度的NP难性)。 层调度是*NP*难的；此外，除非 *P = NP*，否则无法在多项式时间内将其近似到任何常数因子以内。

证明。我们通过从图带宽问题（GBP）的判定版本进行多项式时间归约，证明单QPU层调度问题（LSP）的判定版本本身已是NP难的。GBP提出的问题是：给定一个图 G =(V, E) 和一个整数 k，是否存在一种线性布局 f，使得带宽至多为 k?

给定一个任意的GBP实例 (G, k)，我们在多项式时间内构造一个单QPU LSP实例。我们定义一个主任务集合 $\mathcal{J} = \{J_u, \forall u \in V\}$，并针对每条边 ($u, v$) $\in E$，在相应的执行层上引入一对融合体。该构造的规模关于输入图的大小是多项式的。

现在我们证明，如果所构造的 LSP 实例存在一种有效调度，且所需的最大生命周期不超过 k，那么图 G 的带宽至多为 k。假设存在一个解 $j : \mathcal{J} \to \mathcal{T}$，我们可以将 GBP 的布局定义为：对于所有 $u \in V$， $f(u) = j(J_u)$。对于任意边 (u， v) $\in E$,| f (u) $- f(v)| \leq k$，由于 J_u 和 J_v 上的融合光子对所导致的所需生命周期至多为 k。该布局表明， G 的带宽至多为 k。

因此，由于 GBP 的 NP难性，GBP $\leq_p$ LSP，LSP 是 NP难的。此外，在任何常数近似比 [57] 内找到近似解也是 NP难的。□

这一 NP难性促使我们开发高性能启发式算法，以实现高效且高质量的调度。我们首先介绍一种标准的基于优先级的列表调度启发式算法作为基线，随后提出由瓶颈驱动的迭代优化（BDIR）算法，该算法利用模拟退火（SA），通过识别并解决性能瓶颈，系统地改进初始调度。标准的基于优先级的列表调度 [45] 是一个自然的起点，因为主任务具有顺序局部性，对应于单QPU编译器遍次生成的执行层。然而，这种方法往往会因忽略QPU间同步且缺乏对所需光子寿命的全局视角而引入代价高昂的瓶颈。一种简单策略（例如随机移动任务）无法解决性能瓶颈，并且经常违反依赖约束。由于以下原因，设计有效的邻域生成方法颇具挑战：

- 如何选择瓶颈任务及其目标目的地？由于全局最小最大目标函数，不同成本来源带来了相互冲突的优化压力，限制了瓶颈任务的移动。
- 如何在不违反严格同步约束、也不严重破坏现有调度结构的情况下执行此移动？同步约束使标准交换算子失效，因为同步任务必须在两个量子处理单元上同时调度，因此扩大了交换算子的影响范围。

为应对这两项挑战，我们设计了瓶颈驱动的迭代精化（BDIR）算法（算法3），这是一种嵌入轻量级模拟退火（SA）框架中的“智能”邻域生成函数。GENERATENEIGHBOR函数并非随机执行，而是精准地针对调度中的主要瓶颈。首先，FINDBOTTLENECKTASK会识别导致当前所需光子寿命最长的任务 N 。其次，CALCULATEBALANCEPOINT启发式方法不会随机移动 N ，而是确定一个近乎最优的目标时间槽。具体而言，该方法假设所有其他任务保持在当前时隙$L_{current}$不变，并评估不同时间槽下的局部成本（N对 τ_{photon}的贡献）。所选时隙 t 即为“时间平衡点”，可最佳平衡 N之前的依赖任务与其远程融合伙伴带来的成本压力。第三，PINANDRESCHEDULE函数执行移动：将 N固定在目标时间 t ，并使用标准的基于优先级的列表调度对所有其他任务重新调度。至关重要的是，为保留现有调度结构，每个重新调度任务的优先级都设为其在原始调度中的开始时间（$L_{current}$）。这既能在解决移动 N 所导致的约束冲突的同时，维持调度中的相对顺序。最后，这个“智能”GENERATENEIGHBOR函数被封装在SA框架中。初始调度 L_{init} 通过列表调度生成，其中主任务$J_{i,\ j}$的优先级为其索引j；同步任务 S_k （针对$J_{i,\ j}$，$J_{i',\ j'}$）的优先级为 $_{1\,2}$（$j + j'$）。

–

V. 评估

A. 实验设置

基线。 我们比较的基线方法是 OneQ 编译器 [61]，，这是一个专为光子测量基量子计算架构量身打造的近期编译框架。尽管 OnePerc [59] 和 OneAdapt [60] 等较新的框架引入了进一步的优化，OneQ 仍确立了将测量基量子计算程序映射到三维资源网格的基础方法。关键在于，这些后续框架均建立在 OneQ 所定义的核心编译阶段之上。关于与 OnePerc 和 OneAdapt 的比较，请参见第 V-C 节。

指标。 如第一节所述，在延迟线中长时间存储光子会大幅增加光子

算法 3 瓶颈驱动的迭代精化（BDIR）

输入： 模拟退火初始温度 T_0、冷却率 α、最大迭代次数 $I_{\max}$、初始调度 L_{init}
输出： 优化后的调度 L_{best}

函数 BDIR($T_0, \alpha, I_{\max}, L_{\text{init}}$)
  $L_{\text{current}} \leftarrow L_{\text{init}}$，$T \leftarrow T_0$
  $L_{\text{best}} \leftarrow L_{\text{init}}$，$c_{\text{best}} \leftarrow$ 计算寿命(L_{best})
  对于 $i = 1 \rightarrow I_{max}$，**执行 do**
    $L_{\text{new}} \leftarrow$生成邻居($L_{\text{current}}$)
    $c_{\text{current}} \leftarrow$计算寿命($L_{\text{current}}$)
    $c_{\text{new}} \leftarrow$计算寿命($L_{\text{new}}$)
    $\Delta E \leftarrow c_{\text{new}} - c_{\text{current}}$
    如果 $\Delta E \leq 0$ **或** RANDOM(0,1) $< e^{-\Delta E/T}$ **则**
      $L_{\text{current}} \leftarrow L_{\text{new}}$
    结束条件判断
    如果 $c_{\text{current}} < c_{\text{best}}$ **则**
      $L_{\text{best}} \leftarrow L_{\text{current}}$，$c_{\text{best}} \leftarrow c_{\text{current}}$
    结束条件判断
    $T \leftarrow \alpha \cdot T$
  结束循环
  返回 L_{best}
结束函数

函数生成邻域(L_{current})
  $N \leftarrow$查找瓶颈任务(L_{current})
  $t \leftarrow$ 计算平衡点($L_{current}$，N)
  $L_{\text{new}} \leftarrow$ 固定并重新调度(L_{current}，N，t)
  返回 L_{new}
结束函数

损失的概率。这种损失是导致致命错误的主要来源，会严重损害程序保真度，尤其是在实际时钟频率下（见图1）。因此，我们的核心指标是所需光子寿命（第三节），它是衡量这种由存储引起的保真度下降情况的关键高层代理指标。我们还将执行时间 视为一项重要指标，即运行一个 MBQC 程序所需的时钟周期数。

基准测试程序。 我们选取量子近似优化算法 (QAOA) [21], 变分量子本征求解器 (VQE) [44], 量子傅里叶变换 (QFT) [16], 和脉动进位加法器 (RCA) [18] 作为基准测试，它们包括用于量子编程的算法原语（QFT 和 RCA），以及面向应用的程序（用于优化问题的 QAOA 和用于量子化学模拟的 VQE）。表II列出了量子比特数、空间网格大小（即每个二维逻辑资源层的大小）、双量子比特门数, 以及融合操作数（即 OneQ 计算图中的边）。对于 QAOA，电路由随机图上的最大割问题生成。生成图时，会从所有可能的边中随机选取一半。对于 VQE，我们采用具有完全纠缠层的硬件高效试探态 [31]，其中每对量子比特都通过一个受控非门连接，因此双量子比特门的数量会随着量子比特数呈二次增长。QFT 程序的门数量也呈现出相似的增长趋势。我们评估的最大案例是一个含 196 个量子比特的 QAOA 程序，其中包含 7528 个双量子比特门和 68141 个融合操作。

评估平台。 所有编译运行时间均在一台配备 Intel(R) Xeon(R) Gold 6459C、运行 Ubuntu 22.04.5 LTS 操作系统的工作站上测量。所有报告的编译时间均为五次运行的平均值，以确保测量结果稳定。

参数设置。 对于自适应图划分（算法 2），我们设置改进阈值 $\epsilon_Q = 0.01$ 和步长因子 $\gamma = 1.02$。对于 BDIR调度器（算法 3），我们使用初始温度 $T_0 = 10$、冷却率 $\alpha = 0.95$，以及 $I_{max} = 20$ 次迭代。两个关键参数，连接容量 K_{max} 和最大不平衡因子 α_{max}，在所有主要实验中均采用默认值 $K_{max} = 4$ 和 $\alpha_{max} = 1.5$，除非另有说明。第V-E节详细分析了这两个参数的敏感性。

B. 实验结果

我们首先在表 III 中展示了基线 OneQ 编译器与我们的 DC-MBQC 编译器（使用 4 个量子处理单元和五角星资源状态）在执行时间和所需光子寿命方面的对比结果。QneQ 编译器是为单个量子处理单元设计的，而所提出的编译器设计则利用了多个量子处理单元。可以看出，DC-MBQC 在执行时间和所需光子寿命两方面均始终优于基线，最高可达 3.97×（针对 36 量子比特的变分量子本征求解器程序）。

性能提升分析。相较于单体基线，性能显著提升，源于分布式架构固有的并行性；我们的框架通过协同优化局部计算与全局同步，使这种并行性得以发挥。首先，我们的自适应图划分（算法 2）在平衡工作负载的同时优化模块度，从而生成结构经过优化的子图。这降低了单QPU编译器的复杂度，使其能够为每个量子处理单元找到更紧凑的局部

表II 基准测试程序。

程序	#量子比特	网格大小	#双量子比特门	#融合
VQE	16	7 × 7	120	408
	36	11 × 11	630	2178
	81	17 × 17	3240	11280
	144	23 × 23	10296	35928
QAOA	16	7 × 7	47	487
	64	15 × 15	799	7316
	121	21 × 21	2843	25826
	196	27 × 27	7528	68141
QFT	16	7 × 7	120	408
	36	11 × 11	630	2178
	81	17 × 17	3240	11280
	100	19 × 19	4950	64450
RCA	16	7 × 7	209	1108
	36	11 × 11	529	2808
	81	17 × 17	1249	6633

表 III 使用 4 个量子处理单元和五角星 RSG 时，DC-MBQC 的结果及其相对于基线的性能。

程序-量子比特数	基线执行时间	我们的执行时间	改进因子	基准寿命	我们的寿命	改进因子
VQE-16	46	21	2.19	38	14	2.71
VQE-36	155	46	3.37	143	36	**3.97**
VQE-81	462	128	3.61	441	112	3.94
VQE-144	1022	278	3.68	1000	258	3.88
QAOA-16	45	16	2.81	36	10	3.60
QAOA-64	284	83	3.42	265	70	3.79
QAOA-121	731	198	3.69	717	183	3.92
QAOA-196	1401	394	3.56	1370	369	3.71
量子傅里叶变换-16	126	35	3.60	107	28	3.82
量子傅里叶变换-36	364	101	3.60	333	81	4.11
QFT-81	1000	293	3.41	975	264	3.69
QFT-100	1295	386	3.35	1272	344	3.70
RCA-16	103	27	**3.81**	90	26	3.46
RCA-36	121	34	3.56	123	57	2.16
RCA-81	158	44	3.59	218	135	1.61

表 IV DC-MBQC 的结果及其相对于使用 8 个量子处理单元和四星RSG的基线的性能。

程序-量子比特数	基线执行时间	我们的执行时间	改进因子	基准寿命	我们的寿命	改进因子
VQE-16	54	14	3.86	45	10	4.50
VQE-36	155	40	5.17	140	23	6.09
VQE-81	471	72	6.54	441	66	6.78
VQE-144	1016	149	**6.82**	991	137	7.23
QAOA-16	48	14	3.43	39	11	3.55
QAOA-64	273	56	4.88	253	43	5.88
QAOA-121	712	116	6.14	696	106	6.57
QAOA-196	1401	204	6.87	1372	186	7.38
量子傅里叶变换-16	120	21	5.71	110	18	6.11
量子傅里叶变换-36	356	54	6.59	329	46	7.15
量子傅里叶变换-81	1036	152	6.81	993	133	**7.46**
QFT-100	1250	186	6.72	1221	167	7.31
RCA-16	103	22	4.68	90	29	3.10
RCA-36	121	18	6.72	123	54	2.28
RCA-81	138	27	5.11	203	126	1.61

调度（即更少的执行层）。其次，我们的框架系统性地降低了同步开销。分区算法将量子处理单元间连接器（割边）的数量降至最低，而我们的 BDIR 层调度器（算法 3）则通过启发式方法降低其调度成本（所需光子寿命）。

不同的资源态。我们的框架也支持第 II-B 节介绍的不同资源态。我们在图 7 中比较了 4 种资源态对执行时间和所需光子寿命改进的影响。在大多数基准测试中，四环资源态图优于其他候选方案。在所需光子寿命方面，六环资源态图的表现始终差于其他类型的 RSG。这是由其独特的拓扑结构导致的。假设移除六元环资源态中一对对角光子，则会剩下两个双量子比特资源态，两者都可用于路由。因此，六元环资源态可用作路由资源两次，而其他资源态只能使用一次。由于单个量子处理单元编译面临的主要挑战是处理复杂的路由需求，六环资源态图尤其适用于单个量子处理单元。

量子处理单元数量。我们还在表 IV 中进一步研究了使用 8 个量子处理单元进行分布式 MBQC 的影响。与表 III 中使用 4 个量子处理单元的情况相比，改进效果始终更好；对于 81 量子比特的 QFT，所需光子寿命最多可提升 $7.46\times$。

C. 与最先进编译器的比较

正如第 II-C 节所述，OneQ、OnePerc 和 OneAdapt 等单体编译器的核心任务，是解决将计算图映射到受限三维时空资源网格这一复杂问题。我们的 DC-MBQC框架专为围绕这一抽象实现模块化而设计（见图 2）。本节将阐明该框架与 OnePerc 的关系，并给出与 OneAdapt 对比的新数据。

OnePerc OnePerc 的主要贡献是提出了一种新颖的、感知随机性的在线执行策略（例如重整化），以应对概率性融合失败 [59]。这一在线问题

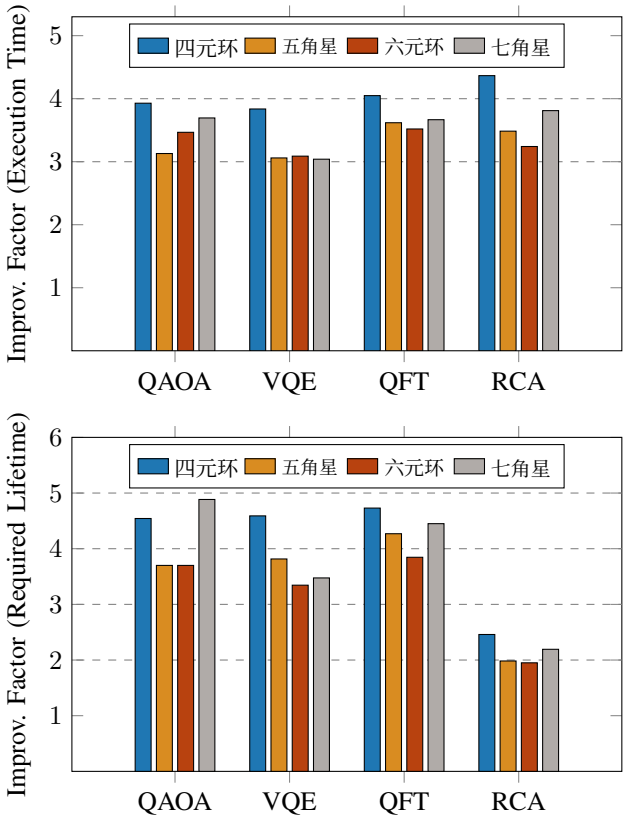

图 7. 使用 4 个量子处理单元和图 4(a) 所示的不同资源态，在 36 量子比特量子近似优化算法、量子傅里叶变换、RCA 和变分量子本征求解器上，DC-MBQC 相对于基线的改进。改进因子定义为 $f \equiv \tau^{\text{OneQ}}/\tau^{\text{DC-MBQC}}$。

与我们的框架正交。对于其离线编译过程（即将逻辑图映射到三维资源网格的阶段），OnePerc 的作者表示，他们“采用”并“扩展”了 OneQ 映射算法这一基础方法。因此，我们在第 V-B 节中以 OneQ 为基线进行的比较，直接评估了这一核心编译阶段；所得结果也代表了 OneQ 和 OnePerc 的离线编译性能。

OneAdapt OneAdapt 引入了一种动态刷新技术，以减轻光子损失。为了将这一单体式编译器调整用于我们的分布式比较，我们通过边界资源预留对量子处理单元间通信开销进行建模。具体而言，我们将每层的边界资源态专用于充当通信接口，从而使每个维度的网格大小减少 2。该方法能够一致地估算通信开销，使我们可以专注于分布式架构与单量子处理单元基线之间的算法性能比较。我们的实验结果汇总于表 V，表明我们的分布式方法仍能在 OneAdapt 的基础上带来显著的额外收益。例如，在使用 8 个量子处理单元时，DC-MBQC 的执行时间加速比最高可达 $5.74\times$，所需光子寿命可减少 $4.33\times$ 。

D. 组件分析

我们将 BDIR 算法与一种基于标准优先级的列表调度基线进行比较。在该实验中，我们使用完整的 DC-MBQC框架，仅替换最终的层调度组件。表VI 表明，与基线相比，BDIR 算法能够持续降低所需光子寿命，改进幅度介于 4.62% 至显著的 15.12% 之间。

E. 敏感性分析

本节对框架中的两个关键参数进行敏感性分析：硬件的连接容量 $K_{\max}$ 以及自适应图划分算法（算法 2）中使用的最大失衡因子$\alpha_{\max}$ 。我们选择 36 量子比特量子傅里叶变换程序作为代表性基准测试，因为表II中的量子傅里叶变换需要最多的双量子比特门和融合操作。对于 $K_{\max}$，我们还对 25 量子比特量子傅里叶变换实例进行了敏感性分析，以展示研究结果的一致性。

对于 $K_{\max}$，图 8 显示，随着$K_{\max}$ 增加，执行时间和所需生命周期的边际收益递减趋势十分明显，拐点出现在大约$K_{\max} = 4$ 至 7 附近。这一趋势表明，只有在 $K_{\max}$ 较小时，量子处理单元间通信才是主要瓶颈，说明我们的 DC-MBQC 框架无需极高的连接容量即可实现显著的性能提升。

参数 $\alpha_{\max}$ 控制着我们自适应图划分算法中，量子处理单元之间严格负载均衡与所得子图结构质量之间的权衡。图 9 展示了执行时间和所需生命周期方面的性能提升对 $\alpha_{\max}$ 具有稳健性，仅在较窄范围内波动。值得注意的是，我们发现，在测试的整个 $\alpha_{\max}$ 范围内（从 1.05 开始），算法 2 生成的图划分始终相同，割大小恒为 60，模块度为 0.74。因此，观察到的性能差异并非源于划分质量，而完全是由后续局部单量子处理单元编译和全局层调度的随机性导致的。

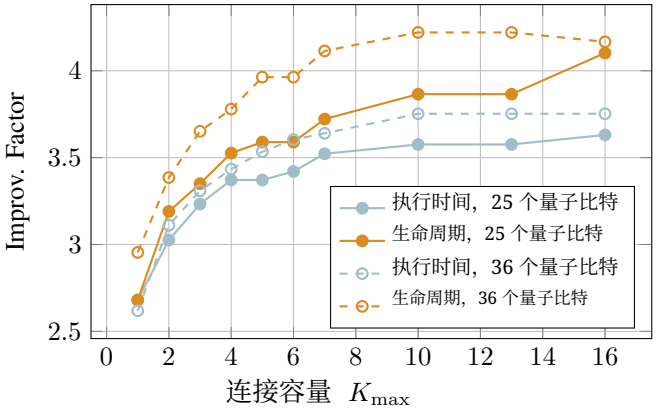

图 8。连接容量 $K_{\max}$ 的影响。

表 V DC-MBQC 在使用 4 个和 8 个量子处理单元时的结果及相对于 OneAdapt 的性能表现。

量子处理单元数	程序-量子比特数	OneAdapt 执行时间	我们的执行时间	改进因子	OneAdapt 生命周期	我们的寿命	改进因子
4	变分量子本征求解器-64	306	92	**3.32**	13	6	2.17
	变分量子本征求解器-100	552	198	2.79	20	14	1.43
	QAOA-64	332	111	2.99	14	7	2.00
	QAOA-121	705	246	2.87	20	13	1.54
	QFT-36	313	101	3.10	10	9	1.11
	QFT-64	715	216	3.31	20	6	**3.33**
8	VQE-64	327	57	**5.74**	20	6	3.33
	VQE-100	599	108	5.55	20	10	2.00
	QAOA-64	326	71	4.59	9	3	3.00
	QAOA-121	706	175	4.03	20	5	4.00
	QFT-36	319	76	4.20	13	3	**4.33**
	QFT-64	673	138	4.88	20	5	4.00

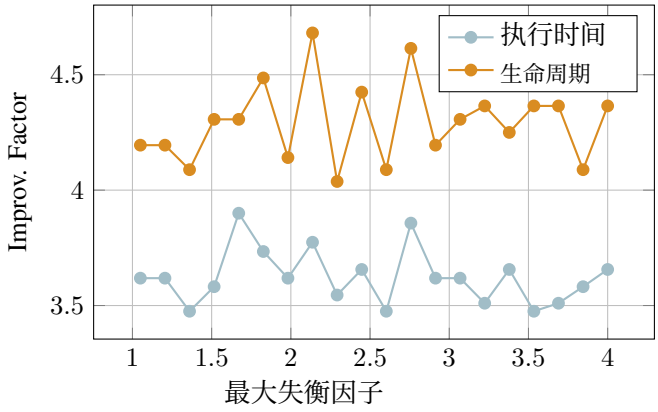

图 9. 最大不平衡因子 $\alpha_{\max}$ 的鲁棒性。

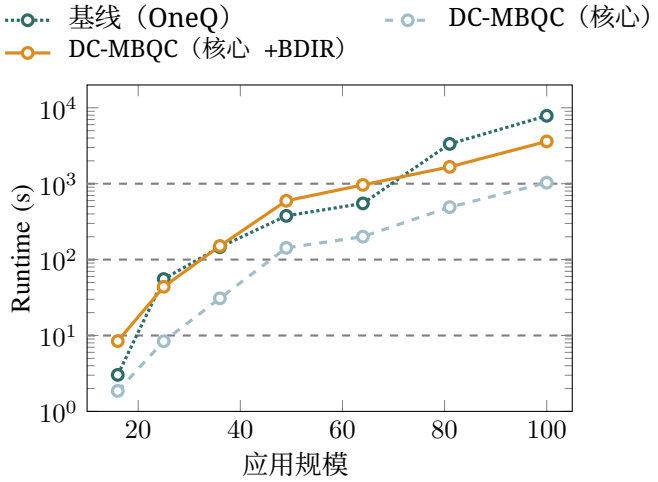

图 10. 编译 QFT 程序的运行时间扩展情况（不包括共同的预处理开销）。与基线相比，DC-MBQC 展现出更好的可扩展性。DC-MBQC 的高效率使其能够在相同的时间预算内编译规模更大的量子程序。

F. 可扩展性

编译时间是执行大规模应用的关键因素 [8]。为评估编译启发式算法的可扩展性，图10比较了最多包含 100 个量子比特的量子傅里叶变换基准测试中，核心编译阶段（不包括通用预处理）的运行时间。我们将单体式（OneQ）基线与 8-QPU框架的两个版本进行比较：DC-MBQC（核心 + BDIR）（采用完整的 BDIR优化，并使用 $I_{\max} = 20$）和 DC-MBQC（核心）（仅采用基本的基于优先级的列表调度，不使用 BDIR）。图10显示，随着问题规模增大，所有方法的编译时间都在增加，而我们的分布式方法展现出更好的扩展性。与基线相比，DC-MBQC（核心 + BDIR）的增幅较小。此外，DC-MBQC（核心）版本提供了一个颇具吸引力的折中方案：省略 BDIR 优化遍后，它能够实现更快的编译，同时仍受益于核心框架的分区和基本调度优势。这凸显了我们框架的灵活性，用户可以据此在编译开销与最终质量之间进行权衡。

表VI BDIR 的有效性

程序-量子比特数	基准寿命	BDIR 生命周期	改进因子
量子傅里叶变换-16	33	31	6.06%
量子傅里叶变换-25	65	62	4.62%
量子傅里叶变换-36	86	73	15.12%
QFT-49	140	124	11.43%
QFT-64	196	184	6.12%

六、结论

在本研究中，我们提出了 DC-MBQC，这是首个专为基于测量的量子计算设计的分布式量子编译框架。我们的框架通过自适应图划分算法以及一种解决层调度问题的新方法，应对分布式 MBQC 面临的挑战。我们引入了所需光子寿命这一指标，用于评估实际性能。实验

结果显示，在 8 个全连接量子处理器上，所需光子寿命降低了 7.46× ，计算加速达到 6.82× ，验证了我们方法的有效性。

致谢

我们感谢 Zhixin Song 就分布式架构开展富有洞见的讨论并提出富有启发性的建议。我们也衷心感谢他在文献综述、图表制作以及初稿引言撰写方面给予的大力协助。我们感谢匿名审稿人提出的建设性意见，这些意见显著提升了本文的质量。本工作得到国家自然科学基金委员会（项目编号：92365117）的资助。

参考文献

[1] H. Aghaee Rad, T. Ainsworth, R. N. Alexander, B. Altieri, M. F. Askarani, R. Baby, L. Banchi, B. Q. Baragiola, J. E. Bourassa, R. S. Chadwick, I. Charania, H. Chen, M. J. Collins, P. Contu, N. D'Arcy, G. Dauphinais, R. De Prins, D. Deschenes, I. Di Luch, S. Duque, P. Edke, S. E. Fayer, S. Ferracin, H. Ferretti, J. Gefaell, S. Glancy, C. Gonz´ alez-Arciniegas, T. Grainge, Z. Han, J. Hastrup, L. G. Helt, T. Hillmann, J. Hundal, S. Izumi, T. Jaeken, M. Jonas, S. Kocsis, I. Krasnokutska, M. V. Larsen, P. Laskowski, F. Lau- denbach, J. Lavoie, M. Li, E. Lomonte, C. E. Lopetegui, B. Luey, A. P. Lund, C. Ma, L. S. Madsen, D. H. Mahler, L. Mantilla Calder´on, M. Menotti, F. M. Miatto, B. Morrison, P. J. Nadkarni, T. Nakamura, L. Neuhaus, Z. Niu, R. Noro, K. Papirov, A. Pesah, D. S. Phillips, W. N. Plick, T. Rogalsky, F. Rortais, J. Sabines-Chesterking, S. Safavi- Bayat, E. Sazhaev, M. Seymour, K. Rezaei Shad, M. Silverman, S. A. Srinivasan, M. Stephan, Q. Y. Tang, J. F. Tasker, Y. S. Teo, R. B. Then, J. E. Tremblay, I. Tzitrin, V. D. Vaidya, M. Vasmer, Z. Vernon, L. F. S. S. M. Villalobos, B. W. Walshe, R. Weil, X. Xin, X. Yan, Y. Yao, M. Zamani Abnili, and Y. Zhang, "扩展并联网模块化光子量子计算机", *Nature*，第 638 卷，第 1–8 页，2025。[2] Google Quantum AI, "低于表面码阈值的量子纠错", *Nature*，第 638 卷，第 8052 期，第 920 页，2024。[3] P. Andr´es-Mart´ınez 和 C. Heunen，"通过超图划分自动分配量子电路", *Phys. Rev. A*，第 100 卷，第 032308 页，2019 年 9 月。[4] J. M. Baker、C. Duckering、A. Hoover 和 F. T. Chong，"面向模块化架构的时间切片量子电路划分"，载于 *Proceedings of the 17th ACM International Conference on Computing Frontiers*，CF '20 系列。美国纽约州纽约市：Association for Computing Machinery，2020，第 98–107 页。[5] D. Barral、F. J. Cardama、G. D´ıaz-Camacho、D. Fa´ılde、I. F. Llovo、M. Mussa-Juane、J. V´azquez-P´erez、J. Villasuso、C. Pi˜neiro、N. Costas、J. C. Pichel、T. F. Pena 和 A. G´omez，"分布式量子计算综述：从单个量子处理单元到高性能量子计算"，*Computer Science Review*，第 57 卷，第 100747 页，2025 年 8 月。[6] S. D. Barrett 和 P. Kok，"利用物质量子比特和线性光学实现高效高保真度量子计算"，*Physical Review A*，第 71 卷，第 6 期，2005 年 6 月。[7] S. Bartolucci、P. Birchall、H. Bombin、H. Cable、C. Dawson、M. Gimeno- Segovia、E. Johnston、K. Kieling、N. Nickerson、M. Pant、F. Pastawski、T. Rudolph 和 C. Sparrow，"基于融合的量子计算"，*Nature Communications*，第 14 卷，第 1 期，第 912 页，2023。[8] M. E. Beverland、P. Murali、M. Troyer、K. M. Svore、T. Hoefler、V. Kliuchnikov、G. H. Low、M. Soeken、A. Sundaram 和 A. Vaschillo，"评估扩展至实用量子优势的需求"，*arXiv preprint arXiv:2211.07629*，2022。[9] V. D. Blondel、J.-L. Guillaume、R. Lambiotte 和 E. Lefebvre，"大型网络中社区的快速展开"，*Journal of Statistical Mechanics: Theory andExperiment*，第 2008 卷，第 10 期，第 P10008 页，2008 年 10 月。

[10] D. Bluvstein、S. J. Evered、A. A. Geim、S. H. Li、H. Zhou、T. Manovitz、S. Ebadi、M. Cain、M. Kalinowski、D. Hangleiter、J. P. Bonilla Ataides、N. Maskara、I. Cong、X. Gao、P. Sales Rodriguez、T. Karolyshyn、G. Semeghini、M. J. Gullans、M. Greiner、V. Vuleti´c 和 M. D. Lukin，《基于可重构原子阵列的逻辑量子处理器》，《自然》，第626卷，第7997期，第58–65页，2023年12月。[11] H. Bombin、I. H. Kim、D. Litinski、N. Nickerson、M. Pant、F. Pastawski、S. Roberts 和 T. Rudolph，《交错：容错光子量子计算的模块化架构》，*arXiv*预印本 *arXiv:2103.08612*，2021年。[12] H. J. Briegel、D. E. Browne、W. D¨ur、R. Raussendorf 和 M. Van den Nest，《基于测量的量子计算》，《自然·物理》，第5卷，第1期，第19–26页，2009年。[13] A. Broadbent 和 E. Kashefi，《量子电路并行化》，《理论计算机科学》，第410卷，第26期，第2489–2510页，2009年。[14] D. E. Browne 和 T. Rudolph，《资源高效的线性光学量子计算》，《物理评论快报》，第95卷，第1期，2005年6月。[15] M. Caleffi、M. Amoretti、D. Ferrari、J. Illiano、A. Manzalini 和 A. S. Cacciapuoti，《分布式量子计算：综述》，《计算机网络》，第254卷，文章110672，2024年。[16] D. Coppersmith，《一种用于量子因数分解的近似傅里叶变换》，2002年，arXiv预印本 arXiv:quant-ph/0201067。[17] J. P. Covey、H. Weinfurter 和 H. Bernien，《采用中性原子处理节点的量子网络》，《*npj*量子信息》，第9卷，第1期，文章90，2023年。[18] S. A. Cuccaro、T. G. Draper、S. A. Kutin 和 D. P. Moulton，《一种新的量子行波进位加法电路》，*arXiv*预印本 *quant-ph/0410184*，2004年。[19] A. M. Dalzell、S. McArdle、M. Berta、P. Bienias、C.-F. Chen、A. Gily´en、C. T. Hann、M. J. Kastoryano、E. T. Khabiboulline、A. Kubica、G. Salton、S. Wang 和 F. G. S. L. Brand˜ao，《量子算法：应用与端到端复杂度综述》，*arXiv*预印本 *arXiv:2310.03011*，2023年。[20] F. Eltes、G. E. Villarreal-Garcia、D. Caimi、H. Siegwart、A. A. Gentile、A. Hart、P. Stark、G. D. Marshall、M. G. Thompson、J. Barreto、J. Fompeyrine 和 S. Abel，《一种工作于低温下的集成光学调制器》，《自然·材料》，第19卷，第11期，第1164–1168页，2020年。[21] E. Farhi、J. Goldstone 和 S. Gutmann，《量子近似优化算法》，2014年，arXiv预印本 arXiv:1411.4028。[22] D. Ferrari、A. S. Cacciapuoti、M. Amoretti 和 M. Caleffi，《分布式量子计算的编译器设计》，《*IEEE*量子工程汇刊》，第2卷，第1–20页，2021年。[23] D. Gao、D. Fan、C. Zha、J. Bei、G. Cai、J. Cai、S. Cao、X. Zeng、F. Chen、J. Chen、K. Chen、X. Chen、X. Chen、Z. Chen、Z. Chen、Z. Chen、W. Chu、H. Deng、Z. Deng、P. Ding、X. Ding、Z. Ding、S. Dong、Y. Dong、B. Fan、Y. Fu、S. Gao、L. Ge、M. Gong、J. Gui、C. Guo、S. Guo、X. Guo、T. He、L. Hong、Y. Hu、H.-L. Huang、Y.-H. Huo、T. Jiang、Z. Jiang、H. Jin、Y. Leng、D. Li、D. Li、F. Li、J. Li、J. Li、J. Li、J. Li、N. Li、S. Li、W. Li、Y. Li、Y. Li、F. Liang、X. Liang、N. Liao、J. Lin、W. Lin、D. Liu、H. Liu、M. Liu、X. Liu、X. Liu、Y. Liu、H. Lou、Y. Ma、L. Meng、H. Mou、K. Nan、B. Nie、M. Nie、J. Ning、L. Niu、W. Peng、H. Qian、H. Rong、T. Rong、H. Shen、Q. Shen、H. Su、F. Su、C. Sun、L. Sun、T. Sun、Y. Sun、Y. Tan、J. Tan、L. Tang、W. Tu、C. Wan、J. Wang、B. Wang、C. Wang、C. Wang、J. Wang、L. Wang、R. Wang、S. Wang、X. Wang、Z. Wei、J. Wei、D. Wu、G. Wu、J. Wu、S. Wu、Y. Wu、S. Xie、L. Xin、Y. Xu、C. Xue、K. Yan、W. Yang、X. Yang、Y. Yang、Y. Ye、Z. Ye、C. Ying、J. Yu、Q. Yu、W. Yu、S. Zhan、F. Zhang、H. Zhang、K. Zhang、P. Zhang、W. Zhang、Y. Zhang、Y. Zhang、L. Zhang、G. Zhao、P. Zhao、X. Zhao、X. Zhao、Y. Zhao、Z. Zhao、L. Zheng、F. Zhou、L. Zhou、N. Zhou、N. Zhou、S. Zhou、S. Zhou、Z. Zhou、C. Zhu、Q. Zhu、G. Zou、H. Zou、Q. Zhang、C.-Y. Lu、C.-Z. Peng、X. Zhu 和 J.-W. Pan，《使用105量子比特祖冲之3.0处理器在量子计算优势方面建立新基准》，《物理评论快报》，第134卷，第9期，文章090601，2025年。[24] M. R. Garey，载于《*Johnson*：计算机与难解问题：*NP*完全性理论指南》，1979年。[25] C. Gidney 和 M. Eker˚a，《使用2000万个噪声量子比特在8小时内分解2048位RSA整数》，《量子》，第5卷，文章433，2021年。[26] L. K. Grover，《一种用于数据库搜索的快速量子力学算法》，载于《第二十八届*ACM*计算理论年会论文集》，1996年，第212–219页。[27] Y.-P. Guo、G.-Y. Zou、X. Ding、Q.-H. Zhang、M.-C. Xu、R.-Z. Liu、J.-Y. Zhao、Z.-X. Ge、L.-C. Peng、K.-M. Xu、Y.-Y. Lou、Z. Ning、L.-

J. Wang、H. Wang、Y.-H. Huo、Y.-M. He、C.-Y. Lu 和 J.-W. Pan，《渗流阈值以上用于可扩展图态生成的增强融合门》，*arXiv*预印本 *arXiv:2412.18882*，2024年。[28] M. Hein、J. Eisert 和 H. J. Briegel，《图态中的多方纠缠》，《物理评论*A*：原子、分子与光学物理》，第69卷，第6期，文章062311，2004年。[29] D. Hucul、I. V. Inlek、G. Vittorini、C. Crocker、S. Debnath、S. M. Clark 和 C. Monroe，《使用光子和声子实现原子量子比特的模块化纠缠》，《自然·物理》，第11卷，第1期，第37–42页，2014年11月。[30] M. Jing、C. Zhu 和 X. Wang，《电路拼接面临受纠缠成本约束的指数级采样开销增长》，《物理评论 *A*》，第111卷，第1期，文章012433，2025年。[31] A. Kandala、A. Mezzacapo、K. Temme、M. Takita、M. Brink、J. M. Chow 和 J. M. Gambetta，《用于小分子和量子磁体的硬件高效变分量子本征求解器》，《自然》，第549卷，第7671期，第242–246页，2017年。[32] G. Karypis 和 V. Kumar，《非规则图的多层k路分区方案》，《并行与分布式计算杂志》，第48卷，第1期，第96–129页，1998年。[33] N. Leung、Y. Lu、S. Chakram、R. Naik、N. Earnest、R. Ma、K. Jacobs、A. Cleland 和 D. Schuster，《超导量子比特之间的确定性双向通信与远程纠缠生成》，《*npj*量子信息》，第5卷，第1期，文章18，2019年。[34] Y. Li 和 J. D. Thompson，《中性原子量子处理器之间的高速率、高保真度模块化互连》，《*PRX Quantum*》，第5卷，第2期，文章020363，2024年。[35] C. L¨oschnauer、J. M. Toba、A. Hughes、S. King、M. Weber、R. Srinivas、R. Matt、R. Nourshargh、D. Allcock、C. Ballance、C. Matthiesen、M. Malinowski 和 T. P. Harty，《囚禁离子量子比特的可扩展、高保真度全电子控制》，*arXiv*预印本 *arXiv:2407.07694*，2024年。[36] D. Main、P. Drmota、D. Nadlinger、E. Ainley、A. Agrawal、B. Nichol、R. Srinivas、G. Araneda 和 D. Lucas，《跨光网络链路的分布式量子计算》，《自然》，第1–6页，2025年。[37] Z. Mo、Y. Li、A. Pawar、X. Tang、J. Yang 和 Y. Zhang，《FCM：一种面向融合的基于测量的量子计算线路切割方法》，载于《第*61*届*ACM/IEEE*设计自动化会议》，2024年，第1–6页。[38] S. A. Moses、C. H. Baldwin、M. S. Allman、R. Ancona、L. Ascarrunz、C. Barnes、J. Bartolotta、B. Bjork、P. Blanchard、M. Bohn、J. G. Bohnet、N. C. Brown、N. Q. Burdick、W. C. Burton、S. L. Campbell、J. P. Campora、C. Carron、J. Chambers、J. W. Chan、Y. H. Chen、A. Chernoguzov、E. Chertkov、J. Colina、J. P. Curtis、R. Daniel、M. DeCross、D. Deen、C. Delaney、J. M. Dreiling、C. T. Ertsgaard、J. Esposito、B. Estey、M. Fabrikant、C. Figgatt、C. Foltz、M. Foss-Feig、D. Francois、J. P. Gaebler、T. M. Gatterman、C. N. Gilbreth、J. Giles、E. Glynn、A. Hall、A. M. Hankin、A. Hansen、D. Hayes、B. Higashi、I. M. Hoffman、B. Horning、J. J. Hout、R. Jacobs、J. Johansen、L. Jones、J. Karcz、T. Klein、P. Lauria、P. Lee、D. Liefer、S. T. Lu、D. Lucchetti、C. Lytle、A. Malm、M. Matheny、B. Mathewson、K. Mayer、D. B. Miller、M. Mills、B. Neyenhuis、L. Nugent、S. Olson、J. Parks、G. N. Price、Z. Price、M. Pugh、A. Ransford、A. P. Reed、C. Roman、M. Rowe、C. Ryan-Anderson、S. Sanders、J. Sedlacek、P. Shevchuk、P. Siegfried、T. Skripka、B. Spaun、R. T. Sprenkle、R. P. Stutz、M. Swallows、R. I. Tobey、A. Tran、T. Tran、E. Vogt、C. Volin、J. Walker、A. M. Zolot 和 J. M. Pino，《一种赛道式囚禁离子量子处理器》，《物理评论*X*》，第13卷，第4期，文章041052，2023年。[39] M. E. J. Newman，《用于快速检测网络社区结构的算法》，《物理评论*E*》，第69卷，文章066133，2004年6月。[40] M. A. Nielsen，《使用簇态的光量子计算》，《物理评论快报》，第93卷，第4期，2004年7月。[41] S. Paesani、Y. Ding、R. Santagati、L. Chakhmakhchyan、C. Vigliar、K. Rottwitt、L. K. Oxenløwe、J. Wang、M. G. Thompson 和 A. Laing，《硅芯片上光量子态的生成与采样》，《自然·物理》，第15卷，第9期，第925–929页，2019年。[42] C. H. Papadimitriou，《带宽最小化问题的NP完全性》，《计算》，第16卷，第3期，第263–270页，1976年。[43] T. Peng、A. W. Harrow、M. Ozols 和 X. Wu，《在小型量子计算机上模拟大型量子电路》，《物理评论快报》，第125卷，第15期，2020年10月。[44] A. Peruzzo、J. McClean、P. Shadbolt、M.-H. Yung、X.-Q. Zhou、P. J. Love、A. Aspuru-Guzik 和 J. L. O'brien，《光子量子处理器上的变分本征值求解器》，《自然·通讯》，第5卷，第1期，文章4213，2014年。

[45] M. Pinedo，《调度：理论、算法与系统》，第 6 版。Cham：施普林格，2022。[46] J. Preskill，“超越含噪中型量子：百万量子运算机器”，*arXiv* 预印本 *arXiv:2502.17368*，2025。[47] PsiQuantum，“光子量子计算的可制造平台”，《自然》，第 1–3 页，2025。[48] R. Raussendorf 和 H. J. Briegel，“单向量子计算机”，《物理评论快报》，第 86 卷，第 22 期，第 5188 页，2001。[49] R. Raussendorf、D. E. Browne 和 H. J. Briegel，“簇态上的基于测量的量子计算”，《物理评论 *A*》，第 68 卷，第 2 期，第 022312 页，2003。[50] S. Ritter、C. N¨olleke、C. Hahn、A. Reiserer、A. Neuzner、M. Uphoff、M. M¨ucke、E. Figueroa、J. Bochmann 和 G. Rempe，“光学腔中单原子构成的基本量子网络”，《自然》，第 484 卷，第 7393 期，第 195–200 页，2012。[51] P. W. Shor，“量子计算机上用于素因数分解和离散对数的多项式时间算法”，《*SIAM*评论》，第 41 卷，第 2 期，第 303–332 页，1999。[52] J. Sinclair、J. Ramette、B. Grinkemeyer、D. Bluvstein、M. Lukin 和 V. Vuleti´c，“面向中性原子阵列的容错光互连”，*arXiv* 预印本 *arXiv:2408.08955*，2024。[53] L. Stephenson、D. Nadlinger、B. Nichol、S. An、P. Drmota、T. Ballance、K. Thirumalai、J. Goodwin、D. Lucas 和 C. Ballance，“在基本量子网络中实现量子比特间高速率、高保真度纠缠”，《物理评论快报》，第 124 卷，第 11 期，第 110501 页，2020。[54] R. Stockill、M. Stanley、L. Huthmacher、E. Clarke、M. Hugues、A. Miller、C. Matthiesen、C. Le Gall 和 M. Atat¨ure，“远距离自旋量子比特间的相位调谐纠缠态生成”，《物理评论快报》，第 119 卷，第 1 期，第 010503 页，2017。[55] W. Tang、T. Tomesh、M. Suchara、J. Larson 和 M. Martonosi，“CutQC：利用小型量子计算机评估大型量子电路”，载于第 *26* 届 *ACM* 编程语言与操作系统架构支持国际会议论文集，ASPLOS '21 系列。ACM，2021 年 4 月，第 473–486 页。[56] V. A. Traag、L. Waltman 和 N. J. van Eck，“从 Louvain 到 Leiden：确保社区强连通”，《科学报告》，第 9 卷，第 1 期，2019 年 3 月。[57] W. Unger，“带宽问题近似的复杂性”，载于第 *39* 届计算机科学基础年度研讨会论文集（目录号 *98CB36280*），1998 年，第 82–91 页。[58] C. Young、A. Safari、P. Huft、J. Zhang、E. Oh、R. Chinnarasu 和 M. Saffman，“中性原子处理器量子联网的架构”，《应用物理 *B*》，第 128 卷，第 8 期，第 151 页，2022。[59] H. Zhang、J. Ruan、H. Shapourian、R. R. Kompella 和 Y. Ding，“OnePerc：面向光子量子计算的随机性感知编译器”，载于第 *29* 届 *ACM* 编程语言与操作系统架构支持国际会议论文集，第 *3* 卷，2024 年，第 738–754 页。[60] H. Zhang、J. Ruan、D. Tullsen、Y. Ding、A. Li 和 T. S. Humble，“OneAdapt：面向资源受限的光子单向量子计算的自适应编译”，2025。[61] H. Zhang、A. Wu、Y. Wang、G. Li、H. Shapourian、A. Shabani 和 Y. Ding，“OneQ：光子单向量子计算的编译框架”，载于第 *50* 届计算机体系结构年度国际研讨会论文集，2023 年，第 1–14 页。