一种量子定价列生成框架，用于解决困难的组合问题

Wesley da Silva Coelho,[1], * Lo¨ıc Henriet,[1] 和 Louis-Paul Henry[1]
[1] *PASQAL SAS, 7 rue Léonard de Vinci, 91300 Massy, France*
（日期：2023年4月5日）

本文提出了一种完整的经典-量子混合算法，其中包含一个基于中性原子平台的量子采样器。该方法受到《运筹学》领域所开发的经典列生成框架的启发，展示了量子程序如何协助经典求解器处理困难组合问题。我们在最小顶点着色问题上对该方法进行了基准测试，并表明所提出的经典-量子混合列生成算法能够在相对较少的迭代次数内得到良好的解。我们将结果与最先进的经典与量子方法进行了比较。

arXiv:2301.02637v2 [quant-ph] 4 Apr 2023

引言

组合优化是许多现实世界问题的核心。其目标是在有限但规模大到难以处理的选项集合中，找出“最佳”选项。列生成 [1] 是一种迭代方法，旨在解决这类困难的数学问题，例如线性形式中的问题，因为此类问题可能过于庞大，无法显式考虑所有选项。在这种方法中，每个选项都对应一个变量。算法首先使用有限的一组变量（或选项）求解所考虑的问题，这称为受限主问题（RMP），随后迭代地添加变量，以改进目标函数。新变量由专门针对这一任务设计的算法生成：在每次迭代中，需要求解一个相关的新子问题，通常称为定价子问题（PSP）；它依托对偶理论 [2] 生成新变量（如果存在），且仅当这些变量能够改进受限主问题当前解时才予以生成。当新变量不再能改进目标函数时，迭代过程便会停止，而这一点可以通过数学证明。然而，求解定价子问题通常是列生成（CG）方法的瓶颈，因为该方法需要求解多个较简单但仍然困难的优化问题。因此，设计高效的定价子问题求解方法，是在尽量减少时间和资源消耗的同时，确保受限主问题获得高质量解的关键。

过去几年，学术界和工业界都投入了大量精力，致力于设计能够相较于经典计算机提供实际优势的量子硬件和算法。正如 [3],所指出的，这种优势可以表现为结果更准确、收敛更快，甚至能耗更低。这类量子算法可以与最先进的经典解决方案（例如列生成算法）结合使用，从而构建强大的经典-量子混合框架。目前，人们正在开发各种量子计算平台，它们采用不同类型的二能级系统作为量子比特，包括约瑟夫森结 [4, 5],、囚禁离子 [6, 7], 、光子 [8], 或中性原子 [9, 10]。这些平台各自支持不同的量子处理单元（QPU）架构，在量子比特的连通性以及易于实现的操作类型方面各有优缺点。深入了解这些平台有助于开发专为各个平台设计的硬件高效方法

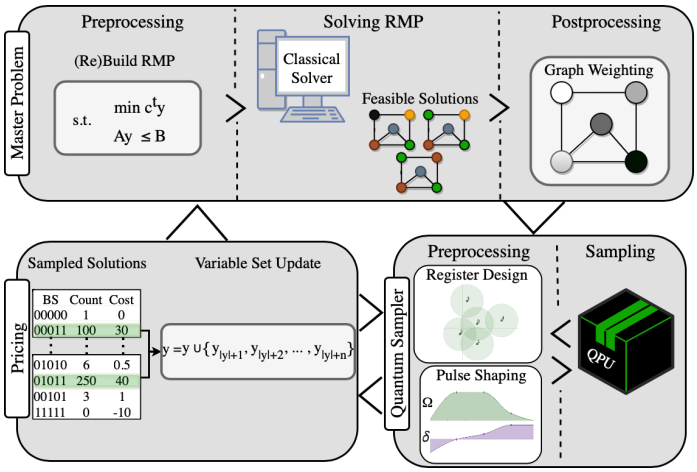


图 1：经典-量子混合列生成方法的工作流程。首先，生成一个最小变量子集 y，以确保缩减主问题有可行解（例如，仅包含图中的单元素集合）。随后由经典求解器求解受限主问题。接下来的步骤与定价子问题有关：利用已求解的受限主问题中的对偶值，寻找可能改善当前受限主问题解的更多变量。如果存在此类变量，就将这些新变量加入受限主问题并再次求解。新变量由量子采样器搜索，该采样器经过专门设计，可处理与每次定价迭代相关的不同输入。重复这些最后的步骤，直到定价子问题不再生成任何列。

其中的部分内容。特别是，这使我们能够识别最适合由量子方法取代的经典瓶颈。

在本研究中，我们提出了一种完整的经典-量子混合列生成框架，其定价子问题可在中性原子量子处理器上高效求解。与量子近似优化算法等其他混合方法不同，求解的核心部分由经典求解器完成，而量子处理单元则用作采样器，以缩小搜索空间。对于给定的图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$，仅需 $|V|$ 个量子比特，即可由一种专门用于改进相关主问题当前解的中性原子采样器求解相应的定价子问题。与经典与量子贪心方法相比，我们通过数值实验表明，所提出的混合列生成方法能够显著提高解的质量，同时减少量子设备上的迭代次数。最后，通过利用一些量

* wesley.coelho@pasqal.com

特征（例如，状态叠加），我们发现，混合列生成方法比经典方法更快地返回（近似）最优解（即，不涉及量子处理单元（QPU）的情况）。图1 总结了我们提出的方法。

本文结构如下：我们首先在第 I 节介绍图论的主要方面及相关组合问题。随后在第II 节回顾相关研究，再在第 III 节简要介绍基于中性原子的量子计算。第 IV 节将介绍列生成方法的主要思想。在第 V 节，我们将深入介绍所提出的用于求解最小顶点着色问题（MVCP）的经典-量子混合方法，并在第 VI 节讨论数值实验结果。

I. 背景

组合问题 [11]已受到学术界和工业界的广泛研究，并在现实世界的系统中有着广泛的应用。这些问题可以自然地定义在图上。图是一种数据结构，由一组称为顶点 （也称为节点）的元素组成，这些元素之间可能存在连接。这些连接称为边 ，并可能编码不同的信息，例如连接的重要性（以权重表示）或连接端点之间的距离。同样，也可以为顶点赋予不同的标签和权重，以便对其加以区分。形式上，图 $\mathcal{G}=(\mathcal{V},\mathcal{E})$ 由顶点集 $\mathcal{V}$ 和边集$\mathcal{E}\in\mathcal{V}^2$ 组成，其中边集表示顶点 u 与 v 之间存在连接，且这些顶点属于 $\mathcal{V}$。

来自广泛领域的诸多现实世界优化问题，都可以映射为图问题。例如，图可用于对社会实验 [12],电信网络 [13], 和物理系统 [14]进行编码。相关优化问题通常要求选择顶点和/或边的一个子集，使其在满足特定规则的前提下达到最优解。这类离散优化问题与量子计算（QC）密切相关，尤其是在含噪中等规模量子（NISQ）时代的平台上 [15, 16]。在这种情况下，通常通过对系统最终状态进行重复测量来获取结果。随后，通过计算成本低廉的经典后处理，从采样得到的状态中选出最佳状态，以此推断解。

对于中性原子量子处理单元，可以对量子比特进行空间排布，使描述系统中相互作用的伊辛哈密顿量与待最小化的给定成本函数密切相关。这使得该平台特别适合求解图组合问题 [10, 17–19]。由于测量量子比特时所处的计算基状态与图问题的解直接对应，这类量子计算尤其能抵御噪声（噪声甚至可能成为一种优势 [20]）。例如，通过绝热退火 [17, 22] 或类似方法逼近量子系统的基态，可以高效求解最大独立集 [21]和最大割 [3] 问题。下文将正式定义一些图问题，并展示如何使用基于量子的方法求解这些问题。

A. 组合问题

下文将介绍最大独立集，它是我们混合方法概念验证的重要组成部分。随后，我们将正式定义顶点着色问题，并介绍相关的数学表述。

1. *Maximum Independent Set problem*

图 $\mathcal{G}=(\mathcal{V},\mathcal{E})$中的独立集是顶点 $\tilde{\mathcal{V}}\subset\mathcal{V}$ 的一个子集，其中任意一对元素都不通过边相连。 $\mathcal{G}$ 的独立集可形式化定义如下：

$$IS_{\mathcal{G}}=\left\{\tilde{\mathcal{V}}\subset\mathcal{V}\;\middle|\;\tilde{\mathcal{V}}^2\cap\mathcal{E}=\emptyset\right\} \tag{1}$$

其中， $\tilde{\mathcal{V}}^2$ 是连接$\tilde{\mathcal{V}}$中各顶点的所有可能边。因此，最大独立集（MIS）是最大的 IS_G集合：

$$\mathrm{MIS}(\mathcal{G})=\underset{\tilde{\mathcal{V}}\in IS_G}{\operatorname{argmax}}\,|\tilde{\mathcal{V}}|. \tag{2}$$

最大独立集问题也可以用二次无约束二进制优化（QUBO）形式来描述。考虑一个图 $\mathcal{G}=(\mathcal{V},\mathcal{E})$。令$x_u$ 为与每个顶点 $u\in\mathcal{V}$对应的二元变量：若选择顶点 u 加入独立集，则该变量取值为 1，否则取值为 0。因此，二进制向量 $\mathbf{x}=\{x_1,\ldots,x_{|\mathcal{V}|}\}$ 与顶点集 $\mathcal{V}$ 的划分 $\mathcal{V}$ 之间可以建立一一对应关系，具体如下：

$$\tilde{\mathcal{V}}(\mathbf{x})=\left\{u\in\mathcal{V}\;\middle|\;x_u=1\right\} \tag{3}$$

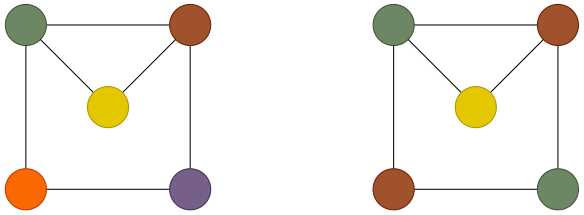
最大（加权）独立集问题的解为：

$$\mathrm{MIS}(\mathcal{G})=\underset{\mathbf{x}\in\{0,1\}^{|\mathcal{V}|}}{\operatorname{argmin}}\left(-\sum_{u\in\mathcal{V}}w_u x_u+\alpha\sum_{\{u,v\}\in\mathcal{E}}x_u x_v\right) \tag{4}$$

其中，参数 w_u 表示与每个顶点 $u\in\mathcal{V}$相关联的权重，而 $\alpha>0$ 是用于惩罚不可行解的任意系数。请注意，对于无权图，所有 w 参数都设为 1，而惩罚系数 α 必须取大于或等于 2 的值。

2. *Minimum Vertex Coloring problem*

顶点着色问题在网络设计 [23]和任务调度 [24]等现实世界优化问题中有多种应用。顶点着色是指为图中的每个顶点分配一种颜色（或标签），使任意两个颜色相同的顶点之间没有边相连。最小顶点着色问题就是在寻找一种可行着色方案的同时，尽可能减少分配的颜色（或标签）数量；为给定图 $\mathcal{G}$ 中的所有顶点着色所需的最少颜色数称为其色数，以下记作 $\mathcal{X}(\mathcal{G})$。最小顶点着色问题的形式化定义如下：



(a) (b)

图2：同一个图的两种着色解，该图有5个顶点、6条边，且 $\mathcal{X}(\mathcal{G}) = 3$。集合 C 有5种可用颜色：绿色、棕色、橙色、紫色和黄色。图2a展示了一种平凡着色，其中每种颜色恰好映射到一个顶点；而图2b 则展示了同一实例的最优解。

定义1 设 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ 是一个图，其顶点集为 $V = \{u_1, .., u_{|\mathcal{V}|}\}$，边集为 $\mathcal{E} \subset \mathcal{V}^2$。另设 C 为可用颜色集。最小顶点着色问题要求为 $\mathcal{G}$中的每个顶点恰好分配一种来自 C 的颜色，使所用颜色数量最少，同时确保任意两个相邻顶点的颜色都不相同。

图2 展示了同一个图的两种可能着色解。采用平凡着色（见图 2a）时，每种颜色恰好映射到一个顶点；这种简单的方法总能为该问题提供一个可行解。然而，如图 2b 所示，相关的色数（即最优解）可以降至3。值得注意的是，对于顶点着色问题的一个可行解，任何由相同颜色着色的顶点子集也都是一个独立集。因此，寻找能够覆盖给定图中所有顶点的最小独立集子集 $\mathcal{G}$，等价于在同一个图中求解 MVCP。然而，求一个图的色数是Karp的21个NP完全问题之一 [25]。

B. 最小顶点着色问题的扩展形式

现在，我们给出最小顶点着色问题的扩展形式[1] ，并在所提出的混合方法中使用该形式。首先，令 S 为图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$中所有可能的独立集的集合。再令 b_{us}为一个二元参数：若顶点 $u \in V$ 属于独立集 $s \in S$，则其值为 1；否则为 0。最后，为每个独立集 $s \in S$关联一个二元变量 y_s；若选中相应的独立集，则该变量取值为 1；否则为 0。因此，求解 MVCP 就是求解以下扩展形式：

$$min \sum_{s \in S} y_s \tag{5}$$

满足条件：

$$\sum_{s \in S} b_{us} y_s = 1, \qquad \forall u \in \mathcal{V} \tag{6}$$

$$y_s \in \{0, 1\}, \qquad \forall s \in S \tag{7}$$

其中（5）被设定为：在确保图中的每个顶点恰好存在于其中一个独立集中的同时，最小化所选独立集的数量（见公式（7））。请注意，将每个独立集 $s \in S$ 视为一种颜色分配后，与最小顶点着色问题相关的邻接约束会自动得到满足（见定义（1））。

图中所有独立集的数量，以及相应的 y_s 变量数量，可能极其庞大，并且会随着图中顶点数的增加而呈指数增长。因此，正如预期的那样，在给定图中找出所有此类独立集是一项非常困难的任务，即使对于较小的实例，也可能耗费大量时间和资源。为克服上述局限性，我们提出了一种混合经典-量子列生成框架，通过仅枚举少量独立集来高效求解所提出的扩展形式。下文将介绍相关工作，并讨论如何将量子采样器集成到经典框架中。

II. 相关工作

过去几年中，人们提出了多种用于求解图着色问题的量子算法，其中大多数依赖量子退火方法。在 [26],中，作者研究了一种基于实时量子动力学的量子退火方法，其相关哈密顿量经过设计，无需添加惩罚项即可自然满足所有与问题相关的约束。另一方面， [27], 中的作者提出了一种基于遗传算法的量子方法，在多种高度可配置的基于电路的模型中求解顶点着色和边着色问题。Titiloye 和 Crispin [28] 比较了求解图着色问题的经典方法与量子退火方法。作者指出，基于路径积分蒙特卡罗的量子退火（QA）算法优于相应的经典算法。

[29] 中的作者提出了另一种方法：将与问题相关的一组约束转化为能量最小化问题，得到 QUBO 公式，再在量子退火器平台上求解。然而，通过运行多种数值模拟，并比较标准版与增强版基于电路的 QAOA 算法所得结果， [30] 中的作者指出，现有的量子退火硬件方案在求解顶点着色问题方面存在局限。另据 Silva 等人 [31]，他们比较了模拟退火与量子退火方法，用于求解针对顶点着色问题提出的 QUBO 公式。作者将 D-Wave 2X 用作简单贪心框架中的独立集采样器，并指出，由于硬件限制，在小规模图上，所提出的量子采样器有很大概率能够改善结果。

Fabrikant 和 Hogg [32] 提出了一种用于图着色的量子启发式算法，适用于最多可用 3 种颜色求解的实例。每个图顶点使用两个量子比特，近似渐近分析表明，其时间复杂度为多项式时间

[1] 扩展形式是指数型数学模型，其中变量数会随着输入规模的增加而呈指数增长。

即可求解相关 3着色问题的开销。此外， [33] 提出了一种指数空间量子算法，可在 $O(1.9140^{|\mathcal{V}|})$运行时间内求解 MVCP。他们提出了一种基于 Ambainis 量子动态规划的量子随机存取存储器框架 [34]，并将 Grover 搜索应用于分支算法。此外， [35]提出了一种贪心量子算法，通过迭代计算最大独立集问题的解来求解 MVCP。他们在经典计算机上模拟一个框架，以再现中性原子量子处理器上的里德堡阻塞现象，并表明该方法总能找到该问题的可行解。关于该方法的更多详情，请参见附录A。

最后，作者在 [36] 中提出了一种基于量子退火的方法，用于求解炼油厂调度问题的基于列生成的方法中的两个定价子问题之一。他们首先将该问题分解为一个主问题和两个定价子问题，并将其中一个表述为 QUBO 模型。虽然只有与 QUBO 相关的定价子问题使用量子退火方法求解，但主问题和另一个定价子问题则由经典求解器求解。该量子系统基于 D-Wave 的量子退火器构建，旨在仅返回最优解。由于受到问题规模或其变量连通性等方面的一些严重限制，作者只能针对少数小规模图实例保证获得高质量解。

需要注意的是，尽管这些研究提出了用于求解组合问题（有时仅涉及判定版本）的有趣方法，但人们对混合与模拟方法关注甚少，尤其是对使用基于中性原子的量子处理单元的研究。下文将介绍一个完整的量子定价框架，该框架基于中性原子量子处理单元构建，可以轻松嵌入列生成算法，用于求解困难组合问题，例如图着色问题。

III. 中性原子量子处理器

在基于中性原子的量子处理器中，使用激光器或微波来诱导碱金属（通常为铷）原子价电子的电子态之间发生跃迁。可以选用不同的电子能级对作为量子比特。在此，我们仅关注这样一种情形：这两个态分别是电子基态 $|g\rangle \equiv |0\rangle$ 和一个 $s-$ 里德堡能级 $|r\rangle \equiv |1\rangle$。在这种情况下，原子可以任意放置在空间中，因此，原子在时间 t 的有效哈密顿量可写为

$$H(t) = \Omega(t)\sum_{u=1}^{|\mathcal{V}|}\hat{\sigma}_u^x - \Delta(t)\sum_{u=1}^{|\mathcal{V}|}\hat{n}_u + \sum_{u<v=1}^{|\mathcal{V}|}U_{uv}\hat{n}_u\hat{n}_v, \quad (8)$$

其中，激光的振幅（决定拉比频率）$\Omega(t)$和失谐 $\Delta(t)$均可随时间调控，而相互作用强度 $U_{uv} \propto |\mathbf{r}_u - \mathbf{r}_v|^{-6}$ 是原子 u 与原子 v之间距离的函数。在本文中，我们设定 $\hbar = 1$。请注意，在当前工作中，我们仅考虑对这些原子进行均匀全局激光控制。

里德堡物理的一个关键特性是所谓的里德堡阻塞机制 [9]：式（8）中的二体相互作用项禁止两个距离近于某一特定阈值的原子同时被激发。给定一组原子及其位置，便可定义一个图，使每个原子对应一个顶点，并且当且仅当相关原子之间的距离小于给定阈值时，两个顶点之间才有一条边。这类图称为单位圆盘图，是最适合在中性原子量子处理器上编码的图。对于这一图类，可以确保量子系统的演化被限制在完整希尔伯特空间的一个子空间内，该子空间对应于图的独立集。通过设定 $\Omega(t) = 0$ 并调整 Δ 的值，可以确保基态对应于一个最大独立集。基于这些特性，人们探索了将量子退火作为求解图上优化问题的一种方式 [37]。然而，对于非单位圆盘图，可以构造替代方法，类似于量子近似优化算法或变分量子本征求解器中采用的方法 [38]。

IV. 问题分解

本节将完整介绍所提出的扩展形式（5）-（7）的分解，这是使用基于列生成的算法求解相关组合优化问题的关键步骤。之所以需要采用这种数学策略，是因为在大多数情况下，生成与扩展形式变量相关的所有元素是一项非常困难的任务。例如，即使对于规模较小的实例，枚举图中的所有独立集也可能不切实际。为解决上述问题，我们将所研究的问题分解为两个问题，分别称为受限主问题 （RMP）和定价子问题 （PSP）。前者仅包含求解问题所需变量的一小部分，而后者旨在生成满足 RMP 所施加的所有技术约束的新元素（例如独立集）。随后，这些新元素将作为新变量（也称为列）添加到与 RMP 相关的数学模型中，因为它们有可能提高解的质量（例如，减少求解顶点着色问题所需的颜色数量）。

基于列生成的方法依赖于对偶理论 [39], 该理论指出，优化问题可以从两个不同的视角进行处理：原问题 或其对应的对偶问题。这两个问题之间的关系如下：(i) 原问题中的每个变量（分别对应每个约束）在对偶问题中都有一个相关的约束（分别对应变量）；(ii) 对偶问题中的优化方向（例如，最大化或最小化）与其原问题对应方向相反。对于原问题中满足所有约束的每个次优解，至少存在一个移动方向，使目标函数得到改善。这些改进方向由对偶变量向量表示，对其进行优化等价于收紧原问题的界。关于原问题与对偶问题关系的深入讨论，可参阅 [39]。求解与原问题相关的线性模型后，只需调用所用求解器提供的某个内置函数，即可轻松获得这样的方向向量（即对偶变量）。

为了设计高效的列生成框架，随后将定价子问题（PSP）表述为能够纳入

受限主问题（RMP）当前解所提供的对偶信息，这意味着每次调用子问题时都要解决一个不同的定价实例。因此，通过应用对偶理论，定价子问题会搜索能够改进受限主问题目标函数的新变量；一旦从数学上证明不再可能生成此类变量（即新列），基于循环的过程便会停止。当给定组合优化问题的最优解中通常只有少数变量处于激活状态（即取非零值）时，这种方法非常有效。因此，采用这种技术时，定价例程只需生成数量极少的变量。

接下来，我们将介绍一种分解方案，用于求解所提出的 MVCP 扩展形式。其核心思路是构建一个仅包含部分独立集的受限模型，并通过迭代添加有望改进当前解的新变量（即列）来更新该模型。

A. 受限主问题

由于 $\mathcal{G}$ 所对应的集合 S 中的独立集表示了数量呈指数级增长的潜在适用着色，因此扩展形式（5）-（7）包含数量呈指数级增长的变量。为克服生成S的困难，我们提出仅生成求解主问题所需的一小部分变量 $S' \subseteq S$ 。例如，一种简单的初始化方法是令 S' 仅包含输入图中的单元素集合。下文将该缩减模型称为受限主问题 （RMP），其定义如下：

$$min \sum_{s \in S'} y_s \tag{9}$$

满足条件：

$$\sum_{s \in S'} b_{us} y_s = 1, \qquad \forall u \in \mathcal{V} \tag{10}$$

$$0 \le y_s \le 1, \qquad \forall s \in S' \tag{11}$$

请注意，为应用对偶理论，我们必须求解 RMP 的线性松弛，这意味着该形式中的 y 变量不再是二元变量。当 RMP 包含了为松弛后的 RPM 提供最优解所需的全部变量后，再加入整数性约束（7）重新求解 RMP；这一点可以通过数学方法证明，证明将在下一节给出。

B. 定价子问题

我们首先给出与松弛受限主问题（9）-（11）相关的对偶公式。为 w_u （10）中的每个约束关联一个对偶变量，我们将对偶问题定义如下：

$$max \sum_{u \in \mathcal{V}} w_u \tag{12}$$

满足约束：

$$\sum_{u \in \mathcal{V}} w_u b_{us} \le 1, \qquad \forall s \in S' \tag{13}$$

$$w_u \in \mathcal{R}, \qquad \forall u \in \mathcal{V} \tag{14}$$

不等式分离（13）表示与扩展形式（9）-（11）相关的定价子问题。因此，PSP 的任务是找到一个新的着色集，使其满足所有邻接约束，同时改善受限主问题的解的代价（即，降低目标函数（9）的值）。为此，令$\bar{w}_u$ 为受限主问题中与约束（10）相关的当前对偶解的分量。将每个对偶变量 $\bar{w}_u$ 设为相应顶点 $u \in \mathcal{V}$的权重后，在这些条件下寻找新的独立集，便转化为求解以下最大权独立集公式：

$$max \sum_{u \in \mathcal{V}} \bar{w}_u x_u \tag{15}$$

满足：

$$x_u + x_v \le 1, \qquad \forall (u,v) \in \mathcal{E} \tag{16}$$

$$x_u \in \{0,1\}, \qquad \forall u \in \mathcal{V} \tag{17}$$

其中， x_u 是一个二元变量，若该顶点属于独立集则取值为 1，否则取值为 0。不等式（16）确保新的独立集满足邻接约束。注意，定价公式（15）-（17）是 QUBO 公式（4）的整数规划（IP）版本。

与定价问题的解相关联的新变量所带来的净收益由约化成本给出。根据不等式分离（13），我们计算 r_s 任意独立集 s 的约化成本，该独立集由求解公式（15）-（17）得到，具体如下：

$$r_s = 1 - \sum_{u \in \mathcal{V}} \bar{w}_u x_u \tag{18}$$

由于本文中的主问题是最小化问题，任何约化成本为负的解都可能改进受限主问题的解。因此，定价问题在于寻找一个新的独立集 s，其总权重严格大于 1；任意独立集的总权重均按成本函数（15）计算。因此，公式（15）-（17）的任何成本大于 1 的解都可以加入子集 S'；如果不存在这样的解，则无法改进受限主问题的解，因此，使用当前与子集相关联的变量 S'即可得到松弛受限主问题的最优解。

V. 求解方法

如前所述，求解定价子问题通常是基于列生成的算法中的瓶颈，因为这需要多次求解一个困难组合优化问题的不同实例。为了解决这一问题，我们提出了一种量子定价算法，它能够比经典算法更快地找到（近似）最优解（即，不涉及 QPU 的算法）。为此，让

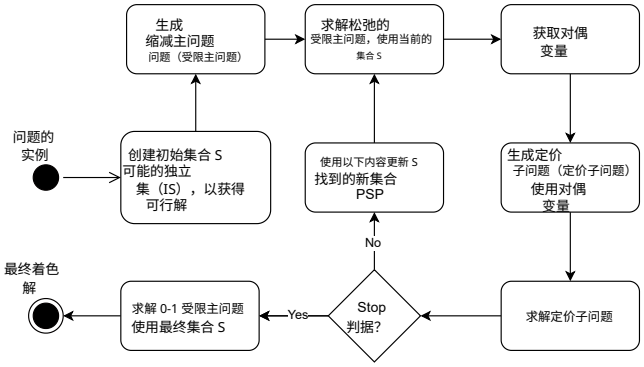


图 3：受限主问题与子问题之间的交互

现在描述为求解最小顶点着色问题而提出的基于列生成的框架；该框架总结于图 3中。

首先，以确保扩展形式（ $S' \subseteq S$ 存在可行解的方式生成一组最小的独立集5）-（7）。如前所述，构建初始独立集集合 S' 最简单的方法是在图中仅生成单元素集合；这一简单方法始终能为受限主问题提供一个解。

所提出混合方法中的经典部分与受限主问题有关。创建初始集合 S' 后，构建受限主问题，然后由经典求解器（见公式（9）-（11））求解其线性松弛形式，例如GPLK。经典求解器还会在求解受限主问题的每个版本（即变量子集各不相同的版本）后，通过运行内置例程给出对偶变量的值。

接下来的步骤与定价子问题有关，其中通过将已求解的受限主问题中的相关对偶变量值代入，求解定价子问题。如前所述，这一步旨在寻找权重严格大于 1 的独立集。如果存在此类元素，就将其添加到 S'。如下文所述，我们提出了一种专门设计的量子采样器，可在上述条件下输出多个独立集。对于求解相关定价子问题时找到的每个新独立集，都会创建一个新变量，并将其添加到子集 S'中。然后，使用新增的列（即转换为变量的独立集）再次求解受限主问题。重复上述步骤，直到定价子问题不再生成列。最后，在加入所有生成的变量（即独立集）及整数性约束（7）后，求解最终受限主问题，如前所述。

A. 实例演示

表 I 展示了一个实例演示，说明如何在图 4所示的图上应用列生成框架，其中受限主问题由整数规划求解器采用经典方法求解。第一列显示在得到最终解之前求解了多少个定价子问题。第二列展示了作为受限主问题公式（9）-（11）的解而选出的独立集。第三列给出了变量 w 对应的对偶解（参见对偶公式（12）-（14））。最后一列展示了最大权重独立集，这些独立集是在每次运行定价子问题后生成的，其中公式（ $\bar{w}$15）-(17)成本函数中的参数 w取对偶变量的值；随后将生成的加权独立集添加到集合 S' 中，再次运行受限主问题。最终着色由受限主问题的最后一个解表示，每个选出的独立集对应一种颜色。

在此实例中，我们先将图中的全部五个单元素集合生成为独立集的第一个子集 S'，然后求解第一个版本的受限主问题。如观察到的那样，为找到最优解，只需再生成 4 个独立集（剩余 10 个待生成独立集中的 4 个）。事实上，尽管闭环可在第 4 次迭代后停止，但直到第 5 次迭代才证明最优性，因为此时已无法生成总权重大于 1 的独立集（见最后一行的对偶解，该解用作顶点权重）。请注意，第 i 次迭代中生成的独立集很可能会在下一次迭代的受限主问题解中被选中。

通过求解对偶公式，可以计算出对偶变量的值（12）-（14），其中成本函数设为同一次交互中受限主问题的解的代价。在此示例中，第一、二、三、四和五次迭代的对偶公式成本函数分别设为 5、3、3、2 和 2。然而，大多数商业求解器会并行求解对偶公式，以证明受限主问题原问题解的最优性。因此，内置求解器的函数（例如 $solver.get\ dual()$）可以轻松提供对偶变量的最终值。但请注意，求解相关定价公式（15）-（17）只会生成一个独立集（权重最大的那个）；并且由于这种方法具有确定性，只要输入（即图和顶点权重）不变，最终解总是相同的。

现在，我们通过实例演示量子采样器如何应用于列生成框架中的定价子问题；表 II 展示了这样一个实例演示。正如表 I所示的前一个实例演示中所介绍的，我们考虑图 4 初始子集 S' 仅由单元素集合组成。可以看到，只需三次迭代即可找到最终解。事实上，在每次定价迭代中采样更多独立集，可以显著提升列生成算法的性能。在这个例子中，10 个可能的独立集中生成了 9 个。实际上，由于随机性，量子采样器能够针对相同的输入高效地提供多个集合，从而加快受限主问题收敛至最优解的速度。

接下来，我们将介绍如何设计基于中性原子的系统，使其考虑来自受限主问题的对偶值，从而输出多个加权独立集，并高效求解相关的定价子问题。

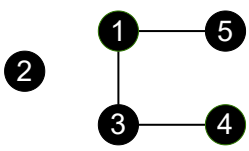


图 4：展示一个包含 5 个顶点和 3 条边的图；由五个单元素集合和子集 [1,2], [1,4], [2,3], [2,4], [2,5], [3,5], [4,5], [1,2,4], [2,3,5], [2,4,5] 组成的所有独立集集合为 S 。

B. 嵌入策略

将随机图嵌入 UD 圆盘表示已被证明是一个 NP 困难问题 [40]。此外，并非所有图都能实现这种表示。例如，在二维平面上，不可能为任何 K_{1n} 具有 $n > 6$的星图找到 UD 圆盘表示。为了设计一个近似最优的寄存器来表示任意给定的输入图，我们采用 [3], 中提出的嵌入策略，其中使用一种基于力导向原理的算法将图嵌入平面，使得相连的（相应地，不相连的）顶点彼此靠近（相应地，远离），两者之间的距离最小（相应地，最大化其与平面中心之间的距离）。更多详情请参见附录B。

如前所述，创建初始寄存器后，其原子数与初始图中的顶点数相同 $\mathcal{G}$。然而，在列生成框架中求解定价子问题，可能需要在子图 $\mathcal{G}'$ 上寻找一个或多个独立集：该子图仅由权重为正的顶点构成（即已求解的受限主问题给出的正对偶值）。将原子固定在原位的光图案由空间光调制器生成，而确定该设备的正确设置需要进行耗时的校准 [19]。

因此，使用剩余顶点运行 Fruchterman-Reingold 算法来创建新寄存器会耗费过多时间。我们不只是移除映射到已不再属于子图 $\mathcal{G}'$ 的顶点上的原子，而是在此提出一种顶点-原子重新映射策略，同时考虑顶点的权重（对偶值）。

算法1总结了我们提出的方法的核心思想：输入一个子图 $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$、一个由原子及其在量子处理单元（QPU）中的位置组成的寄存器 R，以及表示 $\mathcal{V}'$ 中所有顶点权重的向量 W。首先，令 $\mathcal{V}'$ 为 $\mathcal{G}'$ 的顶点列表，并按 W 给出的权重降序排列。然后，从 $\mathcal{V}'$ 中移除权重小于或等于零的所有顶点（步骤2），将权重最大的顶点映射到距离寄存器中心最远的原子（步骤3），并从 $\mathcal{V}'$ 中移除重新映射的顶点（步骤4）。接着，将 $\mathcal{V}'$ 中权重最大的顶点嵌入到与所有已映射至 $\mathcal{G}'$ 中顶点的原子距离最远的位置，并将其从 $\mathcal{V}'$ 中移除；重复这些步骤，直到将 $\mathcal{G}'$ 中的所有顶点映射到寄存器 R 中的原子。最后，如步骤9所示，从 R 中移除所有未映射到任何顶点的原子。需要再次指出，由于所提出的重新映射策略用于求解定价子问题，因此权重向量 W 是根据当前受限主问题（RMP）解提供的对偶值生成的，

迭代	RMP解	对偶解	MWIS
1	[1] , [2] , [3] , [4] , [5]	[1.0, 1.0, 1.0, 1.0, 1.0]	[1, 2, 4]
2	[3] , [5] , [1, 2, 4]	[1.0, 1.0, 1.0, -1.0, 1.0]	[2, 3, 5]
3	[3] , [5] , [1, 2, 4]	[1.0, -1.0, 1.0, 1.0, 1.0]	[1, 4]
4	[1, 4] , [2, 3, 5]	[1.0, -1.0, 1.0, 0.0, 1.0]	[3, 5]
5	[3, 5], [1, 2, 4]	[1.0, 0.0, 1.0, 0.0, 0.0]	None

表 I：在图 4 所示图上应用列生成框架的实例演示。仅生成 5 个独立集（共 10 个）即可找到最优解。4。

迭代 RMP解		对偶解 IS已生成	
1	[1],[2],[3],[4],[5]	[1.0, 1.0, 1.0, 1.0, 1.0]	[1, 2, 4],[2, 4, 5] [2, 3, 5] , [1, 2] [2, 5] , [2, 3] [4, 5] , [3, 5]
2	[3, 5], [1, 2, 4]	[1.0, -0.5, 1.0, 0.5, 0.0]	[1, 4]
3	[1, 4],[2, 3, 5]	[1.0, 0.0, 1.0, 0.0, 0.0]	None

表 II：将量子采样器应用于求解与列生成框架相关的定价子问题的实例演示。考虑图 4 所示的图，在得到最终解之前，仅求解了 3 个不同的定价子问题。

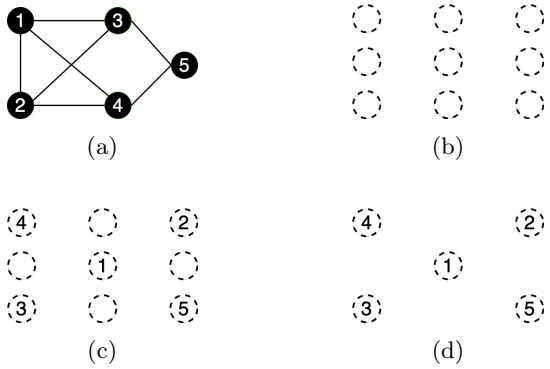


图 5：5顶点图在含 9 个原子的寄存器上的顶点-原子映射。图 5a 展示了一个含 5 个顶点的图，其权重等于对应的索引（例如，顶点 1（分别为顶点 5）的权重等于 1（分别为 5））。图 5b 展示了一个含 9 个原子的寄存器及其位置。图 5c 描绘了一种映射，其中权重最大的顶点彼此相距较远。最后，图 5d 展示了最终的寄存器，以及它们所代表的子图中的顶点。

如前所述，所提出算法的实例演示如图5所示。

C. 独立集量子采样器

我们提出了一种基于当前一代中性原子量子处理单元的独立集采样器，其灵感来自 [41] 所描述的量子绝热方法。后者的核心思想是将系统从易于制备的基态缓慢演化至最终代价哈密顿量 H_C 的基态。通过缓慢演化系统——

算法1 顶点-原子重新映射**输入：** 一个图 $\mathcal{G}'$、一个寄存器 R 和顶点权重 W。**输出：** 缩减后的寄存器 R。1：令 $\bar{\mathcal{V}}'$ 为由 $\mathcal{G}'$ 中的顶点构成的列表，并按 W给出的权重降序排列。2：从 $\bar{\mathcal{V}}'$ 中移除所有权重小于或等于零的顶点。3：将顶点 $\bar{\mathcal{V}}'.first$ 映射到距离 R 中心最远的位置。4：$\bar{\mathcal{V}}' \leftarrow \bar{\mathcal{V}}' \backslash \bar{\mathcal{V}}'.first$ ：**当** 仍有顶点**时，执行** 6：将顶点 $\bar{\mathcal{V}}'.first$ 映射到距离 7 $\bar{\mathcal{V}}' \leftarrow \bar{\mathcal{V}}' \backslash \bar{\mathcal{V}}'.first$ 中已映射到原子的所有顶点最远的位置。8：**结束循环**9：从寄存器 R中移除所有未映射到任何顶点的原子。10：**返回**寄存器 R

，原子会保持在瞬时基态 [42]。在这里，我们的目标只是让系统接近瞬时基态，同时允许它在低能态中获得一些分量。

为了实现这样的演化，我们随时间连续改变失谐 $\delta(t)$ 和拉比频率 $\Omega(t)$：从 $\Omega(t_0) = 0$、$\delta(t_0) < 0$开始，最终达到 $\Omega(t_f) = 0$、$\delta\ t_f\ > 0$ $H\ t_0$ $|00000\rangle$ ( )。因此，( ) 的初始基态为，而 $H(t_f)$ 的低能空间包含代价哈密顿量 H_C 的低能空间。该协议最初旨在制备基态并解决最大独立集问题。在本文中，我们利用它对总权重等于或接近最大权独立集总权重的独立集进行采样。

为确保我们不会将系统激发到不构成独立集的状态，我们必须估计图中不相邻原子之间的最小距离（这给出 Ω_d），并估计两个相连原子之间的最大距离，这给出 Ω_c。为此，令 d_{uv} 表示来自 $\mathcal{V}$ 的节点 u 与 v 之间的距离，而 C_6 是与量子设备相关的相互作用系数。于是，Ω_d 和 Ω_c 分别定义如下：

$$\Omega_d = \underset{(u,v)\notin\mathcal{E}}{\operatorname{argmax}} \quad C_6 d_{uv}^{-6} \tag{19}$$

$$\Omega_c = \underset{(u,v)\in\mathcal{E}}{\operatorname{argmin}} \quad C_6 d_{uv}^{-6} \tag{20}$$

在单位圆盘（UD）图中，保持 $\Omega \in [\Omega_d, \Omega_c]$ 可确保动力学中只出现独立集（前提是动力学不能过快）[37, 43]。较大的 Ω 值有利于加快系统的动力学，并到达与初始状态相距较远的状态。这就是我们将 $\Omega_{max} = max(\Omega_c, \Omega_d)$设为拉比频率在脉冲序列中可以取到的最大值的原因。

对于非单位圆盘图，这个 Ω_{max} 值仍然是一个很好的折中。事实上，如果 $\Omega < \Omega_c$，则编码在系统中的有效图 $\mathcal{G}'$ 包含的边多于目标图 $\mathcal{G}$。于是，$\mathcal{G}'$ 的独立集并不严格包含于 $\mathcal{G}$的独立集，定价可能因此遗漏一些变量。另一方面，如果 $\Omega > \Omega_d$，则 $\mathcal{G}'$ 的所有边也是 $\mathcal{G}$的边，因此，$\mathcal{G}'$ 的独立集严格包含于 $\mathcal{G}$的独立集。这确保定价仍能探索所有独立集，从而提供新的变量来改进当前的受限主问题解。

量子采样器的输入是原子寄存器以及代表所设计哈密顿量的脉冲。随后，在运行所提出的基于中性原子的量子算法后，可能得到不同的输出：

1. 仅返回最大的独立集；2. 返回所有独立集；3. 返回权重大于给定阈值的所有独立集；4. 仅返回权重最大的独立集；

要应用最后两种策略，还必须将权重向量和成本函数作为输入提供给量子采样器。在本研究中，我们通过应用成本函数 (15) 来评估任意独立集，从而采用上述第三种输出情况；请注意，对于无权图，将所有权重设为 1 即可应用此成本函数。

D. 求解定价子问题

令 $\bar{w}$ 为前文所述、与已求解的松弛受限主问题相关的对偶值向量。然后，我们以如下方式生成图 $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \bar{w})$：$\mathcal{V}' = \{u \in \mathcal{V} | \bar{w}_u > 0\}$ 且 $\mathcal{E}' = \{(u, v) \in \mathcal{E} | \bar{w}_u \bar{w}_v > 0\}$。此外，对于每个顶点 $u \in \mathcal{V}'$，其对应的权重为 $\bar{w}_u \in \bar{w}$。

我们在相关图 $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', \bar{w})$ 上运行前文所述的量子算法，其中 $\bar{w}$ 是当前原问题求解结果中的对偶向量。虽然我们只保留与 $\mathcal{V}'$中的顶点相关的原子，但仍可能按照算法 1中的描述对其位置进行置换。此外，还可以通过计算新的 Ω来调整脉冲形状$_{max}$，具体做法是计算每一对量子比特之间的距离，正如前文所述。通过应用目标函数（15）对量子处理单元采样得到的每个比特串进行评估，随后将 S' 更新为所有采样得到的、总权重严格大于 1 的加权独立集。需要注意的是，$\bar{w}$可能由非常小的值组成（即，$\bar{w}_u \ll 1, \forall u \in \mathcal{V}'$包括零值和负值），这意味着任何独立集，包括最大独立集，其总权重都可能严格小于 1：在这种情况下，不会有独立集被添加到 S'中。

E. 停止准则

当求解相关定价子问题后未生成任何列时，算法便停止内层循环。这意味着，当前子集 S' 已经包含了求解松弛形式最优解所需的全部独立集（若 PSP 得到精确求解）（9)-(11)；或者，定价子问题求解器在给定约束下未找到任何解（若 PSP 通过启发式方法求解）。内层循环停止后，将使用所有已生成的列（即变量/独立集）和整数性约束（7）求解最终受限主问题的 ILP 版本。

需要注意的是，相关线性规划形式（9)-(11）给出的最终解，可能与 ILP 形式（5)-(7）的解不同（甚至可能不是最优解）（即，未满足整数性约束（7））。为确保最优性，应将所提出的混合列生成嵌入分支定价框架中。然而，这种方法不在本文的讨论范围之内。更多信息可参阅 [1]。

VI. 数值模拟

首先，我们说明数值模拟中使用的设置。随机图实例由 Vladimir-Brandes 算法生成 [44], ，该算法生成 Erd˝os–R´enyi 图；保证为单位圆盘图的图则按照 [45]中提出的方法生成。值得一提的是，随机图很可能不是单位圆盘图。事实上，随着顶点数量增加且密度保持稳定，随机图为单位圆盘图的 Erd˝os–R´enyi 图的概率会迅速趋近于零。因此，本文以下将这一图类（即随机图）称为非单位圆盘图。对于每个图类，我们针对三种不同的图密度各生成了 30 个实例，设置 p 任意一对顶点之间存在边的概率分别为 20%、50% 和 80%。此外，每个实例的最优解均通过精确求解最小顶点着色问题的紧凑表述而获得

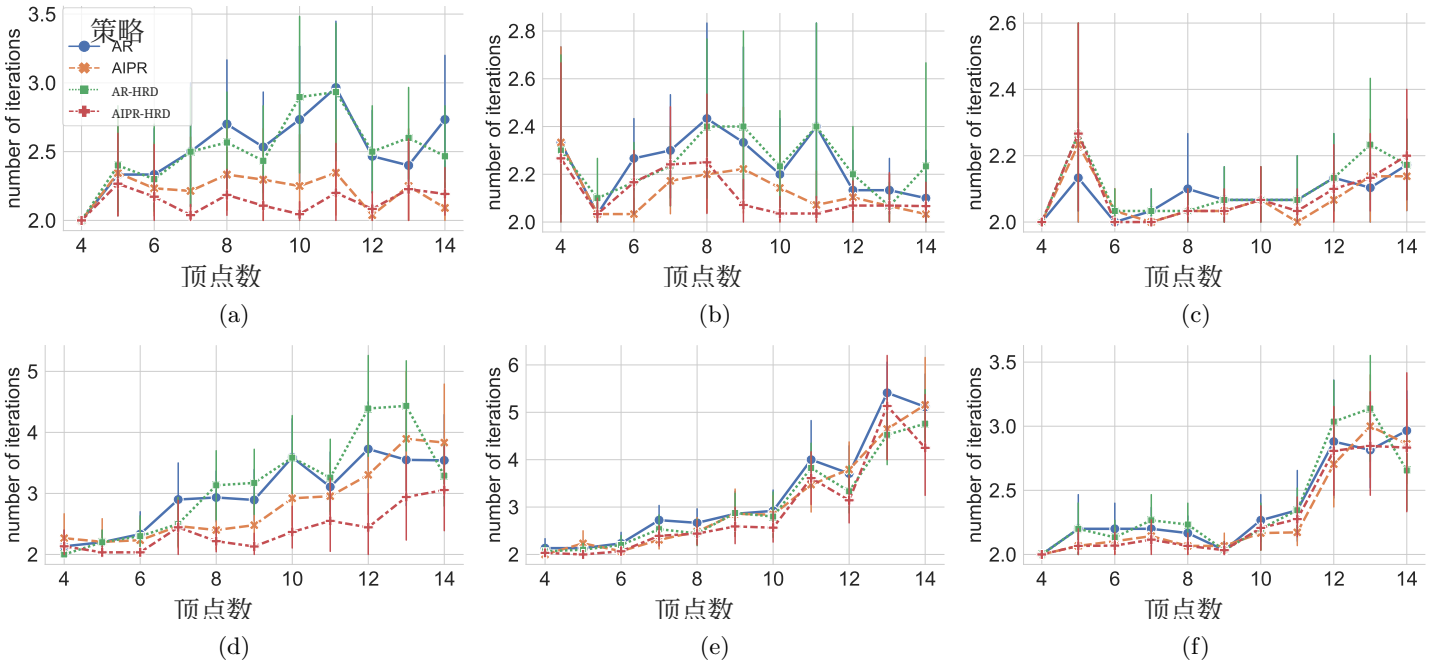


图 6：在不同图类（UD 和非 UD）、阶数（从 4 个到 14 个顶点）和密度（所有可能连接的 20%、50% 和 80%）下，采用不同的寄存器和哈密顿量重新设计策略（仅原子移除（AR）、原子索引排列与原子移除（AIPR）、原子移除与哈密顿量重新设计（AR-HDR），以及 AIPR 与哈密顿量重新设计（AIPR-HDR））时，QPU 模拟器达到停止准则前的迭代次数。图 6a-6c（分别为 6d-6f）展示了单位圆盘（分别为非 UD）图上的结果。此外，图 6a 和 6d （分别为 6b 和 6e）展示了密度为 20%（分别为 50%）的图上的结果。最后，图 6c 和 6f 展示了密度为 80% 的图上的结果。

通过精确求解最小顶点着色问题的紧凑表述 [46]。

我们将提出的混合算法与若干经典和量子领域的最先进方法进行了比较。事实上，可以使用经典求解器求解定价问题（15）-（17），其中 $\mathcal{G}'$ 给定为顶点加权图。不过请注意，每次迭代只会生成最优加权独立集。如前所述，由于该方法具有确定性，以下称为 经典 *CG*，只要输入（即图和顶点权重）保持不变，最终解也不会改变。我们还将提出的方法与附录*A*中描述的经典贪心算法进行了比较，其中每次迭代都由经典求解器提供一个最大独立集。我们使用 GLPK [47]作为线性求解器，以求解（缩减后的）主问题和经典定价子问题。

为了将我们的中性原子量子采样器与其他随机方法进行比较，我们实现了一个贪心生成器，下文称为贪心列生成，它可以随机生成多个加权独立集。更多详情请参见附录 C。我们还将我们的方法与基于模拟退火（SA）的求解器进行了比较，下文称为 模拟退火列生成；该求解器使用经典 D-Wave QUBO 采样器 [48]，最小化式（4）中描述的相关最大权独立集 QUBO 矩阵；权重取自每次迭代中已求解受限主问题的对偶值，而 α 设为权重绝对值之和。对于所有随机子程序，最大尝试次数设为 1000。此外，在受限主问题解得到改进之前，可能需要进行多次定价迭代。这是由于与问题实例相关的解空间具有内在对称性，促使算法生成成本相同的独立集，驱使算法生成成本相同的独立集。在所有基于列生成的方法中，允许在受限主问题解未得到任何改进的情况下进行的最大定价迭代次数设为 3。最后，我们还将混合方法与附录 A中所述贪心算法的量子版本进行了比较；该版本在每次迭代中仅返回第 VC 节所述的量子采样器采样得到的最大独立集。下文将此方法称为 量子贪心算法，它基于 Vitali 等人 [35]提出的工作。

每个定价子问题对应的寄存器均通过应用第VB节介绍的嵌入策略创建。为此，我们使用 弹簧布局函数计算每个原子的位置，该函数来自*Networkx* 软件包 [49]：我们将每个位置向量乘以 40，以满足设备施加的距离约束。此外，我们还应用了第VC节所述的哈密顿量设计策略。

随后，使用 Pulser 的仿真模块 [50]，模拟给定寄存器在预测脉冲作用下的量子系统演化。我们进行了无噪声仿真和含噪声仿真。无噪声仿真通过求解含时薛定谔方程来实现。无噪声仿真的输出是希尔伯特空间中的一个向量，可以对其进行有限次采样，以模拟真实

SPAM	错误制备 η	0.005
	假阳性 ϵ	0.03
	假阴性 ϵ'	0.08
温度		$30\mu K$
激光束腰		$148\mu m$

表III：用于含噪模拟。

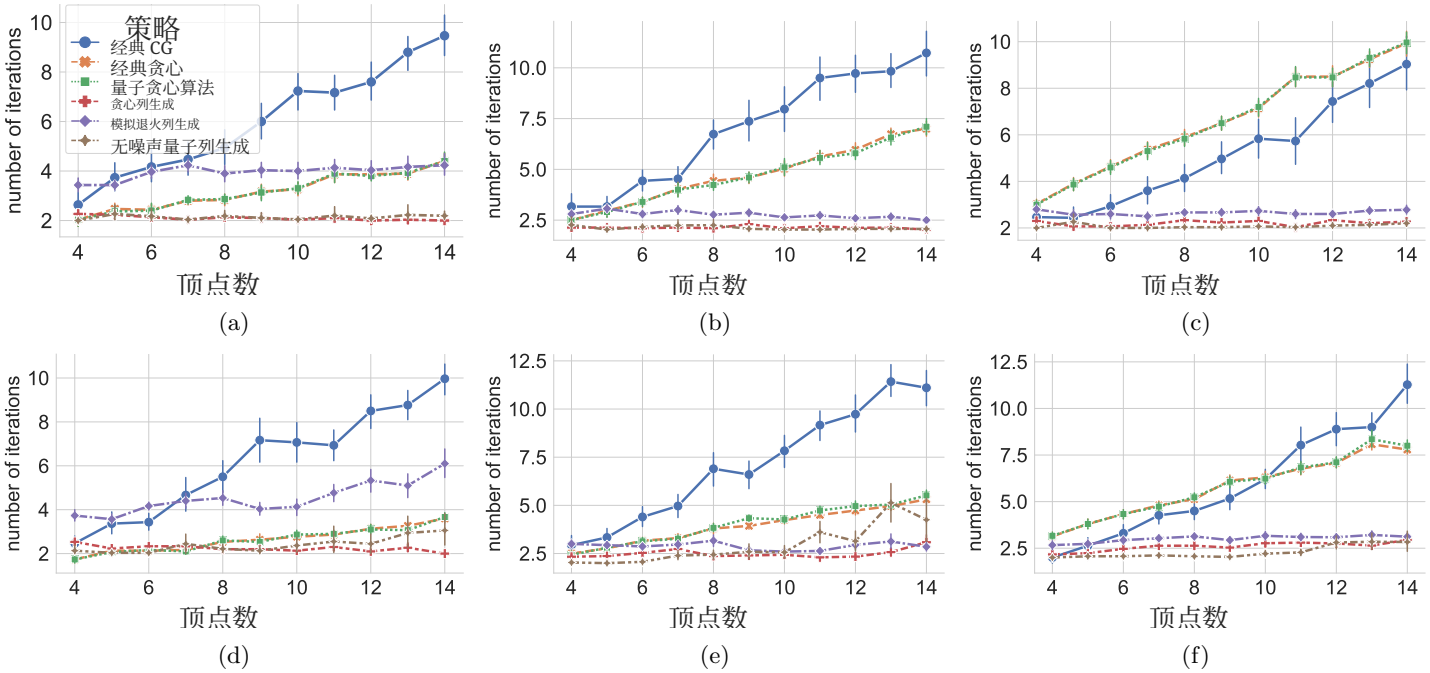


图 7：采用不同方法——经典列生成（CG）、贪心列生成、模拟退火列生成、量子列生成、经典贪心和量子贪心算法——在不同图类（单位圆盘图和非单位圆盘图）、阶数（4 至 14 个顶点）及密度（所有可能连接的 20%、50% 和 80%）上，达到停止准则前的迭代次数。图7a-7c（分别7d-7f）展示了单位圆盘图（分别为非单位圆盘图）的结果。此外，图7a 和7d（分别7b 和7e）展示了密度为 20%（分别为 50%）的图的结果。最后，图7c 和7f 展示了密度为 80% 的图的结果。

实验装置在测量预算有限的情况下的表现。为了评估我们的方法对噪声的鲁棒性，我们还进行了考虑这些系统当前噪声水平的含噪声仿真。 [19, 51]这些计算在数值上开销很大，因此我们将研究范围限定为态制备与测量（SPAM）误差。这些误差预计是噪声的主要来源，可以通过对无噪声结果进行后处理获得（详情请参见 [38]）。我们将噪声参数设定为基于当前硬件规格的现实数值，汇总于表 III。寄存器和哈密顿量的设计分别按照第VB 节和VC节中介绍的方法进行。

A. 寄存器和脉冲重新设计策略

我们首先分析四种不同的寄存器和脉冲（即，哈密顿量）重新设计策略对我们提出的经典-量子混合列生成方法性能的影响：原子移除（AR）策略仅移除相关顶点权重小于或等于零的原子，而原子索引置换与原子移除（AIPR）策略还会应用所提出的顶点-原子重新映射算法（见算法1）。除这两种策略外，我们还测试了原子移除与哈密顿量重新设计（AR-HDR）策略，该策略会在应用 AR 策略后重新计算新的最大拉比频率。最后，在原子索引置换、原子移除与哈密顿量重新设计（AIPR-HDR）策略中，也可以在应用 AIPR 策略后重新计算新的最大拉比频率。

图 6 展示了之前的平均定价迭代次数（以及标准差和 95% 置信区间）在不同图类（UD 和非 UD）、阶数（顶点数从 4 到 14）以及密度（所有可能连接的 20%、50% 和 80%）下达到停止准则。正如观察到的，只需调用量子采样器少数几次（即定价迭代），即可达到与每个图实例相关的主问题松弛版本的最优解[2] 。事实上，非 UD 图（UD 图）的平均迭代次数始终少于 6 次（3 次）；如图6c （图6f）所示，在采用 AIRP（AR-HDR）策略时，平均只需进行 2 次（3 次）采样过程，即可求解顶点数为 4（13）、密度为 80% 的 UD（非 UD）图。尽管非 UD 图似乎更难求解，尤其是密度为 50% 的图（见图6e），采用不同的寄存器和脉冲重设计策略可以加快求解过程。我们观察到，仅从寄存器中移除无用原子后重设计脉冲，并无显著影响（蓝线和绿线分别对应 AR 和 AR-HDR 策略）；而在重新排列原子索引后重新计算拉比频率的最大值（即采用 AIPR-HDR 方法），可将采样过程次数减少 44%（见图6d）。事实上，如图6a 和6d所示，该方法的总体表现最佳，对稀疏图的影响尤为显著。

[2] 请回想一下，对偶变量只能由线性规划生成，这意味着整数性约束（7）被约束（11）取代。

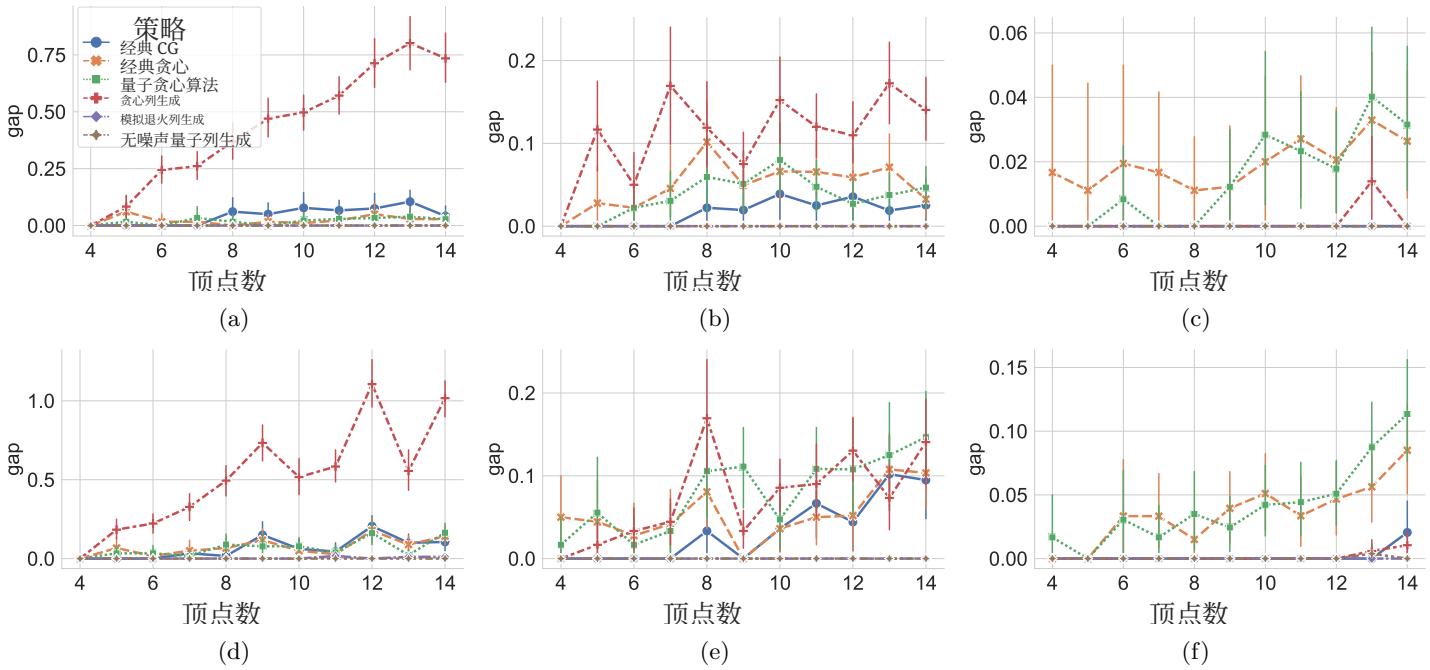


图 8：采用不同方法——经典列生成（CG）、贪心列生成、模拟退火列生成、无噪声量子列生成、经典贪心和量子贪心算法——在不同图类（单位圆盘图和非单位圆盘图）、不同规模（4 至 14 个顶点）及不同密度（所有可能连接的 20%、50% 和 80%）下，所找到的最佳解与最优解之间的差距。图 8a-8c （分别是 8d-8f）展示了单位圆盘图（分别是非单位圆盘图）上的结果。此外，图 8a 和8d （分别是 8b 和 8e）展示了密度为 20%（分别为 50%）的图上的结果。最后，图 8c 和 8f 展示了密度为 80% 的图上的结果。

B. 量子与经典方法

现在，我们比较采用不同方法时，在不同图类、阶数和密度下所需的迭代次数：经典列生成（CG）、贪心列生成、模拟退火列生成、无噪声量子列生成、经典贪心和量子贪心算法。我们在量子列生成框架中，采用 AIPR-HDR 策略对每个定价子问题进行重设计。对于给定图的着色问题，该指标表示在经典和量子列生成框架中求解定价子问题的次数；正如前文所述，它也表示在算法2 的 while 循环中，使用经典和量子方法生成的独立集数量。图7展示了如上所述随机生成的 30 个图的平均迭代次数和标准差（含 95% 置信区间）。

首先，我们观察到，量子贪心方法的整体表现与其经典对应方法相同，这表明我们的量子采样器能够高效地求解最大独立集问题。此外，两种策略的表现都随图的规模（即其所含边数）呈线性变化，且最受稠密图的影响（见图 7c 和 7f）。这种表现符合预期，因为随着边数增加，每个独立集的规模会变小。因此，通常需要进行更多迭代，才能覆盖稠密图中的所有顶点。此外，经典贪心算法和量子贪心算法在几乎所有图类上的表现均优于经典 CG 方法（按迭代次数衡量），但在 UD 图上的表现略有下降。最后，利用相关的叠加特性，所提出的量子列生成在所有图类上的表现均优于其他方法。例如，对于所有稀疏和稠密（以及密度为 0.5 的）非单位圆盘图，量子列生成算法所需的采样迭代次数不到 4 次（分别不到 6 次），而量子和经典贪心方法（以及经典 CG 算法）则需要最多 10 次（分别最多 12 次）定价交互才能求解同一图类。对于所有稀疏和稠密（分别为密度 0.5）的非单位圆盘图，量子定价算法所需的采样迭代次数少于 4 次（分别少于 6 次）；而量子和经典贪心方法（分别为经典 CG 算法）则最多需要 10 次（分别最多 12 次）定价交互来求解同一图类。

图 8展示了平均间隙[3] （以及 95% 置信区间下的标准差），即最优解与采用所述方法找到的最佳解之间的差距。首先可以看出，我们提出的量子列生成方法总体表现最佳。事实上，它几乎能在所有实例中找到最优解；如图 8f所示，我们的方法仅未能在部分 13 个顶点的非单位圆盘图中找到最优解。此外，与其他所有方法不同，量子列生成不受图类影响；经典列生成在稠密图上的表现更好（见图 8c 和8f），而经典与量子贪心方法在单位圆盘图上的表现则更稳定。此外，如图 8d（分别是图 8a）所示，与量子贪心方法（分别是经典列生成）相比，所提出的量子列生成算法将 12 个顶点的非单位圆盘图（分别是 13 个顶点的单位圆盘图）的平均间隙从约 19%（分别是 11%）降至 0%。

我们提出的基于量子定价的方法，在大多数

[3] 本文中，差距按如下方式计算：对于给定的问题实例，设最优解的值为 x_o （即为图着色所需的最少颜色数，也称为图的色数），运行给定方法后得到的解的值为 x_m （即用于图着色的颜色数），则差距为 $x_o - x_m$。换言之，x_o差距表示任一方法找到的给定解与最优解之间的距离。

实例中也优于两种随机经典方法，尤其是在与 UD 图和稀疏图相关的实例中。尽管解的质量保持不变（见图 8），与基于 SA 的定价相比，无噪声量子列生成在稀疏图上的迭代次数可减少 50%，如图 7a 和 7d所示。尽管贪心列生成在某些图类上的定价迭代次数较少，例如密度为 20% 和 50% 的较大非单位圆盘图（见图 7d 和 7e），但在相同图类上应用我们提出的量子列生成后，平均间隙可减少 80%，如图 8a 和 8d所示。这表明，通过随机采样寻找独立集无法有效求解定价子问题。

C. 含噪与无噪模拟

现在，我们展示数值模拟的结果，其中加入了前文讨论过的模拟噪声。图 9展示了采用不同方法时，在达到停止准则前的迭代次数，以及最终解与最优解之间的差距。图阶范围为 4 至 14 个顶点，所采用的方法包括：经典 CG、贪心列生成、模拟退火列生成、无噪声量子列生成，以及含噪量子列生成。所示结果针对各方法应用于所有具有相同图阶的图实例，不论其密度如何，也不论它们是否为单位圆盘。首先，如图 9a所示，向量子独立集采样器添加噪声并未产生影响。事实上，最终总体差距与无噪声模型相近，且优于经典列生成。最后，如图 9b所示，与无噪声模型相比，含噪模型找到松弛受限主问题最优解所需的迭代次数平均仅增加了 6%，即使对于大型图也是如此。

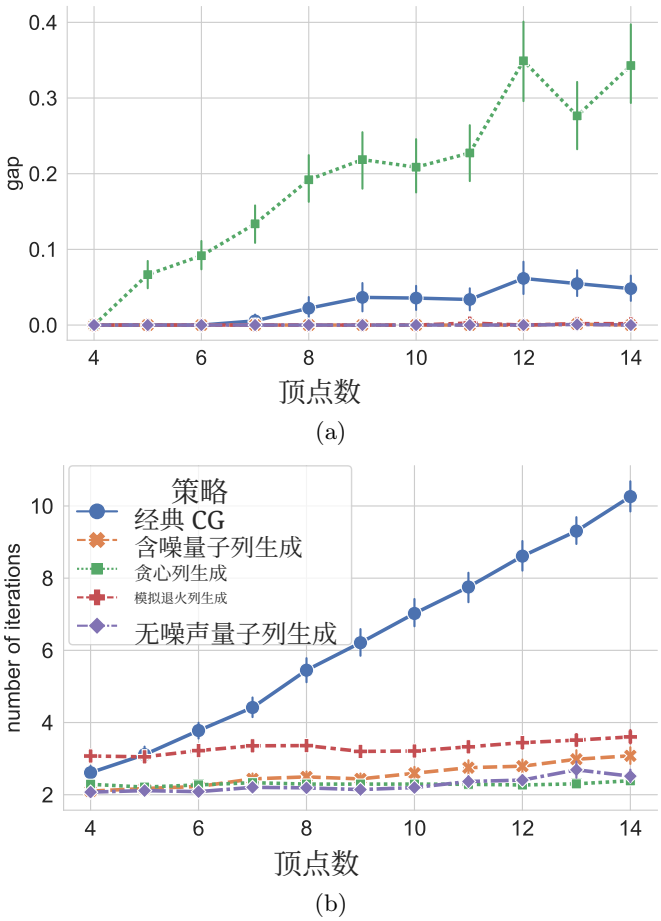


图 9：采用不同方法——经典列生成（列生成）、贪心列生成、模拟退火列生成、无噪声量子列生成和含噪量子列生成——处理不同图阶（顶点数从 4 到 14）时，达到停止准则前的迭代次数，以及最终解与最优解之间的差距。图 9a 展示了最终解与最优解之间的差距，图 9b 展示了达到停止准则前的迭代次数。

结论

本研究表明，可以将量子元素融入经典的先进算法，以提升其性能。与经典列生成相比，我们基于中性原子的量子定价最多可将求解最小顶点着色问题所需的迭代次数减少 83%。此外，与经典方法和贪心方法相比，我们提出的混合方法可将平均间隙从 19% 降至 0%。而且，与确定性方法不同，我们基于量子定价的列生成对所有图类都具有鲁棒性，包括所有测试阶数和规模的非单位圆盘图。这表明，经过一些简单的基于转换的预处理后，所提出的混合算法能够高效求解其他组合优化问题（例如，最小边着色问题和最小聚类问题）。所提出的框架也优于其他随机算法，尤其是在稀疏图上。事实上，与基于模拟退火和基于贪心的定价方法相比，我们基于量子定价的列生成在图上最多可将定价迭代次数减少 50%，并将最终解的差距最多减少 80%。最后，我们观察到所提出的混合框架对噪声具有鲁棒性。与无噪声模型相比，最终解的质量未受影响，这充分表明模拟运行模式具有很强的抗噪能力。我们提出的方法可直接在中性原子量子计算硬件上实现。

我们需要指出，我们提出的算法仍然是用于求解组合问题的启发式方法。尽管它能够为多种实例类别提供高质量的解，但要保证针对所研究问题的任意实例都达到最优解，仍有必要将该方法嵌入分支定价框架中。由于为受限主问题提供初始变量子集是任何基于列生成的算法中的重要步骤，因此，可以使用所提出的量子采样器作为热启动工具，生成这样的元素子集（例如独立集）。在求解最小顶点着色问题时，将所有顶点权重设为 1 后，当量子处理器资源有限时，这种方法可能会很有用。此外，该策略还有可能加速经典和量子列生成方法。更进一步，可以针对每个输入实例采用基于最优控制的策略来设计脉冲，从而有可能解决具有挑战性的定价子问题。同样，也可以开发不同的寄存器嵌入方法，以考虑输入数据中的不同信息。

致谢

感谢 Julien Bernos 和 Vincent Elfving 富有洞察力的讨论。在完成本文期间，我们了解到一项相关工作 [52]。

附录A：最小顶点着色问题的贪心算法

此处介绍一种基于 [35]提出的最小顶点着色框架的贪心启发式。其主要思路是仅使用给定图中的一部分顶点，以交互方式求解最大独立集问题。算法2概述了所提出的方法。

算法2 贪心算法

输入： 图 $\mathcal{G} = (\mathcal{V},\ \mathcal{E})$ 和一组可用颜色 C。
输出： 颜色-顶点分配。

```
G' ← G
c ← 1
当 G' 有顶点时
    在 G' 中找到一个（最大）独立集 IS
    将颜色 c ∈ C 分配给 G 中来自 IS 的所有顶点
    从 G' 中移除每个顶点在 IS 中关联的所有边。
    从 G' 中移除顶点集合 IS。
    c = c + 1
当条件不成立时结束循环
```

作为输入，算法 2 接收一个图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ 和一组 C 可用颜色。请注意，为始终获得可行的颜色分配，必须满足 $|\mathcal{V}| \leq |C|$。首先，创建 $\mathcal{G}$ 的一个副本，并构造辅助图 $\mathcal{G}' \equiv G$。同时创建变量 k，用于记录第一个可用颜色的索引。随后，重复执行步骤 3–7，直至 $\mathcal{G}'$ 中不再有顶点。每次迭代中，都会在 $\mathcal{G}'$ 中生成一个可行独立集（可能是极大独立集）。例如，可以使用经典求解器求解公式（4）；也可以使用第 VC 节所述的量子采样器，并将算法设置为仅输出采样得到的最大独立集。然后，将图 G 中索引与所找到的独立集 IS 中顶点相同的所有顶点，赋予 C 中第一个可用的颜色，其索引由变量 c 给出。接着，从 IS 中移除 $\mathcal{G}'$ 中的所有顶点（以及与之相关的关联边），然后更新第一个可用颜色的引用（见步骤 8）。

值得一提的是，尽管与其他方法相比，在量子处理单元上运行该算法所需的量子比特数量有所减少[4]，但独立集的生成顺序可能会显著影响其性能。此外，在最坏情况下，$|\mathcal{V}|$ 在找到最终解之前，量子设备上可能需要进行多次迭代（以完全图为例）。有关其性能的更多详情，请参阅第VI节。

[4]尽管某些顶点着色问题的 QUBO公式包含 $|V|^2$ 个变量，因此至少需要 $|V|^2$ 个量子比特来编码与之对应的 [27, 53]（例如，不使用辅助量子比特时），但 [35] 中提出的贪心算法最多只使用 $|V|$ 个变量/量子比特。

附录 B：力导向算法

正如 [3], 所指出的，基于力导向原理的算法通常用于将图嵌入平面，使得相连的顶点彼此靠近，而不相连的顶点彼此远离；顶点之间的距离应尽可能小，而它们与平面中心的距离则应尽可能大。在此背景下，为了体现固有的对称性，Fruchterman和Reingold [54] 还提出了一种高效算法，使顶点均匀分布在平面上，从而让边的长度保持一致。为此，图中的每条边都被视为一根弹簧，将其两端的顶点拉近；同时施加相互竞争的斥力，将所有顶点彼此推开，即使它们在原图中并不由边相连。经过足够多次迭代后，整个系统将达到平衡，使所有吸引力与斥力之间的差异最小化。

两个顶点之间的排斥力和吸引力分别 f_r 和 f_a 由公式（B1）和（B2）给出。虽然$k = \sqrt{A/|\mathcal{V}|}$ 设定为与欧几里得平面的面积 A 相关，d_{uv} 表示顶点u与 $v \in \mathcal{V}$之间的距离。最后，系统的总能量 f_t 通过对所有顶点对之间的力求和得到，如式（B3）所示。因此，f_t 随着系统趋近平衡而趋于零。此处所述的寄存器嵌入下文称为弹簧布局。

$$f_r(u,v) = -k^2/r_{uv} \tag{B1}$$

$$f_a(u,v) = d_{uv}^2/k \tag{B2}$$

$$f_t = \sum_{u,v\in\mathcal{E}} f_a(u,v) + \sum_{u\in\mathcal{V}} \sum_{v\in\mathcal{V}:u\neq v} f_r(u,v) \tag{B3}$$

图5a展示了将算法直接应用于一个含有 5 个顶点和 7 条边的图时，可能得到的一种嵌入。关于该算法的详细说明，请参阅 [3, 54]。如果存在这样的实现，那么只要迭代次数足够多（即，持续迭代直至系统达到平衡），所讨论的图自然会嵌入为 UD 图。还值得一提的是，正如 [3],指出的，只有在满足设备的技术约束时，所提出的嵌入策略才能在中性原子量子处理器上实现，例如寄存器中心的最大距离以及原子之间的最小距离。如果违反了任何技术约束，可以将每个位置向量乘以一个因子重新缩放 $\alpha > 0$。

附录 C：经典贪心定价算法

算法 3概述了所提出的随机加权独立集生成器。算法的输入是一个图 $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$，其中 W是顶点权重向量。首先，创建一个辅助图 $\mathcal{G}'$，用于接收输入图的副本 $\mathcal{G}$（步骤 3）。然后，从图$\mathcal{G}'$ 中随机选择一个顶点，并将其加入独立集 S（见步骤 6-7）。接着，在步骤 8 中，从 $\mathcal{G}'$ 中移除所选顶点及其邻居。重复步骤 6-8，直至没有顶点剩余。随后，根据顶点权重向量 W计算最终解的总成本（见步骤 11）。由于所提出算法本身具有随机性，因此会得到不同的加权独立集

算法3 随机独立集生成器 输入：一个图 $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ W、权重阈值 w_{min}，以及最大尝试次数 t。输出：加权独立集的集合。1: $S \leftarrow \emptyset$ 2: 当尚未达到最大尝试次数 t 时，执行循环 3: $\mathcal{G}' \leftarrow G$ 4: $IS \leftarrow \emptyset$ 5: 当 $\mathcal{G}'$ 中仍有顶点时，执行循环 6: 在 $\mathcal{G}'$ 中随机选择一个顶点 7: 将所选顶点加入 IS 8: 从 $\mathcal{G}'$ 中移除所选顶点及其邻居。 9: 循环结束 10: 如果此前未生成过该独立集（IS），且其总权重大于 w_{min}，则 11: $S \leftarrow S \cup IS$ 12: 如果结束 13: 循环结束 14: 返回 S

可能由相同的输入生成；运行步骤 3–11 的过程可以重复多次（参见步骤 2 中的条件）。

[1] C. Barnhart、E. L. Johnson、G. L. Nemhauser、M. W. Savelsbergh 和 P. H. Vance，《分支定价：用于求解大型整数规划的列生成》，*Operations research*，第 46 卷，第 3 期，第 316–329 页，1998年。[2] A. Bachem 和 W. Kern，《线性规划对偶》，载于 *Linear Programming Duality*。Springer，1992年，第 89–111 页。[3] W. d. S. Coelho、M. D'Arcangelo 和 L.-P. Henry，《在中性原子量子处理器上求解组合图问题的高效协议》，2022年。 [在线]。可访问：https://arxiv.org/abs/2207 13030 。M. Kjaergaard、M. E. [4] Schwartz、J. Braumuller、P. Krantz、J. I.-J. Wang、S. Gustavsson 和 W. D. Oliver，《超导量子比特：当前发展状况》，*Annual Review of Condensed Matter Physics* 11, 1, 369 395, 2020 ，第 卷，第 页，–。

[5] M. H. Devoret、A. Wallraff 和 J. M. Martinis，《超导量子比特：简要综述》，2004年。 [在线]。可访问：https://arxiv.org/abs/cond-mat/0411174 [6] I. Pogorelov、T. Feldker、C. D. Marciniak、L. Postler、G. Ja- cob、O. Krieglsteiner、V. Podlesnic、M. Meth、V. Negnevit- sky、M. Stadler、B. H¨ofer、C. W¨ achter、K. Lakhmanskiy、R. Blatt、P. Schindler 和 T. Monz，《紧凑型离子阱量子计算演示器》，*PRX Quantum*，第 2 卷，第 020343 页，2021年6月。[7] J. Benhelm、G. Kirchmair、C. F. Roos 和 R. Blatt，《迈向基于囚禁离子的容错量子计算》，*Nature Physics*，第 4 卷，第 6 期，第 463–466 页，2008年4月。[8] C. Ant´on、J. C. Loredo、G. Coppola、H. Ollivier、N. Vig- gianiello、A. Harouri、N. Somaschi、A. Crespi、I. Sagnes、A. Lemaˆıtre、L. Lanco、R. Osellame、F. Sciarrino 和 P. Senellart，《对接可扩展光子平台：可重构玻璃芯片上的固态多光子干涉》，*Optica*，第 6 卷，第 12 期，第 1471–1477 页，2019年12月。[9] L. Henriet、L. Beguin、A. Signoles、T. Lahaye、A. Browaeys、G.-O. Reymond 和 C. Jurczak，《中性原子量子计算》，*Quantum*，第 4 卷，第 327 页，2020年。[10] J. Wurtz、P. Lopes、N. Gemelke、A. Keesling 和 S. Wang，《中性原子量子计算在求解独立集问题中的行业应用》，2022年。 [在线]。可访问：https://arxiv.org/abs/2205 08500 。J. A. Bondy、U. S. R. Murty、[11] *et al.*，*Graph theory with applications* 1976, 290 。Macmillan London，第 卷。[12] M. Roth、A. Ben-David、D. Deutscher、G. Flysher、I. Horn、A. Leichtberg、N. Leiser、Y. Matias 和 R. Merom，《利用隐式社交图推荐好友》，载于 *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining* 2010 233 242 ，第 页，–。

[13] A. Kershenbaum，*Telecommunications network design algorithms* 1993 . McGraw-Hill, Inc., . [14] F. Barahona, M. Gr¨otschel, M. J¨unger, and G. Reinelt, "将组合优化应用于统计物理和电路布局设计，" *Operations Research*, vol. 36, no. 3, pp. 493–513, 1988年. [15] K. Bharti, A. Cervera-Lierta, T. H. Kyaw, T. Haug, S. Alperin-Lea, A. Anand, M. Degroote, H. Heimonen, J. S. Kottmann, T. Menke, W.-K. Mok, S. Sim, L.-C. Kwek, and A. Aspuru-Guzik, "含噪中等规模量子算法，" *Reviews of Modern Physics*, vol. 94, no. 1, feb 2022年. [16] H. Nishi, T. Kosugi, and Y. ichiro Matsushita, "通过引入非局域近似，在含噪中等规模量子设备上实现量子虚时间演化方法，" *npj Quantum Information* 7, 1, 2021 , vol. no. jun .

[17] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Sama- jdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi, S. Sachdev, N. Gemelke, L. Zhou, S. Choi, H. Pichler, S.- T. Wang, M. Greiner, V. Vuleti´c , and M. D. Lukin, "利用里德堡原子阵列优化最大独立集，" *Science*, vol. 376, no. 6598, pp. 1209–1215, jun 2022年.[18]M. Kim, K. Kim, J. Hwang, E.-G. Moon, and J. Ahn, "用于非平面高阶图最大独立集问题的里德堡量子线，" 2021年. [ ] 2109 03517 在线 . 可访问：[19] https://arxiv.org/abs/ . B. Albrecht, C. Dalyac, L. Leclerc, L. Ortiz-Guti´errez, S. Thabet, M. D'Arcangelo, V. E. Elfving, L. Lassabli`ere, H. Silv´erio, B. Ximenez, L.-P. Henry, A. Signoles, and L. Henriet, "用于中性原子量子处理器上图机器学习的量子特征映射，" 2022年. [20] L. Novo, S. Chakraborty, M. Mohseni, and Y. Omar, "环境辅助模拟量子搜索，" *Physical Review A* 98, 2, 2018 , vol. no. aug .

[21] C. Dalyac, L. Henriet, E. Jeandel, W. Lechner, S. Perdrix, M. Porcheron, and M. Veshchezerova, "评估针对困难工业优化问题的量子方法：智能电动汽车充电领域的案例研究，" *EPJ Quantum Technology* 8, 1, 12, 2021 , vol. no. p. .

[22] A. Rajak, S. Suzuki, A. Dutta, and B. K. Chakrabarti, "量子退火：综述，" 2022 年 . [在线]. 可访问：https://arxiv.org/abs/2207 01827 . A. Cornejo and F. [23] Kuhn, "使用蜂鸣信号部署无线网络，" in *International Symposium on Distributed Computing* 2010, 148 162 . Springer, pp. – .

[24] D. Marx, “图着色问题及其在调度中的应用,” *Periodica Polytechnica Electrical Engineering (Archives)* 48, 1 2, 11 16, 2004, vol. no. - pp. – . [25] R. M. Karp, “组合问题之间的可归约性,” in *Complexity of computer computations*. Springer, 1972年, pp. 85–103. [26] K. Kudo, “图着色的受约束量子退火,” *Physical Review A*, vol. 98, no. 2, p. 022301, 2018年. [27] S. M. Ardelean and M. Udrescu, “使用约简量子遗传算法进行图着色,” *PeerJ Computer Science*, vol. 8, p. e836, 2022年. [28] O. Titiloye and A. Crispin, “图着色问题的量子退火,” *Discrete Optimization*, vol. 8, no. 2, pp. 376–384, 2011年. [29] C. Silva, A. Aguiar, P. Lima, and I. Dutra, “将图着色映射到量子退火,” *Quantum Machine Intelligence* 2, 2, 1 19, 2020, vol. no. pp. – . [30] Z. Tabi, K. H. El-Safty, Z. Kallus, P. H´aga, T. Kozsik, A. Glos, and Z. Zimborás, “采用空间高效嵌入的图着色问题量子优化,” in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* 2020, 56 62. IEEE, pp. – . [31] J. Kwok and K. Pudenz, “ 利用量子退火进行图着色 ,” 2020 年 . [在线]. 可访问：https://arxiv.org/ abs/ 2012 04470 . A. [32] Fabrikant and T. Hogg, “利用量子启发式算法进行图着色,” in *AAAI/IAAI*, 2002年, pp. 22–27. [33] K. Shimizu and R. Mori, “图着色问题的指数时间量子算法,” *Algorithmica* 1 19, , pp. – . [34] 2022 A. Ambainis, K. Balodis, J. Iraids, M. Kokainis, K. Pr¯usis, and J. Vihrovs, “指数时间动态规划算法的量子加速,” in *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms* 2019, 1783 1793. SIAM, pp. – . [35] G. Vitali, P. Viviani, C. Vercellino, A. Scarabosio, A. Scionti, O. Terzo, E. Giusto, and B. Montrucchio, “利用里德堡原子实现最优图着色,” in *The International Conference for High Performance Computing, Networking, Storage and Analysis Research posters* 2021 [ ] , , , . 在线 . 可访问：https://sc21.supercomputing.org/presentation/ ?id = 113 = 278 rpost &sess sess J. Ossorio-Castillo [36] and F. Pena-Brage, “结合列生成与量子退火器优化炼油厂调度流程,” *Optimization and Engineering*, vol. 23, no. 3, pp. 1471–1488, 2022年. [37] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, “利用里德堡原子阵列优化最大独立集,” 2018年. [在线]. 可访问：https://arxiv.org/abs/ 1808 10816 . S. [38] Martiel, T. Ayral, and C. Allouche, “以应用为中心、与硬件无关且可扩展的量子协处理器基准测试方法,” *IEEE Transactions on Quantum Engineering* 2, 1 11, 2021, vol. pp. – . [39] M. Balinski and A. W. Tucker, “ 线性规划的对偶理论：一种具有应用的构造性方法,” *Siam Review*, 第 11 卷, 第 3 期, 第 347–377 页, 1969年。[40] H. Breu 和 D. G. Kirkpatrick, 《单位圆盘图识别是 NP 困难问题》, *Computational Geometry*, 第 9 卷, 第 1-2 期, 第 3–24 页, 1998年。[41] 《使用 QAOA 和 QAA 求解单位圆盘最大独立集问题》, https://pulser.readthedocs.io/en/latest/tutorials/qaoa mis.html, 访问日期: 2022-11-05。[42] T. Albash 和 D. A. Lidar, 《绝热量子计算》, *Reviews of Modern Physics* 90, 1, 015002, , 卷、期、页码缺失。2018 [43] D. Bluvstein、A. Omran、H. Levine、A. Keesling、G. Semeghini、S. Ebadi、T. T. Wang、A. A. Michailidis、N. Maskara、W. W. Ho、S. Choi、M. Serbyn、M. Greiner、V. Vuleti´c 和 M. D. Lukin, 《控制受驱动里德堡原子阵列中的量子多体动力学》, *Science*, 第 371 卷, 第 6536 期, 第 1355–1359 页, 2021年。 [在线]。获取地址: https://www.science.org/doi/abs/ 10 1126 2530 。/ [44] science.abg V. Batagelj 和 U. Brandes, 《高效生成大型随机网络》, *Physical Review E*, 第 71 卷, 第 3 期, 文章编号 036113, 2005年。[45] M. Penrose, *Random geometric graphs*。牛津大学出版社, 2003年, 第 5 卷。[46] E. Malaguti 和 P. Toth, 《顶点着色问题综述》, *International transactions in operational research*, 第 17 卷, 第 1 期, 第 1–34 页, 2010年。[47] 《GLPK——GNU 线性规划工具包》, https://www.gnu.org/software/glpk/, 访问日期: 2022-11-05。[48] 《 D-Wave 模拟退火 QUBO 采样器》, h t t p s:// d ocs. ocean.dwavesys.com/projects/neal/en/latest/reference/sampler.html, 访问日期: 2022-12-21。[49] A. Hagberg、P. Swart 和 D. S Chult, 《使用 NetworkX 探索网络结构、动力学和功能》, 洛斯阿拉莫斯国家实验室（LANL）, 美国新墨西哥州洛斯阿拉莫斯, 技术报告, 2008年。[50] H. Silv´erio、S. Grijalva、C. Dalyac、L. Leclerc、P. J. Karalekas、N. Shammah、M. Beji、L.-P. Henry 和 L. Henriet, 《Pulser：用于设计可编程中性原子阵列脉冲序列的开源软件包》, *Quantum*, 第 6 卷, 文章编号 629, 2022年。[51] L. Leclerc、L. Ortiz-Guitierrez、S. Grijalva、B. Albrecht、J. R. K. Cline、V. E. Elfving、A. Signoles、L. Henriet、G. Del Bimbo、U. A. Sheikh、M. Shah、L. Andrea、F. Ishtiaq、A. Duarte、S. Mugel、I. Caceres、M. Kurek、R. Orus、A. Seddik、O. Hammammi、H. Issel nane 和 D. M’tamon, 《中性原子量子处理器上的金融风险管理》, 2022年。[在线]。获取地址：https://arxiv.org/abs/ 2212 03223 。M. Veshchezerova, 《用于能源管理优化问题的量子算法》, 博士论文, Ecole Doctorale IAEM, 2022年。[52] [53] F. Glover、G. Kochenberger、R. Hennig 和 Y. Du, 《量子桥接分析 I: QUBO 模型的构建与使用教程》, *Annals of Operations Research*, 第 314 卷, 第 1 期, 第 141–183 页, 2022年。[54] T. M. Fruchterman 和 E. M. Reingold, 《力导向布局图绘制》, *Software: Practice and experience* 21, 11, 1129 1164, 1991, 卷、期、页码缺失。