Quantum compilation toolkit for Rydberg atom arrays with implications for problem hardness and quantum speedups

Martin J. A. Schuetz [1,2,*] Ruben S. Andrist [1,*] Grant Salton [1,2] Romina Yalovetzky,[3] Rudy Raymond [3] Yue Sun [3] Atithi Acharya,[3] Shouvanik Chakrabarti,[3] Marco Pistoia,[3,†] and Helmut G. Katzgraber [1,†]

[1]*Amazon Advanced Solutions Lab, Seattle, Washington 98170, USA*

[2]*AWS Center for Quantum Computing, Pasadena, California 91125, USA*

[3]*Global Technology Applied Research, JPMorgan Chase, New York, New York 10017, USA*

(Received 16 January 2025; accepted 30 June 2025; published 1 August 2025)

We propose and implement a comprehensive quantum compilation toolkit for solving the maximum independent set (MIS) problem on quantum hardware based on Rydberg atom arrays. Our end-to-end pipeline involves three core components to efficiently map generic MIS instances onto Rydberg arrays with unit-disk connectivity, with modules for graph reduction, hardware compatibility checks, and graph embedding. The first module (reducer) provides hardware-agnostic and deterministic reduction logic that iteratively reduces the problem size via lazy clique removals. We find that real-world networks can typically be reduced by orders of magnitude on subsecond timescales, thus significantly cutting down the eventual load for quantum devices. Moreover, we show that reduction techniques may be an important tool in the ongoing search for potential quantum speedups, given their ability to identify hard problem instances. In particular, for Rydberg-native MIS instances, we observe signatures of an easy-hard-easy transition and quantify a critical degree indicating the onset of a hard problem regime. The second module (compatibility checker) implements a hardware compatibility checker that quickly determines whether a given input graph may be compatible with the restrictions imposed by Rydberg quantum hardware. The third module (embedder) describes hardware-efficient graph embedding routines to generate (approximate) encodings with controllable overhead and optimized ancilla placements. We exemplify our pipeline with experiments run on the QuEra Aquila device available on Amazon Braket. In aggregate, our work provides a set of tools that extends the class of problems that can be tackled with near-term Rydberg atom arrays.

DOI: 10.1103/7dkh-crjj

I. INTRODUCTION

The field of combinatorial optimization (CO) involves the search for the extremum of an objective function (such as a cost or profit value) within a finite (but usually large) set of candidate solutions. Despite often being simple to conceptualize, many CO problems are hard to solve and even NP-hard, representing some of the most exquisite yet challenging computational problems known [1,2]. Given its close ties to the maximum clique, minimum vertex cover, and set packing problems [3], the maximum independent set (MIS) problem represents an important paradigmatic CO problem, with practical applications in virtually any industry sector. These include network design [4], vehicle routing [5], and quantitative finance [6–8], to name a few.

A. Quantum optimization

Since the late 1990s, quantum optimization (QO) has emerged as a novel paradigm for solving such discrete optimization problems [9] in pursuit of quantum speedups over the best-known classical algorithms for practically relevant problems. Prominent QO algorithms include quantum annealing algorithms (QAA) [10–14] as well as hybrid (quantum-classical) algorithms such as the quantum approximate optimization algorithm (QAOA) [15,16], among others. In these approaches, typically the optimal solution to the classical optimization problem at hand is encoded in the ground state of quantum many-body systems that can be implemented and controlled with (special-purpose) quantum hardware [17,18].

B. Optimization with Rydberg arrays

Since the mid-2010s, analog neutral-atom quantum machines in the form of Rydberg atom arrays have emerged as a novel class of programmable and scalable special-purpose quantum devices geared towards optimization workloads [16,19–27]. In particular, recent experiments with up to 289 Rydberg atoms have reported a potential superlinear quantum speedup over classical simulated annealing for the MIS problem, with pioneering implementations of both QAA and QAOA within the same experimental setup [22,28]. Because

*These authors contributed equally to this work.

†These authors acted as Co-PIs.

Published by the American Physical Society under the terms of the Creative Commons Attribution 4.0 International license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

of the isotropic nature of the Rydberg blockade mechanism [29–31], these first-generation experiments were inherently limited to MIS problem instances on a restricted class of geometric graphs known as unit-disk (UD) graphs [19,20,32]. Ultimately, however, the usefulness of these devices will depend on the scope of problems they can tackle beyond the hardware-native MIS-UD problem set.

C. Embedding schemes

Embedding techniques are designed to expand the scope of problems supported by a given quantum device by mapping *logical* input problems to *physical* representations compatible with connectivity-constrained quantum hardware. Given the local topology of the underlying physical hardware and limitations in natively accommodating long-range couplings, embedding is usually achieved via the introduction of ancilla qubits to effectively distribute logical information across distant locations, thereby boosting the system's connectivity at the expense of an (often quadratic) overhead in the number of qubits [33]. Specifically, for superconducting (SC) annealing devices, so-called *minor embedding techniques* are well established [33–36], providing a (heuristic) mapping from a logical graph to a subgraph of a target hardware graph (with sparse connectivity), with chains of several physical qubits making up single (compound) logical variables. For Rydberg atom arrays, embedding schemes have recently been put forward that are akin to the embedding schemes for SC annealers. These schemes were developed within a larger effort to expand the class of problems that can be tackled with near-term, analog Rydberg devices, beyond the hardware-native MIS problem on UD graphs (dubbed MIS-UD hereafter). Existing proposals, however, are either restricted to a specific class of problem instances [37,38] or come with (potentially demanding) experimental requirements, such as the need for three-dimensional arrays [39], four-body interactions [40], or local detuning [41], with some schemes incurring an (*a priori* undefined) overhead in the number of ancilla qubits [38]. Recently, first experiments have successfully verified the ability to embed and solve maximum weighted independent set problems on non-UD graphs, albeit for relatively simple instances at small scales with five logical and nine physical nodes, respectively [42].

D. Research objectives

Our main goal is to contribute to this line of recent research by providing a set of tools that help expand the scope and size of problems compatible with near-term Rydberg atom arrays. We note that the ability to coherently shuttle around individual atoms—arguably one of the most distinctive features of modern Rydberg arrays [43,44]—with its potential to generate effectively nonlocal interactions in real time, brings exciting possibilities for future Rydberg embedding technology. For the sake of relatively low experimental complexity, however, here we focus on developing tools that are either hardware agnostic or can already be implemented with static Rydberg array positioning.

II. OVERVIEW OF MAIN RESULTS

Here we complement the existing Rydberg embedding toolbox by providing a comprehensive, modular compilation pipeline that maps generic (potentially large-scale) instances of the MIS problem to (potentially much smaller) MIS problems on UD graphs, which are natively encodable with Rydberg atom arrays (see Fig. 1 for a schematic illustration). The pipeline consists of three core modules, with the goal of providing approximate, low-overhead embeddings for Rydberg atom arrays that keep the number of required qubits small and controllable. We now provide a brief description of these modules.

A. Reducer

Our first module, the *reducer*, takes a generic graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, with node set $\mathcal{V}$ and edge set $\mathcal{E}$, as problem input and provides a smaller (*core*, or *kernel*) graph (with one or more components) as output. These output graph components can then be treated independently, thereby distilling the most important logical variables from the original problem definition, while minimizing the eventual workload for the quantum device. Specifically, we implement a procedure known as isolated clique removal [45–49]. Just like peeling an onion, this scheme proceeds by iteratively removing subgraphs of the input graph (such as isolated nodes, dangling bonds, and larger isolated cliques) while maintaining optimality, until no further reduction can be achieved. Optimal solutions for the reduced (core) instance can then be easily expanded to an optimal solution for the original (potentially much larger) instance by reversing previously applied reductions.

For illustration, consider a simple dangling bond (i.e., a clique of size two) involving an exposed (degree-1) node u connected to the remainder of the larger graph via an edge (u, v) to node v only. It is easy to see that node u has to be included in at least a subset of all (degenerate) MIS solutions; leaving both node u as well as node v out of the set is clearly suboptimal, and at least one of them should be marked. By deterministically selecting the exposed node u and removing the entire clique from the graph (in accordance with the independence constraint), our reduction logic is minimally invasive to the remainder of the graph, ensuring that at least one optimal MIS solution can be found, while systematically shrinking the problem size. The underlying clique checks are performed *lazily*, because the subgraph removals may require a do-over. The scheme then starts over, applying the same simplification logic to the reduced graph where nodes u and v have now been removed, with the exposed node u tracked as member of the set.

We provide systematic numerical experiments, testing the performance of this reduction technique for both synthetic and real-world graphs, with up to tens of thousands of nodes. In particular, we find that structured, real-world graphs can often be reduced by orders of magnitude, in some cases even down to an empty null graph (in which case the reducer amounts to an MIS solver), typically in linear time (as demonstrated below for both synthetic and real-world graphs). For example, we find that the well-known Cora [50] and Pubmed [51] citation graphs can be reduced from $\sim$2700 and $\sim 2\times 10^4$ nodes to small core graphs with just 9 and 11 nodes, respectively, on subsecond timescales on a laptop. We also discuss and analyze reduction in the context of the ongoing search for potential quantum speedups, based on the reducer's ability

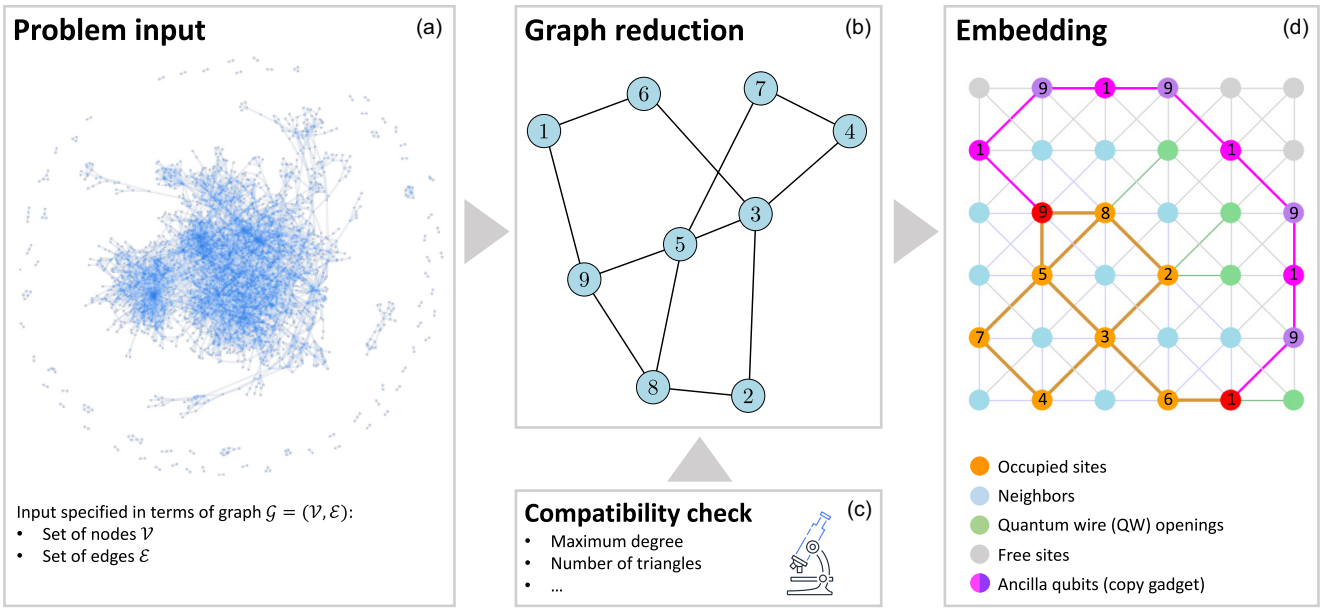


FIG. 1. Schematic illustration of the proposed compilation pipeline, shown here for the Cora citation graph with ∼2700 nodes and ∼5300 edges. (a) Problem input: The computational problem is defined in terms of an input graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The ultimate goal is to solve the MIS problem on this graph, using Rydberg atom arrays. The scale and connectivity of the graph $\mathcal{G}$ is arbitrary, and, in general, incompatible with near-term Rydberg arrays. (b) Graph reduction: Using an iterative and deterministic reduction algorithm, subgraphs of the original graph can be removed, while maintaining optimality, leaving a graph kernel with one or more smaller components that can be solved individually. After reduction, the largest component for the Cora graph consists of the graph shown with 9 nodes and 12 edges. (c) Compatibility check: Using a fixed set of criteria (based on graph properties such as the maximum degree, number of triangles, etc.), the graph compatibility checker quickly assesses whether the reduced graph is (in principle) compatible with the restrictions imposed by the Rydberg quantum hardware. (d) Embedding: Heuristic, random-key algorithms are used to find a hardware-efficient mapping from the logical graph in (b) to the Rydberg quantum hardware with unit-disk connectivity (with Rydberg atoms corresponding to the nodes of the graph). Here we assume a simple, two-dimensional square lattice with Union-Jack- (UJ) like connectivity to nearest and next-nearest neighbors only, but generalizations are straightforward. Our (bottom-up) embedding logic first optimizes the similarity between the logical input graph and the physical (embedded) graph, given the limited hardware connectivity. For the example graph in (b) an optimized atom placement (shown as occupied sites in orange and red) can be found with an edit distance of one (only missing the edge between nodes 1 and 9), without incurring any overhead. This overhead-free embedding can be further refined using additional ancilla atoms arranged into a quantum wire (shown in pink and purple), the placement of which is optimized with a breadth-first search algorithm solving a constrained shortest-path problem on the underlying square lattice between the start and end nodes marked in red. Overall, the MIS problem on the original Cora graph has been mapped to the MIS problem on a hardware-native graph with nine nodes, 11 UJ edges, and one quantum wire with eight ancilla nodes.

to efficiently single-out hard problem instances. Specifically, for Rydberg-native MIS instances, we observe signatures of an easy-hard-easy transition, as evidenced in experiments analyzing the achievable reduction as a function of the lattice filling, with high (low) reduction values identifying easy (hard) parameter regions, and quantify the corresponding transition points.

B. Checker

Our second module, the *compatibility checker*, takes a graph as an input and checks if the graph is, in principle, compatible with an overhead-free embedding on a Rydberg array. The checker can determine if a graph *cannot* be natively embedded, and it outputs a Boolean flag to this effect. The flag constitutes a necessary but not sufficient condition for native encoding; a graph may pass the compatibility check but may not be natively encoded in hardware, but a graph that fails the check *cannot* be encoded without overhead. Thus, the checker provides information that can be utilized in any downstream embedding scheme. For concreteness, we will focus on UD graphs compatible with Rydberg atoms arranged on a two-dimensional square lattice with nearest and next-nearest (diagonal) couplings, generating a Union-Jack- (UJ) like connectivity pattern [cf. Fig. 1(d)], as realized experimentally in, e.g., Ref. [22] and currently available on Amazon Braket [52]. Generalizations to alternative geometries, however, should be straightforward. For a simple illustration, consider the maximum degree of a given graph. As evident from Fig. 1(d), the UJ Rydberg hardware can only accommodate graphs with maximum degree $d_{\max} \leqslant 8$. Therefore, for any input graph with $d_{\max} > 8$, the compatibility checker will raise a flag, classifying the input graph as a non-native graph that will require ancilla qubits for any perfect embedding. Beyond the maximum degree, additional graph properties (such as the number of triangles) are evaluated in the compatibility assessment, providing a simple yet fine-grained (binary) classification for any given input graph.

C. Embedder

Our third module, the *embedder*, takes a given (reduced) graph as input and provides an efficient embedding, with optimized overhead and ancilla wiring. Specifically, we design two complementary embedding strategies, including a (bottom-up) approach that *learns* approximate embeddings with minimal resource requirements, as well as a (top-down) scheme that constructively generates exact embeddings with MIS optimality guarantees, while minimizing the required qubit overhead. See Fig. 1(d) for an illustration of our bottom-up embedder. Here we design efficient assignment heuristics that *learn* hardware-efficient embeddings by optimizing a given similarity metric (such as the edit-distance) between the (logical) input graph and the (physical) embedded graph, while adhering to the connectivity constraints imposed by the hardware. This hardware-efficient embedding can then be further improved by surgically placing vertex wires [39,41] to generate long-distance edges. In our scheme, the placement of the latter is optimized with a breadth-first search algorithm solving a constrained shortest-path problem to efficiently connect distant nodes. Our second (top-down) embedding scheme builds on the generic (exact) embedding scheme outlined in Ref. [41] and systematically reduces the required overhead using a custom simulated annealing solver that identifies optimized qubit arrangements.

After reduction and embedding, quantum algorithms (e.g., those shown in Ref. [22]) can be used to approximately solve the MIS on the embedded (physical) graph, with site-resolved projective measurements for readout of the final quantum many-body state involving both system and ancilla atoms. Simple postprocessing can then distill the solution for the original (logical) input graph.

D. Modular integration

The modular nature of our scheme allows for plug-and-play solutions, combining a subset of the building blocks detailed here with some of the existing schemes discussed above. For example, it should be straightforward to integrate the reduction logic proposed here with the embedding scheme described in Ref. [41]. Finally, for an experimental realization, our scheme is tailored towards neutral atom experiments supporting two-dimensional atom trapping and the Rydberg blockade mechanism, as successfully demonstrated with hundreds of atoms in, e.g., Ref. [22] and available in the cloud on Amazon Braket via QuEra's Aquila device [52]. These architectures have the potential to scale to systems with thousands of atoms in the near future [53,54].

E. Structure

The remainder of this paper is organized as follows. In Sec. III, we first formalize the problem we consider and provide some more background. Next, in Sec. IV, we describe in detail our larger pipeline with an in-depth discussion for all core modules. In Sec. VI, we then assess the performance of our pipeline with a series of numerical experiments for both synthetic and real-world graphs. Finally, in Sec. VII, we draw conclusions and give an outlook on future directions of research.

III. PRELIMINARIES AND PROBLEM SPECIFICATION

Our work is motivated by recent advances in experiments involving neutral-atom quantum machines [55,56], in particular in the field of quantum optimization [9,22], and our main goal is to extend both the size and scope of problems compatible with these devices.

A. Rydberg atom arrays

In Rydberg atom arrays, atomic qubits are (typically) defined by an internal ground state $|0\rangle_i$ and a highly excited, long-lived Rydberg state $|1\rangle_i$, where the label $i = 1, \ldots, n$ runs over the n qubits. Within these two-dimensional subspaces, we define the usual Pauli matrices $\hat{\sigma}_i^{\alpha}$, where $\alpha = x, y, z$, and the corresponding lowering operators $\hat{\sigma}_i^{-} = |0\rangle_i\langle 1|$. The quantum dynamics of Rydberg atom arrays is then described by the Hamiltonian $\hat{H} = \hat{H}_{\text{drive}} + \hat{H}_{\text{cost}}$, with ($\hbar = 1$)

$$\hat{H}_{\text{drive}} = \frac{\Omega(t)}{2} \sum_i (e^{i\phi(t)}\hat{\sigma}_i^{-} + \text{H.c.}),$$
$$\hat{H}_{\text{cost}} = -\sum_i \Delta_i(t)\hat{n}_i + \sum_{i<j} V_{ij}\hat{n}_i\hat{n}_j, \tag{1}$$

where $\hat{n}_i := |1\rangle_i\langle 1| = (\mathbb{1}_2 + \sigma_i^z)/2$ is the number operator counting the number of Rydberg excitations for atom i, $\Omega(t)$ is the (time-dependent) global Rabi frequency, $\phi(t)$ is the laser phase of the Rabi drive, $\Delta_j(t)$ is the laser detuning for atom j, and V_{ij} is the static pairwise interaction potential between atoms i and j. The interaction term is given by $V_{ij} = C_6/\|\mathbf{x}_i - \mathbf{x}_j\|^6$ for atoms positioned at $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively, where C_6 is the (atom-species-dependent) van der Waals coefficient. Schedules (or drives) specified in terms of the laser parameters $\{\Omega(t), \phi(t), \Delta_i(t)\}$ and atom arrangement $\{\mathbf{x}_i\}$ define the complete problem input for Rydberg-based analog quantum simulators available today with 256 qubits [52]. Specifically, for a given (programmable) atom arrangement, one can then explore the quantum many-body dynamics generated by the Hamiltonian $\hat{H}$ by tuning (for example) quantum-annealing and/or bang-bang-type schedules [19,22,25].

B. Rydberg blockade

The (induced) dipole-dipole interactions V_{ij} in Eq. (1) give rise to the so-called *Rydberg blockade* mechanism [29–31], whereby simultaneous excitation of nearby atoms is strongly suppressed when their spatial separation is smaller than the Rydberg blockade radius $R_b \equiv (C_6/\Omega_{\max})^{1/6}$ with $\Omega_{\max} = \max_t \Omega(t)$. Typically, $R_b \sim 1$–10 μm [55]. With the Rydberg blockade effectively preventing two neighboring atoms from being simultaneously in the excited Rydberg state (provided they are within the Rydberg blockade radius R_b), the dynamics of Rydberg atom arrays are natively restricted to coherent superpositions of independent sets [19], i.e., subsets of nodes in a graph for which no two nodes within the set are adjacent, as pertinent to the MIS problem.

C. Rydberg-native MIS-UD problem

The Rydberg blockade mechanism allows for a hardware-efficient encoding for the (hardware-native) MIS-UD problem

(and the generalized maximum-weight independent set problem on UD graphs), with Rydberg atoms placed at the nodes of the problem (target) graph and Rydberg interactions inducing edges within the (tunable) unit-disk radius $R_d = R_b$ [19,22]. Specifically, setting $\Omega = 0$ and carefully choosing $\Delta_i = \Delta > 0$ in Eq. (1), optimal solutions to the MIS-UD problem are encoded in the ground states of the Hamiltonian $\hat{H}$. In this classical limit, up to corrections from the Rydberg interaction tails, the Rydberg Hamiltonian effectively reduces to the classical MIS cost function

$$H = -\sum_{i \in V} n_i + U \sum_{(i,j) \in E} n_i n_j, \tag{2}$$

with $n_i = 1$ if node $i = 1, \ldots, N$ belongs to the independent set and $n_i = 0$ otherwise. By virtue of the Rydberg blockade mechanism, the problem in Eq. (2) is defined on a *physical* unit-disk graph $G = (V, E)$, with set of nodes V and edges E, respectively. Energetically, this (classical) Hamiltonian favors having each variable in the state $n_i = 1$ unless a pair of nodes is connected by an edge; for $U > 1$, the ground state is guaranteed to be an optimal MIS solution, because it is strictly more favorable to have at most one vertex per edge in the set as opposed to both vertices being marked. As such, the Hamiltonian H captures the MIS problem with membership in the MIS described by nodes marked as $n_i = 1$. As demonstrated in Refs. [22,25], one can then search for the MIS (encoded in the ground state of the Hamiltonian H) via, for example, coherent quantum-annealing-type approaches using quantum tunneling induced by the quantum mixer $\hat{H}_{\text{drive}}$. For atoms arranged on a two-dimensional square lattice with lattice spacing a chosen such that $\sqrt{2}a \leqslant R_d < 2a$ the UD connectivity results in a Union-Jack-like pattern [cf. Fig. 1(d)], as experimentally studied (for example) in Refs. [22,25,27]. If not stated otherwise, then we will focus on this family of hardware-native instances, but generalizations should be straightforward.

D. Generic MIS problem

Generically, graphs of interest for real-world applications will not necessarily be UD graphs and may feature system sizes (specified in terms of the number of *logical* nodes) much larger than the number of qubits provided by near-term Rydberg devices. Here we design and implement a set of tools that help expand the scope and size of problems compatible with near-term Rydberg atom arrays, allowing us to solve (potentially large-scale) MIS instances on *logical* graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ not restricted to the family of UD graphs.

E. Embedding overhead

A comprehensive, exact embedding scheme for Rydberg atom arrays has recently been put forward in Ref. [41]. In particular, this work introduces a set of gadgets whereby logical variables are redundantly encoded in one-dimensional chains of ancilla atoms to effectively distribute logical information across distant locations and couple pairs of distant qubits, while adhering to the underlying unit-disk connectivity. Generically, however, this scheme comes with a quadratic overhead that requires $N \lesssim 4n(n-1)$ physical qubits (with $N = |V|$) for the faithful embedding of a logical problem with $n = |\mathcal{V}|$ variables. Accordingly, relatively small (logical) problems with just $n \sim 10$ variables (requiring $N \sim 360$ physical qubits) can already exceed the capabilities of today's devices with ∼250 qubits [52]. To address this challenge, we propose and implement tools that effectively identify and remove redundant variables in the original problem, and we provide efficient embedding schemes with small and optimized overhead, thereby expanding the scope and size of MIS problems that can be tackled with near-term Rydberg atom arrays.

IV. THEORETICAL FRAMEWORK

In this section, we describe our larger quantum compilation pipeline for Rydberg atom arrays, as illustrated in Fig. 1. While our three core modules are detailed in the context of the full quantum compilation pipeline, we emphasize that they can be utilized independently, given their stand-alone, modular design.

A. Graph reduction

Reduction (or kernelization) techniques are known to play an integral part in state-of-the-art (SOTA) MIS solvers for both efficient exact algorithms and heuristics [45–49,57–59]. By selecting vertices that are provably part of some maximum(-weight) independent set and removing well-defined subgraphs according to relatively simple reduction rules, kernelization techniques are able to efficiently shrink the input graph to an irreducible kernel. If an MIS has been found for the kernel, then each reduction step can be undone, eventually providing a provably optimal MIS for the original graph, thanks to the optimality guarantees built into the reduction rules. For exact MIS solvers, leading approaches implement some form of branch-and-reduce search [49], where a suite of reductions intermixes branching and reduction to shrink the input size. For those (hard) instances that cannot be solved exactly, ReduMIS has emerged as one of the leading heuristic approaches [59]. In ReduMIS, reduction techniques are integrated with a heuristic, evolutionary algorithm, with recent work suggesting potential further improvements via memetic algorithms combining genetic algorithms with local search [60].

To the best of our knowledge, classical reduction techniques have so far not been adopted for quantum computing purposes in the Rydberg community. Here we provide a self-contained implementation of a simple yet efficient, lazy reduction algorithm known as isolated clique removal [45–48]. Our contributions are the following: (i) We position reduction techniques as a new tool for Rydberg atom arrays, and for the larger QO community, with applications in quantum compilation, hybrid (quantum-classical) algorithms integrating classical reduction with quantum algorithms, and the search for quantum speedups. (ii) Beyond the MIS problem, we propose a simple generalization of our reduction logic towards other (NP-hard) optimization problems, such as maximum cut or graph coloring. (iii) We show that our lazy implementation of clique-based reduction has linear run-time scaling across a large set of instances, without any limit in clique size.

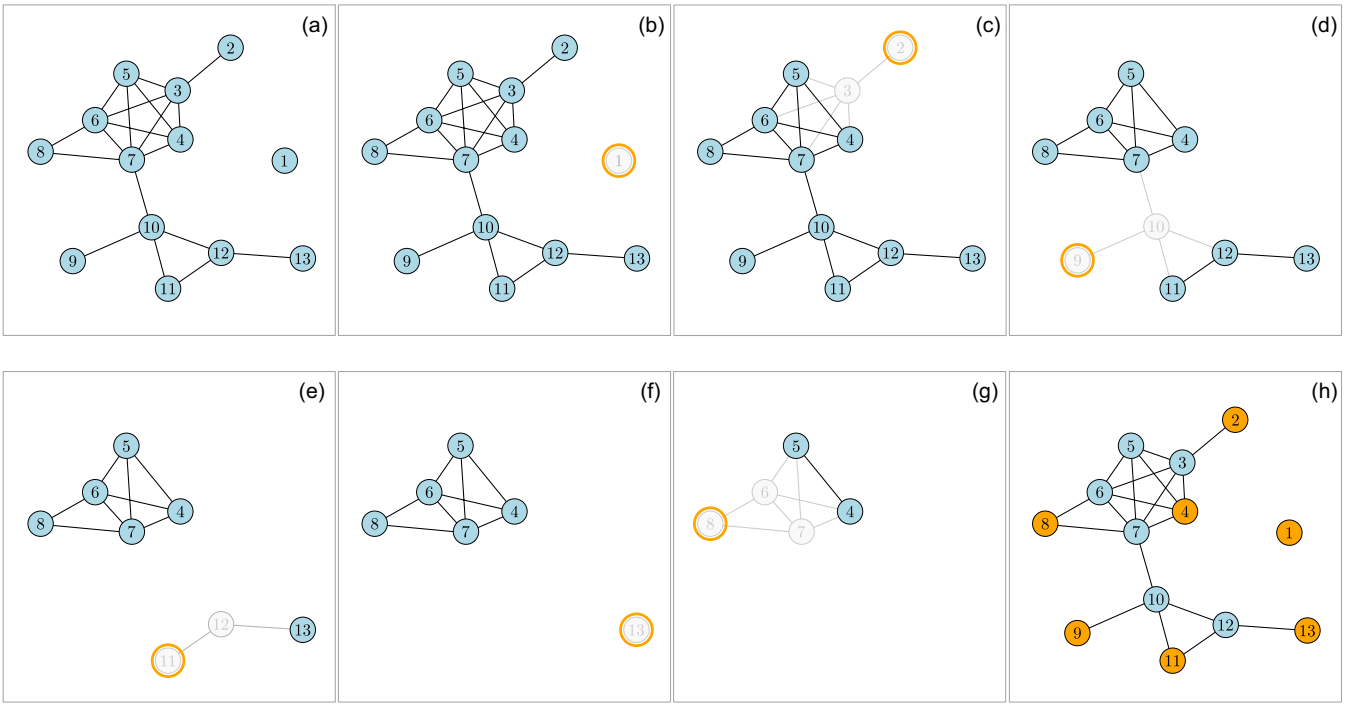


FIG. 2. (a) Schematic illustration of our clique-based graph reduction scheme for an example input graph with 13 nodes. The set of *exposed* nodes is given by {1, 2, 4, 5, 8, 9, 11, 13}. Our reduction algorithm proceeds deterministically by identifying exposed nodes that can be added to the independent set and removed from the graph along with their neighborhood, in ascending order of node degree and node label, while preserving the ability to compute an optimal solution. (b) First, any isolated nodes (i.e., cliques of size one, such as node 1 here) are added to the set and removed from the graph. Next, as shown in steps (c) and (d), dangling bonds (i.e., cliques of size two) are removed, adding nodes 2 and 9 to the set, respectively, while removing adjacent nodes 3 and 10 from the graph without selection, in accordance with the independence constraint. (e) Node 11 is selected as the lowest degree node with smallest label, thereby removing adjacent node 12. (f) Node 13, now isolated, has been added to the set. (g) After selection of exposed node 8, we are left with a simple bond from which node 4 is selected. (h) Because this graph can be reduced completely (amounting to a reduction factor of $\xi = 1$), the MIS with size $|\mathrm{MIS}| = 7$ is fully determined by the set of nodes previously selected in the reduction routine (given here by the set {1, 2, 4, 8, 9, 11, 13} highlighted in orange). The degenerate MIS solution {1, 2, 5, 8, 9, 11, 13} where node 4 is swapped for node 5 within the 5-clique subgraph can be found with the same logic after random relabeling of the nodes, or (alternatively) via simple local search.

1. Reduction scheme

Our classical graph reduction algorithm is illustrated in Fig. 2. The algorithm takes a generic graph $\mathcal{G}$ as input and, after reduction, outputs a kernel $\mathcal{K} = \mathrm{red}(\mathcal{G})$ of smaller or equal size (in polynomial time), thereby helping to identify the hard core of the MIS problem on graph $\mathcal{G}$, while stripping off the simple parts, yet preserving all information required to find an MIS for the input graph $\mathcal{G}$.

2. Reduction factor

The overall amount of reduction is measured in terms of the normalized reduction factor ξ as

$$\xi = \frac{n(\mathcal{G}) - n(\mathcal{K})}{n(\mathcal{G})}, \tag{3}$$

where $n = n(\mathcal{G})$ and $n(\mathcal{K})$ denotes the number of nodes of the original graph $\mathcal{G}$ and graph kernel $\mathcal{K}$, respectively. For $\xi = 0$, no reduction is possible, and the reduced graph is identical to the input graph. For $\xi = 1$, complete reduction has been achieved from the input graph down to the null graph, and a MIS solution has been found by mere reduction; cf. Fig. 2. For $0 < \xi < 1$, we obtain partial reduction, as is the case for the reduced Cora graph displayed in Fig. 1(b). Generically, the larger (smaller) the reduction factor ξ is, the easier (harder) the given problem instance.

3. Exposed corner nodes

Our clique-based reduction scheme is based on the simple observation that many real-world graphs feature some community-like structure in the form of cliques (i.e., complete subgraphs where every two distinct nodes in the clique are adjacent) that, by definition, can feature at most one member in the MIS. For example, see the 2-clique, 3-clique, and 5-clique spanned by nodes {2, 3}, {10, 11, 12}, and {3, 4, 5, 6, 7}, respectively, in Fig. 2. Our algorithm recursively identifies and removes certain cliques, importantly without having to solve the NP-hard maximum clique problem. To this end, we introduce the notion of *exposed* nodes (or, equivalently, *corner* nodes) to identify nodes that *only* have connections within their host clique. In the literature, those nodes are also known as simplicial vertices or isolated vertices [45–49], and corner nodes of degree one are also referred to as pendant vertices [59]. For example, for the graph shown in Fig. 2 all nodes in {1, 2, 4, 5, 8, 9, 11, 13} qualify as exposed nodes.

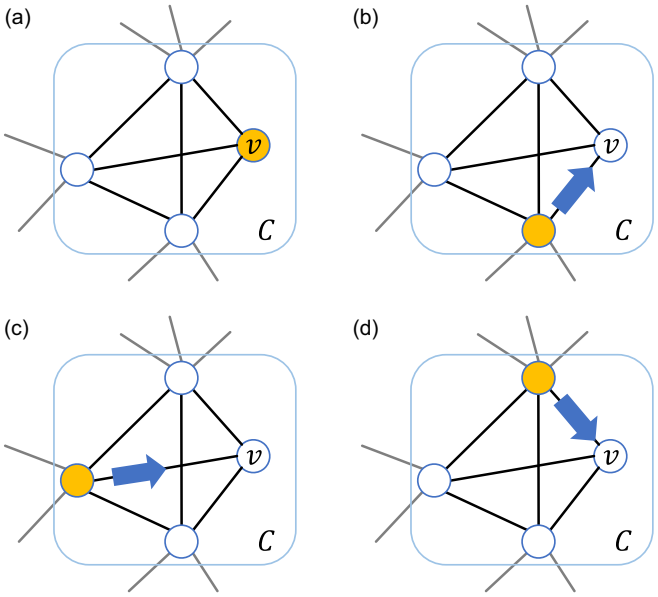


FIG. 3. Schematic illustration of the cut-and-paste argument underlying the provably optimal selection of corner nodes. Node v is part of a 4-vertex clique C. Because v has no neighbors outside of the clique C, it is an (exposed) corner node that must be part of some MIS. At most one node $u \in C$ can be selected. If $u \neq v$, then one can always swap (i.e., cut and paste) the selection of u for the selection of v, without causing any constraint violations while preserving the independent set size. Therefore, per reduction logic we opt to add v to the MIS, and remove v and its neighborhood $\mathcal{N}(v) = \{u|(v,u) \in \mathcal{E}\}$ from the graph.

Conversely, node 7 is not a corner node; given its degree, it is a candidate for a 7-clique but its neighborhood lacks the required complete connectivity. Viewed differently, node 7 is part of the 5-clique $\{3, 4, 5, 6, 7\}$, but, given its outside connections $(7, 8)$ and $(7, 10)$ it does not qualify as a corner node. Recursive selection of corner nodes only then maximizes the number of marked vertices within a given clique without blocking the selection of nodes outside of this clique. For example, selection of node 11 in Fig. 2 maximizes the number of selected nodes in the underlying clique without blocking node selections outside of this clique, whereas a (suboptimal) selection of noncorner node 7 would not only block the underlying 5-clique, but would also block nodes 8 and 10 from further addition to the independent set.

4. Optimality

This repeated selection of corner nodes is *optimal* because those nodes are provably always part of *some* MIS, as follows from a simple cut-and-paste argument [48,49,57,59]; cf. Fig. 3 for a schematic illustration. To see this, we note that at most one node from a given clique can be part of any MIS. Either it is the corner node v, or, if a neighbor of v is in an MIS, then simple reduction opts to select v instead (while preserving the independent set size).

5. Reduction algorithm

Formally, our implementation of isolated clique removal proceeds by considering each node as a potential *corner* node, for which there are only n candidates in total, by starting with low-degree nodes and working in order of increasing node degree. The algorithm proceeds in this way because smaller neighborhoods are cheaper to check and remove, with the added benefit of potentially shrinking more expensive, larger cliques prior to the neighborhood completeness check. For example, in Fig. 2, the largest 5-clique was never evaluated, simply because smaller adjacent cliques had previously been removed (by selecting the exposed nodes 2 and 8, respectively, while removing their neighborhoods). In the case of tie-breaks between exposed nodes of the same degree, we choose to remove by node label in ascending order, noting that node removals can expose more corners but cannot make any corner node irremovable, independent of the removal ordering.

The run-time for clique-based reduction is polynomial rather than exponential as in the case of the maximum clique problem. The reason for this scaling is that we are only looking for exposed cliques (i.e., those where at least one node has minimal degree of $k-1$ within a k-clique for $k = 1, 2, \ldots$) rather than searching over all cliques. For every node, we only need to check if all of its neighbors form a $k-1$ clique. As such, (potentially maximum) cliques without exposure are not detected. Given that there are n corner candidates and that the cost of clique checks are upper-bounded by $d_{\max}^2$ (where $d_{\max}$ refers to the maximum degree) we obtain a rough upper bound on the algorithmic run-time T_{red} given by $T_{\text{red}} \lesssim d_{\max}^2 n$, in the worst case (when $d_{\max} \sim n$) resulting in a run-time given by $\mathcal{O}(n^3)$. However, for *sparse* instances with bounded maximum degree (i.e., $d_{\max} = \text{const}$) we obtain a *linear* run-time scaling. In particular, we find that the average run-time $\bar{T}_{\text{red}}$ can be approximated well by a linear scaling as

$$\bar{T}_{\text{red}} \approx \gamma(\bar{d})n, \tag{4}$$

with a prefactor $\gamma(\bar{d})$ that roughly captures the structure of the instance via the average degree $\bar{d}$. As demonstrated in Secs. V and VI, we have empirically verified Eq. (4) for both synthetic and real-world graphs.

By only selecting corner nodes, our reduction algorithm is able to preserve the MIS size (also referred to as independence number) but cannot generate all possible (degenerate) MIS solutions. However, additional MIS solutions not found by clique-based reduction can potentially be generated via simple local search starting from clique-based solutions. This is done by swapping selected corner nodes for neighbors when possible; for example, in Fig. 2(h) one can swap selection of node 4 for the selection of node 5 without introducing any constraint violation, thereby generating the MIS solution $\text{MIS} = \{1, 2, 5, 8, 9, 11, 13\}$. Alternatively, one can use randomization techniques as described below.

6. Node elimination via exact splitting

Our clique-based reduction algorithm terminates once no further corner nodes can be found; see, for example, the reduced core graph shown in Fig. 1(b), which does not feature any corner nodes, and, as such, cannot be reduced further via clique removals. In these situations, further simplification may be achieved via complementary reduction schemes. For example, one can perform exact node elimination of K nodes via splitting (branching) into 2^K branches to track and evaluate

a bounded number of possibilities. Heuristically, one may choose to split on high-degree nodes, as those splits provide the largest footprint, by eliminating a large number of nodes and edges. We showcase this scheme with a concrete example below in Sec. V, whereby one node is eliminated from an irreducible core in order to unblock the clique-based reduction scheme, which can then continue to iterate with its reduction logic in another round of clique-based reduction via removal of corner nodes. In our numerical experiments, we study and quantify the reduction already possible with only the simple clique-based scheme, leaving potential extensions (such as node splitting) and integration with alternative, existing graph reduction or sparsification schemes (as discussed for example in Refs. [61–66]) to future research.

7. Randomization

Our scheme is designed to be deterministic by recursively identifying and removing exposed nodes and their neighborhoods, in ascending order of node degree and (arbitrary) node label. To generate a larger pool of solutions (if those exist), alternative solutions can potentially be found as needed by simply randomly reshuffling the node labels. Similarly, randomization of our scheme could be implemented by randomly picking a node in the case of tie-breaks between nodes with the same degree.

8. Reduction of weighted problems

Our reduction scheme can also be generalized towards the maximum-weight independent set (MWIS) problem with node weights $\{w_i\}$, e.g., in the form of simple reduction heuristics. Specifically, for instances where the distribution of node weights $\{w_i\}$ is relatively narrow, the graph reduction logic presented here should provide good (heuristic) results that may be analyzed statistically with randomization schemes as discussed above. Conversely, if the distribution of node weights $\{w_i\}$ is broad, then simple (hybrid) greedy heuristics that first add the largest-weight nodes are expected to provide good reduction results. For details on SOTA reduction techniques for the MWIS problem we refer to Refs. [49,60].

9. Problem hardness

The clique-based reduction scheme presented here is conceptually simple and fast, as demonstrated numerically in Sec. VI. Overall, we think that classical reduction schemes are of interest to near-term quantum devices for at least two reasons: First, reduction techniques help minimize the eventual load for quantum devices, thereby effectively giving quantum devices access to problem instances otherwise outside of their hardware capabilities. In particular, this is important in the near term where the number of available qubits is still in the range of hundreds to thousands. As such, integration of reduction techniques is poised to play a pivotal role in the design of new hybrid (quantum-classical) algorithms. Second, with their ability to efficiently distill a problem down to its core, reduction techniques can be an important tool in the ongoing search for classically hard problem instances, for which a potential quantum speedup is more likely to occur [22,24,27,28], thereby furthering the quantum community's ongoing efforts towards quantum advantage. To further qualify this hypothesis, we apply our reduction logic to a family of random (Rydberg-native) UJ instances in Sec. V.

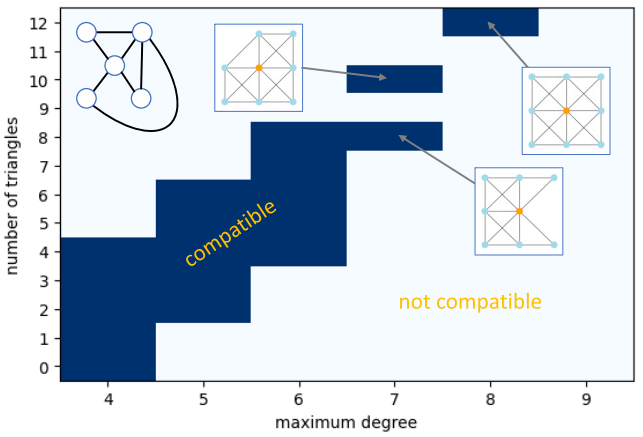


FIG. 4. Hardware compatibility diagram. Graphs are classified according to their maximum degree $d_{\max}$ and the number of triangles adjacent to maximum degree nodes $n_{\triangle}$. Regimes not compatible with the Rydberg hardware are shaded light, while candidates for a native embedding are shaded dark. Example UJ-native graphs are shown for $d_{\max} = 7$, 8, together with corresponding number of triangles. The compatibility check provides a necessary, but not sufficient condition, as illustrated by the simple non-native graph with $d_{\max} = 4$ and commensurate $n_{\triangle}$ shown in the upper left-hand corner.

10. Generalization

The reduction scheme introduced here has been applied to the MIS problem. However, similar considerations can be applied to other (NP-hard) combinatorial optimization problems often discussed in the quantum optimization community as benchmark problems, such as maximum cut (MaxCut) or the graph coloring problem (GCP). For example, just as done above for both MaxCut and the GCP, dangling bonds (exposed corner nodes with degree one) can be removed recursively and labeled optimally in simple postprocessing routines by mere neighborhood association, given the assignment for the reduced core graph. In particular, such reduction schemes could find applications when dealing with (structured) real-world networks that often feature (approximately) power-law degree distributions, with the majority of nodes having a low degree (e.g., one, two, or three) and a few highly connected hubs (e.g., large hub airports in transportation networks) [67].

B. Hardware compatibility checks

We now describe in detail the Rydberg hardware compatibility checker (HCC). The HCC logic performs a series of $l \geqslant 1$ simple checks to determine if a given graph can be encoded natively with Rydberg hardware or not. For concreteness, following experimental demonstrations in Refs. [22,25,27], we focus on setups where Rydberg atoms are placed on a two-dimensional square lattice with nearest and next-nearest interactions, resulting in aUJ-like connectivity with vacancies, as illustrated in Fig. 1(c).

Simple ($l = 2$) compatibility checks can be performed based on the diagram shown in Fig. 4. Here a given graph

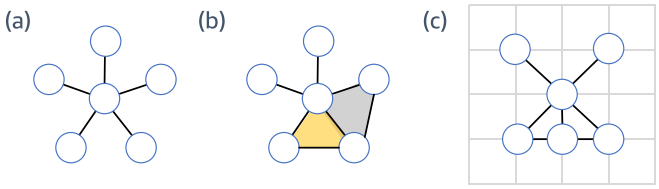


FIG. 5. Example for simple hardware compatibility checks. (a) Simple star graphs with $d_{\max} \geqslant 5$ are not UJ compatible, because triangle-free graphs are supported natively only up to $d_{\max} = 4$. (b) Similar but denser graphs with $d_{\max} = 5$ and $n_{\triangle} = 2$ are UJ compatible, as shown in panel (c).

is classified according to its maximum degree $d_{\max}$ and the number of *triangles* (consisting of three vertices and three edges, in the form of a triangle) adjacent to the maximum degree nodes $n_{\triangle}$. Accordingly, graphs with maximum degree $d_{\max} > 8$ *cannot* be supported natively by Rydberg arrays with UJ connectivity, while graphs with $d_{\max} \leqslant 8$ *may* be compatible only if the number of adjacent of triangles $n_{\triangle}$ is commensurate with the restricted UJ connectivity. For example, as shown in Fig. 4 graphs with $d_{\max} = 8$ can only be UJ compatible if $n_{\triangle} = 12$, while graphs with $d_{\max} = 7$ can only be UJ compatible if $n_{\triangle} = 8$ or $n_{\triangle} = 10$ (depending on whether a boundary or corner node is missing with respect to the unit-filling case). Accordingly, a graph with $d_{\max} = 7$ and $n_{\triangle} = 9$ is not compatible with the UJ connectivity and will require ancilla qubits for a faithful embedding. We emphasize that the maximum degree (or, equivalently, *local* density) is not a good indicator for native embeddability per se. For illustration, see Fig. 5 showing that a simple star graph with $d_{\max} = 5$ is not UJ compatible, while a similar graph with higher density and $n_{\triangle} = 2$ can be embedded natively with UJ connectivity.

The simple HCC outlined above provides a necessary, but not sufficient criterion for native embeddability. For example, the $d_{\max} = 4$ graph shown in the inset of Fig. 4 passes the check but cannot be embedded natively. Such false positives could be caught with extended, more fine-grained checks involving, for example, other graph properties (such as the number of chordless cycles) in a deeper ($l > 2$) catalog of checks.

Finally, we note that the classification outlined here does not have to be aggregated on the graph level but may be performed locally on the node level. Specifically, given its properties (such as degree and number of adjacent triangles) every single node may be classified as (potentially) hardware compatible or not, thereby providing local information that can be utilized in downstream embedding logic. For example, with the central ($d_{\max} = 5$) node in Fig. 5(a) classified as non-compatible, one could "roll-out" this node using a simple copy gadget [41] to effectively support the required connectivity with the help of ancilla qubits.

C. Graph embedding

We now outline two complementary embedding strategies for Rydberg atom arrays. Our first (bottom-up) approach targets *approximate* embeddings with minimal resource requirements, while our second (top-down) scheme provides *exact* embeddings with optimality guarantees, at the expense of a larger, yet optimized overhead.

1. Bottom-up embedding scheme

We first describe an approximate, hardware-efficient (bottom-up) embedding heuristic we refer to as Generative Approximate Graph Embedder (GAGE), in which optimized atomic (vertex) positions and the corresponding physical graph are learned via iterative training. Our scheme makes use of the random-key formalism to efficiently identify a physical (hardware-native) graph G that is optimized to be as similar as possible to the target (input) graph $\mathcal{G}$, according to some desired metric. In the following, we first briefly review the random-key formalism, and then we outline its application to the embedding problem at hand.

(a) Random key optimizer (RKO). A *random key* is a real number in the continuous interval $[0, 1)$, and a vector χ of random keys is an array $\chi \in [0, 1)^D$. The RKO formalism is based on the idea that solutions to optimization problems can be encoded as vectors of random keys, providing an abstract (problem-independent) embedding for solution candidates in the latent (hypercube) space $[0, 1)^D$, with the encoding dimension D chosen by the user. Such a vector χ is mapped to a feasible solution of the optimization problem with the help of a deterministic (problem-dependent) *decoder* that takes as input a vector of random keys and returns a feasible solution $\mathcal{D}(\chi)$ to the optimization problem, as well as the cost of the solution. Based on a clear separation of problem-independent and problem-dependent modules, given by the encoding space $\mathcal{H}_D = [0, 1)^D$ and the decoder $\mathcal{D}(\cdot)$, respectively, RKO provides standardized plug-ins for a plethora of optimization paradigms [68], such as genetic algorithms [69,70], annealing-type algorithms [71], or greedy randomized adaptive search procedures [72], among others, all within one unified framework. For more details on the random key formalism and RKO we refer to Refs. [69,70], and references therein.

(b) RKO embedder. Here we utilize RKO to generate a physical graph G with adjacency matrix A that is similar to the logical graph $\mathcal{G}$ with adjacency matrix $\mathcal{A}$ yet easy to implement on Rydberg atom arrays with native connectivity constraints. We focus on embeddings within two-dimensional square lattices with lattice spacing a, as realized experimentally in Ref. [22]; generalizations to other settings (such as those with continuous atomic positions) should be straightforward. Our decoder takes a vector χ of $N_{\text{sites}} = L_x \cdot L_y$ random keys as input (where N_{sites} is the number of available atomic positions in the underlying lattice) and sorts the keys in increasing order. Similarly to canonical optimization problems, such as the traveling salesperson problem or vehicle routing [71], the indices of this sorted vector make up the solution, where $n < N_{\text{sites}}$ atoms are placed according to the first n indices associated with the sorted vector. This design ensures placement of at most one atom per available site, as desired, while also adhering to placements within the constrained space with width L_x and height L_y. For a given placement, one can then easily extract the corresponding graph G for a given blockade radius $r = R_b/a$. The cost (or, equivalently, fitness) associated with such a placement can be calculated via graph

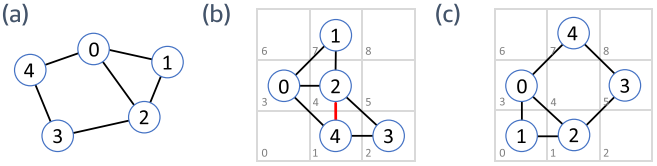


FIG. 6. Example for the RKO-based generative approximate graph embedder (GAGE), for an underlying square lattice with $N_{\text{sites}} = 9$ sites labeled as $i = 0, 1, \ldots, 8$. (a) Input graph $\mathcal{G}$ with five nodes. (b) Approximate (hardware-native) embedding G with edit distance $d_{\text{edit}} = 1$. This node placement corresponds to the example random-key vector given in Eq. (6). Edges are drawn for nearest and next-nearest neighbors corresponding to native UJ connectivity, here resulting in one undesired edge $(2, 4)$ highlighted in bold red. Note that this embedding, even though approximate, supports MIS solutions such as $\{0, 3\}$ or $\{1, 4\}$ that are optimal solutions for the original input graph $\mathcal{G}$. (c) Perfect (hardware-native) embedding G with edit distance $d_{\text{edit}} = 0$, as encoded by the example random-key vector $\chi = (0.16, 0.23, 0.71, 0.05, 0.62, 0.29, 0.79, 0.47, 0.98)$.

similarity measures, such as the standard graph edit distance, e.g.,

$$d_{\text{edit}} = \frac{1}{2}\sum_{ij}(\mathcal{A}_{ij} - A_{ij})^2, \tag{5}$$

which is then fed back to the optimizer as a feedback signal to be optimized over. RKO then learns a hardware-native embedding, guided by this similarity measure as a feedback signal, by traversing the random-key space in the form of, e.g., genetic [69,70] or annealing-based [71] updates. Upon completion, after a series of training steps, RKO outputs an approximate graph G of low cost that is as similar as possible to the input graph $\mathcal{G}$, given the hardware connectivity constraints.

Let us illustrate the GAGE embedding scheme as described above with a simple example; cf. Fig. 6 for illustration. Consider a simple input graph $\mathcal{G}$ with five nodes as illustrated in Fig. 6(a). We search for a faithful embedding of $\mathcal{G}$ on a square lattice with $L_x = L_y = 3$ and Union-Jack connectivity ($\sqrt{2} \leqslant r < 2$). Let us assume, for example, that the current solution candidate is given by the random-key vector

$$\chi = (0.84, 0.34, 0.27, 0.07, 0.18, 0.42, 0.71, 0.13, 0.54). \tag{6}$$

The decoder takes this vector χ and applies simple sorting logic (in ascending order), resulting in $\text{sort}(\chi) = (0.07, 0.13, \ldots, 0.84)$. The corresponding vector of indices is given by $\text{argsort}(\chi) = (3, 7, 4, 2, 1, 5, 8, 6, 0)$. Accordingly, the first node is placed at position 3, the second one at position 7, and so on until all five nodes are placed, leaving sites $(5, 8, 6, 0)$ unoccupied. Assuming hardware-native UJ connectivity, we then readily obtain the corresponding candidate graph shown in Fig. 6(b). This is an example of a low-cost (high-quality) solution candidate G with Union-Jack connectivity, with edit distance $d_{\text{edit}} = 1$. Further (heuristic) optimization in additional training steps and restarts will search for even better solutions, with the potential to ultimately identify the best possible Union-Jack embedding, as shown in Fig. 6(c) for a perfect embedding with $d_{\text{edit}} = 0$. The runtime of the RKO embedder can be controlled with a simple stopping criterion (such as a maximum number of iterations, a maximum number of iterations without improvement, or a maximum run-time).

(c) Embedding refinement. In the simple decoder design outlined above, we systematically search for the best possible hardware-native embedding with zero qubit overhead. The quality of the embedding (measured, e.g., in terms of graph similarity) may, however, be improved with the help of additional ancilla qubits, using (for example) the gadgets described in Ref. [41]. Specifically, within an extended decoder design, one might also be able to learn the optimized placement of ancilla atoms, as encoded in additional ancilla keys. Alternatively, we can take the (approximate) overhead-free embedding solution as generated by GAGE and try to refine this embedding via optimized placement of ancilla quantum wires (QWs); see Fig. 1(d) for an example illustration. To this end, we introduce the notion of *QW openings*, which refer to vacant sites whose neighborhood is empty except for a connection to the original node. In Fig. 1(d), we show an example in which potential QW openings have been highlighted in green and the QW openings for nodes 1 and 9 have been used to effectively couple those through an even numbered ancilla chain at the expense of eight additional qubits. This ancilla chain effectively acts as an edge that is missing in the zero-overhead embedding, thereby improving the embedding quality from $d_{\text{edit}} = 1$ down to $d_{\text{edit}} = 0$. To systematically find the placement of QW wires, we frame the ancilla placement problem as a constrained shortest path problem (SPP) within free (i.e., available) sites connecting two desired QW openings. To solve this SPP, we use a breadth first search algorithm that preserves cardinality (by restricting ancilla wires to be even numbered for proper redundant encoding) and avoids sharp turns via direction tracking(so as to not introduce undesired edges within the wire).

(d) Characteristics and limitations. As exemplified in our experiment underlying Fig. 1(c), our RKO-based embedder, in conjunction with potential refinement through quantum wires, can learn and identify (typically approximate) low-overhead embeddings. In particular, the zero-overhead embedding found via RKO can be used as a another tool to assess the hardware *friendlessness* of a given input graph, as quantified by the edit distance d_{edit}. While the example graph displayed in Fig. 1(b) is found to be hardware friendly, further numerical experiments have shown that generic input graphs are typically not hardware-friendly (as expressed by a large edit distance), i.e., not even approximately compatible with the limited UJ hardware connectivity. See, for example, Fig. 7 which showcases an overhead-free, bottom-up embedding result for the largest component of the kernel of the Dolphins network (described below) with a relatively large edit distance $d_{\text{edit}} = 5$. In particular, we observe that the hardware-native embedding introduces undesired extra edges not present in the input graph, thus calling for the use of (copy) gadgets to effectively distribute logical information across distant locations. To address this challenge, we design and implement a complementary top-down embedding scheme described below.

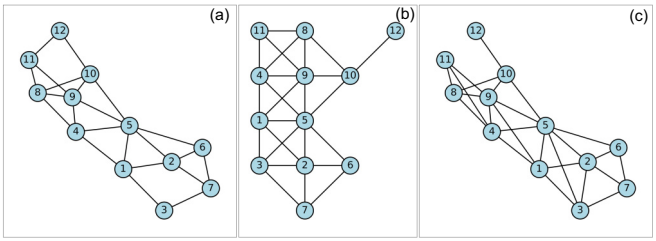


FIG. 7. Example overhead-free embedding for a hardware-unfriendly input graph. (a) Input graph $\mathcal{G}$ with 12 nodes. (b) Approximate (hardware-native) embedding G with edit distance $d_{\text{edit}} = 5$, as found with the RKO embedder. (c) The same graph G with node positions from panel (a) for simple visual comparison with the input. While missing the logical edge (11, 12), the hardware-native UJ connectivity introduces four undesired extra edges, namely (1, 9), (2, 3), (3, 5), and (4, 11).

2. Top-down embedding scheme

Our second embedding approach builds on the generic embedding scheme outlined in Ref. [41]. This scheme is designed to provide a (physical, hardware-native) embedding for any potential (logical) input graph, regardless of its size, edge density or specific interactions, with an exact one-to-one correspondence between the ground states of the logical and embedded graphs. The recipe calls for the representation of all logical nodes in the form of extended qubit chains (i.e., copy gadgets), arranged such that all pairs intersect at exactly one crossing. Edges between logical nodes are implemented by selecting an "interacting" intersection gadget for the crossing, while crossings of node-chains without an edge in the logical graph use the "noninteracting" crossing gadget (see Fig. 8). We employ a version with odd-length chains and detunings adjusted accordingly. For a graph with n nodes and m edges, this scheme then results in an embedding with the following qubit count:

$$N_{\text{Qubits}} \approx 8 \cdot \frac{n(n-1)}{2} - m + 5n, \tag{7}$$

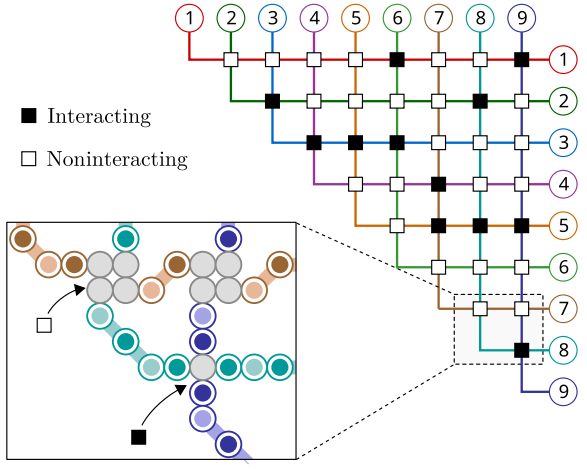


FIG. 8. Representation of the generic embedding scheme from Ref. [41] as a wire representation (main panel), with individual qubits forming intersection gadgets (inset). Filled squares represent interacting crossings, while empty squares are noninteracting gadgets. The graph shown here corresponds to the logical graph in Fig. 1(b).

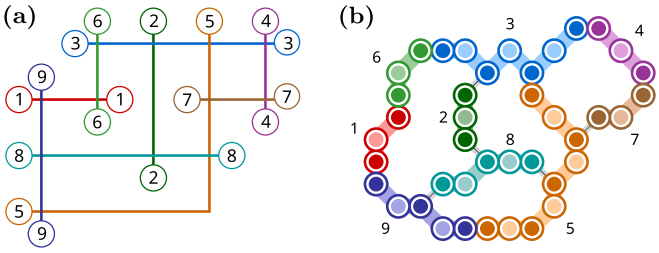


FIG. 9. (a) Rearranged and shortened chains for the graph from Fig. 8. Here a geometry has been identified where no interacting gadgets are necessary: All interactions happen at the end of at least one of the chains involved. As a result, usage of costly crossing gadgets can be completely avoided. (b) Visualization of the optimized qubit placement for chains with the interactions dictated by the simplified chain geometry. For each terminal crossing in Fig. 9(a), the placed chains are touching at odd-numbered offsets (indicated by darker-filled circles), while ancilla qubits (lighter filling) only touch odd-numbered qubits of the same chain.

with eight qubits per crossing (reduced by m because interacting gadgets use seven vs eight qubits) and five extra qubits per chain for the ends and corners. Accordingly, the overhead in the number of qubits scales roughly as $\sim 4n^2$, assembled within an embedding area of $(4n+4)\times(4n+3)$, which scales as $\sim 16n^2$.

(a) Overhead reduction. While some simple overhead reduction strategies are discussed in Appendix A of Ref. [41], here we aim to systematically reduce this overhead while preserving the ability to accurately represent a given graph—in particular for lower densities and graphs typically found after reduction. To this end, we make use of the following four observations (cf. Fig. 9):

First, the suggested arrangement of chains with all pairs intersecting allows for an embedding scheme with the same placement for any graph. However, for a specific graph, we are free to choose a different chain geometry. Starting from the upper-right triangle arrangement in Fig. 8, we can change the ordering of node labels on the right/top such that some pairs never intersect. As long as we only drop noninteracting intersections with this reordering, we preserve the ability for an exact representation, while reducing the overhead through removal of redundant noninteracting gadgets.

Second, while chains extending to the top/right of the arrangement are convenient for readout, this is not a necessity. With odd-length chains, we can instead choose to shorten each chain from both ends up to their first/last *interacting* crossing.

Third, interactions at the end of a chain can be implemented without a gadget, by placing the terminal qubit next to an odd-numbered qubit of the chain it needs to interact with, thereby allowing us to avoid additional interacting intersections.

Fourth, the resulting chain representation can be translated into an actual qubit arrangement with straights represented by copy gadgets. We can further reduce the qubit overhead by relaxing gadget positioning and rewriting chain paths (provided that parity and spacing requirements are observed).

(b) Embedding algorithm. In practice we employ simulated annealing [73] to identify an optimized end-point ordering, using the number of gadgets needed after steps 1–3 as the

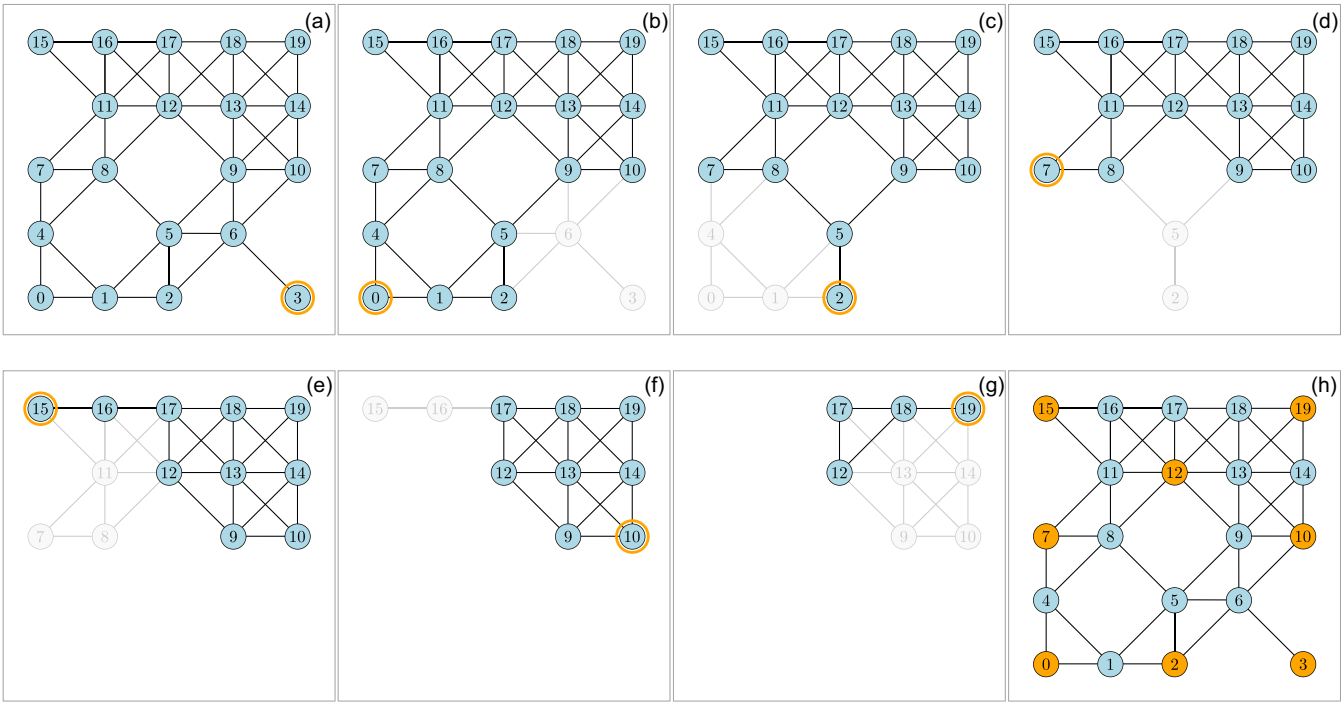


FIG. 10. Example application of our general-purpose reducer to a hardware-native random UJ instance with $n = 20$ nodes on a square lattice with $L = 5$ (i.e., 25 sites) and filling $\varrho = 0.8$. The logic follows the one detailed in Fig. 2. By iteratively removing exposed corner nodes, the input graph in panel (a) can be fully reduced [panels (b)–(g)], deterministically providing an optimal solution with independence number $|\mathrm{MIS}| = 8$, as highlighted in panel (h). Additional MIS solutions can be generated via local search (e.g., swapping node 2 for node 5 or node 12 for node 17). Given that this example instance can be solved to optimality by reduction alone, it is classified as *easy*.

merit function. Empirically, the best results were achieved using a combination of nonergodic swap moves that maintain necessary crossings, along with frequent restarts from randomized initial orders. After translation to a qubit arrangement, we further reduce the overhead by adjusting gadget positions in a secondary Markov-chain process: At each step, a gadget and all the chains connecting to direct neighbors are removed, followed by a proposed location adjustment and rewiring. The new connections are constructed using a biased breadth-first search that takes qubit interactions into account (both within the chain itself and with existing chains/gadgets), while ensuring that the resulting length has the correct parity. If successful, then the new placement is accepted based on a standard Metropolis criterion on the change in qubits used.

V. IMPLICATIONS FOR PROBLEM HARDNESS AND QUANTUM SPEEDUPS

To illustrate the stand-alone value of the reduction module outlined above, we now first apply our reducer to the Rydberg-native family of random UJ instances, as previously studied in, e.g., Refs. [22,28,25,27]. We find that a large fraction of these instances can be reduced substantially or even solved to optimality in polynomial time by reduction alone, thereby providing us with an efficient tool to classify instances as easy or (potentially) hard. We also pinpoint signatures of an "easy-hard-easy" transition that allows us to tune problem hardness. Based on this evidence, we conclude that classical reduction routines (and generalizations thereof) provide a useful tool in the ongoing search for hard problem instances and quantum speedups and as such should be integrated into future hybrid (quantum-classical) end-to-end MIS and MWIS solvers.

A. Easy UJ example instance

For illustration, first consider the small random UJ instance shown in Fig. 10 (taken from Ref. [23]), with $n = 20$ nodes on a square lattice with $L = 5$ and filling fraction $\varrho = 0.8$. By iteratively removing exposed corner nodes, we straightforwardly find an optimal solution with independence number $|\mathrm{MIS}| = 8$ via reduction. As such, this example instance can be classified as *easy*, representing a candidate instance that is unlikely to showcase potential future quantum speedups. Quantum speedups are more likely to be found for *hard* instances that feature a large kernel after reduction [48]. We note that such a definition of *hard instances* should involve a full suite of reduction algorithms, beyond the clique-based reducer implemented and tested here. Specifically, it is easy to construct UJ instances that are immune to clique-based reduction by simply removing all four exposed corner nodes. However, those instances could be unblocked with additional reduction logic, e.g., in the form of the splitting techniques discussed above. Nonetheless, here we focus on clique-based reduction only, because it provides a simple tool with which to identify easy instances, and we will assess its reduction performance in detail.

B. Larger UJ example instances

Next, we consider four larger example instances previously studied in Refs. [25] and [74]. While those four instances are all random UJ instances of the same size (with $n = 137$

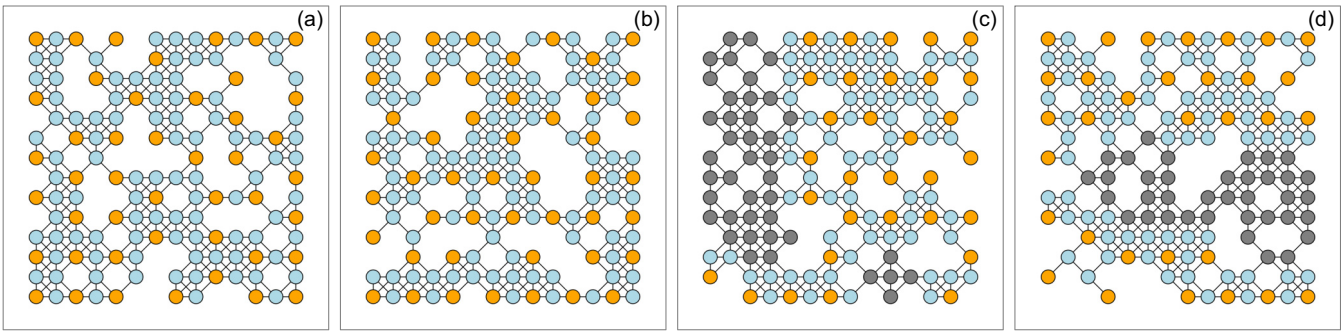


FIG. 11. Reduction results for larger, hardware-native, random UJ instance with $n = 137$ nodes on a square lattice with $L = 14$ and filling fraction $\varrho \sim 0.7$. The hardness parameter of these four instances ranges across three orders of magnitude, with $\mathbb{H} \sim 1.478$ in panel (a), $\mathbb{H} \sim 14.12$ in panel (b), $\mathbb{H} \sim 125.5$ in panel (c), and $\mathbb{H} \sim 1435$ in panel (d). Orange nodes have been selected ($n_i = 1$), while blue nodes have not been selected ($n_i = 0$). Kernel nodes are shown in dark gray. The instances in panels (a) and (b) can be solved to optimality by simple reduction, with $|\mathrm{MIS}| = 45$ and $|\mathrm{MIS}| = 47$, respectively. The instances in panels (c) and (d) can be reduced by $\xi \sim 70\%$ and $\xi \sim 73\%$, with 31 and 32 nodes selected outside of the kernel, leaving smaller kernel instances of size 41 and 37, respectively. It is straightforward to see that the resulting kernel graphs do not feature any exposed (corner) nodes.

on a square lattice with side length $L = 14$), their *hardness parameters* $\mathbb{H}$ span three orders of magnitude [22,23,28], with $\mathbb{H}$ defined as $\mathbb{H} = D_{|\mathrm{MIS}|-1}/(|\mathrm{MIS}| \cdot D_{|\mathrm{MIS}|})$, where D_α denotes the degeneracy of the independent sets of size α. For both classical and quantum Markov-chain Monte Carlo (MCMC) algorithms, the conductance-like hardness parameter $\mathbb{H}$ captures the likelihood to get stuck in a local minimum of size $|\mathrm{MIS}| - 1$, as opposed to reaching a ground state of size $|\mathrm{MIS}|$ [22]. The corresponding success probability to find the MIS in a single algorithmic run (shot), denoted as P_{MIS}, shows an exponential dependence on $\mathbb{H}$. Specifically, $P_{\mathrm{MIS}} \approx 1 - \exp(-C\mathbb{H}^{-\alpha})$, where α depends on the given MCMC algorithm and C refers to a positive fitted constant that could have polynomial dependence on the system size in general. In particular, $P_{\mathrm{MIS}} \approx 0.478$, $P_{\mathrm{MIS}} \approx 0.004$, $P_{\mathrm{MIS}} \approx 0.019$, and $P_{\mathrm{MIS}} = 0$ has been reported for the four instances shown in Fig. 11, notably with $P_{\mathrm{MIS}} = 0$ for the hardest instance ($\mathbb{H} \sim 1435$), even for an optimized quantum algorithm [74]. Conversely, our reduction algorithm is insensitive to the hardness parameter $\mathbb{H}$ tailored towards MCMC-based algorithms, by design. As demonstrated in Fig. 11, all four instances can be either solved to optimality or reduced substantially using our simple clique-based reduction logic alone. In particular, for two instances we find $\xi \sim 70\%$ and $\xi \sim 73\%$, respectively, while the other two instances are even solved to optimality by reduction alone. As such, given this efficiency of the inexpensive reduction logic, usage of relatively expensive quantum algorithms (both in terms of wall clock run-time or actual monetary cost) can either be made more efficient (with a reduced workload for the QPU and higher success rate given the known size dependence of the hardness parameter [22,28]) or even become unnecessary.

This statement can be generalized through repeated (optimal) reduction, as exemplified in Fig. 12. Here it is shown that the irreducible kernel taken from Fig. 11(d) can be unblocked via node elimination (that exposes new corner nodes) and subsequent repeated reduction. Overall, with just two rounds of reduction and the combination of selected nodes over all reduction steps, we then find the optimal solution for this problem of $|\mathrm{MIS}| = 32 + 13 = 45$, with $P_{\mathrm{MIS}} = 1$, thereby outperforming the optimized quantum algorithm from Ref. [74]. Further extensions of this idea (with potential plug-ins for quantum computers as coprocessors) will be discussed below in Sec. VII.

C. Systematic experiments

Next, we perform systematic experiments to quantify typical reduction values across a large set of random UJ instances. Specifically, we analyze the reduction factor ξ as a function of system size (given in terms of the number of nodes n) for different values of the lattice filling fraction ϱ, with large reduction pinpointing easy instances with a small kernel of size $(1 - \xi)n$ (i.e., the larger ξ, the easier the instance). Our results are displayed in Fig. 13, for UJ instances with $L = 15, 17, \ldots, 49$, resulting in graphs with up to $n = 1921$ nodes (still beyond the scale of today's quantum hardware with $L \approx 16$ [52]). All reduction experiments completed on subsecond timescales per instance on a laptop (taking on average only ∼0.02 s for

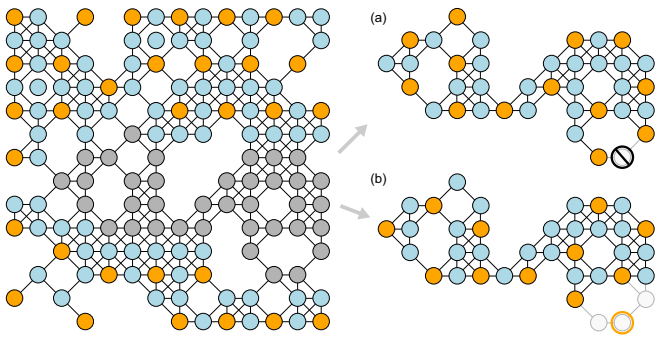


FIG. 12. Optimal MIS solution through repeated reduction post elimination via exact splitting. The irreducible kernel shown in Fig. 11(d) has been split based on the node in the lower right-hand corner, given that this node (like any other node) can only be selected (b) or not (a). Removal of the selected node (and its neighborhood if selected) unblocks clique-based reduction. Post splitting, the remaining kernels can be solved to optimality via mere reduction, with 13 (11) nodes selected in scenarios (a) and (b), respectively.

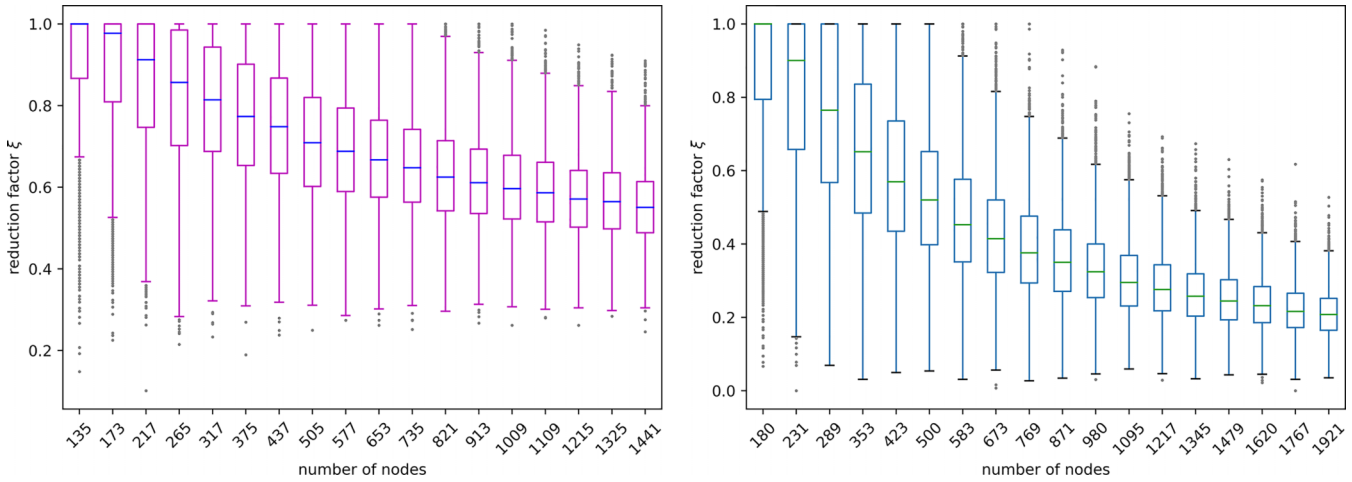


FIG. 13. Box plot for the reduction factor ξ as a function of the number of nodes for random UJ instances with filling fractions $\varrho = 0.6$ (left) and $\varrho = 0.8$ (right), respectively. The underlying sizes of the square lattices range from $L = 15$ up to $L = 49$ (with $n = \varrho L^2$), with 5000 random seeds per size. Boxes extend from the first quartile (25th percentile) to the third quartile (75th percentile), with a horizontal colored line at the median. Whiskers extend from the box to the farthest data point lying within 1.5 times the interquartile range from the box. Outliers outside of the whiskers are shown individually as dots.

instances with ∼1000 nodes), following an approximately linear run-time scaling. For more details on algorithmic run-time, we refer to Appendix. Within the range of instances probed, we find that typical instances can be reduced substantially, with strong instance-to-instance variations even for fixed system size, somewhat reminiscent of the strong spread in hardness reported in Refs. [22,28]. For small instances compatible with today's quantum hardware (with $L \leqslant 15$), the median reduction is found to amount to $\xi = 1$, meaning that a given random (small) UJ instance can be solved to optimality via mere reduction with at least 50% probability (akin to the example shown in Fig. 10). For larger instances with $L > 15$, we observe a slow decrease in the average reduction, suggesting a potential plateau with a nonzero (density-dependent) fixed point in the asymptotic limit. This density dependence is already evident from Fig. 13, with denser instances with $\varrho = 0.8$ typically showing smaller reduction values than sparser instances with $\varrho = 0.6$ for graphs of comparable size n.

D. Easy-hard-easy transition

To probe this density dependence further, we analyze the reduction factor ξ as a function of the filling fraction ϱ for a fixed number of nodes n. Our results for $n = 1000$ are shown in Fig. 14. We observe indications of an easy-hard-easy crossover, as expected, because both sparse and complete UJ instances are prone to complete reduction (with $\xi = 1$). The smallest average reduction (corresponding to the hardest instances for reduction logic) is observed for filling fractions around ∼80%, resulting in an average degree of $\bar{d} \sim 6$, which is consistent with the results in Ref. [28] reporting the hardest random UJ instances at similar filling values. For filling fractions in the range ∼0.8–0.92, we observe a large instance-to-instance spread in reduction values, featuring UJ instances effectively immune to reduction (with $\xi \approx 0$) as well as easy instances (with $\xi \approx 1$), even for a fixed filling fraction ϱ. This is reminiscent of the phase transition and empirical hardness spike observed for random propositional satisfiability (SAT) problems [75]; however, we did not determine in detail if this was an actual phase transition or a crossover. A more detailed analysis thereof will be left for future research.

E. Implications for problem hardness and quantum speedups

While we have focused on random UJ instances, similar conclusions should hold for other instances previously studied with Rydberg atom arrays (e.g., Cayley tree instances [76]), simply because those instances feature exposed corner nodes making them prone to reduction. Given that reduction algorithms effectively reduce the problem size as $n \to (1 - \xi)n$

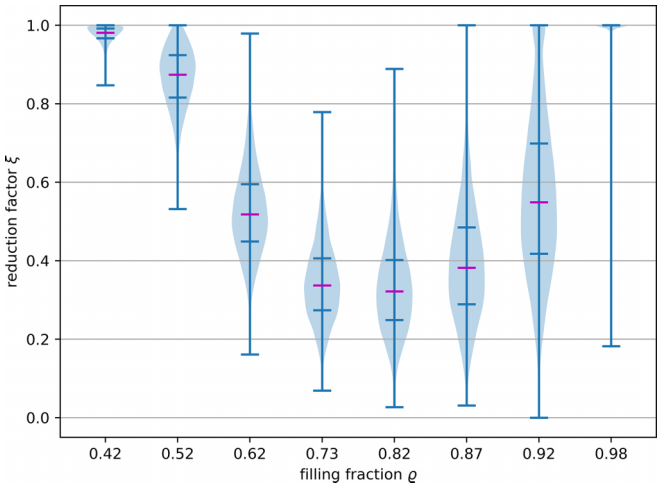


FIG. 14. Violin plot for the reduction factor ξ as a function of the filling fraction ϱ for random UJ instances with $n = 1000$ nodes. The underlying sizes of the square lattices range from $L = 32$ for dense instances with $\varrho \approx 0.98$ up to $L = 49$ for sparser instances with $\varrho \approx 0.42$, with 10^4 random seeds per size. Extrema, first quartile (25th percentile), median (50th percentile), and third quartile (75th percentile) are marked with horizontal lines. The data suggest an easy-hard-easy crossover as a function of the filling fraction.

with polynominal run times, they can reduce the generically exponential run-time of exact downstream solvers from $\sim 2^{\alpha n}$ to $\sim 2^{\tilde{\alpha} n}$ with $\tilde{\alpha} = (1 - \xi)\alpha$, meaning that a reduction of, for example, $\sim$50% can effectively result in a quadratic speedup. As such, reduction algorithms can have a strong impact on problem hardness as measured by, e.g., the overall time to solution for generic pipelines combining reduction techniques with downstream solvers. Therefore, adopting the notion put forward in the classical literature according to which only those MIS instances with a large kernel size are considered *hard* [48], our results should help the Rydberg and larger quantum community zero in on truly hard instances, thereby helping to direct efforts in the ongoing search for potential quantum advantage.

VI. NUMERICAL EXPERIMENTS

We now provide additional numerical results illustrating the performance of our larger compilation pipeline. First, we discuss an end-to-end example for the well-known Cora citation graph, featuring results obtained from the QuEra Aquila device available on Amazon Braket. Next, we report on systematic experiments providing a comprehensive assessment of the achievable reduction factor for both real-world networks and families of synthetic graphs. Finally, we provide numerical experiments to quantify the typical embedding overhead for a family of graph instances.

A. End-to-end example

For illustration, we first consider the well-known Cora citation graph with $\sim$2700 nodes and $\sim$5300 edges [50]; see Fig. 1. The size of this network goes well beyond the scale supported by today's quantum hardware (which is limited to hundreds of qubits with restricted connectivity [52]).

In an effort to make instances of this scale potentially compatible with near-term quantum hardware, we first apply our clique-based reduction scheme and find that the Cora graph can be reduced by $\sim$97% to a core with 79 nodes and 94 edges, with maximum degree $d_{\max} = 4$ (compared to $d_{\max} = 168$ for the original graph), within subsecond run-time on a laptop. Moreover, this core is found to decompose into 16 connected components that can each be solved individually in parallel. The majority of these components (9 of 16) are simple cycle graphs that are immune to clique-based reduction but are trivial to solve (i.e., every other node can be added to the independent set). The largest component of the core graph is displayed in Fig. 1(b) and has nine nodes, with the second largest having just six nodes, overall resulting in a large reduction in complexity and load for any downstream MIS solver.

After reduction, the core graph can be solved with a variety of MIS solvers. Here we utilize our embedding logic to make the remaining core graph compatible with Rydberg atom arrays that support the implementation of *quantum* algorithms to approximately solve the MIS problem on the embedded (physical) graph [22]. The individual components of the core can be embedded and solved individually, with the potential to fit more than one component on an annealing device, provided both enough space and qubits are available. For illustration, we focus on the largest component. As displayed in Fig. 1(c), we find that this core graph (with nine nodes and twelve edges) can be embedded onto a hardware-native graph with nine nodes, 11 UJ edges, and one quantum wire with eight ancilla nodes. We then solve the MIS problem on this embedded graph via quantum-annealing techniques that leverage neutral atoms [19,25]. To this end, nodes are mapped to qubits subject to analog quantum dynamics under the many-body Hamiltonian given in Eq. (1).

Our results are shown in Fig. 15, with the underlying simple, piecewise linear annealing schedule detailed in Fig. 15(c). In particular, we set $a = 4.5$ μm and drive parameters were chosen such that $\sqrt{2} \leqslant R_b/a < 2$ as required for UJ connectivity. Here we outline two complementary experimental strategies: (i) *Relaxed graph*: To keep the embedding overhead minimal, we first solve the MIS problem on a *relaxed*, hardware-native graph that is optimized to approximate the original input graph up to one missing edge (resulting in an edit distance of one) while adhering to UJ connectivity constraints. Exact (noise-free) simulations of this nine-qubit system's quantum dynamics show that the Rydberg solver acts as a *biased* MIS solver [77], favoring those MIS solutions that minimize the Rydberg interaction tails. Specifically, we find that of eight possible MIS solutions for this instance, the Rydberg solver predominantly finds three MIS solutions with probabilities of $\sim$61% [shown in Fig. 15(a)], $\sim$37%, and $\sim$1%, whereas an unbiased MIS sampler should generate these with uniform probabilities of $1/8 = 12.5\%$. It is possible, however, to potentially generate additional MIS solutions through simple postprocessing routines, taking the solutions generated by the Rydberg solver as initial seed for local search heuristics. Specifically, additional MIS solutions may found by swapping nodes in the set with available neighbors. For example, starting from the MIS solution $\{1, 2, 4, 9\}$ shown in Fig. 15(a) it is easy to identify the swap candidates $1 \leftrightarrow 6$, $4 \leftrightarrow 7$, and $5 \leftrightarrow 9$, generating in total $\binom{3}{1} + \binom{3}{2} + \binom{3}{3} = 7$ potential MIS candidates that can be accepted if feasible or dismissed otherwise. Similar logic can be applied in an effort to achieve feasibility on the original input graph for solutions found on the companion relaxed graph, as is the case for the solution shown in Fig. 15(a) when mapped to the original graph shown in Fig. 1(b) with an additional edge $(1, 9)$. (ii) *Full embedded graph*: Such postprocessing routines can be avoided via an *exact* embedding, albeit at the expense of an increased overhead, as exemplified in our second experiment; see Fig. 15(b). Here we solve for the MIS on the embedded graph, using an analog Hamiltonian simulation (AHS) program for 17 qubits representing nine logical nodes and 8 additional ancillas. We find that the Rydberg solver favors the MIS solution $\{2, 6, 7, 9\}$ shown in Fig. 15(b) with a probability of $p \approx 25\%$, with the ancilla quantum wire copying the selection of node 9 across every other node along the wire as desired.

Finally, we have run our AHS program on real quantum hardware provided by QuEra available on Amazon Braket. Given the limited laser power budget for driving the ground-Rydberg transition [52], here we used slightly different parameters, with reduced Rabi frequency $\Omega_{\max}/2\pi = 2.5$ MHz, while adjusting the lattice spacing to $a = 4.8$ μm

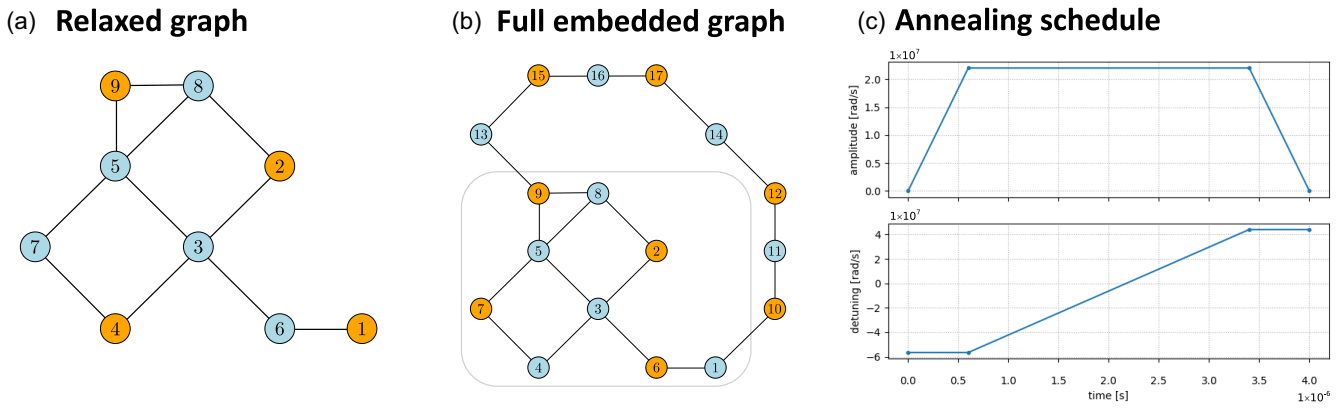


FIG. 15. MIS solutions obtained for the largest component of the Cora core graph via analog Hamiltonian simulation (AHS) on Amazon Braket. (a) Relaxed graph: For the overhead-free, relaxed graph, we predominantly find the MIS solution $\{1, 2, 4, 9\}$ with a probability of $p \approx 61\%$. This solution is infeasible for the original problem graph, because the edge $(1, 9)$ is missing in the relaxed graph but is present in the original graph. This issue can be easily fixed via local search (e.g., by swapping selection of node 1 with selection of node 6 and/or swapping node 9 for node 5). (b) Full embedded graph: For the full embedded graph involving 17 qubits, we predominantly find the feasible MIS solution $\{2, 6, 7, 9\}$ (with probability of $p \approx 25\%$), with the 8-qubit quantum wire (outside of the gray box) effectively suppressing the simultaneous selection of nodes 1 and 9. (c) Annealing schedule: All results were obtained using a piecewise linear schedule and 1000 shots. Numerical parameters: Maximum Rabi frequency $\Omega_{\max}/2\pi = 3.5$ MHz, minimal detuning $\Delta_{\min}/2\pi = -9$ MHz, maximum detuning $\Delta_{\max}/2\pi = 7$ MHz, laser phase $\phi = 0$, total annealing time $\tau = 4$ μs, and lattice spacing $a = 4.5$ μm.

so as to keep $r = R_b/a \sim 1.74$, comparable to $r \sim 1.7$ as used in Ref. [22]. All other parameters are listed in the caption of Fig. 15. The most likely outcome is found to be identical to our result shown in Fig. 15(b) based on noise-free simulations, with a reduced success probability of $p \approx 9.80\%$, a drop that we attribute to noise present on the quantum hardware.

B. Reduction experiments

We now complement the reduction results presented above with additional experiments on both real-world networks and synthetic (random) graphs. We observe that substantial reduction can typically be achieved up to some critical average degree, with structured graphs being more prone to reduction than structure-less, random graphs.

1. Real-world networks

We have applied our reduction logic to 10 well-known benchmark networks, involving small social networks with $\sim$10 nodes up to relatively large citation networks with $\sim 10^4$ nodes (including the Cora citation network featured above as our guiding example). Specifically, we consider the following publicly available benchmark instances. The Florentine graph describes a social network of families in 15th-century Florence related by marriage. The Zachary graph refers to another social network of a university karate club. The Dolphins graph represents a social network of frequent associations between 62 dolphins in a community living off New Zealand. The Les Miserables graph is a coappearance network of characters in the novel Les Miserables. The Jazz graph is a social collaboration network of jazz musicians. The C. Elegans graph is a metabolic network of the nematode (roundworm) *Caenorhabditis elegans*. The Email graph describes a university email network. The Cora and Citeseer graphs describe networks of computer science publications (with nodes representing publications and edges referring to citations), and the PubMed graph is another citation network based on articles related to diabetes from the PubMed database. All graphs can be sourced directly from the NetworkX library or downloaded from the network repository website [78].

Our numerical results are summarized in Table I. Across our set of 10 network instances, we consistently observe significant reduction, with $0.58 \leqslant \xi \leqslant 1$, an average reduction of $\sim$87%, and a relatively strong Pearson correlation coefficient of ~ -0.73 between the reduction factor ξ and the average degree of the input graph. These results suggest that graphs with a larger average degree may be more resilient to clique-based reduction. In particular, we find that the largest graph considered (PubMed) can be reduced by $\sim$99.9% from $\sim 2\times 10^4$ nodes to a core with 16 nodes in subsecond run-time on a laptop. For some graphs (such as Cora, Citeseer, and PubMed) the core is found to decompose into several, smaller components which can then be solved downstream individually in parallel. In line with our experiments for random UJ graphs, all reduction experiments completed on subsecond timescales on a laptop, with the run-time following an approximately linear run-time scaling $\sim n$; see Appendix A 2 for more details.

2. Synthetic graphs

We have performed additional, systematic benchmark experiments across three families of synthetic (random) graphs, including random geometric (RG) graphs, Erdős-Rényi (ER) graphs, and Barabási-Albert (BA) graphs. Here we focus on our results for Rydberg-native RG graphs. Qualitatively identical results have been obtained for ER and BA graphs, and quantitative differences across these families of random graphs will be discussed below.

RG graphs belong to the larger class of spatial networks, as found in infrastructure networks in transportation and telecommunication systems, where the probability of a connection between two nodes is a decreasing function

TABLE I. Numerical reduction results for real-world graphs. The first block describes the input graph $\mathcal{G}$ in terms of number of nodes, number of edges, and average degree. The second block details the kernel of the graph $\mathcal{K}$ after clique-based reduction in terms of nodes and edges (post reduction), the number of components, and the size of the largest component. The reduction result is summarized in terms of the reduction factor ξ, and the run-time of the reduction logic (on a laptop). Note that nodes with self loops have been removed. Further details are provided in the main text.

	Input graph $\mathcal{G}$			Kernel $\mathcal{K}$ of input graph $\mathcal{G}$				Run-time	Reduction
Network	Nodes	Edges	Average degree	Nodes	Edges	Components	Largest component	[ms]	ξ
Florentine	15	20	2.67	0	0			0.38	100%
Zachary Karate	34	78	4.59	4	4	1	4	0.97	88.2%
Dolphins	62	159	5.13	20	30	2	12	1.51	67.7%
Les Miserables	77	254	6.60	0	0			2.21	100%
Jazz	198	2742	27.70	83	580	1	83	17.74	58.1%
C. Elegans	438	1519	6.94	19	26	1	19	14.32	95.7%
Email	1133	5451	9.62	315	818	1	315	44.05	72.2%
Cora	2708	5278	3.90	79	94	16	9	116.93	97.1%
Citeseer	3264	4536	2.78	217	341	26	83	67.30	93.4%
PubMed	19714	44281	4.49	16	23	2	11	665.06	99.9%

of the distance between those nodes [79]. In particular, d-dimensional RG graphs are graphs where each node is assigned a random coordinate in the unit box $[0, 1]^d$, and only nodes within the connection distance R are connected by an edge. Here we focus on two-dimensional RG graphs as native to Rydberg atom arrays with arbitrary positioning in two space dimensions and $R = R_b$.

Our reduction results for RG graphs in two dimensions (with up to 10^4 nodes) are shown in Fig. 16. By tuning the parameter R we can effectively tune the number of edges M and, as such, the average degree $\bar{d} = 2M/N$. We observe large reduction up to a certain (critical) average degree $\bar{d}^* \sim 6$, with a clear transition from macroscopic reduction to minimal reduction in the asymptotic limit of large system sizes. Moreover, this reduction is accompanied by a notable run-time peak around the critical degree, akin to phase transitions in uniform-random 3-SAT instances around the critical clause-to-variable ratio [75]. However, additional studies would have to be performed to determine if this is an actual phase transition or an easy-hard-easy crossover. Consistent with our results for random UJ instances and real-world networks, reduction run times are subsecond for sparse instances as large as $\sim 10^4$ nodes, with a scaling that is approximately linear in the number of nodes.

Similarly to our findings for random UJ instances, our results suggest that Rydberg-native RG instances with an average degree below the critical degree $\bar{d}^*$ are easy and, as such, unlikely to display potential future quantum speedups. Conversely, RG instances with sufficient connectivity are immune to clique-based reduction and have a large kernel, making them interesting candidates for future quantum experiments. Beyond Rydberg-native UJ and RG graphs, we have observed a similar critical degree of $\bar{d}^* \sim 6$ for scale-free BA graphs that feature hubs and power-law degree distribution [80], approximately similar to many real-world networks. For sufficiently sparse graphs, such structured networks are likely prone to reduction, e.g., via simple removal of simple dangling bonds (with degree one). Conversely, given their

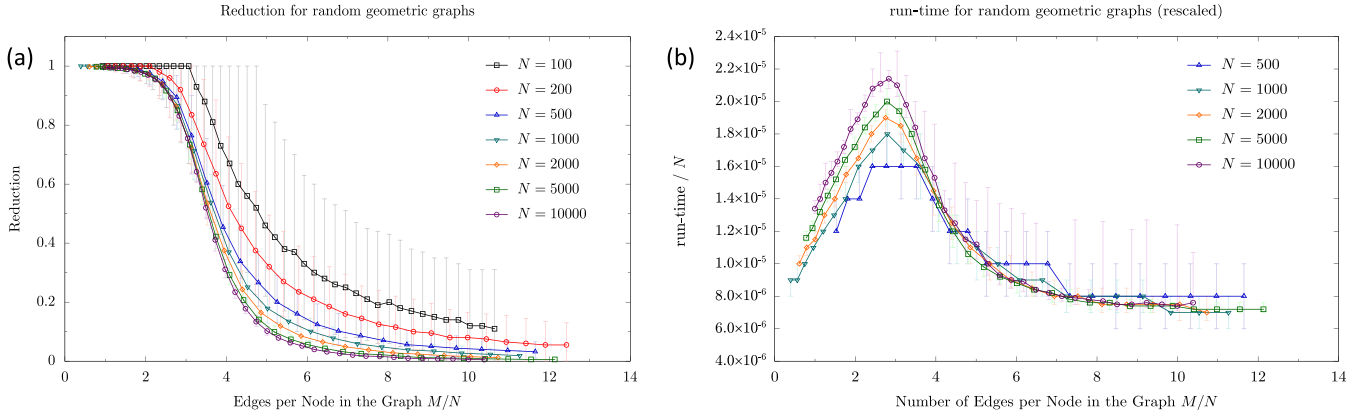


FIG. 16. Reduction applied to random geometric graphs in two space dimensions. (a) Reduction factor ξ as a function of the number of edges M per node, as can be tuned via the radius R. (b) Reduction run-time (in seconds, scaled in terms of the number of nodes) as a function of the number of edges per node. We observe a peak in run-time at the critical average degree of $\bar{d}^* = 2M/N \sim 6$. Substantial reduction is possible for sparse instances with average degree smaller than $\bar{d}^*$, while denser instances with average degree larger than $\bar{d}^*$ are immune to clique-based reduction. The average run-time is well approximated by $\bar{T}_{\text{red}}/n \approx \gamma(\bar{d})$, in agreement with Eq. (4).

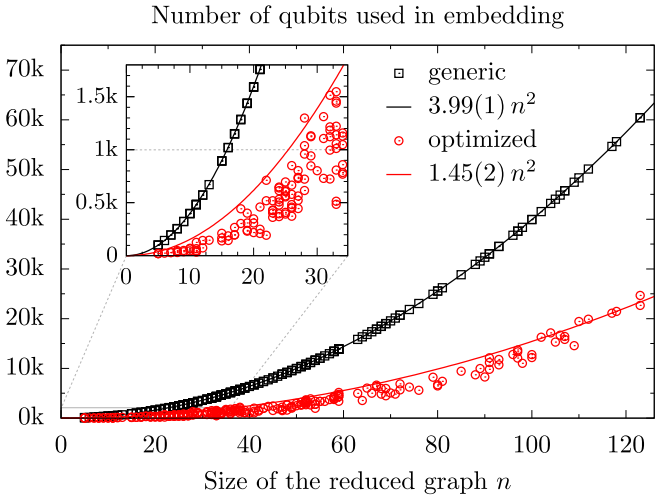


FIG. 17. Number of qubits required for hardware-native embedding for the generic scheme (black squares) and optimized approach (red circles). Each point indicates the overhead for a *residual graph kernel* with n nodes, as obtained by applying reduction to random Erdős–Rényi graphs with 1.7 edges per node. The fitted functions indicate a reduction in the prefactor from 3.99 to 1.45. Inset: Number of qubits required for (small) logical graphs $n < 35$, with the horizontal line indicating hyptothetical quantum hardware with 10^3 physical qubits.

inherent lack of structure, ER graphs with a Poissonian degree distribution appear to be the most robust against reduction with a smoother transition towards irreducibility appearing for average degree values as low as ∼3 to 4; see Appendix A 3 for more details.

C. Embedding experiments

In our proposed pipeline, embedding techniques are used to make the residual kernel graph compatible with Rydberg quantum hardware (after reduction has been applied), ultimately allowing us to solve for the MIS with quantum algorithms. Here, using our top-down embedding scheme, we perform numerical experiments to assess the qubit overhead for a set of graphs involving 250 *reduced* graphs. These are generated by applying reduction to large ER graphs with a density that corresponds to 1.7 edges per node. The resulting residual (kernel) graphs are smaller than the original ER graphs and used as a representation of typical irreducible cores for sparse graphs. Our results are shown in Fig. 17, demonstrating that the required number of qubits can be reduced from $\sim 3.99(1)n^2$ for the generic embedding strategy per Ref. [41] down to $\sim 1.45(2)n^2$ when adopting our optimized top-down embedding scheme. As shown in Appendix A 4 the run-timescales approximately as $\sim n^3$. Accordingly, the overhead remains quadratic in the number of nodes n, yet the smaller prefactor allows us to embed graphs with ∼1.66 as many nodes. For example, for the set of graphs studied here and hypothetical hardware with 1000 physical qubits, the optimized top-dowm method is able to embed all instances with up to $n = 27$ nodes (and several larger instances), while the full generic scheme reaches this limit already at $n = 15$. While the specific prefactors are likely dependent on the class of graphs studied, this indicates that optimized graph-specific top-down embedding can be leveraged efficiently for larger sparse instances.

VII. CONCLUSION AND OUTLOOK

In summary, we have proposed and analyzed a comprehensive suite of tools to support the implementation of quantum optimization algorithms with near-term Rydberg atom arrays, for generic (non-native) MIS instances, potentially at problem scales previously inaccessible. Our toolbox features three main modules.

The reducer module provides an efficient tool to shrink the size of generic MIS problem instances while preserving optimality, by iteratively removing subgraphs via selection of exposed nodes that are provably part of some optimal MIS solution. Optimal solutions for the reduced (core) instance can then be easily expanded to an optimal solution for the original instance by unrolling previously applied reductions. We have shown that this simple reduction technique can be a powerful asset in the ongoing search for potential quantum speedups, because it allows us to identify hard problem instances (where quantum speedups are more likely to be found) as those instances with a large kernel. In particular, for Rydberg-native MIS instances we have observed signatures of an easy-hard-easy transition, showing that problem hardness could be tuned as a function of atomic density.

The checker module quickly assesses the hardware "friendliness" of a given input graph via simple checks on graph properties, such as the maximum degree and the number of triangles. We have used this tool to generate a hardware compatibility diagram that can inform downstream embedding logic.

Last, we have described in detail two complementary embedding strategies for Rydberg atom arrays, including a (bottom-up) approach that *learns* approximate embeddings with minimal resource requirements, as well as a (top-down) scheme that constructively generates exact embeddings with MIS optimality guarantees, while minimizing the required qubit overhead. Using this module, we have systematically analyzed the typical overhead for a class of generic input instances and quantified problem scales accessible with near-term devices.

In aggregate, these three modules can be integrated to set up an end-to-end pipeline for solving generic (i.e., non-native) instances of the MIS problem on quantum hardware based on Rydberg atom arrays. We have exemplified this pipeline with a demo run on quantum hardware provided by QuEra on Amazon Braket for the Cora citation network with ∼2700 nodes, after optimality-preserving reduction to a kernel with a largest component of just nine nodes.

Finally, we now highlight possible extensions of research going beyond our present work. First, in an effort to streamline ongoing work towards potential quantum advantage, it would be interesting to use state-of-the-art reduction procedures (involving, for example, generalized dominance reduction instead of isolated clique removal [58]) to establish a standard testbed featuring provably hard MIS instances as a go-to resource for benchmarking new quantum algorithms. Second, one could study extended embedding schemes in

which atomic positions are not necessarily restricted to sites on an underlying square lattice, but are allowed to take on arbitrary (continuous) coordinates. More broadly, our work prompts deeper research into novel embedding schemes that could further reduce the qubit overhead. Potential avenues include the following: (i) The ability to trap Rydberg atoms in three dimensions [81,82] should allow us to reduce the number of crossing gadgets, thus reducing the qubit overhead. (ii) Further, the ability to coherently shuttle Rydberg atoms [43,44] would allow us to tune the connectivity in real time, possibly opening up avenues for entirely new dynamic embedding schemes. (iii) Moreover, one could explore regimes with larger (tunable) Rydberg radii, beyond the UJ connectivity regime, thus allowing us to realize larger all-to-all connected cliques, and with local control one might even adjust the connectivity pattern on a local scale. (iv) One could also leverage recent, alternative parity-based embedding schemes as proposed in Refs. [83,84], that could be integrated with our reducer logic to form an alternative end-to-end compilation pipeline. Third, given the wide-ranging applications of the MIS problem, e.g., in network design [4], vehicle routing [5], or quantitative finance [7], it would be interesting to apply and systematically test the performance of our pipeline for such real-world problems. Finally, one could expand the set of accessible optimization problems through a series of extensions, including, for example, (i) the generalization to the MWIS based on reductions for the MWIS problem [49], with real-world applications in map-labeling, (ii) generalized reduction logic for other graph-based optimization problems (such as maximum cut or graph coloring) where similar subgraph removal techniques should apply, and (iii) the integration of state-of-the-art reduction logic with non-native hybrid quantum solvers as presented in Ref. [85]. In particular, inspired by the results presented in Figs. 11 and 12, it would be interesting to consider *quantum* versions of state-of-the-art classical MIS heuristics such as ReduMIS [59] in which exact reduction is applied repeatedly in tandem with a quantum algorithm (rather than a classical evolutionary heuristic) to select a subset of independent kernel vertices to include, thereby opening up the reduction space after removal of the selected vertices and their neighbors. Similarly to classical state-of-the-art MIS heuristics [59,60], the quantum computer would then be used as a coprocessor to inform the classical wrapper about vertices that are likely to be in large independent sets.

ACKNOWLEDGMENTS

We thank Victor Bocking, Alexander Buts, Peter Komar, Lou Romano, Peter Sceusa, and Rajagopal Ganesan for their support. We thank Darren Strash for useful comments on the manuscript.

This paper was prepared for informational purposes with contributions from the Global Technology Applied Research center of JPMorgan Chase & Co. This paper is not a product of the Research Department of JPMorgan Chase & Co. or its affiliates. Neither JPMorgan Chase & Co. nor any of its affiliates makes any explicit or implied representation or warranty and none of them accept any liability in connection with this paper, including, without limitation, with respect to the completeness, accuracy, or reliability of the information contained herein and the potential legal, compliance, tax, or accounting effects thereof. This document is not intended as investment research or investment advice, or as a recommendation, offer, or solicitation for the purchase or sale of any security, financial instrument, financial product or service, or to be used in any way for evaluating the merits of participating in any transaction.

DATA AVAILABILITY

The data that support the findings of this article are openly available [77].

APPENDIX: ADDITIONAL INFORMATION

Here we provide further details for selected aspects discussed in the main text.

1. Classical hardware

In this section, we specify the classical (laptop) hardware on which our classical experiments were run. We used a MacBook Pro (2021) with Apple M1 Pro chip and 32-GB memory. The M1 Pro chip has a 10-core CPU with eight performance cores and two efficiency cores. The high-performance cores are clocked at 3228 MHz, and the high-efficiency cores are clocked at 2064 MHz.

2. Reduction run-times

In this section, we provide results for the algorithmic run times of our clique-based reduction algorithm when applied to both random UJ instances and real-world networks. Our results are displayed in Figs. 18 and 19, respectively. For random UJ instances, we find that all reduction runs finish on subsecond timescales per instance on a laptop (taking on

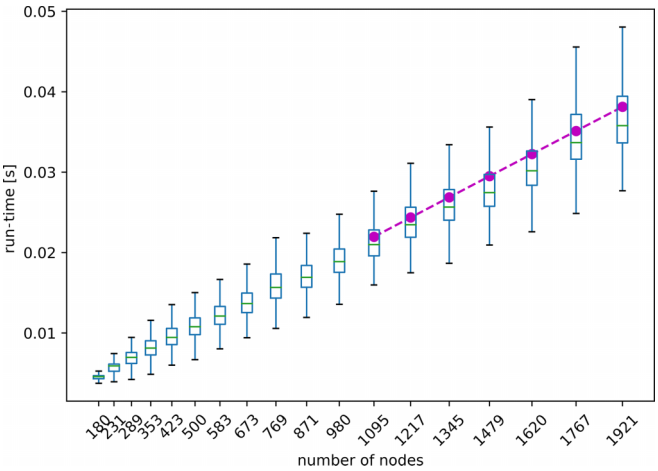


FIG. 18. Box plot for algorithmic run-time (in seconds) for our reduction logic as a function of the number of nodes for random UJ instances. Data and the definition of boxes and whiskers are identical to those shown in Fig. 13 for filling fraction $\varrho = 0.8$. Outliers are omitted for clarity. The best power-law fit $\sim n^{\alpha}$ for all data for larger instances with $n > 1000$ (including outliers) is shown as a dashed line (magenta), resulting in an approximately linear scaling with $\alpha \sim 0.981(5)$.

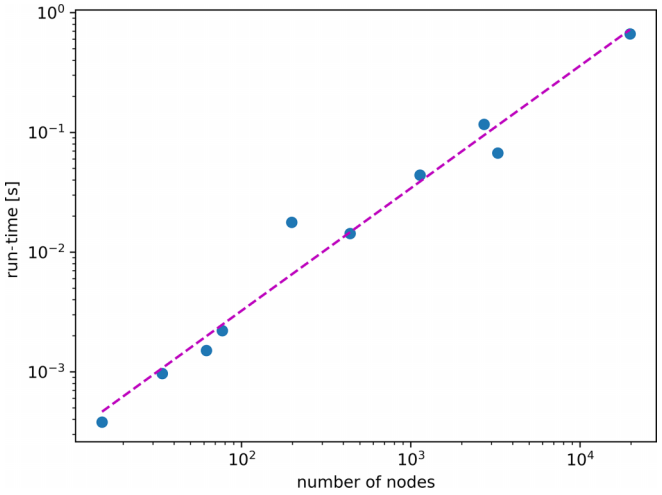


FIG. 19. Algorithmic run-time (in seconds) for our reduction logic as a function of the number of nodes for the 10 networks considered in Table I. The best power-law fit $\sim n^{\alpha}$ is shown as a dashed line (magenta), resulting in an approximately linear scaling with $\alpha \sim 1.02(6)$.

average $\sim$0.02 s for instances with $\sim$1000 nodes). In good agreement with Eq. (4) and in line with other numerical results discussed in Sec. VI, we also observe an approximately linear scaling of the run-time with $\sim n$, for large instances with $n > 1000$. We observe similar results for our set of 10 real-world networks as described in Table I. Again, we find subsecond run times for instances as large as $n \sim 2\times10^4$ (for the Pubmed graph), with the data fitting best to an approximate linear scaling $\sim n^{1.02(6)}$, in good agreement with Eq. (4).

3. Reduction of synthetic (random) graphs

In this section, we provide additional results on the clique-based reduction of synthetic (random) graphs similar to the results shown for RG graphs in the main text. Our results for random BA and ER graphs are shown in Fig. 20.

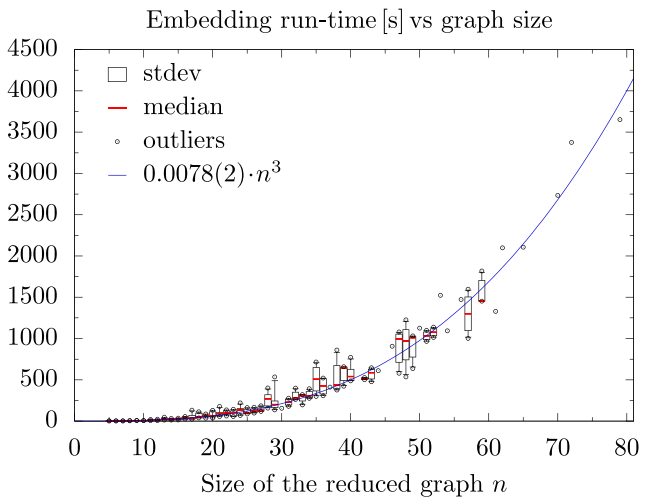


FIG. 21. Run-time (in seconds) for our top-down embedder as a function of the number of nodes n for random ER graphs as studied in Fig. 17. Our fits support a run-time scaling $\sim n^3$.

4. Embedding run-times

In this section, we provide results for the algorithmic run-time of our top-down embedder when applied to (reduced) random ER graphs as studied in Fig. 17. Our results are shown in Fig. 21, as a function of the number of nodes n. We observe that our data fit well to a run-time scaling $\sim n^3$, as expected given that the embedded graph involves $\sim n^2$ physical qubits and our simulated annealing solver is set to scale the number of sweeps linearly as $\sim n$. These experiments were performed on an AMD Ryzen 9 5950X @ 4.90 GHz.

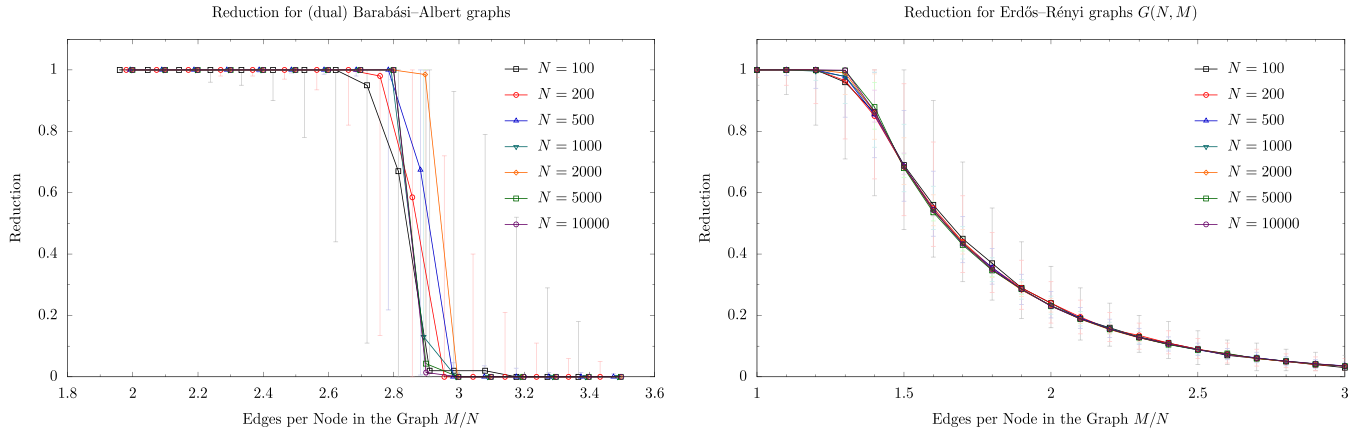


FIG. 20. Reduction factor ξ as a function of the number of edges M per node for random BA (left) and ER (right) graphs with up to 10^4 nodes. For scale-free BA graphs, we observe a sharp transition at a critical average degree of $\bar{d}^* \sim 6$ similar to our results for random RG graphs. For structureless ER graphs, we observe a smoother transition at relatively small average degree values.

[1] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity* (Courier Corporation, North Chelmsford, MA, 1998).

[2] B. Korte and J. Vygen, *Combinatorial Optimization* (Springer, New York, 2012), Vol. 2.

[3] J. Wurtz, P. L. S. Lopes, N. Gemelke, A. Keesling, and S. Wang, Industry applications of neutral-atom quantum computing solving Independent Set problems, arXiv:2205.08500.

[4] W. K. Hale, Frequency assignment: Theory and applications, Proc. IEEE **68**, 1497 (1980).

[5] Y. Dong, A. V. Goldberg, A. Noe, N. Parotsidis, M. G. C. Resende, and Q. Spaen, A metaheuristic algorithm for large maximum weight independent set problems, Networks **85**, 91 (2025).

[6] V. Boginski, S. Butenko, and P. M. Pardalos, Statistical analysis of financial networks, Comput. Stat. Data Anal. **48**, 431 (2005).

[7] A. Kalra, F. Qureshi, and M. Tisi, Portfolio asset identification using graph algorithms on a quantum annealer, SSRN (2018).

[8] D. Herman, C. Googin, X. Liu, Y. Sun, A. Galda, I. Safro, M. Pistoia, and Y. Alexeev, Quantum computing for finance, Nat. Rev. Phys. **5**, 450 (2023).

[9] A. Abbas, A. Ambainis, B. Augustino, A. Bärtschi, H. Buhrman, C. Coffrin, G. Cortiana, V. Dunjko, D. J. Egger, B. G. Elmegreen *et al.*, Challenges and opportunities in quantum optimization, Nat. Rev. Phys. **6**, 718 (2024).

[10] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse ising model, Phys. Rev. E **58**, 5355 (1998).

[11] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Quantum computation by adiabatic evolution, arXiv:quant-ph/0001106.

[12] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem, Science **292**, 472 (2001).

[13] A. Das and B. K. Chakrabarti, *Colloquium*: Quantum annealing and analog quantum computation, Rev. Mod. Phys. **80**, 1061 (2008).

[14] P. Hauke, H. G. Katzgraber, W. Lechner, H. Nishimori, and W. Oliver, Perspectives of quantum annealing: Methods and implementations, Rep. Prog. Phys. **83**, 054401 (2020).

[15] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[16] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices, Phys. Rev. X **10**, 021067 (2020).

[17] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, 5 (2014).

[18] F. Glover, G. Kochenberger, and Y. Du, Quantum bridge analytics I: A tutorial on formulating and using QUBO models, 4OR **17**, 335 (2019).

[19] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, Quantum optimization for maximum independent set using Rydberg atom arrays, arXiv:1808.10816.

[20] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, Computational complexity of the Rydberg Blockade in two dimensions, arXiv:1809.04954.

[21] M. F. Serret, B. Marchand, and T. Ayral, Solving optimization problems with Rydberg analog quantum computers: Realistic requirements for quantum advantage using noisy simulation and classical benchmarks, Phys. Rev. A **102**, 052617 (2020).

[22] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar *et al.*, Quantum optimization of maximum independent set using Rydberg atom arrays, Science **376**, 1209 (2022).

[23] M. Cain, S. Chattopadhyay, J.-G. Liu, R. Samajdar, H. Pichler, and M. D. Lukin, Quantum speedup for combinatorial optimization with flat energy landscapes, arXiv:2306.13123.

[24] B. F. Schiffer, D. S. Wild, N. Maskara, M. Cain, M. D. Lukin, and R. Samajdar, Circumventing superexponential runtimes for hard instances of quantum adiabatic optimization, Phys. Rev. Res. **6**, 013271 (2024).

[25] J. R. Finžgar, M. J. A. Schuetz, J. K. Brubaker, H. Nishimori, and H. G. Katzgraber, Designing quantum annealing schedules using bayesian optimization, Phys. Rev. Res. **6**, 023063 (2024).

[26] J. R. Finžgar, A. Kerschbaumer, M. J. Schuetz, C. B. Mendl, and H. G. Katzgraber, Quantum-informed recursive optimization algorithms, PRX Quantum **5**, 020327 (2024).

[27] K. Kim, M. Kim, J. Park, A. Byun, and J. Ahn, Quantum computing dataset of maximum independent set problem on king lattice of over hundred Rydberg atoms, Sci. Data **11**, 111 (2024).

[28] R. S. Andrist, M. J. A. Schuetz, P. Minssen, R. Yalovetzky, S. Chakrabarti, D. Herman, N. Kumar, G. Salton, R. Shaydulin, Y. Sun *et al.*, Hardness of the maximum-independent-set problem on unit-disk graphs and prospects for quantum speedups, Phys. Rev. Res. **5**, 043277 (2023).

[29] M. D. Lukin, M. Fleischhauer, R. Cote, L. M. Duan, D. Jaksch, J. I. Cirac, and P. Zoller, Dipole Blockade and quantum information processing in mesoscopic atomic ensembles, Phys. Rev. Lett. **87**, 037901 (2001).

[30] H. Levine, A. Keesling, G. Semeghini, A. Omran, T. T. Wang, S. Ebadi, H. Bernien, M. Greiner, V. Vuletić, H. Pichler *et al.*, Parallel implementation of high-fidelity multiqubit gates with neutral atoms, Phys. Rev. Lett. **123**, 170503 (2019).

[31] M. Saffman, T. G. Walker, and K. Mølmer, Quantum information with Rydberg atoms, Rev. Mod. Phys. **82**, 2313 (2010).

[32] B. N. Clark, C. J. Colbourn, and D. S. Johnson, Unit disk graphs, Discr. Math. **86**, 165 (1990).

[33] M. S. Könz, W. Lechner, H. G. Katzgraber, and M. Troyer, Embedding overhead scaling of optimization problems in quantum annealing, PRX Quantum **2**, 040322 (2021).

[34] P. I. Bunyk, E. M. Hoskinson, M. W. Johnson, E. Tolkacheva, F. Altomare, A. J. Berkley, R. Harris, J. P. Hilton, T. Lanting, A. J. Przybysz *et al.*, Architectural considerations in the design of a superconducting quantum annealing processor, IEEE Trans. Appl. Supercond. **24**, 1 (2014).

[35] W. Vinci, T. Albash, G. Paz-Silva, I. Hen, and D. A. Lidar, Quantum annealing correction with minor embedding, Phys. Rev. A **92**, 042310 (2015).

[36] Y. Sugie, Y. Yoshida, N. Mertig, T. Takemoto, H. Teramoto, A. Nakamura, I. Takigawa, S.-I. Minato, M. Yamaoka, and T. Komatsuzaki, Minor-embedding heuristics for large-scale annealing processors with sparse hardware graphs of up to 102,400 nodes, Soft Comput. **25**, 1731 (2021).

[37] A. Byun, M. Kim, and J. Ahn, Finding the maximum independent sets of platonic graphs using Rydberg atoms, PRX Quantum **3**, 030305 (2022).

[38] A. Byun, J. Jung, K. Kim, M. Kim, S. Jeong, H. Jeong, and J. Ahn, Rydberg-atom graphs for quadratic unconstrained binary optimization problems, Adv. Quantum Technol. **7**, 2300398 (2024).

[39] M. Kim, K. Kim, J. Hwang, E.-G. Moon, and J. Ahn, Rydberg quantum wires for maximum independent set problems, Nat. Phys. **18**, 755 (2022).

[40] C. Dlaska, K. Ender, G. B. Mbeng, A. Kruckenhauser, W. Lechner, and R. van Bijnen, Quantum optimization via four-body Rydberg gates, Phys. Rev. Lett. **128**, 120503 (2022).

[41] M.-T. Nguyen, J.-G. Liu, J. Wurtz, M. D. Lukin, S.-T. Wang, and H. Pichler, Quantum optimization with arbitrary connectivity using Rydberg atom arrays, PRX Quantum **4**, 010316 (2023).

[42] A. G. de Oliveira, E. Diamond-Hitchcock, D. M. Walker, M. T. Wells-Pestell, G. Pelegrí, C. J. Picken, G. P. A. Malcolm, A. J. Daley, J. Bass, and J. D. Pritchard, Demonstration of weighted graph optimization on a Rydberg-atom array using local light shifts, PRX Quantum **6**, 010301 (2025).

[43] D. Bluvstein, H. Levine, G. Semeghini, T. T. Wang, S. Ebadi, M. Kalinowski, A. Keesling, N. Maskara, H. Pichler, M. Greiner *et al.*, A quantum processor based on coherent transport of entangled atom arrays, Nature (London) **604**, 451 (2022).

[44] D. Bluvstein, S. J. Evered, A. A. Geim, S. H. Li, H. Zhou, T. Manovitz, S. Ebadi, M. Cain, M. Kalinowski, D. Hangleiter *et al.*, Logical quantum processor based on reconfigurable atom arrays, Nature (London) **626**, 58 (2024).

[45] S. Butenko, P. Pardalos, I. Sergienko, V. Shylo, and P. Stetsyuk, Finding maximum independent sets in graphs arising from coding theory, in *Proceedings of the ACM Symposium on Applied Computing (SAC'02)* (ACM Press, New York, 2002), pp. 542–546.

[46] S. Butenko and S. Trukhanov, Using critical sets to solve the maximum independent set problem, Oper. Res. Lett. **35**, 519 (2007).

[47] D. Strash, in *Computing and Combinatorics*, edited by T. N. Dinh and M. T. Thai (Springer International, Cham, 2016), pp. 345–356.

[48] D. Hespe, C. Schulz, and D. Strash, Scalable kernelization for maximum independent sets, ACM J. Exp. Algor. **24**, 1 (2019).

[49] S. Lamm, C. Schulz, D. Strash, R. Williger, and H. Zhang, Exactly solving the maximum weight independent set problem on large real-world graphs, in *2019 Proceedings of the Meeting on Algorithm Engineering and Experiments (ALENEX)* (Society for Industrial and Applied Mathematics, 2019), pp. 144–158.

[50] A. K. McCallum, K. Nigam, J. Rennie, and K. Seymore, Automating the construction of internet portals with machine learning, Inf. Retriev. **3**, 127 (2000).

[51] G. Namata, B. London, L. Getoor, and B. Huang, in *Proceedings of the 2012 International Workshop on Mining and Learning with Graphs* (Edinburgh, UK, 2012), p. 8.

[52] J. Wurtz, A. Bylinskii, B. Braverman, J. Amato-Grill, S. H. Cantu, F. Huber, A. Lukin, F. Liu, P. Weinberg, J. Long *et al.*, Aquila: QuEra's 256-qubit neutral-atom quantum computer, arXiv:2306.11727.

[53] F. Gyger, M. Ammenwerth, R. Tao, H. Timme, S. Snigirev, I. Bloch, and J. Zeiher, Continuous operation of large-scale atom arrays in optical lattices, Phys. Rev. Res. **6**, 033104 (2024).

[54] H. J. Manetsch, G. Nomura, E. Bataille, K. H. Leung, X. Lv, and M. Endres, A tweezer array with 6100 highly coherent atomic qubits, arXiv:2403.12021.

[55] C. S. Adams, J. D. Pritchard, and J. P. Shaffer, Rydberg atom quantum technologies, J. Phys. B: At. Mol. Opt. Phys. **53**, 012002 (2019).

[56] L. Henriet, L. Beguin, A. Signoles, T. Lahaye, A. Browaeys, G.-O. Reymond, and C. Jurczak, Quantum computing with neutral atoms, Quantum **4**, 327 (2020).

[57] S. Butenko, P. Pardalos, I. Sergienko, V. Shylo, and P. Stetsyuk, *Estimating the Size of Correcting Codes Using Extremal Graph Problems* (Springer, New York, 2009), pp. 227–243.

[58] L. Chang, W. Li, and W. Zhang, Computing a near-maximum independent set in linear time by reducing-peeling, in *Proceedings of the ACM International Conference on Management of Data (SIGMOD'17)* (Association for Computing Machinery, New York, 2017), pp. 1181–1196.

[59] S. Lamm, P. Sanders, C. Schulz, D. Strash, and R. F. Werneck, Finding near-optimal independent sets at scale, J. Heurist. **23**, 207 (2017).

[60] E. Großmann, S. Lamm, C. Schulz, and D. Strash, Finding near-optimal weight independent sets at scale, in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'23)* (ACM Press, New York, 2023), pp. 293–302.

[61] Y. Liu, T. Safavi, A. Dighe, and D. Koutra, Graph summarization methods and applications: A survey, ACM Comput. Surv. **51**, 1 (2018).

[62] M. Hashemi, S. Gong, J. Ni, W. Fan, B. A. Prakash, and W. Jin, A comprehensive survey on graph reduction: Sparsification, coarsening, and condensation, in *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence* (International Joint Conferences on Artificial Intelligence Organization, 2024), pp. 8058–8066.

[63] M. Lewis and F. Glover, Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis, Networks **70**, 79 (2017).

[64] F. Glover, M. Lewis, and G. Kochenberger, Logical and inequality implications for reducing the size and difficulty of quadratic unconstrained binary optimization problems, Eur. J. Oper. Res. **265**, 829 (2018).

[65] A. Narimani, S. S. C. Rezaei, and A. Zaribafiyan, Combinatorial optimization by decomposition on hybrid CPU–non-CPU solver architectures, arXiv:1708.03439.

[66] E. Pelofske, G. Hahn, and H. Djidjev, Decomposition algorithms for solving np-hard problems on a quantum annealer, J. Signal Proc. Syst. **93**, 405 (2020).

[67] M. E. J. Newman, *Networks: An Introduction* (Oxford University Press, Oxford, UK, 2010).

[68] A. A. Chaves, M. G. C. Resende, M. J. A. Schuetz, J. K. Brubaker, H. G. Katzgraber, E. F. de Arruda, and R. M. A. Silva, A random-key optimizer for combinatorial optimization, arXiv:2411.04293.

[69] M. A. Londe, L. S. Pessoa, C. E. Andrade, and M. G. Resende, Biased random-key genetic algorithms: A review, Eur. J. Oper. Res. **321**, 1 (2025).

[70] M. A. Londe, L. S. Pessoa, C. E. Andrade, and M. G. C. Resende, Early years of biased random-key genetic algorithms: A systematic review, J. Glob. Optim. (2024).

[71] M. J. Schuetz, J. K. Brubaker, H. Montagu, Y. van Dijk, J. Klepsch, P. Ross, A. Luckow, M. G. Resende, and H. G.

Katzgraber, Optimization of robot-trajectory planning with nature-inspired and hybrid quantum algorithms, Phys. Rev. Appl. **18**, 054045 (2022).

[72] A. A. Chaves, M. G. C. Resende, and R. M. A. Silva, A continuous-GRASP random-key optimizer, in *Metaheuristics: 15th International Conference (MIC 2024)* (Springer-Verlag, Berlin, 2024), pp. 15–20.

[73] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by simulated annealing, Science **220**, 671 (1983).

[74] S. Perseguers, Hardness-dependent quantum adiabatic schedules for the maximum-independent-set problem, Phys. Rev. Appl. **23**, 064023 (2025).

[75] K. Leyton-Brown, H. H. Hoos, F. Hutter, and L. Xu, Understanding the empirical hardness of np-complete problems, Commun. ACM **57**, 98 (2014).

[76] Y. Song, M. Kim, H. Hwang, W. Lee, and J. Ahn, Quantum simulation of cayley-tree Ising Hamiltonians with three-dimensional rydberg atoms, Phys. Rev. Res. **3**, 013286 (2021).

[77] Y. Matsuda, H. Nishimori, and H. G. Katzgraber, Quantum annealing for problems with ground-state degeneracy, J. Phys.: Conf. Ser. **143**, 012003 (2009).

[78] R. A. Rossi and N. K. Ahmed, The network data repository with interactive graph analytics and visualization, in *AAAI* (2015), https://networkrepository.com.

[79] M. Barthelemy, Spatial networks, Phys. Rep. **499**, 1 (2011).

[80] A.-L. Barabási and R. Albert, Emergence of scaling in random networks, Science **286**, 509 (1999)

[81] D. Barredo, V. Lienhard, S. de Léséleuc, T. Lahaye, and A. Browaeys, Synthetic three-dimensional atomic structures assembled atom by atom, Nature (London) **561**, 79 (2018).

[82] D. Barredo, V. Lienhard, P. Scholl, S. de Léséleuc, T. Boulier, A. Browaeys, and T. Lahaye, Three-dimensional trapping of individual Rydberg atoms in ponderomotive bottle beam traps, Phys. Rev. Lett. **124**, 023201 (2020).

[83] M. Lanthaler, C. Dlaska, K. Ender, and W. Lechner, Rydberg-blockade-based parity quantum optimization, Phys. Rev. Lett. **130**, 220601 (2023).

[84] M. Lanthaler, K. Ender, C. Dlaska, and W. Lechner, Quantum optimization with globally driven neutral atom arrays, arXiv:2410.03902.

[85] J. Wurtz, S. H. Sack, and S.-T. Wang, Solving non-native combinatorial optimization problems using hybrid quantum-classical algorithms, IEEE Trans. Quantum Eng. **5**, 1 (2024).