COMPUTER-ASSISTED GADGET DESIGN AND PROBLEM REDUCTION OF UNWEIGHTED MAXIMUM INDEPENDENT SET

JIN-GUO LIU*, JONATHAN WURTZ†, MINH-THI NGUYEN‡, MIKHAIL D. LUKIN§, HANNES PICHLER¶, AND SHENG-TAO WANG†

Abstract. Reducing constraint satisfaction problems defined on graphs with general connectivity to geometrically constrained energy models is a crucial step towards utilizing physics-inspired computing devices for solving computationally hard problems. Inspired by recent progress on variational quantum algorithms [Ebadi et al., Science, 376, 1209 (2022)] for solving the maximum independent set problem using two-dimensional, programmable quantum processors, Nguyen et al. [PRX Quantum 4, 010316 (2023)] proposed a framework for reducing a weighted maximum independent set problem on a general graph to that on a unit disk graph. However, the same technique is not applicable to unweighted reduction, which is more readily implementable on such programmable quantum devices. Here, using computer-assisted gadget design, we show how to reduce the unweighted maximum independent set problem on a general graph $G = (V, E)$ to that on a King's subgraph. The transformed graph has a size $O(|V| \times \mathrm{pw}(G))$, where $\mathrm{pw}(G)$ is the pathwidth of G. This reduction scheme is optimal up to a constant factor, assuming the exponential time hypothesis is true.

1. Introduction. Problem reduction plays a central role in the field of computational complexity. It is not only a strategy for proving the computational hardness of a problem set [20] but is also practically useful [10]. One application of the reduction schemes is the re-formulation of problems of interest in a way that matches the native hardware requirements and constraints of some specific hardware. This work is highly inspired by recent experimental progress of solving computationally hard problems with neutral-atom quantum processors [7, 14], which allow for a native implementation of variational quantum algorithms for solving the NP-complete [6] maximum independent set (MIS) problem on unit disk (UD) graphs [23, 22, 7, 2]. In order to expand the applicability of these neutral-atom quantum processors to MIS problems on more general graphs while retaining reasonable vertex overhead, Ref. [21] proposed a scheme to reduce the MIS problem on general weighted graphs to that on *weighted* UD graphs using at most quadratic vertex overhead. In practice, realizing quantum algorithms for the weighted version of the MIS problem requires additional experimental control, so finding similarly efficient reduction schemes from unweighted MIS problems on general graphs to *unweighted* MIS problems on UD graphs would be desirable from the hardware perspective [7, 26]. Moreover, the unweighted MIS problem is interesting in its own right since being a subclass of the weighted problem, it is one of the most popular NP-hard problems in the literature [13, 18, 12], and it plays an important role in coding theory [4, 8] and other industry relevant problems [27].

In this paper, we address this challenge and generalize the framework of Ref. [21], developing a reduction scheme from unweighted MIS on general graphs to unweighted MIS on UD graphs, which requires at most quadratic vertex overhead. To this end, we introduce a computer-assisted gadget design framework based on graph rewriting

*Hong Kong University of Science and Technology (Guangzhou), Guangzhou, China; Department of Physics, Harvard University, Cambridge, Massachusetts 02138, USA; QuEra Computing Inc., 1284 Soldiers Field Road, Boston, MA, 02135, USA (jinguoliu@hkust-gz.edu.cn).

†QuEra Computing Inc., 1284 Soldiers Field Road, Boston, MA 02135, USA

‡Department of Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA; QuEra Computing Inc., 1284 Soldiers Field Road, Boston, MA 02135, USA

§Department of Physics, Harvard University, Cambridge, Massachusetts 02138, USA

¶Institute for Theoretical Physics, University of Innsbruck, Innsbruck A-6020, Austria; Institute for Quantum Optics and Quantum Information, Austrian Academy of Sciences, Innsbruck A-6020, Austria

for the unweighted MIS reduction, in which the unweighted MIS gadgets play the role of essential building blocks. With a computer-assisted exhaustive search, we can explore the super-exponentially large graph space and find several optimal gadgets that we employ in the problem reduction.

The content of the paper is ordered as follows. Section 2 reviews general concepts of the MIS problem and introduces the basic notations we use throughout the paper. Section 3 introduces the graph-rewriting framework for the unweighted MIS reduction, including the principle of computer-assisted gadget design, followed by some specific gadgets found for the unweighted MIS problem reduction. Section 4 introduces a scheme to reduce a general MIS problem to that on a class of UD graphs and discusses the optimality of the reduction. Finally, Section 5 concludes the paper with discussions on some future research directions.

2. Background and notation. In this work, we are interested in the MIS problem. An instance of the (unweighted) MIS problem is specified by a graph $G = (V, E)$, consisting of a set of vertices V and a set of edges E. An independent set of G is a subset of V such that no pair of vertices in the subset is connected by an edge in E. An independent set with the largest size is called a maximum independent set and its size is denoted by $\alpha(G)$.

It is often convenient to formulate the MIS problem as a binary optimization problem. For that purpose, one can employ an identification between bit-strings and (sub)sets, an association that we will use repeatedly below. Specifically, we associate a boolean variable $s_v \in \{0, 1\}$ with each vertex $v \in V$ and identify each bitstring $\boldsymbol{s} = s_1 s_2 \dots s_{|V|} \in \{0, 1\}^{|V|}$ with a subset S of V via

$$v \in S \Leftrightarrow s_v = 1 \text{ and } v \notin S \Leftrightarrow s_v = 0.$$

The unweighted MIS problem can then be formulated as a minimization problem over the energy function

$$H_{\text{MIS}}(\boldsymbol{s}) = -\sum_{v \in V} s_v + \sum_{(u,v) \in E} U s_u s_v \tag{2.1}$$

where U is typically taken to be ∞ to satisfy the independent set constraint, but, in practice, $U > 1$ is sufficient to guarantee that the lowest energy states of H_{MIS} correspond to the MISs. Similarly, an instance of the maximum weight independent set (MWIS) problem is specified by a weighted graph $G = (V, E, \delta)$, where δ is a function assigning each vertex $v \in V$ a weight δ_v. The MWIS problem is to find an independent set that has the largest total weight. It can be similarly written as the minimization problem of the energy function

$$H_{\text{MWIS}}(\boldsymbol{s}) = -\sum_{v \in V} \delta_v s_v + \sum_{(u,v) \in E} U s_u s_v \tag{2.2}$$

where $U > \max_{v \in V} \delta_v$ guarantees that the lowest energy states correspond to the MWISs. Clearly, the MWIS problem contains the MIS problem (by setting all weights to be one).

Both the MIS problem and the MWIS problem can be formulated on arbitrary classes of graphs. In this work, we are particularly interested in UD graphs that can be implemented natively on neutral-atom quantum processors [23]. These are geometric graphs that can be drawn by placing the vertices in the 2D plane and connecting two vertices by an edge if and only if they are within a unit distance.

A particular subclass of UD graphs are subgraphs of the King's graphs. A King's subgraph (KSG) can be obtained by restricting the positions of the vertices to points on a 2D square lattice and connecting vertices by an edge if they are nearest-neighbors or next-nearest neighbors on the square lattice. Since the MIS problem on KSGs is NP-complete [7], any problem in NP can be formulated as a MIS problem on a KSG with at most a polynomial vertex overhead. In this work, we are interested in minimizing this overhead. Ref. [21] developed an efficient reduction scheme to map any problem in NP to a weighted MIS problem on a KSG. However, that procedure results necessarily in a graph with nonuniform weights. In this paper, we develop an alternative framework, based on which we find an efficient reduction scheme to map any problem in NP to an unweighted MIS problem on a KSG.

3. Unweighted MIS gadget design. Our framework for reducing the unweighted MIS problem on a general graph to that on a KSG is based on graph-rewriting techniques. The basic element of a rewriting system is a *graph rewrite rule* of the form $R \to R'$, where R is called the pattern graph and R' the replacement graph (see Figure 1). If a graph G contains R as a subgraph, we can apply the rewrite rule $R \to R'$ to G by replacing R with R'. The graph obtained in this way is denoted by $G[R \to R']$. Note that a well-defined graph rewrite rule $R \to R'$ requires a specification of how edges between R and $G \setminus R$ are mapped to edges between R' and $G \setminus R$. This motivates the definition of the boundary of a pattern graph, ∂R, consisting of the subset of vertices of R that can be connected to vertices in $G \setminus R$. The boundary of a replacement graph R' is defined analogously and denoted by $\partial R'$. Note that in many cases considered below the boundaries of R and R' coincide (e.g. Figure 1).

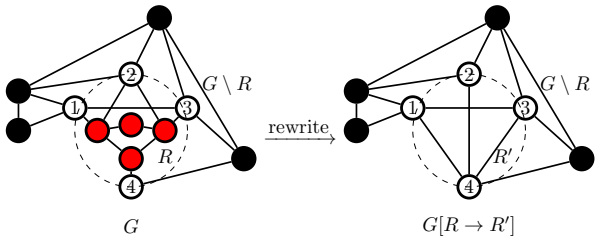


Fig. 1: Rewriting a subgraph R of G with R'. The red-faced vertices are interior vertices of R, the white faced vertices are boundary vertices $\partial R = \partial R' = \{1, 2, 3, 4\}$, and the black vertices are the remaining vertices $G \setminus R$. Note that both R and R' are allowed to contain interior vertices in a general setting, but in this specific example, R' does not contain any interior vertex.

The central idea of this work is to design rewrite rules that allow us to replace a graph G with another graph $G[R \to R']$ with more desirable properties, such as being (closer to) a UD graph. Importantly, we are only interested in rewrite rules that preserve the essential structure of the associated MIS problem, meaning that a MIS of the graph of interest, G, can be found efficiently if one knows any MIS of the rewritten graph, $G[R \to R']$. This motivates the following definition.

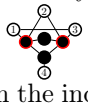
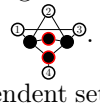
DEFINITION 3.1 (MIS-gadget). *A rewrite rule $R \to R'$ is a MIS gadget if the following holds for any graph G that contains R as a subgraph: for any MIS of $G[R \to R']$, there exists a corresponding MIS of G, such that these two MISs coincide on $G \setminus R$.*

Based on this definition, in order to obtain a MIS of G from a MIS of $G[R \to R']$, we only need to consider vertices in R. This can be achieved by the *solution extraction rule*, which tells us which vertices in R belong to the corresponding MIS of G if we are given the MIS configuration of $G[R \to R']$ on R'. We note that this solution extraction rule is not necessarily bijective, since multiple MISs of $G[R \to R']$ can be mapped to the same MIS of G. It is easy to see that a MIS gadget also satisfies the following requirement: For any graph G, the MIS size of G differs from the MIS size of $G[R \to R']$ only by a constant c independent of G, i.e., $\alpha(G) = \alpha(G[R \to R']) - c$ for any G; c is determined only by the rewrite rule and does not depend on G. This means that the MIS size of a graph of interest can be easily obtained from the MIS size of a rewritten graph.

Our goal is to find a set of properly designed MIS gadgets that allow us to progressively rewrite any graph to a UD form or even as a KSG. However, searching for unweighted MIS gadgets in the super-exponentially large graph space is challenging. In the next subsection, we therefore derive sufficient conditions for a rewrite rule to be a valid MIS gadget that can be easily checked in an automated way. This allows for a computer-assisted, automated search and forms the basis for our computer-assisted framework. Some useful gadgets found with this framework are shown in Subsection 3.2.

3.1. Conditions for MIS gadgets. In this subsection, we formulate sufficient conditions that allow us to identify valid unweighted MIS gadgets in an automated way. To this end, we introduce the concept of the α-*tensor* of a pattern or replacement graph. The α-tensor can be interpreted as a generalization of the scalar $\alpha(G)$, which denotes the MIS size of a graph G.

DEFINITION 3.2 (α-tensor). *Let R be a pattern or replacement graph and ∂R its boundary vertices. The α-tensor of R, $\boldsymbol{\alpha}(R)$, is a tensor of rank $|\partial R|$, whose element $\boldsymbol{\alpha}(R)_{\boldsymbol{s}_{\partial R}}$ is the size of the largest independent set of R, while fixing the boundary configuration $\boldsymbol{s}_{\partial R} \in \{0,1\}^{|\partial R|}$. If the boundary-vertex configuration $\boldsymbol{s}_{\partial R}$ itself violates the independent set constraint, the corresponding α-tensor element is set to $\boldsymbol{\alpha}(R)_{\boldsymbol{s}_{\partial R}} = -\infty$.*

To illustrate this concept, let us consider the pattern graph R in Figure 1, which has 4 boundary vertices. Its α-tensor, denoted by $\boldsymbol{\alpha}(R)$, is therefore a rank-4 tensor whose elements are indexed by the bit strings $\boldsymbol{s}_{\partial R} = s_1 s_2 s_3 s_4$. Specifically, the 16 elements of this tensor are listed in the second column of Table 1. Let us examine a few tensor elements to further illustrate this. Consider, for instance, the tensor element $\boldsymbol{\alpha}(R)_{0000}$: its value is 2 corresponding to the size of the largest independent set of R, given the boundary configuration $s_1 = s_2 = s_3 = s_4 = 0$; the corresponding independent sets are  and  . Here, we use a red edge color (○ or ●) to denote the vertex is in the independent set. Similarly, the entry $\boldsymbol{\alpha}(R)_{0100} = 3$ informs us that the largest independent set of R with boundary configuration $s_2 = 1$ and $s_1 = s_3 = s_4 = 0$ has size 3; the corresponding independent set is  . Lastly, the entry $\boldsymbol{\alpha}(R)_{1010} = -\infty$ indicates that the boundary configuration $s_1 = s_3 = 1$ and $s_2 = s_4 = 0$ violates the independent set constraint. Indeed, vertices 1 and 3 are connected by an edge in R, and thus cannot be both in an independent set.

The α-tensor can be used as a first tool to identify unweighted MIS gadgets, according to the following lemma.

$\boldsymbol{s}_{\partial R}$	$\boldsymbol{\alpha}(R)_{\boldsymbol{s}_{\partial R}}$	$\tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{s}_{\partial R}}$	reduced by	$\boldsymbol{\alpha}(R')_{\boldsymbol{s}_{\partial R}}$
0000	2	2	-	0
0001	3	3	-	1
0010	3	3	-	1
0100	3	3	-	1
1000	3	3	-	1
0011	3	$-\infty$	0001	$-\infty$
0101	3	$-\infty$	0001	$-\infty$
0110	4	4	-	2
1001	3	$-\infty$	0001	$-\infty$
1010	$-\infty$	$-\infty$	-	$-\infty$
1100	4	4	-	2
0111	4	$-\infty$	0110	$-\infty$
1011	$-\infty$	$-\infty$	-	$-\infty$
1101	4	$-\infty$	1100	$-\infty$
1110	$-\infty$	$-\infty$	-	$-\infty$
1111	$-\infty$	$-\infty$	-	$-\infty$

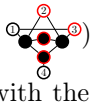
Table 1: The α-tensor (Definition 3.2) $\boldsymbol{\alpha}(R)$ (second column) and reduced α-tensor (Definition 3.6) $\tilde{\boldsymbol{\alpha}}(R)$ (third column) for the pattern graph R and for the replacement graph R' (fifth column) in Figure 1. Each row corresponds to a local MIS size associated with the boundary-vertex configuration $\boldsymbol{s}_{\partial R} \in \{0,1\}^4$ (first column). The fourth column lists the corresponding relevant boundary-vertex configurations used to remove the irrelevant boundary configurations in the corresponding rows of the first column (Definition 3.6), to form the corresponding reduced α-tensor $\tilde{\boldsymbol{\alpha}}(R)$.

LEMMA 3.3. *Let $R \to R'$ be a rewrite rule. If the boundary vertices of R and R' coincide and their α-tensors differ by a constant, i.e., $\boldsymbol{\alpha}(R) = \boldsymbol{\alpha}(R') - c$, then the rewrite rule $R \to R'$ is a valid MIS gadget.*

Proof. Consider an arbitrary graph G that contains R as a subgraph. It is easy to see that $\boldsymbol{\alpha}(R) = \boldsymbol{\alpha}(R') - c$ implies that $\alpha(G) = \alpha(G[R \to R']) - c$. To see this, select a specific configuration of the boundary vertices of R and R' and consider the largest independent sets for G and for $G[R \to R']$ with this boundary configuration. Both sets coincide on $G \setminus R$ and $G \setminus R'$ since they are exactly the same with the same boundary configuration, while on R and R' their sizes differ by c. This holds for any choice of the configuration of the boundary. Since a MIS of G and a MIS of $G[R \to R']$ can be obtained by optimizing over the boundary configuration, their sizes also differ by c. This procedure also defines the solution extraction rule: considering any MIS of $G[R \to R']$, replace all vertices in R' by those forming the largest independent set in R with the same boundary configuration. By construction, this is an independent set of G and its size is $\alpha(G[R \to R']) - c = \alpha(G)$, i.e., it is a MIS of G. ∎

Note that this lemma formulates a sufficient condition for a MIS gadget. However, it is not a necessary condition, and in fact many rewrite rules corresponding to valid MIS-gadgets do not satisfy this condition. For instance, consider again the replacement rule Figure 1, whose α-tensors are given in Table 1, and since $\boldsymbol{\alpha}(R) \neq \boldsymbol{\alpha}(R') - c$ the assumptions of Lemma 3.3 are not met. Nevertheless, this rewrite rule is a valid MIS-gadget. The reason is that the two tensors differ only by a constant on all *rele-*

vant boundary configurations. Below, we will provide a formal definition of relevant and irrelevant boundary configurations, but before formal definition, we find it instructive to first develop these concepts using the example of Figure 1. Consider for example the pattern graph R with the boundary configuration $\boldsymbol{s}_{\partial R} = 0111$. This is an irrelevant boundary configuration, in the sense that it is unnecessary to consider this configuration for finding a MIS on any graph G that contains R as a subgraph, for the reasons given below. Assume that G has a MIS, whose configuration on ∂R is indeed given by $\boldsymbol{s}_{\partial R} = 0111$. Since $\boldsymbol{\alpha}(R)_{0111} = 4$, this MIS contains exactly 4 vertices in R (corresponding to the configuration 

). Importantly, it is easy to see that if G has such a MIS, then G has another MIS, whose configuration on ∂R is given by $\boldsymbol{s}_{\partial R} = 0110$, which also contains $\boldsymbol{\alpha}(R)_{0110} = 4$ vertices in R (the corresponding configuration on R is 

). Moreover, these two MISs coincide on $G \setminus R$. In other words, for any MIS with the boundary configuration 0111 on ∂R, we can construct another MIS with the boundary configuration 0110 on ∂R. Importantly, the reverse is not true. This is because the boundary configuration 0110 is less restrictive than the boundary configuration 0111: any configuration on $G \setminus R$ that is independent with the boundary configuration 0111 on ∂R is also independent with the configuration 0110 on ∂R, but not vice versa. Since the boundary configuration 0111 is not needed to construct a MIS of G, we call this configuration irrelevant. This notion is formalized with the following two definitions.

DEFINITION 3.4 (less restrictive relation). *Consider two bitstrings of equal length n, $\boldsymbol{s} \in \{0,1\}^n$ and $\boldsymbol{t} \in \{0,1\}^n$. We say that $\boldsymbol{s}$ is less restrictive than $\boldsymbol{t}$ if $s_i \leq t_i$ for all $i \in \{1, \ldots, n\}$. We denote this by $\boldsymbol{s} \prec \boldsymbol{t}$.*

Clearly, the least restrictive boundary configuration is the one containing all zeros. For any other boundary configuration, one can always find less restrictive ones by flipping some number of ones to zeros.

DEFINITION 3.5 (irrelevant boundary configuration). *Let R be a pattern or replacement graph, ∂R its boundary, and $\boldsymbol{\alpha}(R)$ its α-tensor. A boundary-vertex configuration $\boldsymbol{t}$ is irrelevant if there exists another boundary-vertex configuration $\boldsymbol{s}$ such that $\boldsymbol{s} \prec \boldsymbol{t}$ and $\boldsymbol{\alpha}(R)_{\boldsymbol{s}} \geq \boldsymbol{\alpha}(R)_{\boldsymbol{t}}$, or if $\boldsymbol{\alpha}(R)_{\boldsymbol{t}} = -\infty$. A boundary configuration is called relevant if it is not irrelevant.*

These concepts are crucial in defining the reduced α-tensor, which plays the central role in our approach to find MIS-gadgets below.

DEFINITION 3.6 (reduced α-tensor). *Let $\boldsymbol{\alpha}(R)$ be an α-tensor for a pattern or replacement graph R. The corresponding reduced α-tensor is defined by setting all entries in $\boldsymbol{\alpha}(R)$ that correspond to irrelevant boundary-vertex configurations to $-\infty$.*

In the following, we use the notation $\tilde{\boldsymbol{\alpha}}(R)$ to denote the reduced α-tensor for the pattern or replacement graph R. As an example, we list in Table 1 the reduced α-tensor for the pattern graph R in Figure 1. This table also identifies the irrelevant boundary configurations and lists the corresponding relevant boundary-vertex configurations used to reduce the α-tensor.

With this, we are prepared to state the following theorem, which formulates a useful condition to identify MIS-gadgets.

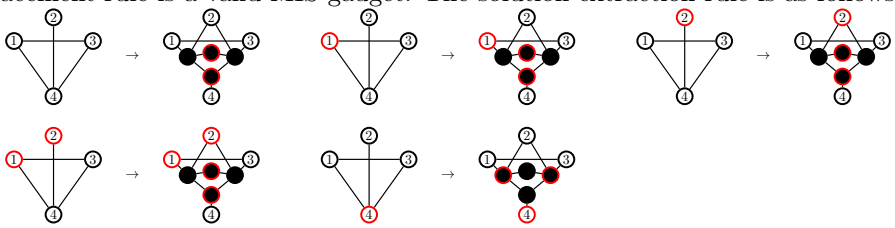
THEOREM 3.7. *Let $R \to R'$ be a rewrite rule, such that the boundaries of R and R' coincide. The rewrite rule $R \to R'$ is a valid MIS gadget if and only if the reduced*

α-tensors of the pattern and replacement graphs R and R' differ by a constant, i.e., $\tilde{\boldsymbol{\alpha}}(R) = \tilde{\boldsymbol{\alpha}}(R') - c$.

Proof. The proof of sufficiency is a simple variant of the one outlined in the proof of Lemma 3.3. Consider an arbitrary graph G containing R as a subgraph. It is easy to see that $\tilde{\boldsymbol{\alpha}}(R) = \tilde{\boldsymbol{\alpha}}(R') - c$ implies that $\alpha(G) = \alpha(G[R \to R']) - c$. For this, consider the relevant boundary vertex configurations of the replacement graph, R'. For each such boundary configuration, consider the largest independent sets for G and for $G[R \to R']$ fixing the specific boundary configuration. We can choose these sets such that they coincide on $G \backslash R$, while on R and R' their size differs by c. This hold for any relevant configurations of the boundary. Since maximum independent sets of G and $G[R \to R']$ are obtained by optimizing over the relevant boundary configurations, their sizes also differ by c. The solution extraction rule is given as follows. Consider any MIS of $G[R \to R']$. If its boundary configuration $\boldsymbol{t}$ is irrelevant, construct the corresponding MIS that has a relevant boundary configuration $\boldsymbol{s}$ such that $\boldsymbol{s} \prec \boldsymbol{t}$. Note that such $\boldsymbol{s}$ exists by definition. We then replace all vertices in R' by those forming the largest independent set in R with the same boundary configuration $\boldsymbol{s}$. By construction, this is an independent set of G and its size is $\alpha(G[R \to R']) - c = \alpha(G)$, i.e., it is a MIS of G.

The necessity condition is proved in Appendix A, where we show that the reduced α-tensors of the pattern and replacement graphs R and R' cannot be further reduced without invalidating the theorem. ∎

To conclude this discussion, we return to the example of Figure 1. As we can see from Table 1, the reduced α-tensors of R and R' differ by 2, and therefore this replacement rule is a valid MIS-gadget. The solution extraction rule is as follows:

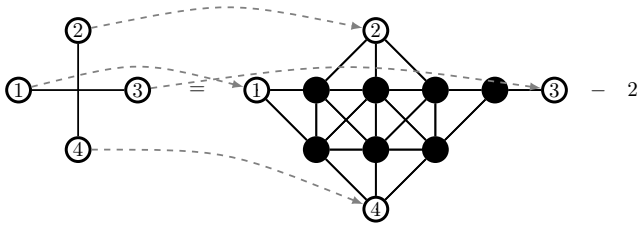


We note that only mappings corresponding to relevant boundary configurations are displayed, since those are the only ones that are necessary. We further note that we dropped mappings for configurations that can be obtained from the above via the vertical reflection symmetry. With the reduced α-tensors, we can reliably identify unweighted MIS gadgets by searching over the feasible graph space.

3.2. Some Useful Gadgets. In the following, we show several gadgets that will be used in Section 4 for reducing the MIS problem on a general graph to that on a UD graph and a KSG. These gadgets are found by exhaustive search over all UD graphs up to a certain size, and their solution extraction rules are listed in Appendix B. The first two gadgets that we introduce below is useful for removing crossings from a graph. In general, crossings are not allowed in UD graphs, as they often cannot be removed by moving the vertices in two dimensions, in particular when the graph is not planar. In our graph rewriting based scheme, a crossing can be removed by applying the following gadget.

Corollary 3.8.

(3.1)



Let CROSS *be the graph on the left-hand side of Equation (3.1) and* BATOIDEA *be the KSG on the right-hand side. The correspondence between their boundary vertices is indicated by the dashed arrows.* CROSS $\to$ BATOIDEA *is a valid MIS gadget, and it is optimal in terms of the number of added vertices,* 7*. The constant difference in the MIS size is* 2*.*

Proof. The reduced α-tensors for CROSS and BATOIDEA are listed in Table 2. By comparing the second and third columns, we can see the corresponding reduced α-tensors differ by a constant 2, proving Equation (3.1) via Theorem 3.7. The optimality is shown by the exhaustive search algorithm detailed in Appendix C. ∎

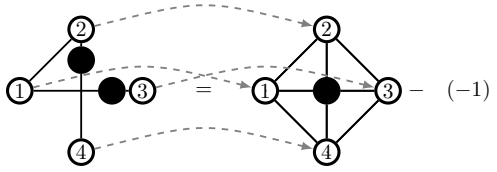
$\boldsymbol{s}_{\partial R}$	$\tilde{\boldsymbol{\alpha}}(\text{CROSS})_{\boldsymbol{s}_{\partial R}}$	$\tilde{\boldsymbol{\alpha}}(\text{BATOIDEA})_{\boldsymbol{s}_{\partial R}}$	difference
0000	0	2	2
0001	1	3	2
0010	1	3	2
0100	1	3	2
1000	1	3	2
0011	2	4	2
0110	2	4	2
1100	2	4	2
0101	$-\infty$	~~3~~	-
1010	$-\infty$	~~3~~	-
1001	2	4	2
0111	$-\infty$	~~3~~	-
1110	$-\infty$	~~4~~	-
1101	$-\infty$	~~4~~	-
1011	$-\infty$	~~4~~	-
1111	$-\infty$	~~4~~	-

Table 2: The reduced α-tensor elements for CROSS and BATOIDEA. Tensor indices $\boldsymbol{s}_{\partial R}$ are the boundary vertex configurations $s_1 s_2 s_3 s_4 \in \{0,1\}^4$. Irrelevant entries are set to $-\infty$ with /.

Some crossings with extra features can be removed by gadgets with less vertex overhead. For example, to rewrite a crossing with two additional vertices and one additional edge, we introduce the following gadget.

Corollary 3.9.

(3.2)



Let CROSS + EDGE *be the graph on the left-hand of Equation (3.2), which consists of a crossing, two additional vertices and one additional edge, and* PIRAMID *be the KSG on the right-hand side. The correspondence between their boundary vertices is indicated by the dashed arrows.* CROSS + EDGE → PIRAMID *is a valid MIS gadget, and it is optimal in terms of the number of vertices. The constant difference in the MIS size is* −1.

Proof. The reduced α-tensors for CROSS + EDGE and PIRAMID are listed in Table 3. By comparing the second and third columns, we can see the corresponding reduced α-tensors differ by a constant -1, proving Equation (3.2) via Theorem 3.7. Again, the optimality is proved by our exhaustive searching program detailed in

$\boldsymbol{s}_{\partial R}$	$\tilde{\boldsymbol{\alpha}}(\text{CROSS} + \text{EDGE})_{\boldsymbol{s}_{\partial R}}$	$\tilde{\boldsymbol{\alpha}}(\text{PIRAMID})_{\boldsymbol{s}_{\partial R}}$	difference
0000	2	1	−1
0001	~~2~~	~~1~~	-
0010	~~2~~	~~1~~	-
0100	~~2~~	~~1~~	-
1000	~~2~~	~~1~~	-
0011	~~2~~	$-\infty$	-
0110	~~2~~	$-\infty$	-
0101	3	2	−1
1010	3	2	−1
1001	~~2~~	$-\infty$	-
0111	~~3~~	$-\infty$	-
1011	~~3~~	$-\infty$	-

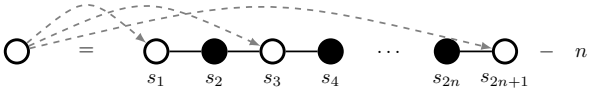
Table 3: The reduced α-tensor elements for CROSS + EDGE and PIRAMID. Tensor indices $\boldsymbol{s}_{\partial R}$ are the boundary vertex configurations $s_1 s_2 s_3 s_4 \in \{0,1\}^4$. Irrelevant entries are set to $-\infty$ with /.

Appendix C. ∎

In the following, we introduce the copy gadget, which is not a single gadget but a class of gadgets that can be used to copy the information of a vertex to other vertices.

Corollary 3.10.

(3.3)



Let K_1 *be the single-vertex graph on the left hand side of Equation (3.3), and* COPY^n *be the path graph on the right hand side with* $(2n+1)$ *vertices for some integer* n.

The boundary vertices of COPY^n *are those annotated with odd indices. The correspondence between their boundary vertices is indicated by the dashed arrows, which is one-to-many. Then* $K_1 \to \mathrm{COPY}^n$ *forms a MIS gadget, and the constant difference in the MIS size is* n*.*

Proof. This proof that this rewrite rule is a valid MIS gadget requires a bit more care, since it does not satisfy the conditions of Theorem 3.7, since the boundaries of $R = K_1$ and $R' = \mathrm{COPY}^n$ do not coincide. While R consists of a single vertex that is also its boundary, R' consists of $2n+1$ vertices, out of which $n+1$ vertices form its boundary. Nevertheless, we show it is a valid MIS gadget according to Definition 3.1. This rewrite rule can be understood as follows: If we replace a vertex R in a graph G with R', we can connect any edge $e \in E$ that connects a vertex v in $G \setminus R$ with R by an edge that connects v with any one of the boundary vertices in R'. Note that different choices for these edges result in different rewritten graphs $G[R \to R']$. However, for each choice, the replacement rule is a valid MIS gadget.

To see this, let us first note that COPY^n has only two relevant boundary configurations. These are $s_i = 0$ for all odd i on one hand, and $s_i = 1$ for all odd i on the other hand. The corresponding largest independent sets on R' have size n and $n+1$ respectively. Importantly, the reduced α tensors for R and R' thus differ by a constant n on all relevant boundary configurations (see Table 4). Let us consider any MIS of $G[R \to R']$. It is easy to see that we can construct a MIS of G from this set by replacing the vertices in R' with the single vertex in R if the boundary configuration of R' is $s_i = 1$ for all boundary vertices. For any other boundary configuration, we do not add R to the independent set to obtain a MIS of G. ∎

$\boldsymbol{s}_{\partial R}$	$\tilde{\boldsymbol{\alpha}}(K_1)_{\boldsymbol{s}_{\partial R}}$	$\boldsymbol{s}_{\partial R'}$	$\tilde{\boldsymbol{\alpha}}(\mathrm{COPY}^n)_{\boldsymbol{s}_{\partial R'}}$	difference
0	0	$00\ldots0$	n	n
1	1	$11\ldots1$	$n+1$	n
-	-	others	$\cancel{\leq n}$	-

Table 4: The reduced α-tensor elements for K_1 and COPY^n. Indices $\boldsymbol{s}_{\partial R} \in \{0,1\}$ are for $\boldsymbol{\alpha}(K_1)$ while $\boldsymbol{s}_{\partial R'} \in \{0,1\}^{n+1}$ are for $\boldsymbol{\alpha}(\mathrm{COPY}^n)_{\boldsymbol{s}_{\partial R'}}$.

The copy gadget is useful for decomposing a high-degree vertex into low-degree ones or bridging two vertices that are far away from each other. By applying the copy gadget multiple times, we can also rewrite a vertex to a tree graph.

4. Reduction of an unweighted MIS problem on a general graph to that on a KSG. The combination of the three gadgets given in Equation (3.1), Equation (3.2), and Equation (3.3) allow for a straightforward reduction of the MIS problem on a general graph to a MIS problem on a KSG. The reduction follows the general idea introduced in Ref. [21], which is based on the concept of the crossing lattice. Generalizing this idea results in the following theorem, which we refine later by combining it with strategies to reorder the vertices in the source graph.

THEOREM 4.1. *The problem of finding a maximum independent set on a general graph* $G = (V, E)$ *can be reduced to that on a KSG with* $O(|V|^2)$ *vertices.*

Proof. We prove this theorem by constructing a two-step reduction scheme as illustrated in the top panel of Figure 2: In step ①, we construct a crossing lattice as shown in subplot (b). We first align the vertices in the source graph into a row and

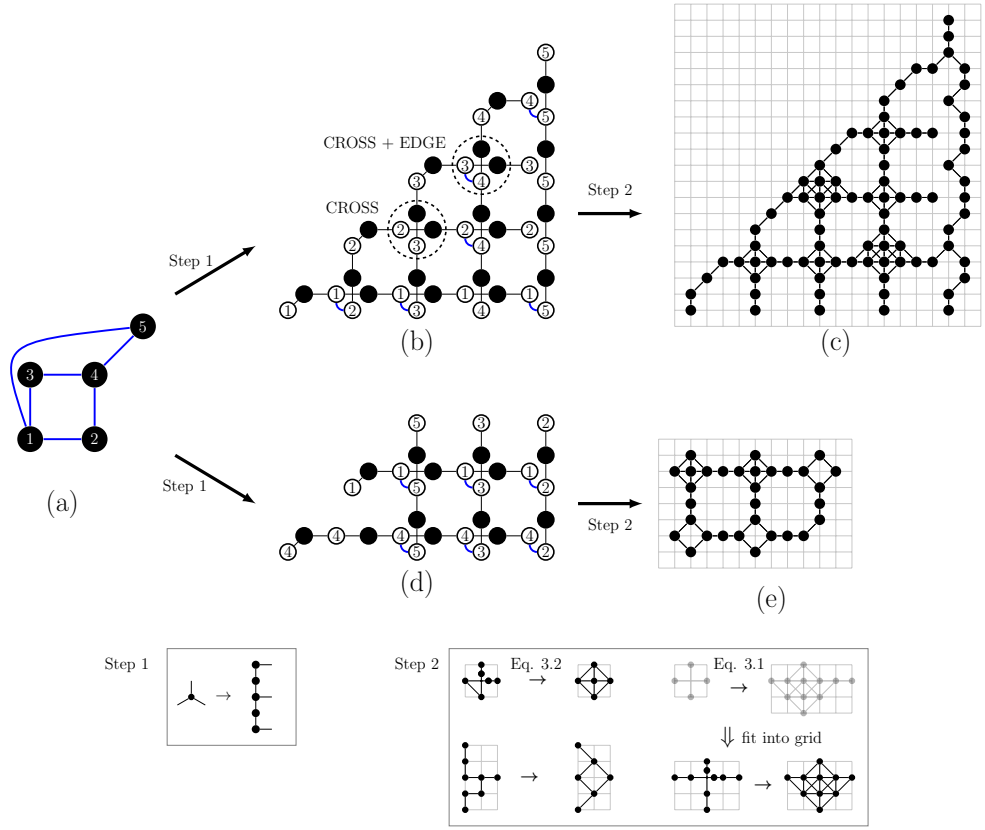


Fig. 2: The reduction schemes for the MIS problem on a general graph G to that on a KSG. Subplots (a), (b), and (c) on the top panel are for the $O(|V|^2)$ mapping without vertex reordering. (a) is the source graph. (b) is the crossing lattice for (a), which is obtained by applying the copy gadgets on the vertices in (a). Black and white circles are interior and boundary vertices respectively. The edges in blue are from the source graph. (c) is the KSG obtained by applying crossing gadgets to the crossings in (b). Subplots (d), (e) on the bottom panel are for the $O(|V| \times \mathrm{pw}(G))$ mapping with vertex reordering, where $\mathrm{pw}(G)$ is the pathwidth of G. (d) is the crossing lattice with vertex reordering. (e) is the KSG obtained by applying crossing gadgets to the crossings in (d). The gadgets applied in each step is indicated by the boxes "Step 1" and "Step 2".

then rewrite each of them with a "Γ" shaped line graph with odd number of vertices (Equation (3.3)). The collection of these line graphs forms a crossing lattice [21], a two dimensional geometric graph that has crossings at certain lattice sites for any $u, v \in V$. By Corollary 3.10, all vertices with odd indices in the same line are "equivalent" to the source vertex, and hence an edge connecting two vertices in the source graph can be redistributed to any pairs of vertices that are equivalent to them. The redistributed edges are shown in the blue lines in subplot (b), where all of them are located at the crossings of the crossing lattice. In step ②, we apply either CROSS → BATOIDEA in Equation (3.1) or CROSS + EDGE → PIRAMID in Equation (3.2) to remove the crossings of the crossing lattice (the dashed circles in subplot (b)), so that the resulting

graph becomes a unit-disk graph. To embed the graph into a grid, extra vertices can be inserted via the copy gadgets, which explains why the gadgets in the figure is slightly different from those in the equations. The resulting KSG is shown in subplot (c). The generated graph can be further simplified by introducing extra gadgets, such as those for trimming the dangling legs (e.g., gadget in the box of step 2). To extract a solution from a MIS of the obtained KSG, one can apply the solution extraction rules detailed in Appendix B in an order that is reverse to the gadget application. □

4.1. Path decomposition and optimal vertex ordering. By relating the layout of the crossing lattice with the path decomposition, we can further reduce the depth of the mapped KSG to $O(\mathrm{pw}(G))$, where $\mathrm{pw}(G)$ is the pathwidth of the source graph $G = (V, E)$, which is bounded above by the number of vertices, $|V|$. This improvement is most significant for sparse graphs, which usually have small pathwidth. For example, the pathwidth of a 3-regular graph is asymptotically bounded by $|V|/6$ [9], and the pathwidth of a tree graph is $O(\log |V|)$.

DEFINITION 4.2 (path decomposition and pathwidth [25]). *Given a graph $G = (V, E)$, its path decomposition is a sequence of bags $X_i \subseteq V$, with the following two properties:*

1. *For each edge of G, there exists an i such that both endpoints of the edge belong to the bag X_i,*
2. *and for every three indices $i \le j \le k$, $X_i \cap X_k \subseteq X_j$.*

The path decomposition defines a mapping from a general graph to a path graph $\{X_1, X_2, \ldots, X_m\}$ and its width is defined as $\max_{j=1}^{m} |X_j| - 1$. The smallest width among all path decompositions is the pathwidth of the graph, which is denoted as $\mathrm{pw}(G)$.

A crossing lattice with a depth $\mathrm{pw}(G) + 1$ can be inferred from the optimal path decomposition of the source graph. Consider the graph in Figure 2 (a), which has an optimal path decomposition

$$\xrightarrow{+4} (4) \xrightarrow{+1} (41) \xrightarrow{+5} (415) \xrightarrow{+3} (413) \xrightarrow{+2} (412). \tag{4.1}$$

It can be diagrammatically represented in Figure 3, where each column is a bag in the path decomposition, and the maximum bag size is equal to the number of rows. Whenever a vertex is added to a bag, a segment with a horizontal span (s_v, f_v) is added to the diagram, where s_v is the step that the vertex is added and f_v is the step that the vertex is removed. It is easy to verify that all edges in the source graph can be mapped to one of the bags in the path decomposition: $(1, 5), (4, 5) \in X_3$, $(1, 2), (2, 4) \in X_5$, and $(1, 3), (3, 4) \in X_4$.

THEOREM 4.3. *The problem of finding a maximum independent set on a general graph $G = (V, E)$ can be reduced to that on a KSG of width $O(|V|)$ and depth* $\mathrm{pw}(G) + 1$, *where* $\mathrm{pw}(G)$ *is the pathwidth of G.*

Proof. We prove this theorem using the example in Figure 2 and Figure 3. The idea generalizes straightforwardly to a general graph G. Given a path decomposition, a vertex ordering can be obtained by sorting the vertices according to the step that they are added to the bags, e.g., $(4, 1, 5, 3, 2)$ in the above example in Figure 3. We assign each column to a vertex and wire its copy gadget into the "⊢" shape as shown in Figure 3 (the red solid lines for example). By Definition 4.2, for any $(u, v) \in E$, copy gadgets of u and v are guaranteed to cross at a certain bag. The edge (u, v) in the source graph can be redistributed to the crossing of the copy gadgets of u and v as shown in Figure 2 (d). Finally, we apply crossing gadgets to the crossings in the

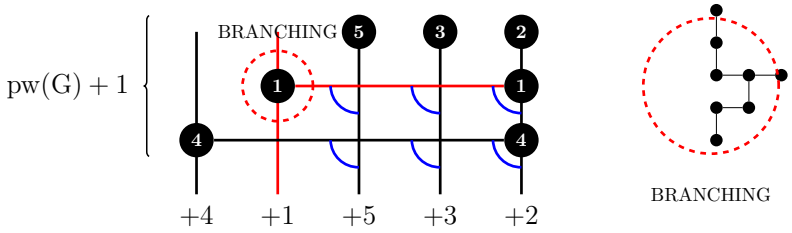


Fig. 3: The optimal path decomposition of the source graph in Figure 2 (a), where each column is a bag in the path decomposition, and the maximum bag size is equal to the number of rows. A segment with a horizontal span (s_v, f_v) represents a vertex v that is added to the bag at step s_v and removed in a future step $f_v + 1$. The blue curves are the edges in the source graph. In the proof of Theorem 4.3, we explain how to connect the vertices with the corresponding edges using the copy gagdets. The copy gadgets are wired in the "⊢" shape as highlighted in the red color. The "BRANCHING" structure appears in the center of the "⊢" shape, which can be rewritten into a KSG by one of the gadgets in the "Step 2" of Figure 2.

crossing lattice and remove the dangling vertices as shown in Figure 2 (e), as well as a new gadget to rewrite the BRANCHING (Figure 3) structure at the center of a "⊢" to a KSG. The correctness of this new gadget can be verified by representing it as a combination of multiple copy gadgets. The resulting graph is a KSG of width $O(|V|)$ and depth $\mathrm{pw}(G) + 1$. □

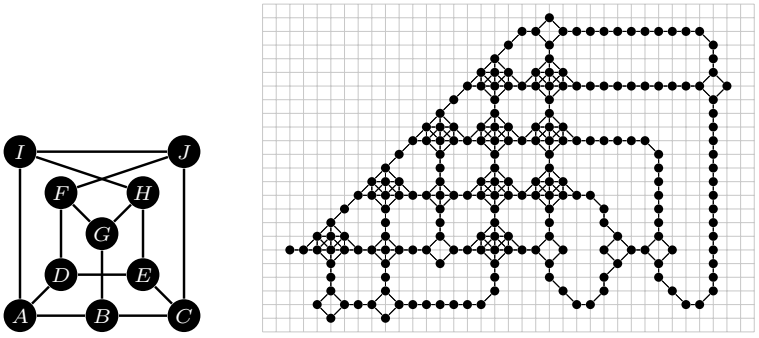
With an optimal path decomposition, the graph in Figure 2 (a) can be mapped to the KSG in Figure 2 (e) with only 31 vertices. The target graph has depth 2 rather than $\mathrm{pw}(G)+1 = 3$ because of the removal of the dangling vertices. Note that removing an even number of dangling vertices is effectively applying the copy gadget in a reversed way, which is also a MIS gadget.

Example 1. Reducing the Petersen graph to a KSG. The Petersen graph is a 3-regular graph with 10 vertices as shown in Figure 4 (a). One of its optimal path decomposition, which has pathwidth 5, is

$$\begin{aligned} &\xrightarrow{+\mathrm{A}} (A) \xrightarrow{+\mathrm{B}} (AB) \xrightarrow{+\mathrm{C}} (ABC) \xrightarrow{+\mathrm{D}} (ABCD) \xrightarrow{+\mathrm{E}} (ABCDE) \\ &\xrightarrow{+\mathrm{G}} (ABCDEG) \xrightarrow[-\mathrm{B}]{+\mathrm{F}} (ACDEGF) \xrightarrow[-\mathrm{D}]{+\mathrm{H}} (ACEGFH) \\ &\xrightarrow[-\mathrm{EG}]{+\mathrm{I}} (ACFHI) \xrightarrow[-\mathrm{AH}]{+\mathrm{J}} (CFIJ). \end{aligned} \tag{4.2}$$

The mapped KSG as shown in Figure 4 (b) has depth 6 and a grid size 23×32. It has 218 vertices and the MIS size of which is larger than that of the Petersen graph by a constant 88. A sample notebook that implements the reduction of this graph is available in our Github repository.

4.2. Lower bound on reduction size. In this subsection, we will show the proposed KSG reduction scheme for the MIS problem is optimal up to a constant or logarithmic factor, otherwise, there will be a sub-exponential time algorithm for finding the maximum independent sets of a general graph, better than any existing classical algorithms [24], in contradiction with the exponential time hypothesis.



(a) (b)

Fig. 4: (a) The Petersen graph and (b) its KSG mapping with an optimal vertex ordering.

DEFINITION 4.4 (area-law graph). *A D-dimensional area-law graph is a geometric graph such that there exists a reference point such that given a ball of radius r centered at the reference point, the number of vertices contained in the ball is upper bounded by its volume αr^D for some α and the number of edges cut by the unit ball is upper bounded by the surface area βr^{D-1} for some β.*

It is easy to see that KSGs are area-law graphs.

LEMMA 4.5. *There exists an algorithm $\mathcal{B}$ which can find a MIS of any area-law graph $\mathcal{G}_a = (V, E)$ in spatial dimension D in a time of at most $O(|E|)2^{O\left(|V|^{\frac{D-1}{D}}\right)}$.*

Proof. The algorithm $\mathcal{B}$ can be the tropical tensor network method in Ref. [15], which can solve the MIS problem on $\mathcal{G}_a$ in time $O(|E|)2^{O(\mathrm{tw}(\mathcal{G}_a))}$. Here, $\mathrm{tw}(\mathcal{G}_a)$ is the treewidth of $\mathcal{G}_a$ and it is upper bounded by the pathwidth since a path decomposition is also a tree decomposition. For an area-law graph, a path decomposition of width $O\left(|V|^{\frac{D-1}{D}}\right)$ can be obtained with the process illustrated in Figure 5. Starting from some reference point, we draw a ball of radius $r = 0$. Then, we increase the radius to include more edges into the ball. Whenever the surface of the ball cuts a new edge, a bag that consists of endpoints of edges at the boundary is added to the sequence. At certain point, all edges will be included, and the generated sequence of bags corresponds to a path decomposition of $\mathcal{G}_a$. By the definition of area-law graphs, the size of the bag is proportional to the surface area of the ball, r^{D-1}. Since the number of vertices included in the ball scales as its volume r^D, the overall time and space complexity to contract this tensor network is $O(|E|)2^{O\left(|V|^{\frac{D-1}{D}}\right)}$, thus proving the lemma. ∎

THEOREM 4.6. *Assuming the exponential time hypothesis is true, no algorithm can reduce the problem of finding a maximum independent set on a general graph $G = (V, E)$ to that on an area-law graph in dimension D of at most $\eta|V|^{\frac{D}{D-1}-\epsilon}$ vertices for some η and any $\epsilon > 0$.*

Proof. We prove the theorem by contradiction. Suppose there exists a polynomial time algorithm $\mathcal{A}$ which reduces the problem of finding a MIS on any graph $G = (V, E)$

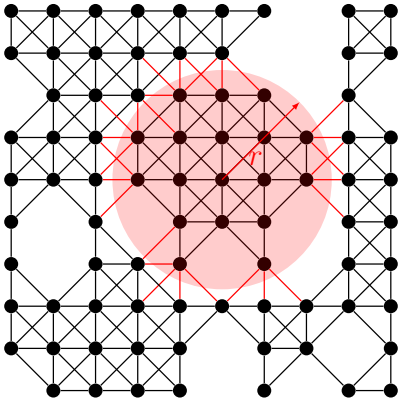


Fig. 5: A way to construct a path decomposition of an area-law graph, e.g. the KSG in the graph. The vertices and edges in the red ball are those already included in the existing bags. The vertices and edges at the cut (in red color) are the elements of the new bag.

onto finding a MIS on an area-law graph $\mathcal{G}_a$ in dimension D with at most $\eta|V|^{\frac{D}{D-1}-\epsilon}$ vertices for some η and $\epsilon > 0$. By Lemma 4.5, there exists an algorithm $\mathcal{B}$ which can solve the MIS on an area-law graph $\mathcal{G}_a = (V_a, E_a)$ in dimension D in a time of at most $O(|E_a|)2^{O\left(|V_a|^{\frac{D-1}{D}}\right)}$. An algorithm $\mathcal{C}$ may be used to solve the MIS on any graph by using $\mathcal{A}$ and $\mathcal{B}$ as subroutines as follows. First, given an arbitrary graph G with $|V|$ vertices, use algorithm $\mathcal{A}$ to compute an area-law graph $\mathcal{G}_a$ of $|V_a| = \eta|V|^{\frac{D}{D-1}-\epsilon}$ vertices. Then, use algorithm $\mathcal{B}$ to find a MIS of $\mathcal{G}_a$, which takes a time of at most $\text{poly}(|V|)2^{O\left(|V|^{1-\epsilon\frac{D-1}{D}}\right)}$. Then, given a MIS solution of $\mathcal{G}_a$, the solution extraction routine that associated with algorithm $\mathcal{A}$ can compute a solution of the source graph G. The exponential time hypothesis (ETH) states that MIS of a generic source graph G cannot be solved in a time lower bounded by $\text{poly}(|V|)2^{O(|V|)}$, by a linear reduction of 3-SAT to MIS [13, 18]. Assuming the ETH is True, it contradicts with the existence of an algorithm $\mathcal{C}$. Therefore, by contradiction, if ETH is True, the algorithm $\mathcal{A}$ does not exist, proving the theorem. □

5. Discussion and outlook. In this work, we introduced a framework of unweighted MIS gadget design based on graph rewriting and use it to search for MIS gadgets in an automated approach. With a few elementary MIS gadgets, we can reduce an unweighted MIS problem on a general graph to that on a KSG with at most a quadratic overhead in the number of vertices. This greatly facilitates the encoding of the MIS problems onto programmable neutral-atom quantum processors, since KSGs can be natively implemented on the quantum hardware. It also removes the need for unequal weights, which were required in the reduction scheme in Ref. [21]. We show that by reordering the vertices, the mapped KSG has a size of $O(|V| \times \text{pw}(G))$, where $\text{pw}(G)$ is the pathwidth of G. We further show that the reduction scheme is optimal up to a constant or logarithmic factor, assuming the exponential time hypothesis holds.

We note that the framework of automated gadget design based on graph rewrite is applicable to broader context beyond MIS reduction on KSGs. The MIS reduction is rather a specific example of the general framework. As a future direction, we are interested in applying the reduction scheme to other constraint satisfaction problems. We remark that the α-tensors and reduced α-tensors in the gadget searching scheme naturally extends to multiple constraint satisfaction problems with the tropical tensor network representations in Ref. [15]. The range of problems include but not limited to the spin-glass problem, the matching problem, the k-coloring problem, the max-cut problem, the binary paintshop problem, the set packing problem, and the set covering problem [15].

The open-source implementation of the MIS reduction scheme is available on Github [1], which is released as a Julia package.

Acknowledgments. Jinguo Liu thanks Dominik Wild for helpful discussion on the exponential time hypothesis. We acknowledge financial support from the DARPA ONISQ program (grant no. W911NF2010021), the ERC Starting grant QARA (grant no. 101041435), the European Unions Horizon 2020 research and innovation program under Grant Agreement No. 101079862 (PASQuanS2), the Army Research Office (grant no. W911NF-21-1-0367), the U.S. Department of Energy (DE-SC0021013) and DOE Quantum Systems Accelerator Center (contract no. 7568717).

REFERENCES

[1] https://github.com/QuEraComputing/UnitDiskMapping.jl.

[2] R. S. Andrist, M. J. A. Schuetz, P. Minssen, R. Yalovetzky, S. Chakrabarti, D. Herman, N. Kumar, G. Salton, R. Shaydulin, Y. Sun, M. Pistoia, and H. G. Katzgraber, *Hardness of the Maximum Independent Set Problem on Unit-Disk Graphs and Prospects for Quantum Speedups*, arXiv e-prints, (2023), arXiv:2307.09442, p. arXiv:2307.09442, https://doi.org/10.48550/arXiv.2307.09442, https://arxiv.org/abs/2307.09442.

[3] H. Breu and D. G. Kirkpatrick, *Unit disk graph recognition is np-hard*, Computational Geometry, 9 (1998), pp. 3–24.

[4] S. Butenko, P. Pardalos, I. Sergienko, V. Shylo, and P. Stetsyuk, *Estimating the size of correcting codes using extremal graph problems*, Optimization: Structure and Applications, (2009), pp. 227–243, https://link.springer.com/chapter/10.1007/978-0-387-98096-6_12.

[5] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, *A limited memory algorithm for bound constrained optimization*, SIAM Journal on Scientific Computing, 16 (1995), pp. 1190–1208, https://doi.org/10.1137/0916069.

[6] B. N. Clark, C. J. Colbourn, and D. S. Johnson, *Unit disk graphs*, Discrete mathematics, 86 (1990), pp. 165–177.

[7] S. Ebadi, A. Keesling, M. Cain, T. T. Wang, H. Levine, D. Bluvstein, G. Semeghini, A. Omran, J.-G. Liu, R. Samajdar, X.-Z. Luo, B. Nash, X. Gao, B. Barak, E. Farhi, S. Sachdev, N. Gemelke, L. Zhou, S. Choi, H. Pichler, S.-T. Wang, M. Greiner, V. Vuletic, and M. D. Lukin, *Quantum optimization of maximum independent set using rydberg atom arrays*, Science, 0, p. eabo6587, https://doi.org/10.1126/science.abo6587.

[8] T. Etzion and P. R. Ostergard, *Greedy and heuristic algorithms for codes and colorings*, IEEE Transactions on Information Theory, 44 (1998), pp. 382–388, https://ieeexplore.ieee.org/abstract/document/651069.

[9] F. V. Fomin and K. Høie, *Pathwidth of cubic graphs and exact algorithms*, Information Processing Letters, 97 (2006), pp. 191–196.

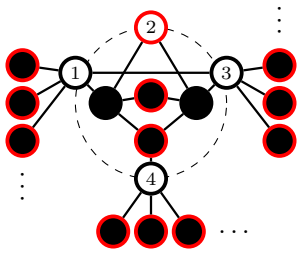
[10] F. Glover, G. Kochenberger, and Y. Du, *A tutorial on formulating and using qubo models*, arXiv preprint arXiv:1811.11538, (2018).

[11] A. Griewank and A. Walther, *Evaluating derivatives: principles and techniques of algorithmic differentiation*, SIAM, 2008.

[12] J. Hastad, *Clique is hard to approximate within* $n^{1-\epsilon}$, in Proceedings of 37th Conference on

Foundations of Computer Science, IEEE, 1996, pp. 627–636.
[13] R. Impagliazzo and R. Paturi, *On the complexity of k-sat*, Journal of Computer and System Sciences, 62 (2001), pp. 367–375.
[14] M. Kim, K. Kim, J. Hwang, E.-G. Moon, and J. Ahn, *Rydberg quantum wires for maximum independent set problems*, Nature Physics, 18 (2022), pp. 755–759, https://www.nature.com/articles/s41567-022-01629-5.
[15] J.-G. Liu, X. Gao, M. Cain, M. D. Lukin, and S.-T. Wang, *Computing solution space properties of combinatorial optimization problems via generic tensor networks*, SIAM Journal on Scientific Computing, 45 (2023), pp. A1239–A1270, https://doi.org/10.1137/22M1501787, https://doi.org/10.1137/22M1501787.
[16] J.-G. Liu, L. Wang, and P. Zhang, *Tropical tensor network for ground states of spin glasses*, Physical Review Letters, 126 (2021), https://doi.org/10.1103/physrevlett.126.090506, https://doi.org/10.1103%2Fphysrevlett.126.090506.
[17] J.-G. Liu and T. Zhao, *Differentiate Everything with a Reversible Programming Language*, (2020), http://arxiv.org/abs/2003.04617, https://arxiv.org/abs/2003.04617.
[18] D. Lokshtanov, D. Marx, S. Saurabh, et al., *Lower bounds based on the exponential time hypothesis*, Bulletin of EATCS, 3 (2013).
[19] B. D. McKay, *Applications of a technique for labelled enumeration*, Congressus Numerantium, 40 (1983), pp. 207–221.
[20] C. Moore and S. Mertens, *The nature of computation*, OUP Oxford, 2011.
[21] 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 (2023), p. 010316, https://doi.org/10.1103/PRXQuantum.4.010316, https://link.aps.org/doi/10.1103/PRXQuantum.4.010316.
[22] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, *Computational complexity of the rydberg blockade in two dimensions*, arXiv preprint arXiv:1809.04954, (2018).
[23] H. Pichler, S.-T. Wang, L. Zhou, S. Choi, and M. D. Lukin, *Quantum Optimization for Maximum Independent Set Using Rydberg Atom Arrays*, arXiv e-prints, (2018), arXiv:1808.10816, p. arXiv:1808.10816, https://doi.org/10.48550/arXiv.1808.10816, https://arxiv.org/abs/1808.10816.
[24] B. Randerath and I. Schiermeyer, *On maximum independent sets in p5-free graphs*, Discret. Appl. Math., 158 (2010), pp. 1041–1044.
[25] N. Robertson and P. Seymour, *Graph minors. i. excluding a forest*, Journal of Combinatorial Theory, Series B, 35 (1983), pp. 39–61, https://doi.org/https://doi.org/10.1016/0095-8956(83)90079-5, https://www.sciencedirect.com/science/article/pii/0095895683900795.
[26] J. Wurtz, A. Bylinskii, B. Braverman, J. Amato-Grill, S. H. Cantu, F. Huber, A. Lukin, F. Liu, P. Weinberg, J. Long, S.-T. Wang, N. Gemelke, and A. Keesling, *Aquila: QuEra's 256-qubit neutral-atom quantum computer*, arXiv e-prints, (2023), arXiv:2306.11727, p. arXiv:2306.11727, https://doi.org/10.48550/arXiv.2306.11727, https://arxiv.org/abs/2306.11727.
[27] 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 e-prints, (2022), arXiv:2205.08500, p. arXiv:2205.08500, https://doi.org/10.48550/arXiv.2205.08500, https://arxiv.org/abs/2205.08500.

Appendix A. The reduced α-tensor is minimal. In this appendix, we show that the further removal of any finite-valued elements in the reduced α-tensors may reduce the MIS size of the host graph, i.e., the number of finite-valued elements in a reduced α-tensor is minimal. Consider a boundary-vertex configuration $\boldsymbol{s}$ such that $\tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{s}} \neq -\infty$. Then, there exists a host graph G of R such that $\boldsymbol{s}$ is consistent with the only MIS of G, i.e., removing $\boldsymbol{s}$ from the search space will result in a smaller MIS size. One such G can be constructed by adding N mutually independent vertices to each $v \in \partial R$ when $s_v = 0$, as shown below.



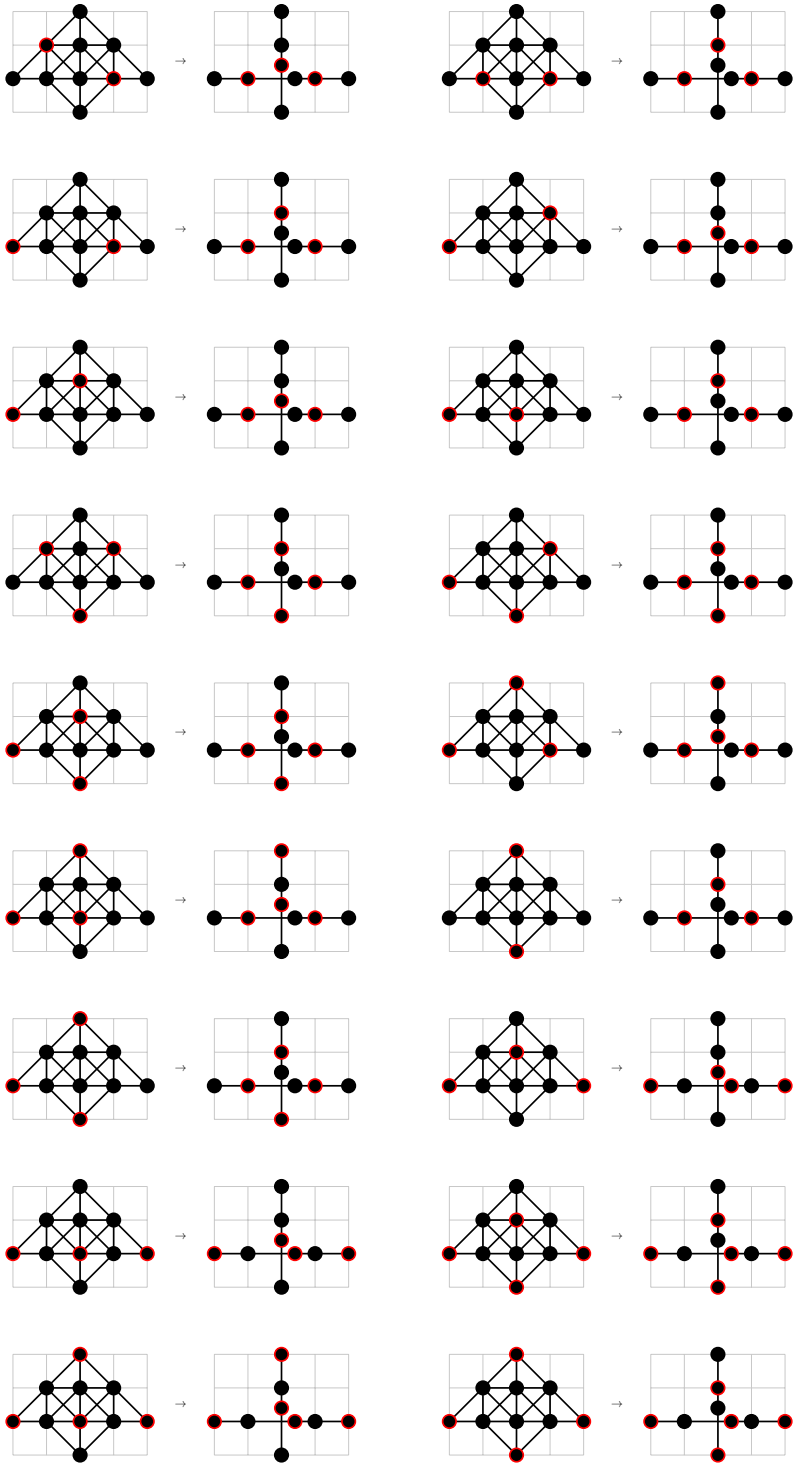
The MIS size of G with the boundary-vertex configuration $\boldsymbol{s}$ is

$$\alpha(G, \boldsymbol{s}) = \tilde{\alpha}(R)_{\boldsymbol{s}} + N\left(|\partial R| - \sum_{v \in \partial R} \boldsymbol{s}_v\right), \tag{A.1}$$

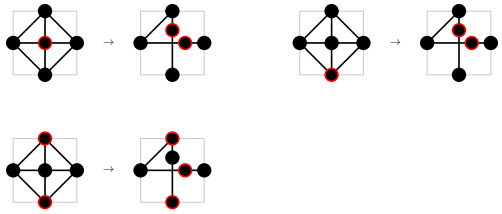
which is a summation of contributions from R and the remaining part of G. In the case that $N > |V_R|$, where $|V_R|$ is the number of vertices in R, we show by contradiction that $\boldsymbol{s}$ is the only boundary-vertex configuration that yields the MISs of G. Assume there exists another boundary-vertex configuration $\boldsymbol{t}$ that produces the same or bigger MIS size, i.e., $\alpha(G, \boldsymbol{t}) \geq \alpha(G, \boldsymbol{s})$. Then, $\boldsymbol{t} \prec \boldsymbol{s}$ must hold, otherwise the MIS size will decrease for at least $N - \tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{s}} > 0$. It implies $\tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{t}} < \tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{s}}$ must hold, otherwise $\boldsymbol{s}$ must be removed by the definition of reduced α-tensor. Therefore, we have $\alpha(G, \boldsymbol{t}) = \tilde{\boldsymbol{\alpha}}(R)_{\boldsymbol{t}} + N(|\partial R| - \sum_{v \in \partial R} \boldsymbol{s}_v) < \alpha(G, \boldsymbol{s})$, which contradicts with the assumption. Hence, by contradiction, $\boldsymbol{s}$ is the only boundary-vertex configuration that is consistent with the only MIS of G, proving the second half of the theorem.

Appendix B. Extracting results. We present the rules to extract the MIS for the source graph as the following table. On the left side of the "$\rightarrow$" symbol, we specify a possible gadget configurations in the MIS of a target graph, and on the right side, we specify a possible replacement.

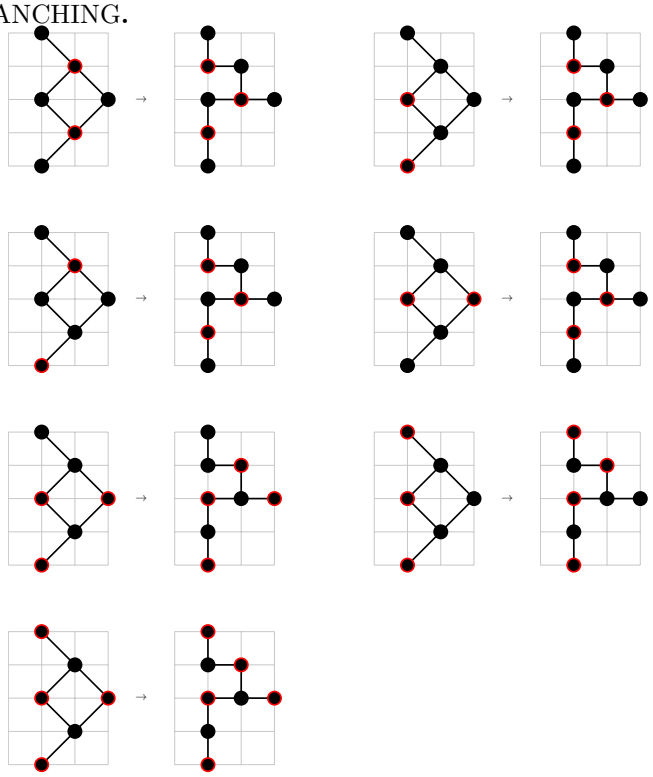
B.1. BATOIDEA **in Equation (3.1).** In order to adapt the gadget to the grid, we transformed the original gadget slightly. We first add two (four) vertices on the vertical (horizontal) line of the pattern graph, which is equivalent to applying the copy gadget. Then, we remove one dangling vertex from both sides.



B.2. PIRAMID **in Equation (3.2).**



B.3. BRANCHING.



Appendix C. Exhaustive search of unweighted MIS gadgets. In this appendix, we show how to search for unweighted MIS gadgets by efficiently enumerating relevant graphs at a certain size. We will focus our discussion on finding a replacement graph to rewrite a crossing graph, CROSS. The searching algorithm is listed in Algorithm C.1, where for each candidate R' we check if it is a valid replacement for CROSS by the following criteria:

1. The reduced α-tensors of CROSS and R' differ only by a constant, i.e., `is_diff_by_constant`$(\tilde{\boldsymbol{\alpha}}(R'), \tilde{\boldsymbol{\alpha}}(\text{CROSS}))$ returns `true` (Theorem 3.7).
2. The replacement graph R' is a unit disk graph, which can be tested by the function `has_unit_disk_embedding` (Appendix C.2).

The function to compute reduced α-tensors is `compute_reduced_alpha_tensor`, which can be implemented with the generic tensor network [16, 15] method. The most challenging part of this algorithm is to enumerate the graph space efficiently. To find the gadget (CROSS, BATOIDEA) that introduced in Subsection 3.2, the graph space

Algorithm C.1 Crossing Gadget Searching

Let CROSS be the pattern graph and $\tilde{\boldsymbol{\alpha}}(\text{CROSS})$ be its reduced α-tensor.
Let n be the size of the replacement graph to be searched.
foreach *non-isomorphic graph* $G = (V, E)$ *with* $|V| = n$ **do**
  foreach *choice of boundary* $\partial R'$ **do**
    $R' \leftarrow (V, E, \partial R')$
    if *pass_filtering_rules*(R') **then**
      $\tilde{\boldsymbol{\alpha}}(R') \leftarrow$ `compute_reduced_alpha_tensor`$(R', \partial R')$
      if *is_diff_by_constant*$(\tilde{\boldsymbol{\alpha}}(R'), \tilde{\boldsymbol{\alpha}}(\text{CROSS}))$ **then**
        if *has_unit_disk_embedding*(R') **then**
          return R'
        end
      end
    end
  end
end

up to size $|V| = 11$ must be searched, which can be as large as $2^{55} \approx 3.6 \times 10^{16}$. For each graph instance, there are $\sim |V|^4$ different choices of boundary vertices, which is beyond the capability of a classical computer. In the following, we focus the discussion on how to reduce the graph space to search by using symmetries. The first trick is to only search for non-isomorphic graphs [19]. The table of non-isomorphic graphs may be found in http://users.cecs.anu.edu.au/ bdm/data/graphs.html. For graph size $|V| = 11$, there are 1018997864 non-isomorphic graphs in total. To further take the advantage of problem symmetries, we designed the function `pass_filtering_rules` to filter out redundant graphs. This function returns `false` if any of the following conditions is met:

1. The boundary vertices either $(1, 3)$ or $(2, 4)$ is directly connected.
2. The set of boundary vertices is related by problem symmetries with an instance already in the search list. For CROSS, the reduced α-tensor is symmetric under exchanges $1 \leftrightarrow 3$, $2 \leftrightarrow 4$ and $(1, 3) \leftrightarrow (2, 4)$.
3. The crossing criteria in Appendix C.1 is not meet.

Graphs up to size $|V| = 11$ are searched on a 72-core Amazon Web Service (AWS) EC2 machine in less than one day. Four valid replacement graphs for CROSS are found, which are listed in Figure 6.

C.1. unit disk crossing criteria for CROSS**.** There are some easy-to-test necessary conditions to tell if a graph can be a unit disk replacement graph for CROSS even before we compute its reduced α-tensor. Let $G = (V, E)$ be a unit disk replacement graph for CROSS and A, B, C and D be the four boundary vertices of G. The edges (A, D) and (B, C) are the two edges that cross each other. For G to be a valid unit-disk replacement graph of CROSS, all paths $\pi_{\text{AD}} \in \text{Paths}(A, D)$ must intersect with all paths $\pi_{\text{BC}} \in \text{Paths}(B, C)$. Then, we have the following lemma.

LEMMA C.1. *Let* π_{AD} *and* π_{BC} *be two paths in a unit disk graph* $G = (V, E)$. *They intersect each other only if at least one of the following statements is true:*

1. *there exists a vertex* v *such that* $v \in \pi_{\text{AD}}$ *and* $v \in \pi_{\text{BC}}$,
2. *there exist a vertex* $w \in \pi_{\text{AD}}$ *and an edge* $(u, v) \in \pi_{BC}$, *both* (w, u) *and* (w, v) *are in* E.
3. *there exist a vertex* $w \in \pi_{\text{BC}}$ *and an edge* $(u, v) \in \pi_{AD}$, *both* (w, u) *and* (w, v) *are in* E.

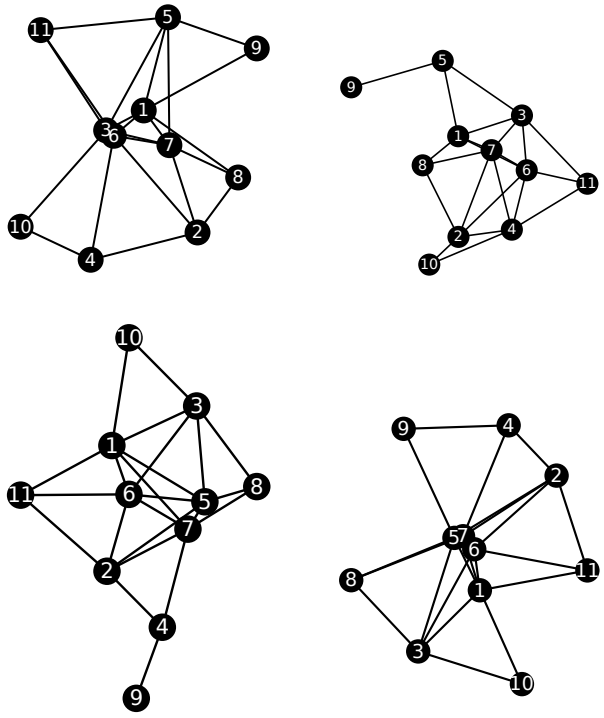
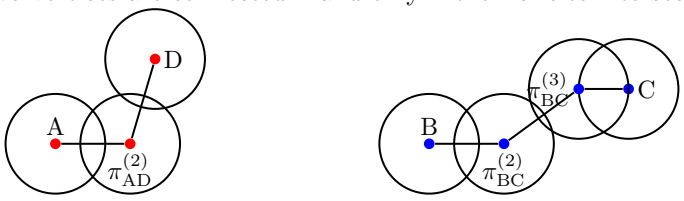


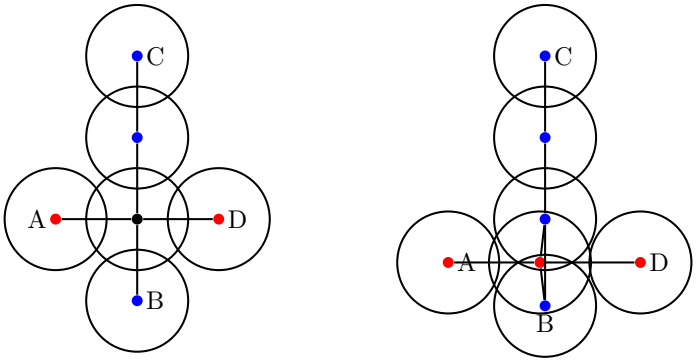
Fig. 6: The four valid unit disk replacement graphs for CROSS. All of them has graph size $|V| = 11$, among which the top right graph (BATOIDEA) can be embedded onto a square grid.

Proof. This lemma is evident if we draw the unit disk graph in the plane. Given a graph, we draw a circle of radius 0.5 around each vertex. By definition of a unit disk graph, two vertices are connected if and only if their circles intersect.



(a) a path from A to D. (b) a path from B to C.

Now we consider π_{AD} and π_{BC} cross each other. Their circles must intersect at least once, because the circles in the same path form a connected region in the two dimensional plane and the two regions must intersect when the two paths cross each other. As illustrated below, the way to cross paths π_{AD} and π_{BC} must be one of the following three cases. One is vertex sharing as shown in (a) and the other is having a vertex in one path connected to an edge in another path as shown in (b).



(a) sharing a vertex. (b) not sharing vertices.

The first case is covered by the first statement in the lemma and the second case are covered by the second and third statements in the lemma respectively. This completes the proof for the lemma. □

C.2. Unit disk embedding. Proving if a general graph has a unit disk embedding is NP-hard [3]. In our case, the graph size is small enough such that we can use a variational approach to find the unit disk embedding of a graph. Let $G = (V, E)$ be the graph to be tested, we define the following loss function

$$\begin{aligned}\mathcal{L}(\boldsymbol{x}) = &\sum_{(i,j)\in E(G)} \texttt{relu}(\|x_i - x_j\|^2 - 0.99) \\ &+ \sum_{(i,j)\notin E(G)} \texttt{relu}(1.01 - \|x_i - x_j\|^2),\end{aligned} \tag{C.1}$$

and variationally optimize this loss function using the automatic differentiation [11, 17] technique. Here, variational parameters $\boldsymbol{x}$ are vertex coordinates and `relu` is the rectified linear unit function

$$\texttt{relu}(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0. \end{cases} \tag{C.2}$$

This loss function is zero only if the graph G is a unit disk graph. To avoid numerical instability, we use 0.99 and 1.01 instead of 1 in the loss function. The optimizer is L-BFGS [5] and we fix one of the vertex coordinates so that we have $2|V| - 2$ free variational parameters in total. We initialize the variational parameters with random numbers and repeat the optimization for 100 times to avoid local minima. With this approach, we can reliably find a unit disk embedding for a graph with up to $|V| = 11$ vertices in a few milliseconds.