Electronic Colloquium on Computational Complexity, Report No. 26 (2023)

Hardness Self-Amplification: Simplified, Optimized, and Unified

Shuichi Hirahara* Nobutaka Shimizu†

Abstract

Strong (resp. weak) average-case hardness refers to the properties of a computational problem in which a large (resp. small) fraction of instances are hard to solve. We develop a general framework for proving hardness self-amplification, that is, the equivalence between strong and weak average-case hardness. Using this framework, we prove hardness self-amplification for popular problems, such as matrix multiplication, online matrix-vector multiplication, triangle counting of Erdős–Rényi random graphs, and the planted clique problem. As a corollary, we obtain the first search-to-decision reduction for the planted clique problem in a high-error regime. Our framework simplifies, improves, and unifies the previous hardness self-amplification results.

Our approach uses a one-query upward self-reduction, that is, a reduction that maps a small instance to a large instance. We demonstrate that this reduction yields hardness self-amplification if the bipartite graph, whose left and right vertices correspond to small and large instances, respectively, has an expansion property. Our key technical contribution is to show the expansion property of the bipartite graph naturally constructed from the planted clique problem by using the coupling method of Markov chains.

*National Institute of Informatics Email: s_hirahara@nii.ac.jp
†Tokyo Institute of Technology Email: shimizu.n.ah@m.titech.ac.jp

ISSN 1433-8092

Contents

1 Introduction

The theory of average-case complexity aims to analyze what fraction of instances of a problem can (or cannot) be solved efficiently. Depending on the fraction of hard instances, we obtain two different notions of average-case complexity. We say that a function f is *δ-weakly average-case hard* if any efficient algorithm fails to compute f on a δ-fraction of inputs for a small parameter $\delta > 0$. We say that a function $f\colon \{0,1\}^n \to \{0,1\}^m$ is *ϵ-strongly average-case hard* if no efficient algorithm can compute f on more than a $(2^{-m} + \epsilon)$-fraction of inputs for a small parameter $\epsilon > 0$. Note that there is a trivial algorithm that computes f on a 2^{-m}-fraction of inputs by outputting a uniformly random element of $\{0,1\}^m$; thus, the strong average-case hardness indicates that no efficient algorithm can compute f significantly better than random guessing.

Strong average-case hardness has had fundamental impacts on the construction of cryptographic primitives and derandomization. To transform a weakly average-case hard problem f to a strongly average-case hard problem g, proof techniques, known as *hardness amplification*, have been developed. For instance, the direct product theorem [Yao82; GNW11] states that for any function $f\colon \{0,1\}^n \to \{0,1\}$, the k-wise direct product $g := f^k\colon (\{0,1\}^n)^k \to \{0,1\}^k$ defined as $f^k(x_1, \dots, x_k) = (f(x_1), \dots, f(x_k))$ is strongly average-case hard if f is weakly average-case hard. However, the resulting function g is often highly artificial, as the main concern in the traditional line of research of hardness amplification is to obtain better trade-offs of parameters (e.g., [Yao82; Imp95; IW97; ODo04; HVV06; IJK09b; IJK09a; IJKW10; GNW11]).

Recently, two sets of authors developed two different frameworks for showing *hardness self-amplification* for natural problems f, i.e., the equivalence between the weak average-case hardness of f and the strong average-case hardness of f.

1. Asadi, Golovnev, Gur, and Shinkar [AGGS22] considered the matrix multiplication, i.e., the function $\mathsf{Mult}\colon \left(\mathbb{F}_p^{n\times n}\right)^2 \to \mathbb{F}_p^{n\times n}$ that takes two matrices $A, B \in \mathbb{F}_p^{n\times n}$ and outputs $A \cdot B$. They showed that if Mult can be solved by an algorithm running in time $T(n)$ on an ϵ-fraction of inputs, then there exists a randomized $2^{O(\log^5(1/\epsilon))} \cdot T(n)$-time algorithm that solves Mult on every input.[1] (The equivalence between the worst-case hardness and weak average-case hardness of Mult is due to Blum, Luby, and Rubinfeld [BLR93].)

2. Hirahara and Shimizu [HS22] considered the problem $\oplus\mathsf{TriangleTripartite}$ of computing the parity of the number of triangles in a random tripartite graph, and showed that ϵ-strong average-case hardness of $\oplus\mathsf{TriangleTripartite}$ is equivalent to δ-weak average-case hardness of $\oplus\mathsf{TriangleTripartite}$ for any constants $\epsilon, \delta > 0$. More generally, they proved that if there exists a circuit of size s that computes $\oplus\mathsf{TriangleTripartite}$ on a $(1/2 + \epsilon)$-fraction of inputs, then there exists a circuit of size s' that computes $\oplus\mathsf{TriangleTripartite}$ on a $(1 - \delta)$-fraction of inputs, where $s' = 2^{-\mathrm{poly}(1/\epsilon, 1/\delta)} \cdot s$. (In particular, by the equivalence between the worst-case and weak average-case hardness [BBB21; Gol20][2], the strong average-case hardness and the worst-case hardness are also equivalent.)

Their proof techniques are very different. On one hand, the framework of [AGGS22] is based on mathematically deep results of additive combinatorics, and relies on the probabilistic version of the quasi-polynomial Bogolyubov–Ruzsa lemma [San12]. The quasi-polynomial overhead $2^{O(\log^5(1/\epsilon))}$ of the reduction can be improved to $\mathrm{poly}(1/\epsilon)$ if the (probabilistic version of) polynomial Bogolyubov–Ruzsa conjecture holds — a major open problem in additive combinatorics. On the other hand, the

[1] When the size p of the field is large, their algorithm runs in time $O((1/\epsilon)^4 \cdot T(n))$.

[2] Although [BBB21; Gol20] focused on Erdős–Rényi graph as the input distribution, their techniques also work for the random tripartite graph.

framework of [HS22] is based on the generalization of the hard-core lemma of Impagliazzo [Imp95], which is one of the most fundamental tools in the literature of hardness amplification.

This state of affairs raises a couple of natural questions. Can we improve the quasi-polynomial overhead of the reduction of [AGGS22] without resolving the polynomial Bogolyubov–Ruzsa conjecture? Can we improve the exponential overhead of [HS22] to a polynomial overhead? Is there a unified framework that enables us to show the hardness self-amplification results for both Mult and $\oplus$TriangleTripartite? Can we prove hardness self-amplification for other popular problems, such as the planted clique problem?

In this paper, we answer all the questions affirmatively. We develop a general framework that improves and unifies the previous hardness self-amplification results of [AGGS22; HS22]. Our proofs are substantially simpler than the previous ones. Moreover, our framework enables us to show the hardness self-amplification theorem for the planted clique problem and gives the first search-to-decision reduction in a high-error regime. We proceed to describe our results in detail.

1.1 Our Results

Our new framework enables us to establish hardness self-amplification theorems for matrix multiplication, online matrix-vector multiplication, triangle counting of Erdős–Rényi random graphs (not only for *tripartite* random graphs), and the planted clique problem. We believe that our framework is versatile and will be used to show hardness self-amplification for other problems in the future. Here, we focus on the four specific problems and, for each problem, we explain its context and our results in the sequel.

1.1.1 Planted Clique

The planted clique problem is one of the most popular average-case problems. To introduce the problem, let $\mathcal{G}_{n,1/2}$ be the distribution of the Erdős–Rényi graph, namely, the distribution of an n-vertex random graph where each possible edge is placed independently with probability $1/2$. For a parameter $k = k(n)$, let $\mathcal{G}_{n,1/2,k}$ be the distribution of an n-vertex graph obtained by first sampling a graph according to $\mathcal{G}_{n,1/2}$, selecting a uniformly random k subset from n vertices, and thereafter placing the k-clique in the selected k vertices. In the decision version of the planted clique problem, an algorithm is given a graph G from either $G \sim \mathcal{G}_{n,1/2}$ or $G \sim \mathcal{G}_{n,1/2,k}$ and is asked to decide whether G comes from $\mathcal{G}_{n,1/2}$ or $\mathcal{G}_{n,1/2,k}$. The search version of the planted clique problem asks to find the planted clique of size k, given a random graph $G \sim \mathcal{G}_{n,1/2,k}$ as input.

The *Planted Clique Conjecture* [Jer92; Kuč95] postulates that there is no polynomial-time algorithm that solves (some version of) the planted clique problem on n-vertex graphs when $2\log n \ll k \ll \sqrt{n}$. This conjecture serves as a key hardness hypothesis in a broad range of fields, including cryptography [JP00; ABW10], the densest subgraph problem [HWX15; MRS21], approximating a Nash equilibrium [HK11], distribution testing [AAKMRX07], submatrix detection [MW15], compressed sensing [KZ14], and the sparse principal component analysis (PCA) [BR13; BBH18].

Note that "the" Planted Clique Conjecture is, in fact, not a single conjecture; rather, it should be considered as a *family of conjectures*. For each variant of planted clique problems, one may consider corresponding hardness conjectures. For example, Hazan and Krauthgamer [HK11] used a search version of the planted clique conjecture to give evidence that there exists no polynomial-time algorithm that finds the best Nash equilibrium of a two-player game. Brennan, Bresler, and Huleihel [BBH18] used a decision version of the planted clique conjecture to obtain evidence of the intractability of numerous average-case problems. Moreover, we obtain different versions

of planted clique conjectures depending on the success probability of average-case algorithms. For example, the strong average-case hardness of planted clique problems is desirable for a cryptographic purpose [JP00]. It is evident that strong average-case hardness implies weak average-case hardness, and that the decision version of the planted clique problem reduces to its search version. The converse was proved in a low-error regime by Alon, Andoni, Kaufman, Matulef, Rubinfeld, and Xie [AAKMRX07]. Specifically, they showed that if the decision variant of the planted clique problem can be efficiently solved on a $(1-1/n^3)$-fraction of n-vertex graphs, then the search version can also be solved efficiently. Apart from this search-to-decision reduction, the relationship among different versions of the planted clique conjecture is not well understood.

We prove that decision and search versions of the planted clique problem admit hardness self-amplification, thereby showing that the planted clique conjecture is robust. In particular, we obtain the first search-to-decision reduction for the planted clique problem in a high-error regime. This is given by combining the search-to-decision reduction of [AAKMRX07] with the following decision-to-decision hardness self-amplification for the planted clique problem.

Theorem 1.1. *For any constants $\delta, \epsilon > 0$, there exists a constant $c = c(\delta, \epsilon)$ that satisfies the following. Let $k = k(n) \geq 3 \log n$. If there exists a polynomial-time algorithm $\mathcal{M}$ that satisfies*

$$\Pr_{G \sim \mathcal{G}_{cn,1/2,k}}[\mathcal{M}(G) = 1] - \Pr_{G \sim \mathcal{G}_{cn,1/2}}[\mathcal{M}(G) = 1] \geq \epsilon,$$

then, there exists a polynomial-time randomized algorithm $\mathcal{M}'$ that satisfies

$$\Pr_{G \sim \mathcal{G}_{n,1/2,k}}[\mathcal{M}'(G) = 1] - \Pr_{G \sim \mathcal{G}_{n,1/2}}[\mathcal{M}'(G) = 1] \geq 1 - \delta.$$

In other words, if it is weakly average-case hard to detect a k-clique in an n-vertex random graph, then it is strongly average-case hard to detect a k-clique in an N-vertex graph, where $N = O(n)$.

Theorem 1.1 makes the planted clique conjecture robust in the sense that different versions of planted clique conjectures can be unified. For example, the hypothesis of [BR13, Hypothesis BPC] states that for any $\epsilon > 0$, for any randomized polynomial-time algorithm $\mathcal{M}$ and $k < n^{1/2-\epsilon}$,

$$\Pr_{G \sim \mathcal{G}_{n,1/2,k}}[\mathcal{M}(G) = 1] - \Pr_{G \sim \mathcal{G}_{n,1/2}}[\mathcal{M}(G) = 1] < 1 - \delta, \tag{Hyp_δ}$$

where $\delta \in (0,1)$ is a parameter. Theorem 1.1 shows that for any constant $\delta \in (0,1)$, (Hyp_δ) is equivalent to the hypothesis of [MW15; GMZ17] that states

$$\Pr_{G \sim \mathcal{G}_{n,1/2,k}}[\mathcal{M}(G) = 1] - \Pr_{G \sim \mathcal{G}_{n,1/2}}[\mathcal{M}(G) = 1] \leq \frac{1}{3}.$$

For simplicity, Theorem 1.1 is stated only for constant error parameters; however, it can be extended to a high-error regime of $\delta(n), \epsilon(n) = o(1)$, at the cost of increasing N. By combining this hardness self-amplification with the search-to-decision reduction of [AAKMRX07], we obtain a search-to-decision reduction in a high-error regime, that is, even if the decision algorithm distinguishes $\mathcal{G}_{n,1/2,k}$ from $\mathcal{G}_{n,1/2}$ with small advantage $\epsilon(n)$.

Theorem 1.2. *Let $\epsilon = \epsilon(n) = n^{-1/2+c_0}$ be a function for a constant $c_0 > 0$. Let c be any constant that satisfies $c > \frac{3}{2c_0}$ and $k = k(n)$ be any function that satisfies $k \geq 108c \log n$. Suppose there exists a polynomial-time algorithm $\mathcal{M}$ such that for every n and $k' \geq k(n)/3$,*

$$\Pr_{G \sim \mathcal{G}_{n^c,1/2,k'}}[\mathcal{M}(G) = 1] - \Pr_{G \sim \mathcal{G}_{n^c,1/2}}[\mathcal{M}(G) = 1] \geq \epsilon(n^c).$$

Then, there exists a randomized polynomial-time algorithm $\mathcal{M}'$ that satisfies

$$\Pr_{G\sim\mathcal{G}_{n,1/2,k}}[\mathcal{M}'(G) \text{ outputs a } k\text{-clique in } G] \geq 1 - O(1/n).$$

For example, for $c_0 = 1/4$, this search-to-decision reduction shows that if it is weakly average-case hard to find a clique of size $k(n) = 648 \log n$ on n-vertex random graphs, then it is strongly average-case hard to distinguish $\mathcal{G}_{n,1/2}$ from $\mathcal{G}_{n,1/2,k'}$ with advantage $\epsilon(n) = n^{-1/4}$ for some $k' \geq 3 \log n$.

1.1.2 Triangle Counting

Subgraph counting is a fundamental task of graph algorithms, and its average-case complexity has been recently investigated actively [GR18; BBB21; DLW20; Gol20; HS21; HS22]. Boix-Adserà, Brennan, and Bresler [BBB21] studied counting k-clique in an Erdős–Rényi graph and presented the following worst-case to average-case reductions in a low-error regime.

Theorem 1.3 (Informal; [BBB21])**.** *Let $0 < p < 1$ be a constant.*

1. *Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ that counts k-clique for an $(1 - 1/\operatorname{polylog}(n))$-fraction of $G \sim \mathcal{G}_{n,p}$. Then, there exists a $T(n) \cdot \operatorname{polylog}(n)$-time randomized algorithm $\mathcal{M}'$ that counts k-clique for every G.*

2. *Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ that computes the parity of the number of k-clique for a $(1 - c)$-fraction of $G \sim \mathcal{G}_{n,p}$, where $c = c(k, p)$ is some constant. Then, there exists an $O(T(n))$-time randomized algorithm $\mathcal{M}'$ that computes the parity of the number of k-clique for every G.*

Whether the reductions of Theorem 1.3 can be extended to a high-error regime has received significant attention, and partial progress has been made in [GR18; Gol20; HS21; HS22]. However, even in the special case of $k = 3$, whether the strong average-case hardness of computing the parity of the number of triangles is equivalent to its worst-case hardness is open. The aforementioned work of [HS22] comes close to resolving this, but slightly falls short of it: The input distribution of the problem $\oplus\mathsf{TriangleTripartite}$, for which they proved the hardness self-amplification, is the *tripartite* variant of the Erdős–Rényi random graph.

Using our framework, we extend the result of [HS22] to the (standard) Erdős–Rényi random graph, thereby resolving the open question for nonuniform algorithms. Moreover, we improve the exponential overhead $2^{(1/\epsilon)^{O(1)}}$ to a polynomial $(1/\epsilon)^{O(1)}$ overhead. Let $\#\mathsf{Triangle}(G)$ (resp. $\oplus\mathsf{Triangle}(G)$) denote the function that maps a graph G to the number (resp. parity) of triangles contained in G.

Theorem 1.4. *Let $p > 0$ be a constant. For any $\epsilon > 0$, there exists $k = O(\epsilon^{-2} \log(1/\epsilon))$ that satisfies the following: If there exists a $T(kn)$-time algorithm $\mathcal{M}$ that computes $\oplus\mathsf{Triangle}(G)$ for a $(1/2 + \epsilon)$-fraction of $G \sim \mathcal{G}_{kn,p}$, then, there exists an $O(T(kn) \cdot \epsilon^{-2} \log(1/\epsilon))$-time nonuniform algorithm that computes $\oplus\mathsf{Triangle}(G)$ for any n-vertex G.*

Shaltiel and Viola [SV10] showed a lower bound ϵ^{-2} on the query complexity of any standard proof of hardness amplification of a Boolean function to ϵ-strong average-case hardness. In other words, the overhead ϵ^{-2} of the running time in Theorem 1.5 is indispensable; thus, our hardness self-amplification achieves the minimum overhead up to a factor of $O(\log(1/\epsilon))$.

We also obtain a hardness self-amplification for counting the number of triangles in an Erdős–Rényi random graph for *errorless* algorithms [BT06a]. We say that an algorithm $\mathcal{M}$ is errorless

for a function f if $\mathcal{M}(x) \in \{f(x), \perp\}$. In other words, an errorless algorithm outputs either the correct value $f(x)$ or a special symbol $\perp$ to indicate "I don't know". Combining our hardness self-amplification with Item 1 of Theorem 1.3, we obtain the equivalence between the worst-case hardness of counting the number of triangles in an Erdős–Rényi random graph and its strong errorless average-case hardness.

Theorem 1.5. *Let $p > 0$ be a constant. Suppose there exists a $T(n)$-time errorless algorithm $\mathcal{M}$ that computes $\#\mathsf{Triangle}(G)$ for an ϵ-fraction of $G \sim \mathcal{G}_{n,p}$. Then, for some $k = \operatorname{polylog}(n)$, there exists an $O(T(kn) \cdot \epsilon^{-1} \operatorname{polylog}(n))$-time nonuniform errorless algorithm that computes $\#\mathsf{Triangle}(G)$ for any n-vertex graph G.*

1.1.3 Matrix Multiplication

Matrix multiplication is a fundamental computational task. A state-of-the-art worst-case algorithm multiplies two $n \times n$ matrices in time $O(n^\omega)$ with $\omega < 2.3728596$ [AW21]. We consider the average-case complexity of the matrix multiplication over the uniform distribution. Using an unexpectedly simple proof technique, we significantly improve the previous worst-case to average-case reduction of [AGGS22].

Theorem 1.6. *Let R be a finite ring. Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ that computes AB for an ϵ-fraction of $A, B \sim R^{n \times n}$. Then, there exists a randomized $O(T(n) \cdot \epsilon^{-1} \operatorname{polylog}(1/\epsilon))$-time algorithm $\mathcal{M}'$ that computes AB for any $A, B \in R^{n \times n}$.*

This result improves the quasi-polynomial overhead $2^{O(\log^5(1/\epsilon))}$ of [AGGS22] to a nearly linear $\widetilde{O}(1/\epsilon)$, without resolving the polynomial Bogolyubov–Ruzsa conjecture. Moreover, our reduction works for matrices over a finite ring, while [AGGS22] works for matrices over a finite field because their proof relies on the Gaussian elimination. Surprisingly, we achieve this by using the k-wise direct product theorem in a black-box way — one of the most fundamental and standard proof techniques in the literature of hardness amplification. We present a proof sketch of Theorem 1.6 in Section 2.1.

1.1.4 Online Matrix-Vector Multiplication

In the *online matrix-vector multiplication* problem (OMv), the goal is to construct a data structure π from a given matrix M so that Mv can be computed efficiently on a given query vector v using the preprocessed data π. The *OMv Conjecture* [HKNS15] states that any data structure on an $n \times n$ matrix over the Boolean semiring cannot answer a query in time $O(n^{2-c})$ for any constant $c > 0$. This conjecture serves as a hypothesis that implies the (worst-case) fine-grained hardness of numerous dynamic problems [HKNS15; BKS17; LMNT15].

Recent progress has been made on the average-case complexity of OMv [AGGS22; HLS22; HS22], where the input matrix M and query vector v are uniformly random (we refer to Section 6 for the formal framework). Based on the idea of [BLR93], Henzinger, Lincoln, and Saha [HLS22] presented a worst-case to average-case reduction for OMv over a finite ring in a low-error regime. Asadi, Golovnev, Gur, and Shinkar [AGGS22] presented a reduction for OMv over a finite field that works for a high-error regime, where the running time overhead is $2^{O(\log^5(1/\epsilon))}$. Hirahara and Shimizu [HS22] obtained a hardness self-amplification result for a slightly different variant known as *online vector-matrix-vector multiplication* over $\mathbb{F}_2$ [HKNS15]. We improve the quasi-polynomial overhead $2^{O(\log^5(1/\epsilon))}$ of [AGGS22] and present a worst-case to average-case reduction for OMv over a finite ring R in a high-error regime.

Theorem 1.7 (Informal)**.** *Suppose there exists a data structure algorithm $\mathcal{M}$ with query time $T(n)$ that succeeds on an ϵ-fraction of $M \sim R^{n\times n}$ and $v \sim R^n$. Then, there exists a randomized data structure algorithm $\mathcal{M}'$ with query time $T(n) \cdot \mathrm{poly}(1/\epsilon)$ that succeeds on any $M \in R^{n\times n}, v \in R^n$.*

1.2 A General Framework for Hardness Self-Amplification

Our general framework is based on the expansion property of a "query graph" — a graph constructed from a one-query reduction. Consider a randomized one-query reduction $\mathcal{R}$ from a problem f to another problem g. Given an instance x of f, the reduction $\mathcal{R}$ produces a query q, obtains the answer $g(q)$ from the oracle, and outputs $f(x)$ with high probability over the internal randomness of $\mathcal{R}$. Such a one-query reduction naturally induces the *query graph* as follows: The query graph $\mathcal{G}$ of a reduction $\mathcal{R}$ is a weighted bipartite graph whose left vertex set consists of the instances of f and right vertex set consists of the instances of g. An edge (x, q) with weight p is placed in $\mathcal{G}$ if the reduction $\mathcal{R}$ makes the query q on input x with probability p.

For example, in order to show hardness self-amplification for the decision version of the planted clique problem f, we consider a simple *upward self-reduction* for f — a one-query reduction $\mathcal{R}$ from f to f itself that takes a graph G of n vertices as input and queries a larger graph G' of N vertices, where $N \gg n$. Here, the graph G' is constructed by embedding G into an N-vertex Erdős–Rényi random graph at a random position. We require an average-case solver that succeeds with low success probability $\frac{1}{2} + \epsilon$ to solve the planted clique problem on the large graph G'. Intuitively, since G' is larger than the original instance G, solving the planted clique problem on G' becomes "more average-case hard." This is a high-level idea of how to amplify weak avergae-case hardness of f to strong average-case hardness of f. We make this idea formal when the query graph has a good expansion property, which is encapsulated in the following.

Theorem 1.8 (A special case of Theorem 4.1)**.** *Let $\mathcal{R}$ be a one-query randomized reduction from f to g whose query graph is a λ-expander. If $\mathcal{M}$ is a randomized algorithm for g that succeeds on a $(1/2 + \epsilon)$-fraction of inputs and $\lambda \le \delta\epsilon^2/100$, then*

$$\Pr_x\left[\Pr_{\mathcal{R}}\left[\mathcal{R}^{\mathcal{M}}(x) = f(x)\right] \ge \frac{1}{2} + \frac{\epsilon}{2}\right] \ge 1 - \delta.$$

Here, we say that a bipartite graph is a λ-expander if the second-largest singular value of its edge-weight matrix is bounded by λ. Theorem 1.8 shows that for a $(1-\delta)$-fraction of instances x of f, the majority vote of the output of the reduction $\mathcal{R}^{\mathcal{A}}(x)$ is the correct output $f(x)$, under the assumption that λ is sufficiently small. The most technical part of this paper is to show the expansion property of the query graph of the reduction for the planted clique problem. We present a proof sketch in Section 2.4.

2 Techniques and Proof Overview

2.1 Matrix Multiplication and Direct Product Theorem

We start with a simple error-tolerant worst-case to average-case reduction for matrix multiplication (Theorem 1.6). This hardness self-amplification can be explained without using a general framework. Let $\mathcal{M}$ be an algorithm that computes AB for an ϵ-fraction of pairs $(A, B) \sim (R^{n\times n})^2$, where $\epsilon > 0$ is a small parameter. Our goal is to construct an efficient algorithm that computes AB for every A and B. The approach is to use the k-wise direct product theorem in a black-box way.

Let $k = O(\log(1/\epsilon))$ and let $d = n/k$. For simplicity, we assume that d is an integer. We partition given matrices A and B into submatrices $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$ as shown in Figure 1.

Specifically, $A_i \in R^{d\times n}$ is the submatrix of A consists of the $((i-1)d+1)$-th to id-th row vectors of A and $B_j \in R^{n\times d}$ is the submatrix of B consists of the $((j-1)d+1)$-th to jd-th column vectors of B. Then, the product AB can be divided into $k \times k$ blocks such that each (i,j)-block is equal to A_iB_j.

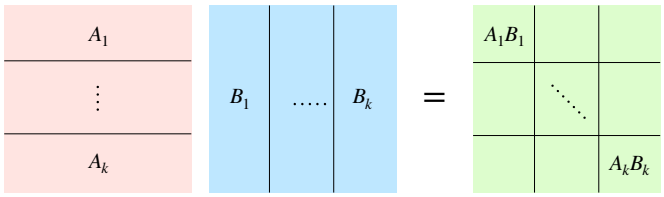

Figure 1: $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$.

Suppose $\mathcal{M}(A, B)$ correctly computes AB. Then, by looking at the diagonal blocks, we obtain A_iB_i for all $i = 1, \ldots, k$. Moreover, if $A, B \sim R^{n\times n}$, then the k pairs $(A_1, B_1), \ldots, (A_k, B_k)$ can be regarded as k independent samples from the uniform distribution over $R^{d\times n} \times R^{n\times d}$. Therefore, using $\mathcal{M}$, we can compute the multiplication of the k pairs of matrices (A_i, B_i) for an ϵ-fraction of the k-tuple of pairs. In other words, we can compute the k-wise direct product of the rectangular matrix multiplication. Then, by the direct product theorem (and Freivalds' verification algorithm [Fre79]), we can multiply two matrices $X \in R^{d\times n}$ and $Y \in R^{n\times d}$ for a 0.99-fraction of $(X, Y) \sim R^{d\times n} \times R^{n\times d}$. This gives a hardness self-amplification for the matrix multiplication.

Combining this algorithm with the random self-reduction of Blum, Luby, and Rubinfeld [BLR93] we obtain a randomized worst-case algorithm that multiplies any two rectangular matrices, $X \in R^{d\times n}$ and $Y \in R^{n\times d}$. This algorithm can be converted into an algorithm that multiplies two square matrices by partitioning given $A, B \in R^{n\times n}$ into $A_1, \ldots, A_k$ and $B_1, \ldots, B_k$ as in Figure 1 and using the worst-case algorithm to compute all A_iB_j. The running time of the reduction is at most $O(\epsilon^{-1}\mathrm{poly}(k))$, where the $O(\epsilon^{-1})$ factor comes from the direct product theorem. Note that this dependence on ϵ^{-1} significantly improves the quasi-polynomial dependence of [AGGS22] with a much simpler proof.

Similar proof ideas can be used to show the hardness self-amplification for the online matrix-vector multiplication problem (Theorem 1.7). Details can be found in Section 6.

We mention in passing that the idea of applying the direct product theorem to show a connection between weakly and strongly average-case hardness for matrix multiplication was also used in [GC20]. They used a reduction somewhat similar to ours, and proved that the weak average-case hardness of Mult with respect to a uniformly random matrix implies the strong average-case hardness of Mult with respect to a random *block-diagonal* matrix, which is far from the uniform distribution.

2.2 General Framework: Query Graph and Sampler

To extend the described idea to a broader class of problems, we further analyze the proof of the direct product theorem, where we consider the following reduction: Given an instance x, sample k independently random instances $x_1, \ldots, x_k$ of f and set $\overline{x} = (x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_k)$ for a random $i \sim [k]$. Run the average-case solver $\mathcal{M}$ for the k-wise direct product f^k and obtain $(z_1, \ldots, z_k) := \mathcal{M}(\overline{x})$. If $f(x) = z_i$, then output z_i. (here we assume that we can efficiently check whether $f(x) = z$ for given x and z.)

In general, the reduction of the direct product theorem can be seen as a one-query randomized reduction $\mathcal{R}^{\mathcal{M}}(x)$, where $\mathcal{M}$ is supposed to be an average-case solver with success probability ϵ.

Given an input x, $\mathcal{R}^{\mathcal{M}}(x)$ produces a random query y and determines the output using $\mathcal{M}(y)$. (In the proof of the direct product theorem, $y = \bar{x}$.) Let $P(x, y)$ be the probability that $\mathcal{R}(x)$ will produce y, where the probability is over the internal randomness of $\mathcal{R}$. This results in an edge-weighted bipartite graph $G = (X, Y, W)$, where X and Y denote the input spaces of x and y, associated with input distributions $\mu \in [0,1]^X$ and $\nu \in [0,1]^Y$, respectively. For each $(x, y) \in X \times Y$, the edge weight of (x, y) is defined to be $W(x, y) := \mu(x)P(x, y)$. Note that W specifies a distribution over $X \times Y$ as $\sum_{(x,y)} W(x, y) = 1$. We refer to G as the query graph of the reduction. We always assume $\mu W = \nu$, meaning that the distribution of the random query y produced by $\mathcal{R}(x)$ for random input $x \sim \mu$ is ν.

In a traditional study on hardness amplification, Impagliazzo, Jaiswal, and Kabanets [IJK09b] and Impagliazzo, Jaiswal, Kabanets, and Wigderson [IJKW10] showed that an expansion property of the query graph yields hardness amplification results. They constructed (artificial) bipartite graphs with good expansion properties, and used these graphs to present simplified, optimized, and derandomized direct product theorems. The present paper (and its title) is largely inspired by their work. The key technical contribution of our work is to show that the proof techniques of [IJK09b; IJKW10] are applicable to natural problems by showing expansion properties of the corresponding query graphs.

We now explain the expansion property of edge-weighted bipartite graphs. Consider an edge-weighted bipartite graph $Q = (X, Y, W)$ associated with distributions $\mu \in [0,1]^X$ and $\nu \in [0,1]^Y$. We say that Q is a (δ, c)-*sampler for density* ϵ if for any ϵ-dense subset $Y' \subseteq Y$ (i.e., $\sum_{y \in Y'} \nu(y) \geq \epsilon$), a $(1-\delta)$-fraction of vertices $x_0 \sim \mu$ satisfies $\Pr[y \in Y' | x_0] \geq (1-c)\epsilon$, where $\Pr[\cdot|x_0]$ refers to the probability over $(x, y) \sim W$ conditioned on $x = x_0$.

Suppose the query graph is a (δ, c)-sampler for density ϵ and let $\mathcal{M}$ be the algorithm that succeeds on an ϵ-fraction of inputs in Y. Then, the set of instances on which $\mathcal{M}$ succeeds is ϵ-dense. By the sampler property, we have that for a $(1-\delta)$-fraction of $x \in X$, the algorithm $\mathcal{M}$ succeeds on at least $(1-c)\epsilon$-fraction of neighbors of x. Therefore, by repeatedly running $\mathcal{R}^{\mathcal{M}}(x)$ for given x, we obtain a list of values such that approximately a $(1-c)\epsilon$-fraction of them is the correct answer with high probability. If we can efficiently verify the correctness of the output, then we can identify the correct one. For example, in the case of matrix multiplication, we can use the Freivalds' verification algorithm [Fre79]. Similarly, if $\epsilon = 1/2 + \epsilon_0$ for some ϵ_0 (e.g., in decision problems, such as the parity of triangle counting), then setting $c = \epsilon_0/3$ (which implies $(1-c)\epsilon \geq 1/2 + \epsilon_0/2$) and taking the majority vote of the values in the list, we may compute the correct answer with high probability.

2.3 Triangle Counting

We now explain how to prove hardness self-amplification for the problem $\oplus\mathsf{Triangle}$ of computing the parity of the number of a random graph. Let $\mathcal{M}$ be an algorithm that computes $\oplus\mathsf{Triangle}(G)$ for a $(1/2+\epsilon)$-fraction of $G \sim \mathcal{G}_{n,p}$, where $0 < p < 1$ is a constant.

We consider the following one-query reduction $\mathcal{R}^{\mathcal{O}}(G)$: Let G be a given n-vertex graph and $t \in \mathbb{N}$ be a parameter. Sample $\overline{G} \sim \mathcal{G}_{tn,p}$ and let $V(\overline{G}) = V_1 \cup \cdots \cup V_t$ be a partition of the vertex set such that $|V_i| = n$ for each $i \in [t]$. Sample $i \sim [t]$ and replace the induced subgraph $\overline{G}[V_i]$ with G (Figure 2). Query the graph $\overline{G}$ to the oracle. We observe that the query distribution is $\mathcal{G}_{kn,p}$ if the input G is drawn from $\mathcal{G}_{n,p}$.

To complete the description of the reduction, we need to compute $\oplus\mathsf{Triangle}(G)$ from the answer $\oplus\mathsf{Triangle}(\overline{G})$ from the oracle. To this end, we use nonuniform advice in a way similar to the proof of [HS22] that $\oplus\mathsf{TriangleTripartite}$ admits a computational design. We color each edge $e \in E(\overline{G})$ red if e is contained in the embedded graph $\overline{G}[V_i] = G$, and blue otherwise (Figure 2). We say that

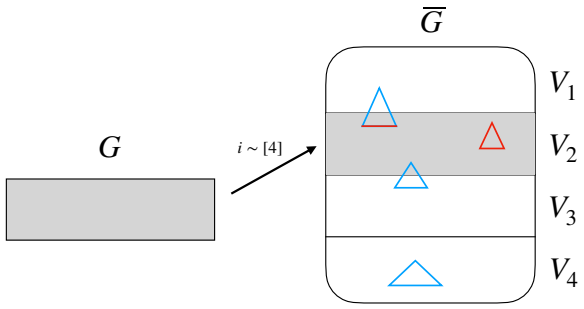

Figure 2: Reduction for triangle counting. Edges in the original graph are red and others are blue. We can compute $\oplus\mathsf{Triangle}(G)$ from $\oplus\mathsf{Triangle}(\overline{G})$ and nonuniform advice.

a triangle is red if all the three edges in the triangle are red. Our goal is to count red triangles, for which it suffices to count non-red triangles, as the parity $\oplus\mathsf{Triangle}(\overline{G})$ of the number of all the triangles is known to the reduction. Observe that any non-red triangle contains at least two blue edges. We count the blue triangles (whose edges are all blue) and the triangles that contain exactly two blue edges separately. Since blue edges are independent of the input G, we may fix these edges as nonuniform advice. Once blue edges are fixed, the number of blue triangles can be given as advice. Moreover, the number of triangles that contain exactly two blue edges can be counted by using the number of blue uv-paths of length 2 for each $u, v \in V(G)$, which can be given as an advice string of length $O(n^2)$. Therefore, we can compute $\oplus\mathsf{Triangle}(G)$ in time $O(n^2)$ from the nonuniform advice and $\oplus\mathsf{Triangle}(\overline{G})$.

Since the query $\overline{G}$ is the graph obtained by combining k independent random graphs with random edges, the expansion property of the query graph follows by the same argument with the proof of the direct product theorem given by [IJK09b; IJKW10]. See Lemma 7.6 for details.

2.4 Planted Clique

Finally, we explain how to prove the hardness self-amplification for the search version of the planted clique problem. For simplicity, we assume that parameters δ and ϵ are constants. Let $k = \omega(\log n)$ be the size of a planted clique.

We consider the following upward self-reduction that takes as input a random graph $G \sim \mathcal{G}_{n,1/2,k}$ with a planted k-clique U and queries a large graph $\overline{G}$ to the oracle. Select an Erdős–Rényi graph $\overline{G_0} \sim \mathcal{G}_{N,1/2}$ and a uniformly random injection $\phi\colon [n] \to [N]$. We define $\overline{G}$ to be the graph constructed by replacing $\overline{G_0}[\phi([n])]$ with G in $\overline{G_0}$; see Figure 3. The reduction queries the graph $\overline{G}$ to the oracle and obtains a k-clique U' in $\overline{G}$. If U' is contained in $\phi([n])$, then the reduction outputs $\phi^{-1}(U')$. It is not hard to see that if $N = n^{O(1)}$ and $k = \omega(\log n)$, then the k-clique in the graph $\overline{G}$ is very likely to be unique. This ensures the correctness of our reduction: If the oracle returns a k-clique in the N-vertex graph $\overline{G}$, then the k-clique clique is the one planted in G with high probability, i.e., $\phi^{-1}(U') = U$.

To prove the hardness self-amplification theorem, it suffices to show that the query graph of the reduction is indeed a sampler. We present two approaches for showing it.

Birthday Paradox and Pairwise Independence. Let $Q = (X, Y, W)$ be the query graph of the reduction associated with input distributions $\mu = \mathcal{G}_{n,1/2,k}$ and $\nu = \mathcal{G}_{N,1/2,k}$. Let $Q^* = (Y, X, W^\top)$ be the graph obtained by swapping X and Y in Q. It is not hard to see that if Q^* is a sampler

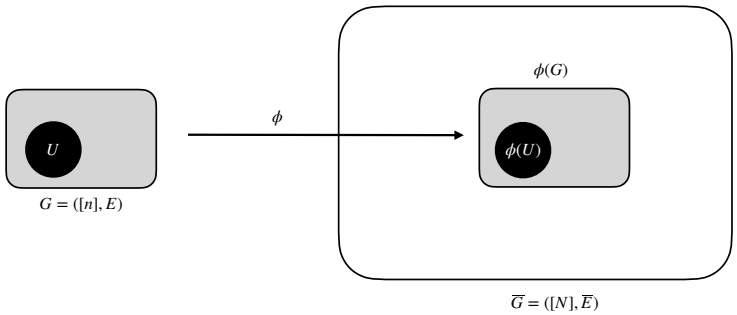

Figure 3: We embed G into a large random graph $\overline{G}$ randomly and run an average-case solver on $\overline{G}$. If $\overline{G}$ and ϕ are given, then we can write G as $G = \phi^{-1}(\overline{G}[\phi([n])])$.

then so does Q with a slight loss of parameters.

We show that Q^* is a sampler. Take a δ-dense subset $X' \subseteq X$. Let $P^* \in [0,1]^{Y \times X}$ be the matrix defined by $P^*(\overline{G}, G) = \frac{\mu(G)P(G,\overline{G})}{\nu(G)}$ so that the weight of an edge (y,x) of Q^* can be written as $W^\top(y,x) = \mu(x)P(x,y) = \nu(y)P^*(y,x)$. For an instance $\overline{G} \in Y$, a random neighbor $G \sim P^*(\overline{G}, \cdot)$ of $\overline{G}$ on Q^* is given by $\phi^{-1}(\overline{G}[\phi([n])])$ (Figure 3), where $\phi\colon [n] \to [N]$ is a uniformly random injection conditional on the event that $\phi([n])$ contains a k-clique (for simplicity, we assume that $\overline{G}$ contains a unique k-clique). Let ϕ_1, ϕ_2 be such independent random injections. By the argument of Birthday Paradox, with probability at least $1 - \frac{2n^2}{N}$ over the choice of ϕ_1, ϕ_2, the vertex sets $\phi_1([n])$ and $\phi_2([n])$ are disjoint except for the k-clique. Therefore, for random $\overline{G} \sim \mathcal{G}_{N,1/2,k}$, the number Z of the neighbors of $\overline{G} \in Y$ in X' can be expressed as the sum of random variables such that at least $(1 - 2n^2/N)$-fraction of pairs are pairwise independent. This bounds the variance of Z. By the Chebyshev inequality, we obtain that the query graph is a sampler if $N \gg n^2$.

The drawback of the argument above is that it requires a *quadratic blow-up* in N. We need to set $N \geq \Omega\left(\frac{n^2}{c^3\delta^2\epsilon}\right)$ to ensure that the query graph is a (δ, c)-sampler for density ϵ. In other words, this reduces the task of finding a k-clique in an n-vertex random graph to the task of finding a k-clique in a $1000n^2$-vertex random graph. This suffices to show Theorem 1.2, but not for Theorem 1.1, in which we need a linear blow-up in N.

Spectral Bounds from Coupling. For the matrix P of the one-query reduction, define its spectral by $\lambda(P) := \max\{\sqrt{|\gamma|}\colon \gamma \neq \pm 1 \text{ is an eigenvalue of } PP^*\}$. It is not hard to see that, for any $c, \epsilon > 0$, the query graph Q is a $\left(\frac{\lambda(P)^2}{c^2\epsilon}, c\right)$-sampler for density ϵ.

To bound $\lambda(P)$, we recall the coupling method, which is a standard technique to show the rapid mixing of Markov chains (see, e.g., [LP17]). Recall that a coupling of two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$ over the space Ω is a pair of random variables (Z_1, Z_2) such that the marginal distribution of each Z_i is $\mathcal{D}_i$. Let $R \in [0,1]^{\Omega \times \Omega}$ be a stochastic matrix[3] associated with a metric space $(\Omega, \mathrm{dist}(\cdot,\cdot))$. For $z \in \Omega$, we denote by $R(z,\cdot) \in [0,1]^\Omega$ the distribution that assigns $x \in \Omega$ a weight $R(z,x)$. For $\theta \in [0,1]$ and $z_1, z_2 \in \Omega$, a *θ-shrinking coupling for z_1 and z_2* is a coupling (Z_1, Z_2) of $R(z_1,\cdot)$ and $R(z_2,\cdot)$ such that $\mathbf{E}[\mathrm{dist}(Z_1, Z_2)] \leq \theta \cdot \mathrm{dist}(z_1, z_2)$. We use Chen's theorem [Che98], which states that $\lambda(R) \leq \theta$ if there exists a θ-shrinking coupling for any $z_1, z_2 \in \Omega$.

Our plan is to apply Chen's theorem to the stochastic matrix $R = PP^*$. For this purpose, we

[3] A matrix $A \in [0,1]^{n \times n}$ is a *stochastic matrix* if all rows sum up to 1.

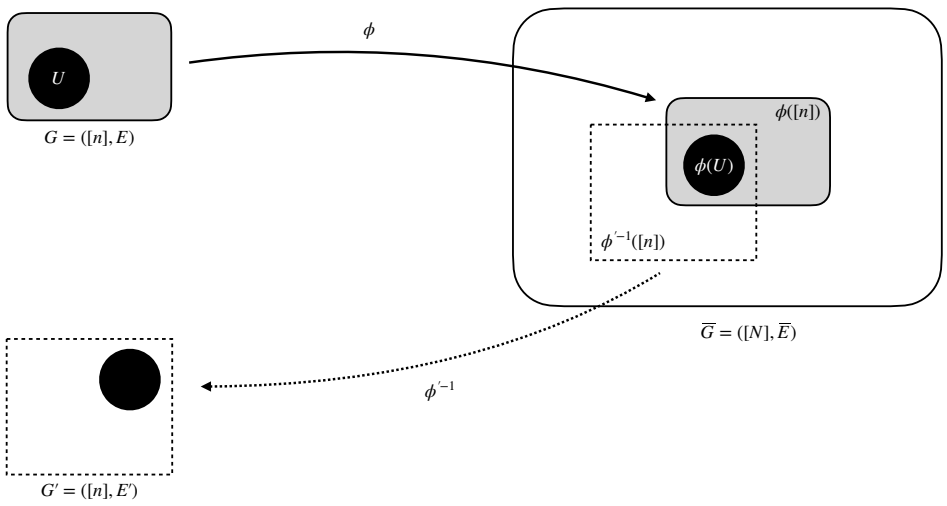

Figure 4: Sampling $G' \sim PP^*(G, \cdot)$ can simulated by using a random injection ϕ to embed $G = ([n], E)$ into a large random graph $\overline{G}$ and thereafter using another random injection ϕ' to extract an n-vertex graph G' randomly from $\overline{G}$. The graphs G and G' share n^2/N vertices on average.

construct an $\frac{n}{N}$-shrinking coupling for the stochastic matrix $PP^* \in [0,1]^{X \times X}$ and carefully define a metric $\mathrm{dist}(\cdot, \cdot)$. Let $G_1, G_2 \in X$ be the initial n-vertex graphs. We assume that G_i contains a unique k-clique $U_i \subseteq [n]$ $(i = 1, 2)$. Let $\pi \colon [n] \to [n]$ be a permutation such that (i) if $U_1 = U_2$, then π is the identity map, and (ii) otherwise, π is any permutation that maps U_2 to U_1. Our coupling simulates the transition of $PP^*(G_1, \cdot)$ and $PP^*(\pi(G_2), \cdot)$ with the same random seed. Let G_1', G_2' be the graphs obtained by the coupling starting from G_1, G_2. The metric $\mathrm{dist}(\cdot, \cdot)$ is defined by $\mathrm{dist}(G_1, G_2) = \infty$ if $U_1 \neq U_2$, and $\mathrm{dist}(G_1, G_2) = d_{\mathrm{ham}}(G_1, G_2)$ otherwise, where d_{ham} denotes the Hamming distance. To prove $\mathbf{E}[\mathrm{dist}(G_1', G_2')] \leq \frac{n}{N}\mathrm{dist}(G_1, G_2)$, we observe that the distribution $PP^*(G, \cdot)$ can be sampled by first embedding G into an N-vertex graph $\overline{G}$ and thereafter randomly extracting an n-vertex graph from it (Figure 4). Thus, if G_1 and G_2 contain the same k-clique, then on average a $(1 - \frac{n}{N})$-fraction of vertices in G will be replaced by new vertices from $\overline{G}$, implying that G_1' and G_2' share the same edges incident to the new vertices. This reduces the Hamming distance. See Section 8.2 for details.

2.5 Related Work

A random self-reduction reduces the evaluation of f on a worst-case input to that on an average-case input. A prototypical example is the permanent of a matrix over a large finite field. Lipton [Lip91] obtained a random self-reduction for the permanent in a low-error regime where the average-case solver must compute the permanent for a $(1-o(1/n))$-fraction of matrices. Subsequent works [FL92; GS92; CPS99] have improved the error regime. In particular, Cai, Pavan, and Sivakumar [CPS99] obtained a random self-reduction with a high-error regime of $1 - 1/\mathrm{poly}(n)$. Blum and Micali [BM84] obtained a random self-reduction for the discrete log function in the literature on pseudo-random generators. Goldreich and Rothblum [GR18] obtained a random self-reduction for counting k-cliques in a high-error regime, although the input distribution is artificial owing to the reduction. Boix-Adserà, Brennan, and Bresler [BBB21] and subsequent papers [Gol20; HS21; DLW20; HS22] obtained random self-reductions for subgraph counting problems in an Erdős–Rényi random graph in a low-error regime. Ball, Rosen, Sabin, and Vasudevan [BRSV17] obtained a random self-

reduction for functions that are closely related to popular problems in fine-grained complexity.

Bogdanov and Trevisan [BT06b] showed that there is no randomized nonadaptive polynomial-time reduction from NP-complete problems to an average-case variant of NP unless the Polynomial Hierarchy collapses. Thus, it is unlikely that the planted clique problem can be reduced from its worst-case variant, i.e., the maximum clique problem.

Samplers and relevant notions have a wide range of applications including randomness-efficient algorithms [BR94; BGG93; Zuc97], error-correcting codes [DHKNT21], and PCP theorem [DR06; GS00]. We refer interested readers to Goldreich [Gol11] for explicit constructions of samplers. In the literature of hardness amplification, Impagliazzo et al. [IJK09b; IJKW10] observed that the direct product theorem follows from the sampler property of the query graph of the well-known reduction.

2.6 Future Direction

This work presents a general framework for hardness self-amplification based on the expansion property of upward self-reductions. Using this framework, we obtain hardness self-amplification for *natural problems* over *natural distributions*. We leave several interesting open questions.

Improved Reductions for Planted Clique Problem. In Theorem 1.2, we presented the first search-to-decision reduction for the planted clique problem in a high-error regime. A natural direction is to improve the blow-up of $N = N(n)$, e.g., $N = n^{1+o(1)}$. This is of particular interest in the literature on the computational limits of statistical problems (e.g., [BBH18]), where we often assume the hardness of distinguishing $\mathcal{G}_{n,1/2,n^{1/2-\epsilon}}$ and $\mathcal{G}_{n,1/2}$.

We mention that the restriction in Theorem 1.2 that $\epsilon(n) \gg n^{-1/2}$ can be removed. We omit the details in this version.

Uniform Reductions for Triangle Counting. In Theorems 1.4 and 1.5, we presented a *nonuniform* hardness self-amplification for triangle counting over an Erdős–Rényi random graph. This improves the error tolerance of the previous *uniform* random self-reduction for clique counting of [BBB21], which is an important open question of the fine-grained average-case complexity of subgraph counting [BBB21; Gol20]. A natural question is whether we can relax the nonuniformity and errorless assumption in Theorem 1.5. Another interesting direction is to obtain a hardness self-amplification result for general subgraph counting (such as k-clique and k-cycle).

2.7 Organization

The remainder of this paper is organized as follows: In Section 3, we introduce the framework of average-case complexity and samplers. In Section 4, we present a general framework of hardness amplification based on samplers. Sections 5 to 8 are independent of each other; the readers may read them in any order. In Section 5, we consider the matrix multiplication and prove Theorem 1.6. In Section 6, we introduce the notion of the data structure algorithms and consider the online matrix-vector multiplication. Then we prove Theorem 1.7. In Section 7, we consider the triangle counting and prove Theorems 1.4 and 1.5. In Section 8, we consider the planted clique problem and prove Theorems 1.1 and 1.2.

3 Preliminaries

For $n \in \mathbb{N}$, we write $[n] = \{1, \ldots, n\}$. We use $x \sim \mu$ to denote that x is drawn from a distribution μ. For a finite set S, we also use $x \sim S$ to denote that x is drawn uniformly at random from S. For a distribution μ, let $\mathsf{supp}(\mu) = \{x \colon \mu(x) > 0\}$ denote the support of μ. We invoke some basic inequalities:

Lemma 3.1 (The Chebyshev inequality)**.** *Let X be a random variable with finite mean and variance. Then, for any $r > 0$,*

$$\Pr[|X - \mathbf{E}[X]| > r\sqrt{\mathbf{Var}[X]}] \leq \frac{1}{r^2}.$$

Lemma 3.2 (The Chernoff inequality)**.** *Let $X_1, \ldots, X_k \in [0,1]$ be independent random variables and $X = \sum_{i \in [k]} X_i$. Then, for any $0 < c < 1$,*

$$\Pr[X < (1-c)\,\mathbf{E}[X]] \leq \exp\left(-\frac{c^2}{2}\,\mathbf{E}[X]\right),$$
$$\Pr[X > (1+c)\,\mathbf{E}[X]] \leq \exp\left(-\frac{c^2}{3}\,\mathbf{E}[X]\right).$$

3.1 Computational Complexity

We identify a problem with a function $f \colon \{0,1\}^* \to \{0,1\}^*$. (The only exception in this paper is the planted clique problem.) In this paper, we usually fix the input size n and consider the complexity of computing $f(x)$ for $x \sim \mu$, where μ is a distribution over the set of all inputs of size n (e.g., $\mathcal{G}_{n,1/2}$). We call such pair (f, μ) a *distributional problem*.

Definition 3.1. *A randomized algorithm $\mathcal{M}$ solves a problem f if $\Pr_{\mathcal{M}}[\mathcal{M}(x) = f(x)] \geq 2/3$ for any input x. A randomized algorithm $\mathcal{M}$ solves a distributional problem (f, μ) with* success probability ϵ *if $\Pr_{x \sim \mu}[\Pr_{\mathcal{M}}[\mathcal{M}(x) = f(x)] \geq 2/3] \geq \epsilon$.*

The constant $2/3$ above can be arbitrary by repetition. Specifically, we sometimes use the following standard result.

Claim 3.3. *If there exists a $T(n)$-time $\mathcal{M}$ that solves (f, μ) with success probability ϵ, then, for any $0 < \gamma < 1$, there exists a $O(T(n)\log(1/\gamma))$-time algorithm $\mathcal{M}'$ that satisfies $\Pr_{x \sim \mu, \mathcal{M}}[\mathcal{M}(x) = f(x)] \geq (1-\gamma)\epsilon$.*

Throughout the paper, we implicitly assume that the complexity of computing parameters ϵ, δ is negligible (or equivalently, we assume that an algorithm $\mathcal{M}$ is given the parameters as input).

3.2 Markov Operator of Bipartite Graphs

Let $Q = (X, Y, W)$ be an edge-weighted bipartite graph whose weights are given by a matrix $W \in [0,1]^{X \times Y}$. We assume that W is normalized such that $\sum_{x,y} W(x,y) = 1$, which specifies a distribution over $X \times Y$. This distribution induces marginal distributions $\mu \in [0,1]^X, \nu \in [0,1]^Y$ as follows:

$$\mu(x) = \sum_{y \in Y} W(x,y), \quad \nu(y) = \sum_{x \in X} W(x,y).$$

Let $\mathcal{L}^2(X) = (\mathbb{R}^X, \langle \cdot, \cdot \rangle_X)$ be a space $\mathbb{R}^X$ associated with the inner product defined by

$$\langle v_1, v_2 \rangle_X = \mathop{\mathbf{E}}_{x \sim \mu}[v_1(x) v_2(x)] = \sum_{x \in X} \mu(x) v_1(x) v_2(x).$$

Let $\| \cdot \|_X$ be the norm induced by the inner product. Let $\mathbb{1}_X \in \mathcal{L}^2(X)$ be the all-one vector. For $v \in \mathcal{L}^2(X)$, we use the standard notation for the mean $\mathbf{E}[v] = \langle v, \mathbb{1}_X \rangle_X$ and variance $\mathbf{Var}[v] = \mathbf{E}[v^2] - \mathbf{E}[v]^2$, where $v^2 \in \mathcal{L}^2(X)$ denotes the component-wise square of v (i.e., $v^2(x) = v(x)^2$ for all $x \in X$). We also define the inner product space $\mathcal{L}^2(Y) = (\mathbb{R}^Y, \langle \cdot, \cdot \rangle_Y)$ in the same way as the definition of $\mathcal{L}^2(X)$. Similarly, we consider the mean $\mathbf{E}[w]$ and variance $\mathbf{Var}[w]$ for $w \in \mathcal{L}^2(Y)$. We sometimes omit the subscript if it is clear from the context (e.g., we use $\langle \cdot, \cdot \rangle$ and $\mathbb{1}$).

Let $P \in [0,1]^{X \times Y}, P^* \in [0,1]^{Y \times X}$ be matrices defined by

$$P(x,y) = \mu(x) W(x,y), \quad P^*(y,x) = \nu(y) W^\top(y,x).$$

We view P, P^* as Markov operators[4] $P \colon \mathcal{L}^2(Y) \to \mathcal{L}^2(X)$ and $P^* \colon \mathcal{L}^2(X) \to \mathcal{L}^2(Y)$ by considering

$$Pw(x) = \mathop{\mathbf{E}}_{y \sim P(x,\cdot)}[w(y)], \quad P^* v(x) = \mathop{\mathbf{E}}_{x \sim P^*(y,\cdot)}[v(x)].$$

In this sense, P^* is the adjoint operator of P. Note that $\langle Pw, v \rangle = \langle w, P^* v \rangle$ for $v \in \mathcal{L}^2(X)$ and $w \in \mathcal{L}^2(Y)$. In particular, $\mathbf{E}[Pw] = \langle Pw, \mathbb{1} \rangle = \langle w, \mathbb{1} \rangle = \mathbf{E}[w]$.

Definition 3.2. *For a Markov operator $P \colon \mathcal{L}^2(Y) \to \mathcal{L}^2(X)$, let*

$$\lambda(P) = \sup_{\substack{w \in \mathcal{L}^2(Y): \\ \langle w, \mathbb{1} \rangle = 0, w \neq 0}} \frac{\|Pw\|}{\|w\|}.$$

We also define $\lambda(P^*)$ for $P^* \colon \mathcal{L}^2(X) \to \mathcal{L}^2(Y)$ in the same way.

Claim 3.4. *For a Markov operator $P \colon \mathcal{L}^2(Y) \to \mathcal{L}^2(X)$ and $w \in \mathcal{L}^2(Y)$, $\mathbf{Var}[Pw] \le \lambda(P)^2 \mathbf{Var}[w]$.*

Proof. Let $w_0 = w - \mathbf{E}[w]\mathbb{1}$. Note that $Pw_0 = Pw - \mathbf{E}[w]\mathbb{1}$. Since the variance does not change by shift and $\langle w_0, \mathbb{1} \rangle = 0$, we have $\mathbf{Var}[Pw] = \mathbf{Var}[Pw_0] = \|Pw_0\|^2 \le \lambda(P)^2 \|w_0\|^2 = \lambda(P)^2 \mathbf{Var}[w]$. □

Lemma 3.5 (Claim 2.7 of [DK17])**.** *For a Markov operator $P \colon \mathcal{L}^2(Y) \to \mathcal{L}^2(X)$ and its adjoint operator $P^* \colon \mathcal{L}^2(X) \to \mathcal{L}^2(Y)$, it holds that $\lambda(PP^*) = \lambda(P)\lambda(P^*) = \lambda(P)^2$.*

Definition 3.3 (λ-expander)**.** *An edge-weighted bipartite graph (X, Y, W) is a* λ-expander *if the associated Markov operator $P \colon \mathcal{L}^2(X) \to \mathcal{R}^2(Y)$ satisfies $\lambda(P) \le \lambda$.*

3.3 Sampler

We consider an edge-weighted bipartite graph $Q = (X, Y, W)$ associated with inner product spaces $\mathcal{L}^2(X), \mathcal{L}^2(Y)$ and the Markov operator $P \colon \mathcal{L}^2(X) \to \mathcal{L}^2(Y)$. A *measure* is a $[0,1]$-valued function. A measure $w \in [0,1]^Y$ is *ϵ-dense* if $\mathbf{E}[w] \ge \epsilon$ (we also define the density of a measure over X). One can think of a measure as the continuous relaxation of an indicator function. Note that $\mathbf{Var}[w] = \mathbf{E}[w^2] - \mathbf{E}[w]^2 \le \mathbf{E}[w](1 - \mathbf{E}[w])$ for any measure.

[4] A Markov operator $M \colon A \to B$ is a linear operator that is nonnegative and $M\mathbb{1}_A = \mathbb{1}_B$.

Definition 3.4 (Sampler)**.** *For $0 < c < \delta$, we say that Q is a* (δ, c)-sampler for density ϵ *if, for any ϵ-dense measure $w \in [0,1]^Y$,*

$$\Pr_{x\sim\mu}\left[Pw(x) \le (1-c)\,\mathbf{E}[w]\right] \le \delta.$$

Lemma 3.6. *If $Q = (X, Y, W)$ is a λ-expander, then, for any $r > 0$ and $w \in \mathcal{L}^2(Y)$,*

$$\Pr_{x\sim\mu}\left[|Pw(x) - \mathbf{E}[w]| \ge r\sqrt{\mathbf{Var}[w]}\right] \le \frac{\lambda^2}{r^2}.$$

In particular, for any $c, \epsilon > 0$, Q is a $\left(\frac{\lambda^2}{c^2\epsilon}, c\right)$-sampler for density ϵ.

Proof. Let $v = P^*w$. Note that $\mathbf{E}[v] = \mathbf{E}[w]$ and $\mathbf{Var}[v] \le \lambda^2\,\mathbf{Var}[w]$ by Claim 3.4. By the Chebyshev inequality, we have

$$\Pr_{x\sim\mu}\left[|Pw(x) - \mathbf{E}[w]| \ge r\sqrt{\mathbf{Var}[w]}\right] \le \Pr_{x\sim\mu}\left[|v(u) - \mathbf{E}[v]| \ge \frac{r}{\lambda}\sqrt{\mathbf{Var}[v]}\right] \le \frac{\lambda^2}{r^2}.$$

In particular, if w is an ϵ-dense measure, $\mathbf{Var}[w] \le \mathbf{E}[w]$. If $\mathbf{Var}[w] > 0$, by setting $r = \frac{c\,\mathbf{E}[w]}{\sqrt{\mathbf{Var}[w]}}$, we have

$$\Pr_{x\sim\mu}\left[Pw(x) \le (1-c)\,\mathbf{E}[w]\right] \le \Pr_{x\sim\mu}\left[|Pw(x) - \mathbf{E}[w]| \ge r\sqrt{\mathbf{Var}[w]}\right] \le \frac{\lambda^2\,\mathbf{Var}[w]}{c^2\,\mathbf{E}[w]^2} \le \frac{\lambda^2}{c^2\epsilon}.$$

If $\mathbf{Var}[w] = 0$, then $w = \mathbf{E}[w] \cdot \mathbb{1}$ and the claim is clear. □

Let $Q^* = (Y, X, W^\top)$ be the bipartite graph obtained by swapping X and Y. Note that the associated Markov operator is P^*.

Lemma 3.7. *Let $Q = (X, Y, W)$ be an edge-weighted bipartite graph. If Q^* is an (ϵ', c')-sampler for density δ, then, for any c, ϵ that satisfy $0 < c < \epsilon$ and $(1-c')\left(1 - \frac{\epsilon'}{\epsilon}\right) \ge 1 - c$, Q is a (δ, c)-sampler for density ϵ. In particular, Q is a (δ, c)-sampler for density ϵ if Q^* is a $(c\epsilon/2, c/2)$-sampler for density δ.*

Proof. The "in particular" part directly follows from the former claim by setting $c' = c/2$ and $\epsilon' = c\epsilon/2$. For any δ-dense measure $v' \in [0,1]^X$, let $S' = \{y \in Y : Pv'(y) \le (1-c')\,\mathbf{E}[v']\}$. For any measure $w' \in [0,1]^Y$, $\sum_{y\notin S'} \nu(y)w'(y) = \mathbf{E}[w'] - \sum_{y\in S'} \nu(y)w'(y) \ge \mathbf{E}[w'] - \epsilon'$ since Q^* is a sampler. Therefore, for any δ-dense measure $v' \in [0,1]^X$ and any measure $w' \in [0,1]^Y$, we have

$$\langle Pv', w'\rangle \ge \sum_{y\notin S'} Pv'(y)\nu(y)w'(y) > (1-c')\,\mathbf{E}[v'](\mathbf{E}[w'] - \epsilon'). \tag{1}$$

Let $w \in [0,1]^Y$ be an ϵ-dense measure and $S = \{x \in X : P^*w(x) \le (1-c)\,\mathbf{E}[w]\}$. Consider the indicator $v = \mathbb{1}_S$ of S. Note that

$$\langle v, P^*w\rangle = \sum_{x\in S} \mu(x)P^*w(x) \le (1-c)\,\mathbf{E}[w]\,\mathbf{E}[v]$$

Our goal is to prove $\Pr_{x\sim\mu}[x \in S] = \mathbf{E}[v] < \delta$. Suppose for contradiction that $\mathbf{E}[v] \ge \delta$. Then, v is a δ-dense measure and by (1) and we have

$$\langle v, P^*w\rangle = \langle Pv, w\rangle > (1-c')\,\mathbf{E}[v](\mathbf{E}[w] - \epsilon') \ge (1-c')(1-\epsilon'/\epsilon)\,\mathbf{E}[w]\,\mathbf{E}[v].$$

Therefore, we have $(1-c')\left(1 - \frac{\epsilon'}{\epsilon}\right) < 1 - c$, which contradicts the assumption on ϵ, c □

4 Hardness Amplification

4.1 Framework

Definition 4.1. *Let (f,μ) and (g,ν) be distributional problems, where μ (resp. ν) is a distribution of inputs of size n (resp. m). A* one-query reduction from (f,μ) to (g,ν) *is a randomized oracle algorithm $\mathcal{R}^{\mathcal{O}}$ that satisfies the following conditions:*

- *Given $x \sim \mu$, $\mathcal{R}^{\mathcal{O}}(x)$ produces a random query y that is an instance of g. Moreover, the distribution of y is ν (over the random seed of $\mathcal{R}$ and choice of $x \sim \mu$).*
- *For any x, $\Pr[\mathcal{R}^{g}(x) = f(x)] \geq 2/3$ where the randomness is over the random seed of $\mathcal{R}$.*

Remark 1. *A reduction making k non-adaptive instances $q_1(x),\dots,q_k(x)$ of (g,ν) can be seen as a one-query reduction to another problem $(g^k,\mathcal{D})$ for some the distribution $\mathcal{D}$ of $(q_1(x),\dots,q_k(x))$ for $x \sim \mu$.*

Let $X = \mathsf{supp}(\mu)$ (resp. $Y = \mathsf{supp}(\nu)$) be the set of all possible inputs of f (resp. g).

Definition 4.2. *For a randomized one-query reduction $\mathcal{R}$ from (f,μ) to (g,ν), the* query graph *is the edge-weighted bipartite graph $Q_{\mathcal{R}} = (X,Y,W)$ associated with the edge weight given by $W(x,y) = \mu(x)\cdot \Pr_{\mathcal{R}}[\mathcal{R}(x)$ produces a query $y]$.*

We simply write Q instead of $Q_{\mathcal{R}}$ if the reduction $\mathcal{R}$ is clear from the context.

Theorem 4.1. *Let $\mathcal{R}^{\mathcal{O}}$ be a randomized reduction from (f,μ) to (g,ν) whose query graph Q is a (δ,c)-sampler for density ϵ. If $\mathcal{M}$ is a randomized algorithm that satisfies*

$$\Pr_{y\sim\nu,\mathcal{M}}[\mathcal{M}(y) = g(y)] \geq \epsilon,$$

then

$$\Pr_{x\sim\mu}\left[\Pr_{\mathcal{R}}\left[\mathcal{R}^{\mathcal{M}}(x) = f(x)\right] \geq (1-c)\epsilon\right] \geq 1-\delta.$$

Proof. The reduction $\mathcal{R}^{\mathcal{M}}(x)$ is divided into two parts: the query-making part $\mathcal{R}_{\mathrm{query}}$ that produces a random query $y \sim \mathcal{R}_{\mathrm{query}}(x)$ and the decision-making part $\mathcal{R}_{\mathrm{decide}}(x,a)$ that determines the output given x and $a = \mathcal{M}(y)$.

Let $w \in [0,1]^Y$ be the measure defined by $w(y) = \Pr[\mathcal{M}(y) = g(y)]$ (the probability is over the random seed of $\mathcal{M}$). By assumption of $\mathcal{M}$, w is ϵ-dense and thus $\mathbf{E}[w] \geq \epsilon$. Consider the Markov operator P associated with the query graph Q. Then, $P(x,y) = \Pr[\mathcal{R}_{\mathrm{query}}(x) = y]$ is the probability that the query-making part produces y. Since Q is a sampler, we have

$$\Pr_{x\sim\mu}\left[\mathop{\mathbf{E}}_{y\sim\mathcal{R}_{\mathrm{query}}(x)}[w(y)] > (1-c)\epsilon\right] = \Pr_{x\sim\mu}\left[Pw(x) > (1-c)\epsilon\right] \geq 1-\delta.$$

Note that $\mathbf{E}_{y\sim\mathcal{R}_{\mathrm{query}}(x)}[w(y)] = \Pr_{\mathcal{M},y\sim\mathcal{R}_{\mathrm{query}}(x)}[\mathcal{M} = g(y)] = \Pr_{\mathcal{R}}[\mathcal{R}^{\mathcal{M}}(x) = f(x)]$. □

Corollary 4.2. *Suppose there exist (i) a randomized $T(n)$-time one-query reduction $\mathcal{R}$ from (f,μ) to (g,ν) whose query graph is a (δ,c)-sampler for density $(1/2+\epsilon)$ for $c \leq \epsilon/4$, and (ii) a $T'(n)$-time randomized algorithm $\mathcal{M}$ for (g,ν) with success probability $1/2+\epsilon$. Then, there exists a randomized $O\left(\frac{T(n)+T'(n)\log(1/\epsilon)}{\epsilon^2}\right)$-time algorithm for (f,μ) with success probability $1-\delta$.*

Proof. By Claim 3.3, we have a $O(T'(n)\log(1/\epsilon))$-time algorithm $\mathcal{N}$ that satisfies $\Pr_{y\sim\nu,\mathcal{N}}[\mathcal{N}(y) = g(y)] \geq 1/2+\epsilon/2$. Run $\mathcal{R}^{\mathcal{N}}(x)$ for $\ell := \lceil 512/\epsilon^2\rceil$ times and take the majority of the outputs (note that the time complexity of computing ϵ is negligible). At each iteration, for a $(1-\delta)$-fraction of $x \sim \mu$, $\mathcal{R}^{\mathcal{N}}(x) = f(x)$ with probability $(1/2+\epsilon/2)(1-c) \geq 1/2+\epsilon/8$. By the Chernoff inequality (Lemma 3.2), the majority is $f(x)$ with probability at least $1-\exp(-\ell\epsilon^2/256) \geq 2/3$. □

4.2 Example: Direct Product

For a distributional problem (f, μ) with $X = \mathsf{supp}(\mu)$ and $k \in \mathbb{N}$, consider the k-wise direct product (f^k, μ^k), where $f^k(x_1, \ldots, x_k) = (f(x_1), \ldots, f(x_k))$ and $\mu^k(x_1, \ldots, x_k) = \prod_{i \in [k]} \mu(x_i)$. Let $Y = X^k$. Consider the following well-known reduction for the direct product problem ([IW97; Imp95; GNW11]).

Reduction $\mathcal{R}^{\mathcal{O}}$ from (f, μ) to (f^k, μ^k)

Given input $x \in X$, sample $i \in [k]$ and $(x_1, \ldots, x_k) \sim L^k$. Let $\overline{x} = (x_1, \ldots, x_{i-1}, x, x_{i+1}, \ldots, x_k)$ be a query. Given $(y_1, \ldots, y_k) = \mathcal{O}(\overline{x})$, output y_i.

Lemma 4.3 (Direct Product Lemma)**.** *The query graph of the reduction $\mathcal{R}$ is a (δ, c)-sampler for density ϵ for any ϵ, δ, c that satisfy $2\exp(-kc^2\delta/8) \le c\epsilon$.*

Proof. Let Q be the query graph associated with the Markov operator P. It follows from [IJK09b, Lemma2.5] that the reversed graph Q^* is a sampler. Indeed, let P^* be the Markov operator associated with Q^*. If $v' \in [0,1]^X$ is a δ-dense measure, then, for a fixed $y = (x_1, \ldots, x_k) \in X^k$, we have $P^*v'(y) = \frac{1}{k}\sum_{i \in [k]} v'(x_i)$. For $y \sim X^k$, by the Chernoff bound (Lemma 3.2), we have

$$\Pr_{y \sim X^k}\left[P^*v'(y) < (1-c')\,\mathbf{E}[v']\right] \le \exp\left(-\frac{kc'^2\delta}{2}\right).$$

Therefore, Q^* is a (ϵ', c')-sampler for density δ if $\exp(-0.5kc'^2\delta) \le \epsilon'$. We obtain the claim by Lemma 3.7 (where we set $\epsilon' = 0.5c\epsilon$ and $c' = 0.5c$). □

5 Matrix Multiplication

We prove Theorem 1.6. Let R be a finite ring. Let Mult be the function that maps two matrices A, B over R to the product AB. Let $\mathcal{U}_{n_1,n_2,n_3}$ be the uniform distribution over $R^{n_1 \times n_2} \times R^{n_2 \times n_3}$. Our computational model is the $O(\log |R|)$-word RAM and thus basic arithmetic operations on R can be performed in a unit of time. We assume that we can sample a uniformly random element of R in a unit of time. We also assume that each algorithm takes at least time $\Omega(n^2)$ to scan the entire input, which allows us to write $O(T(n))$ instead of $O(T(n) + n^2)$.

For $v \in R^n$ and $I \subseteq [n]$, we denote by $v_I \in R^{|I|}$ the subvector of v restricted on the index set I. Similarly, for a matrix $M \in R^{n_1 \times n_2}$ and non-empty $I \subseteq [n_1], J \subseteq [n_2]$, we use $M_{I,J} \in R^{|I| \times |J|}$ to denote the submatrix of M restricted on the row set I and column set J.

5.1 Auxiliary Results

We recall the following worst-case to average-case reduction for matrix multiplication by Blum, Luby, and Rubinfeld [BLR93].

Lemma 5.1. *If there exists a $T(n,m)$-time algorithm $\mathcal{M}$ that solves $(\mathsf{Mult}, \mathcal{U}_{n,m,n})$ with success probability ϵ, then, there exists a randomized algorithm $\mathcal{M}'$ that satisfies $\Pr_{\mathcal{M}'}[\mathcal{M}'(A,B) = \mathsf{Mult}(A,B)] \ge 4\epsilon - 3$ for all $(A,B) \in R^{n \times m} \times R^{m \times n}$.*

Proof. Let (A, B) be the input. Sample $(X_1, Y_1) \sim \mathcal{U}_{n,m,n}$ and let $X_2 = A - X_1$ and $Y_2 = B - Y_1$. Note that the marginal distribution of each (X_i, Y_j) is $\mathcal{U}_{n,m,n}$. By the union bound, $\mathcal{A}$ correctly computes all X_iY_j with probability $4\epsilon - 3$. If this holds, the algorithm $\mathcal{A}'$ outputs $\sum_{i,j \in [2]} \mathcal{A}(X_i, Y_j) = AB$. □

We also recall the randomized verification algorithm for Mult by Freivalds [Fre79].

Lemma 5.2 ([Fre79])**.** *There exists an $O(nm+n^2)$-time randomized algorithm that, given matrices $A \in R^{n\times m}, B \in R^{m\times n}, C \in R^{n\times n}$, decides $AB = C$ correctly with probability $2/3$.*

We say that a function f is $t(n)$-*time verifiable* if there exists a $t(n)$-time randomized algorithm that, given x, y, decides $f(x) = y$ correctly with probability $2/3$. Note that we can amplify the constant $2/3$ arbitrarily through repetition. Lemma 5.2 implies that Mult is $O(nm + n^2)$-time verifiable.

Lemma 5.3 (Direct Product Theorem for Verifiable Function)**.** *Let (f, μ) be a distributional problem such that f is $t(n)$-time verifiable. Let $k \in \mathbb{N}, \delta > 0, \epsilon > 0$ be such that $\epsilon \geq 4\exp(-\delta k/32)$. If there exists a $T(n)$-time algorithm $\mathcal{M}$ for (f^k, μ^k) with success probability ϵ, then, there exists an $O(\epsilon^{-1}(T(n)+t(n)))$-time randomized algorithm $\mathcal{M}'$ that solves (f, μ) with success probability $1-\delta$.*

Proof. By the condition of k, δ, ϵ and Lemma 4.3, the direct product problem (f^k, μ^k) has a reduction $\mathcal{R}^{\mathcal{O}}$ whose query graph is a $(\delta, 1/2)$-sampler for density ϵ. For given input $x \sim \mu$, run the reduction $\mathcal{R}^{\mathcal{A}}(x)$ for $\ell = \lceil 20/\epsilon \rceil$ times using $\mathcal{A}$ as oracle. At each iteration, check the correctness of the output by the $t(n)$-time verifier. This runs in time $O(\epsilon^{-1}(T(n) + t(n)))$.

From Theorem 4.1, for a $(1-\delta)$-fraction of $x \sim \mu$, at each iteration, $\mathcal{R}^{\mathcal{A}}(x) = f(x)$ with probability 0.5ϵ. For such x, with probability at least $1 - (1-\epsilon/2)^{\ell} \geq 0.99$, $\mathcal{R}^{\mathcal{A}}(x) = f(x)$ at least once during the iteration. We can detect this correct output with probability 0.99 using the verifier. □

5.2 Proof of Theorem 1.6

Lemma 5.4. *Let $\epsilon > 0, k \in \mathbb{N}$ be such that $\epsilon \geq 4\exp(-k/3200)$. Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ that solves $(\mathsf{Mult}, \mathcal{U}_{n,n,n})$ with success probability ϵ. Let $d = \lfloor n/k \rfloor$. Then, there exists an $O(\epsilon^{-1}T(n))$-time randomized algorithm $\mathcal{M}'$ that solves $(\mathsf{Mult}, \mathcal{U}_{d,n,d})$ with success probability* 0.98.

Proof. For $i \in [k]$, let $I_i = \{(i-1)d+1, \ldots, id\}$. Let $A_i := A_{I_i,[n]} \in R^{d\times n}$ and $B_i := B_{[n],I_i} \in R^{n\times d}$ be submatrices of A and B, respectively. Note that $(AB)_{I_i,I_j} = A_iB_j \in R^{d\times d}$ for every $i, j \in [k]$ (Figure 1).

If $(A, B) \sim \mathcal{U}_{n,n,n}$, then, the pairs $(A_i, B_i) \sim \mathcal{U}_{d,n,d}$ are independent. If $\mathcal{M}(A, B) = AB$, then we have A_iB_i for all $i \in [k]$. Therefore, there exists an $O(T(n))$-time algorithm $\mathcal{B}$ satisfying

$$\Pr_{(A_1,B_1),\ldots,(A_k,B_k)\sim\mathcal{U}_{d,n,d}}[\mathcal{B}(A_1, B_1, \ldots, A_k, B_k) = (A_1B_1, \ldots, A_kB_k)] \geq \epsilon.$$

Then, from Lemma 5.3 (with setting $\delta = 0.01$), we obtain the algorithm $\mathcal{M}'$ of the claim. □

Proof of Theorem 1.6. Let $A, B \in R^{n\times n}$ be the input. Let $k = \lceil 100\log(1/\epsilon) \rceil$. By padding zeros to A, B, we may assume that k divides n. Divide A, B into $A_1, \ldots, A_k \in R^{d\times n}$ and $B_1, \ldots, B_k \in R^{n\times d}$ as in Figure 1. Repeat the algorithm of Lemma 5.4 for $O(\log k)$ times to amplify the success probability from 0.98 to $1 - 0.01/k^2$. By the union bound over $i, j \in [k]$, we can compute all A_iB_j correctly with probability 0.99. This can be done in time $O(k^2 \cdot \epsilon^{-1}T(n)\log k) = O(\epsilon^{-1}T(n)\log^2(1/\epsilon)\log\log(1/\epsilon))$.

□

6 Online Matrix-Vector Multiplication

We introduce the notion of data structure algorithms and prove Theorem 1.7. Consider a pair $\mathcal{M} = (\mathcal{M}_{\mathrm{pre}}, \mathcal{M}_{\mathrm{ans}})$ of algorithms. The preprocessing algorithm $\mathcal{M}_{\mathrm{pre}}(x)$ takes x as input and returns a string $\pi \in \{0,1\}^*$ in polynomial time, representing the memory of a data structure. The algorithm $\mathcal{M}_{\mathrm{ans}}^{\pi}$ is given oracle access to π and outputs $\mathcal{M}(x;q) := \mathcal{M}^{\pi}(q)$ for a given query q. We call this pair $\mathcal{M}$ a *data structure algorithm*. The *query time* of $\mathcal{M}$ is defined as the running time of $\mathcal{M}_{\mathrm{ans}}$. Unlike Yao's cell-probe model [Yao81], we measure the running time by the number of arithmetic operations and queries.

Let R be a finite ring. For $M \in R^{m\times n}$ and $v \in R^n$, let $\mathsf{OMv}(M,v) = Mv$ and $\mathcal{U}_{m,n}$ be the uniform distribution over $R^{m\times n} \times R^n$. We extend the notion of the success probability (Definition 3.1) as follows: A data structure algorithm $\mathcal{M}$ solves $(\mathsf{OMv}, \mathcal{U}_{m,n})$ with success probability ϵ if $\Pr_{(M,v)\sim\mathcal{U}_{m,n}}[\mathcal{M}(M;v) = Mv] \geq \epsilon$. The main result of this section is the following random self-reduction for OMv:

Theorem 6.1 (formal statement of Theorem 1.7)**.** *Suppose there exists a data structure algorithm $\mathcal{M}$ with query time $T(n)$ that solves $(\mathsf{OMv}, \mathcal{U}_{n,n})$ with success probability ϵ. Then, there exists a randomized data structure algorithm $\mathcal{M}'$ with query time $O(\epsilon^{-2}T(n)\log^4(1/\epsilon)\log\log(1/\epsilon))$ that solves OMv.*

6.1 Auxiliary Results

We observe that OMv is efficiently verifiable in the data structure setting based on the Freivalds' algorithm (Lemma 5.2).

Lemma 6.2. *There exists a randomized data structure algorithm $\mathcal{M} = (\mathcal{M}_{\mathrm{pre}}, \mathcal{M}_{\mathrm{ans}})$ such that, given any matrix $M \in R^{m\times n}$ and vectors $v \in R^n, w \in R^m$, $\mathcal{M}(M; v, w)$ decides in query time $O(tn)$ whether $Mv = w$ correctly with probability $1 - 2^{-t}$ (over the randomness of $\mathcal{M}$). Moreover, $\mathcal{M}$ has a one-sided error (it outputs "Yes" whenever $Mv = w$).*

Proof. For two vectors $u, v \in R^n$, let $\langle u, v\rangle = \sum_{i\in[n]} u(i)v(i)$. The preprocess $\mathcal{M}_{\mathrm{pre}}$ samples t independent random vectors $r_1, \ldots, r_t \sim \{0,1\}^n$ and returns $u_i := r_i^{\top} M$ and r_i for each $i \in [t]$ as a data structure π. In the query phase, for given $v, w \in R^n$, $\mathcal{M}_{\mathrm{ans}}^{\pi}(u, v)$ outputs "No" if $\langle u_i, v\rangle \neq \langle r_i, w\rangle$ for some $i \in [t]$ and outputs "Yes" otherwise. If the given $v, w \in R^n$ satisfies $Mv = w$, then $\langle u_i, v\rangle = r_i^{\top} Mv = r_i^{\top} w = \langle r_i, w\rangle$; thus $\mathcal{M}_{\mathrm{ans}}^{\pi}(v, w)$ always outputs "Yes". Otherwise, for each $i \in [t]$, we have $\Pr[\langle u_i, v\rangle \neq \langle r_i, w\rangle] \geq 1/2$; thus $\mathcal{M}_{\mathrm{ans}}$ outputs "No" with probability at least $1 - 2^{-t}$. □

We also observe that the standard random self-reduction of Blum, Luby, and Rubinfeld [BLR93] works for the data structure algorithm. We do not use Lemma 6.3 directly but use the same idea in the reduction.

Lemma 6.3. *Suppose there exists a data structure algorithm $\mathcal{M} = (\mathcal{M}_{\mathrm{pre}}, \mathcal{M}_{\mathrm{ans}})$ with query time $T(m,n)$ that solves $(\mathsf{OMv}, \mathcal{U}_{m,n})$ with success probability ϵ. Then, there exists a randomized data structure algorithm $\mathcal{M}' = (\mathcal{M}'_{\mathrm{pre}}, \mathcal{M}'_{\mathrm{ans}})$ with query time $O(T(m,n))$ that satisfies $\Pr_{\mathcal{M}'}[\mathcal{M}'(M;v) = Mv] \geq 4\epsilon - 3$ for any $M \in R^{m\times n}, v \in R^n$.*

Proof. Let $\mathcal{M}'$ be as follows: In the preprocessing phase, for a given M, sample $R_1 \sim R^{m\times n}$ and let $R_2 := M - R_1$. Run $\mathcal{M}_{\mathrm{pre}}(R_1)$ and $\mathcal{M}_{\mathrm{pre}}(R_2)$ and obtain two data structures π_1, π_2. Then, return (π_1, π_2) as a data structure. In the query phase, for a given v, sample $s_1 \sim R^n$ and let

$s_2 := v - s_1$. Compute $w_{i,j} := \mathcal{M}^{\pi_i}_{\mathrm{ans}}(s_j)$ for each $i, j \in [2]$ and output $\sum_{i,j\in[2]} w_{i,j}$. By the union bound, with probability $4\epsilon - 3$, we have $w_{i,j} = R_i s_j$ for all $i, j \in [2]$. If this occurs, then the output $\sum_{i,j} w_{i,j} = (R_1 + R_2)(s_1 + s_2) = Mv$ is correct. □

Remark 2. *One may consider the following algorithm: Repeat the algorithm of Lemma 6.3 and use the verifier of Lemma 6.2 to check the correctness at each iteration. Unfortunately, this argument works only for $\epsilon > 3/4$.*

Lemma 6.4 (Reverse Markov's inequality)**.** *Let X be a $[0,1]$-valued random variable and $\mu = \mathbf{E}[X] \in (0,1)$. Then, $\Pr[X \geq \mu/2] \geq \mu/2$.*

Proof. From Markov's inequality, for any $c \in (0,1)$,

$$\Pr[X \leq c] = \Pr[1 - X \geq 1 - c] \leq \frac{1-\mu}{1-c} \leq 1 - (\mu - c).$$

We obtain the claim by substituting $c = \mu/2$. □

6.2 Proof of Theorem 6.1

Fix a data structure algorithm $\mathcal{M}$ with query time $T(n)$ that solves $(\mathsf{OMv}, \mathcal{U}_{n,n})$ with success probability ϵ. Let $k \in \mathbb{N}$ be such that $\epsilon > \exp(-Ck)$ where $C > 0$ is an appropriate small constant (say, $C = 10^{-5}$). Suppose k divides n and let $d = n/k$. We say that a vector $v \in R^n$ is *good* if $\Pr_{M\sim R^{n\times n}}[\mathcal{M}(M; v) = Mv] \geq \epsilon/2$.

Lemma 6.5. *At least $0.5\epsilon|R^n|$ good vectors exist in R^n.*

Proof. For $v \sim R^n$, let $X = \Pr_M[\mathcal{M}(M; v) = Mv]$ be the fraction of M on which $\mathcal{M}(M; v)$ succeeds. Note that $\mathbf{E}_v[X] \geq \epsilon$. Thus, from the reverse Markov inequality (Lemma 6.4), we have $\Pr_{v\sim R^n}[v \text{ is good}] \geq \Pr[X \geq \mathbf{E}[X]/2] \geq \epsilon/2$. □

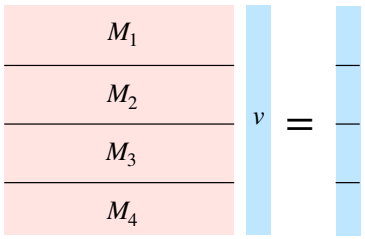

Figure 5: Mv is the concatenation of $M_i v$ for $i \in [k]$.

Lemma 6.6. *There exists a randomized data structure algorithm $\mathcal{M}_1$ with query time $O(\epsilon^{-1}T(n))$ that satisfies $\Pr_{M\sim R^{d\times n}}[\Pr_{\mathcal{M}_1}[\mathcal{M}_1(M; v) = Mv] \geq 0.99] \geq 0.99$ for every n that is divisible by k and every good $v \in R^n$.*

Proof. Note that $d = n/k$ since k divides n. For $i \in [k]$, let $I_i = \{(i-1)d + 1, \ldots, id\}$ and $M_i = M_{I_i,[n]} \in R^{d\times n}$. The vector $Mv \in R^n$ can be obtained by concatenating $M_i v \in R^d$ for $i \in [k]$ (Figure 5). Thus, for every fixed good v, we can compute the function $(M_1, \ldots, M_k) \mapsto (M_1 v, \ldots, M_k v)$ correctly on at least $(\epsilon/2)$-fraction of $M_1, \ldots, M_k$. We view this function as the k-wise direct product of the function $R^{d\times n} \ni M \mapsto Mv \in R^d$ and apply the direct product theorem Lemma 5.3.

Specifically, in the preprocessing phase, $\mathcal{M}_1$ runs as follows: For a given matrix $M \in R^{d\times n}$, sample k independent random matrices $M_1, \dots, M_k \sim R^{d\times n}$ and index $i \sim [k]$. Construct a random matrix $\overline{M} \in R^{n\times n}$ by aligning $M_1, \dots, M_{i-1}, M, M_{i+1}, \dots, M_k$. We repeat this for $\ell = O(1/\epsilon)$ times to create ℓ independent copies $\overline{M}_1, \dots, \overline{M}_\ell$ of $\overline{M}$ and $i_1, \dots, i_\ell$ of $i \sim [k]$. Also, run the preprocess of $\mathcal{M}$ for each $\overline{M}_\ell$ and the preprocess of Lemma 6.2. Output all of them as a data structure π.

In the query phase, $\mathcal{M}_1$ runs as follows: Let $v \in R^n$ be the given query that is supposed to be good. For each $j \in [\ell]$, we simulate $\mathcal{M}^{\pi}_{\mathrm{ans}}(v)$ and obtain a vector $w = \mathcal{M}(\overline{M}_j; v)$. Let $w_1, \dots, w_k$ be the subvector of w where each $w_i \in R^d$ consists of the $((i-1)d+1)$-st to id-th elements of w. Then, use Lemma 6.2 to check whether $\overline{M}_j v = w$. If so, output w_{i_j}. If the algorithm does not halt while iterating over j, output an arbitrary string. This runs in time $O(\ell T(n))$. The correctness of $\mathcal{M}_1$ follows from the proof of Lemma 5.3. □

Corollary 6.7. *There exists a randomized data structure algorithm $\mathcal{M}_2$ with query time $T_2(n)$ that satisfies $\Pr_{\mathcal{M}_2}[\mathcal{M}_2(M; v) = Mv] \geq 0.98$ for every n that is divisible by $k(n)$, $M \in R^{d\times n}$ and good $v \in R^n$, where $T_2(n) = O\left(\epsilon^{-1}T(n)\right)$.*

Proof. We combine Lemma 6.6 and the reduction based on the idea of Lemma 6.3. Specifically, for a given $M \in R^{n\times n}$, sample $R_1 \sim R^{n\times n}$ and let $R_2 = M - R_1$. Use the data structure algorithm $\mathcal{M}_1$ of Lemma 6.6 and output $\mathcal{M}_1(R_1; v) + \mathcal{M}_1(R_2; v)$. By the union bound, this algorithm succeeds with probability 0.98 for any good v. □

Lemma 6.8. *There exists a randomized data structure algorithm $\mathcal{M}_3$ with query time $T_3(n)$ that satisfies $\Pr_{v\sim R^d}[\Pr_{\mathcal{M}_3}[\mathcal{M}_3(M; v) = Mv] \geq 0.99] \geq 0.99$ for every n that is divisible by k and $M \in R^{d\times d}$, where $T_3(n) = O(\epsilon^{-2}T(n))$.*

Remark 3. *Note that the algorithm $\mathcal{M}_3$ of Lemma 6.8 runs in time $T_3(n)$ to compute Mv for a slightly smaller matrix $M \in R^{d\times d}$ and a vector $v \in R^d$.*

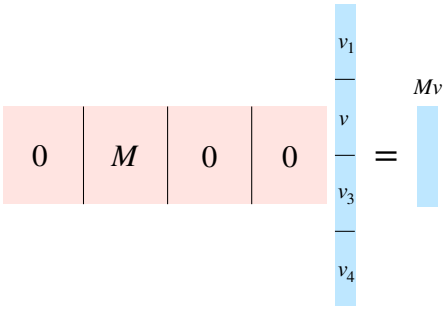

Figure 6: $M'\overline{v} = Mv$ where $M \in R^{d\times d}$ and $v \in R^d$ is embedded in the second block ($i = 2$).

Proof of Lemma 6.8. Let $S \subseteq (R^d)^k$ be the set of k-tuple of vectors $(v_1, \dots, v_k)$ such that the concatenation $(v_1, \dots, v_k) \in R^n$ is good. From Lemma 6.5, $|S| \geq (\epsilon/2)|R^d|^k$. For a vector $v \in R^d$, sample $v_1, \dots, v_k \sim R^d$ and $i \sim [k]$ and thereafter let $\overline{v} = (v_1, \dots, v_{i-1}, v, v_{i+1}, \dots, v_k) \in R^n$. Let $H = \{v \in R^d \colon \Pr[\overline{v} \in S] \geq 0.25\epsilon\}$. From Lemma 4.3 (with letting $\delta = 0.01, c = 0.5, \gamma = 2$), we have $|H| \geq 0.99|R^d|$.

In the query phase, the algorithm $\mathcal{M}_3$ repeats the following for $O(1/\epsilon)$ times: Suppose that a given query v is in H. Then, $\overline{v}$ is good with probability 0.25ϵ. Suppose v is embedded in the i-th block of $\overline{v}$. If $\overline{v}$ is good, the data structure algorithm $\mathcal{M}_2$ of Corollary 6.7 computes $N\overline{v}$ in query time $O(\epsilon^{-1}T(n))$ for any matrix $N \in R^{d\times n}$. Let $M' \in R^{d\times n}$ be the matrix such that $M'_{[d],I_i} = M$

and the other elements are set to zero, where $I_i = \{(i-1)d+1, \ldots, id\}$ (we store M' for all $i \in [k]$ in the preprocessing phase). Note that $\mathcal{M}_2(M'; \overline{v}) = M'\overline{v} = Mv$ if $\overline{v}$ is good (Figure 6). Use Lemma 6.2 to check whether $\mathcal{M}_2(M'; \overline{v}) = Mv$. If so (which occurs with probability 0.25ϵ over the choice of $\overline{v}$ for any $v \in H$), output Mv and terminate. Otherwise, sample $\overline{v}$ and repeat.

If $v \in H$, with probability 0.99 (over the choice of $\overline{v}$), this algorithm terminates within $O(1/\epsilon)$ iterations. The total running time of $\mathcal{M}_3$ is $O(\epsilon^{-1}T_2(n)) = O(\epsilon^{-2}T(n))$. □

Corollary 6.9. *There exists a randomized data structure algorithm $\mathcal{M}_4$ with query time $T_4(n)$ that satisfies $\Pr_{\mathcal{M}_4}[\mathcal{M}_4(M; v) = Mv] \geq 0.98$ for every n that is divisible by $k(n)$, $M \in R^{d\times d}$ and $v \in R^d$, where $T_4(n) = O(\epsilon^{-2}T(n))$.*

Proof. The proof is almost identical to that of Corollary 6.7. Specifically, at the query phase, for a given $v \in R^d$, $\mathcal{M}_4$ samples $r_1 \sim R^d$ and set $r_2 = v - r_1$. Then, $\mathcal{M}_4$ runs $\mathcal{M}_3(M; r_1)$ and $\mathcal{M}_3(M; r_2)$ and then outputs the sum of the two outputs. By the union bound, with probability 0.98, these two outputs are Mr_1 and Mr_2 and thus we have $Mr_1 + Mr_2 = Mv$. □

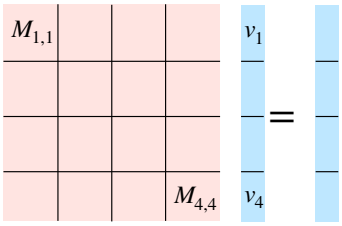

Figure 7: $M_{i,j}$ and v_j for $k = 4$

Proof of Theorem 6.1. We may assume that k divides n (by padding). For given $M \in R^{n\times n}$ and $v \in R^n$, divide them into $M_{i,j} \in R^{d\times d}$ and $v_i \in R^d$ as in Figure 7. Specifically, write $S_i = \{(i-1)d+1, \ldots, id\}$ and let $M_{i,j} = M_{S_i,S_j}$ and $v_i = v_{S_i}$. For each $i, j \in [k]$, repeat running the data structure algorithm $\mathcal{M}_4(M_{i,j}; v_i)$ of Corollary 6.9 for $O(\log k)$ times to obtain $M_{i,j}v_j$ with probability $1 - 0.01/k^2$. Then, by the union bound, with probability 0.99, we have $M_{i,j}v_j$ for all $i, j \in [k]$. Then, compute Mv by $(Mv)_{I_i} = \sum_{j\in[k]} M_{i,j}v_j$. This algorithm runs in time $O(T_4(n) \cdot k^2 \log k) = O(\epsilon^{-2}T(n)\log^2(1/\epsilon)\log\log(1/\epsilon))$ (recall $k = O(\log(1/\epsilon))$). □

7 Triangle Counting

In this section, we prove Theorems 1.4 and 1.5. For a graph G, let $V(G)$ and $E(G)$ denote the vertex and edge sets, respectively. For a vertex subset $S \subseteq V$, let $G[S] = (S, E \cap \binom{S}{2})$ denote the induced subgraph on S. To clarify the input size, we identify the problem #Triangle (resp. $\oplus$Triangle) with the family of functions #Triangle $= (\#\text{Triangle}_n)_{n\in\mathbb{N}}$ (resp. $\oplus\text{Triangle} = (\oplus\text{Triangle}_n)_{n\in\mathbb{N}}$) where each $\#\text{Triangle}_n$ (resp. $\oplus\text{Triangle}_n$) is the function that maps an n-vertex graph G to the number (resp. parity) of triangles in G.

7.1 Nonuniform and Errorless Algorithm

Definition 7.1. *An algorithm $\mathcal{M}$ for a problem f is* errorless *if $\mathcal{M}(x) \in \{f(x), \perp\}$ for all input x. A* nonuniform algorithm *is an algorithm $\mathcal{M}$ that takes x and an auxiliary input α called* advice *as input, where the string α depends only on the size of x.*

An algorithm for problem f can output a string that is not $f(x)$ for input x. In contrast, an errorless algorithm is an algorithm that never outputs an incorrect answer but it may sometimes output $\perp$ to indicate "I do not know".

It is well known that nonuniform algorithms can be derandomized by fixing random seeds as advice.

Lemma 7.1. *Let (f, μ) be a distributional problem and $\mathcal{M}(x; r)$ be a randomized algorithm that is given an input x and a random seed r and satisfies $\Pr_{x\sim\mu}[\Pr_r[\mathcal{M}(x; r) = f(x)] \geq p_1] \geq p_2$. Then, for every $n \in \mathbb{N}$, there exists advice α_n such that $\Pr_{x\sim\mu}[\mathcal{M}(x; \alpha_n) = f(x)] \geq p_1p_2$.*

Proof. For every n, $\mathcal{M}$ satisfies $\Pr_{x\sim\mu,r}[\mathcal{M}(x; r) = f(x)] \geq p_1p_2$ and thus there exists r such that $\Pr_x[\mathcal{M}(x; r) = f(x)] \geq p_1p_2$ by averaging. We use such $\alpha_n = r$ as our advice. □

7.2 Auxiliary Results

We invoke the worst-case to average-case reduction of [BBB21].

Lemma 7.2 (Theorem 2.8 of [BBB21])**.** *Let $0 < p < 1$ be any constant. If there exists a $T(n)$-time algorithm $\mathcal{M}$ that solves $(\#\mathsf{Triangle}_n, \mathcal{G}_{n,p})$ with success probability $1 - O\left(\frac{1}{\log^7 n}\right)$, then, there exists a $T(n)\operatorname{polylog} n$-time randomized algorithm $\mathcal{M}'$ that solves $\#\mathsf{Triangle}_n$.*

Remark 4. *Actually, Boix-Adserà, Brennan, and Bresler [BBB21] reduced computing $\#\mathsf{Triangle}_n(G)$ to solving $(\#\mathsf{Triangle}_{3n}, \mathcal{G}_{3n,p})$ with success probability $1 - 1/\operatorname{polylog}(n)$, which can be done in time $T(3n)$. In other words, the sizes of queries are larger than the original input. This issue can be easily handled with the following lemma, which implies Lemma 7.2.*

Lemma 7.3. *Suppose there exists a $T(n)$-time algorithm that solves $\#\mathsf{Triangle}_n$. Then, for any $\ell \in \mathbb{N}$, there exists an $O(T(\lceil n/\ell \rceil) \cdot \ell^3 \log \ell)$-time algorithm that solves $\#\mathsf{Triangle}_n$. The same holds for $\oplus\mathsf{Triangle}_n$.*

We prove Lemma 7.3 in Appendix A. Our reduction crucially relies on the following observation.

Lemma 7.4. *For an N-vertex graph $\overline{G} = (\overline{V}, \overline{E})$ and nonempty $V \subseteq \overline{V}$, let $G = \overline{G}[V]$ and $G' = \left(\overline{V}, \overline{E} \setminus \binom{V}{2}\right)$ be the graph obtained by removing edges inside V from $\overline{G}$. Let A' be the adjacency matrix of G'. Then, $\#\mathsf{Triangle}_n(G) = \#\mathsf{Triangle}_N(\overline{G}) - \#\mathsf{Triangle}_N(G') - \sum_{\{u,v\}\in E(G)} A'^2_{uv}$.*

Proof. We prove $\#\mathsf{Triangle}_N(\overline{G}) = \#\mathsf{Triangle}_N(G') + \#\mathsf{Triangle}_n(G) + \sum_{e\in E(G)} A'^2_{uv}$. Let $C = \{u, v, w\}$ be a triangle in $\overline{G}$. Then, C is in one of the following three cases: (i) If C forms a triangle in G', then $\#\mathsf{Triangle}_N(G')$ counts such a triangle. (ii) If C forms a triangle in G, then $\#\mathsf{Triangle}_n(G)$ counts such a triangle. (iii) Otherwise, C consists of an edge $uv \in E(G)$ and a 2-path (a path of length 2) uwv for $w \notin V$. As this 2-path is contained in G', A'^2_{uv} counts such a triangle. Therefore, we obtain the claim. □

7.3 Proof of Theorem 1.5

Lemma 7.5. *Let $0 < p < 1$ be a constant. Let $k \in \mathbb{N}, \delta, \epsilon > 0$ be such that*

$$4 \exp\left(-\frac{k\delta}{32}\right) \leq \epsilon. \tag{2}$$

If there exists a $T(kn)$-time errorless algorithm $\mathcal{M}$ that solves $(\#\mathsf{Triangle}_{kn}, \mathcal{G}_{kn,p})$ with success probability ϵ, then, there exists an $O(T(kn)\epsilon^{-1}\log(1/\delta))$-time nonuniform errorless algorithm $\mathcal{M}'$ that solves $(\#\mathsf{Triangle}_n, \mathcal{G}_{n,p})$ with success probability $1-2\delta$.

Proof of Theorem 1.5. Let $\delta = C\log^7 n$ for a sufficiently large constant $C > 0$. We may assume $\epsilon \geq n^{-3}$ (otherwise, we can solve $\#\mathsf{Triangle}_n$ in time $O(n^3) = O(\epsilon^{-1})$). Then, the condition (2) holds for some $k = \mathrm{polylog}(n)$. From Lemmas 7.2 and 7.5, we obtain an $O(T(kn)\epsilon^{-1}\log(1/\delta))$-time randomized algorithm that solves $\#\mathsf{Triangle}_n$. □

The remainder of this subsection is devoted to proving Lemma 7.5.

Definition 7.2. *For k vertex-disjoint graphs $G_1 = (V_1, E_1), \ldots, G_k = (V_k, E_k)$, let $\langle G_1, \ldots, G_k\rangle$ be the random graph H obtained by using the following procedure.*

1. *Start from $H := (V_1 \cup \cdots \cup V_k, E_1 \cup \cdots \cup E_k)$.*
2. *For each $u \in V_i, v \in V_j$ with $1 \leq i < j \leq k$, add the edge $\{u, v\}$ to H with probability p independent of any other pairs. Then, return H.*

Note that, if $G_1, \ldots, G_k$ are i.i.d. samples from $\mathcal{G}_{n,p}$, then $\langle G_1, \ldots, G_k\rangle$ is a sample of $\mathcal{G}_{kn,p}$. We consider the following reduction.

Randomized Reduction $\mathcal{R}^{\mathcal{O}}(G)$

We expect $\mathcal{O}$ to be an errorless algorithm. For $i \in [k]$, let $V_i = \{(i-1)n+1, \ldots, in\} \subseteq [kn]$.

1. Sample $i \sim [k]$ and $G_1, \ldots, G_k \sim \mathcal{G}_{n,p}$ independently. We assume that the vertex set of G_i is V_i.
2. Let $\overline{G} = \langle G_1, \ldots, G_{i-1}, G, G_{i+1}, \ldots, G_k\rangle$, where G is supposed to have vertex set V_i. This graph is the query.
3. let $G' = \left([kn], E(\overline{G}) \setminus \binom{V_i}{2}\right)$ and A' be the adjacency matrix of G'.
4. If $\mathcal{O}(\overline{G}) \neq \perp$, output $\mathcal{O}(\overline{G}) - \#\mathsf{Triangle}_N(G') - \sum_{uv \in E(G)} A'^2_{uv}$. Otherwise, output $\perp$.

The correctness (i.e., $\mathcal{R}^{\#\mathsf{Triangle}_{kn}}(G) = \#\mathsf{Triangle}_n(G)$) follows from Lemma 7.4. The query $\overline{G}$ has the same flavor of the reduction in the direct product theorem (cf. Lemma 4.3). We prove that the query graph of this reduction is a sampler using the same argument as the proof of Lemma 4.3.

Lemma 7.6. *The query graph of the reduction $\mathcal{R}^{\mathcal{O}}$ above is a (δ, c)-sampler for density ϵ for any $\delta, \epsilon, c > 0$ that satisfies*

$$2\exp\left(-\frac{kc^2\delta}{8}\right) \leq c\epsilon.$$

Proof. Let $Q = (X, Y, W)$ be the query graph associated with the Markov operator P, where X (resp. Y) is the set of all n-vertex (resp. kn-vertex) graphs. Write $\mu = \mathcal{G}_{n,p}$ and $\nu = \mathcal{G}_{kn,p}$. Our plan is to show that $Q^* = (Y, X, W^\top)$ is a sampler and then apply Lemma 3.7.

For $G \in X$ and $\overline{G} \in Y$, let $e(G, \overline{G}) = |\{i \in [k] \colon \overline{G}[V_i] = G\}|$. Then, we have $P(G, \overline{G}) = \frac{e(G,\overline{G})}{k} \cdot \frac{\nu(\overline{G})}{\mu(G)}$ and thus the adjoint operator P^* is given by $P^*(\overline{G}, G) = \frac{e(G,\overline{G})}{k}$. In other words, $P^*(\overline{G}, \cdot)$ is the distribution of $\overline{G}[V_i]$ for $i \sim [k]$. Let $v \in [0,1]^X$ be a δ-dense measure. For

$\overline{G} \in Y$, $P^* v(\overline{G}) = \mathbf{E}_{G|\overline{G}}[v(G)] = \frac{1}{k}\sum_{i\in[k]} v(\overline{G}[V_i])$. If $\overline{G} \sim \nu$, then the graphs $\overline{G}[V_1], \ldots, \overline{G}[V_k]$ are i.i.d. sampled from $\mathcal{G}_{n,p}$. Therefore, by Lemma 3.2, we have that Q^* is an (ϵ', c')-sampler for density δ for any $\delta, \epsilon', c' > 0$ that satisfy $\epsilon' \geq \exp(-0.5c'^2\delta k)$. From Lemma 3.7 and setting $\epsilon' = 0.5c\epsilon$ and $c' = 0.5c$, we obtain the claim. □

Unfortunately, the running time of $\mathcal{R}^{\mathcal{O}}$ is $O(T(kn) + (kn)^\omega)$ since we compute $\#\mathsf{Triangle}_{kn}(G')$ and A'^2 at Step 4. To avoid the running time of $O((kn)^\omega)$, we observe that $\#\mathsf{Triangle}_{kn}(G')$ and A'^2 are *independent of the input*. Thus, we can give these to $\mathcal{R}^{\mathcal{O}}$ as advice. We show how to use this advice to simulate $\mathcal{R}^{\mathcal{O}}$ in time $O(T(kn) + (kn)^2)$.

Proof of Lemma 7.5. By Lemma 7.6 and the assumption (2), the query graph of $\mathcal{R}^{\mathcal{O}}$ is a $(\delta, 1/2)$-sampler for density ϵ. Note that $\mathcal{M}$ solves $(\#\mathsf{Triangle}_{kn}, \mathcal{G}_{kn,p})$ with success probability ϵ. Then, from Theorem 4.1, $\mathcal{R}^{\mathcal{M}}$ is an errorless algorithm that computes $\#\mathsf{Triangle}_n(G)$ with probability $\epsilon/2$ (over the random seed of $\mathcal{R}$) for a $(1-\delta)$-fraction of $G \sim \mathcal{G}_{n,p}$. We begin by considering the following algorithm $\mathcal{N}$: Run $\mathcal{R}^{\mathcal{M}}(G)$ for $\ell = O\left(\frac{\log(1/\delta)}{\epsilon}\right)$ times. Whenever $\mathcal{R}^{\mathcal{M}}(\overline{G}) \neq \perp$, $\mathcal{N}$ outputs it and terminates. If $\mathcal{N}$ did not terminate during the iteration, $\mathcal{N}$ outputs $\perp$. Note that $\mathcal{N}$ satisfies $\Pr_G[\Pr_{\mathcal{N}}[\mathcal{N}(G) = \#\mathsf{Triangle}_n(G)] \geq 1-\delta] \geq 1-\delta$. Then, by Lemma 7.1, we can fix the randomness of $\mathcal{N}$ as advice and obtain a nonuniform errorless algorithm $\mathcal{N}'$ that satisfies $\Pr_{G\sim\mathcal{G}_{n,p}}[\mathcal{N}'(G;\alpha_n) = \#\mathsf{Triangle}_n(G)] \geq (1-\delta)^2 \geq 1-2\delta$. Here, the advice α_n consists of ℓ graphs $G'_1, \ldots, G'_\ell$ and ℓ indices $i_1, \ldots, i_\ell \in [k]$. Since $G'_1, \ldots, G'_\ell$ are graphs that do not depend on the input graph G, we can also provide $\mathcal{N}'$ with the adjacency matrices $A'_1, \ldots, A'_\ell$ of $G'_1, \ldots, G'_\ell$, their squares $A'^2_1, \ldots, A'^2_\ell$, and the values $\#\mathsf{Triangle}_{kn}(G'_1), \ldots, \#\mathsf{Triangle}_{kn}(G'_\ell)$ as advice. Using this as advice, our nonuniform algorithm $\mathcal{M}'$ runs as follows.

Nonuniform Errorless algorithm $\mathcal{M}'(G)$

Let $\ell := \left\lceil \frac{2\log(1/\delta)}{\epsilon} \right\rceil$. As advice, $\mathcal{M}'$ is given ℓ graphs $G'_1, \ldots, G'_\ell$ (as adjacency matrices $A'_1, \ldots, A'_\ell$), indices $i_1, \ldots, i_\ell \in [k]$, $\#\mathsf{Triangle}_{kn}(G'_1), \ldots, \#\mathsf{Triangle}_{kn}(G'_\ell) \in \mathbb{N}$, and matrices $A'^2_1, \ldots, A'^2_\ell \in \mathbb{N}^{kn\times kn}$. Here each G'_j is some kn-vertex graph over the vertex set $\overline{V} := [kn]$. For $i \in [k]$, let $V_i = \{(i-1)n+1, \ldots, in\}$. We assume G'_j does not contain any edges inside V_j. The input is an n-vertex graph G.

1. For each $j \in [\ell]$:
   (a) Let $G_0 = G'_j$ and replace $G_0[V_{i_j}]$ with G. Let $\overline{G}$ be the resulting graph.
   (b) If $\mathcal{M}(\overline{G}) \neq \perp$, output $\mathcal{M}(\overline{G}) - \#\mathsf{Triangle}_{kn}(G'_j) - \sum_{uv\in E(G)}(A'^2_j)_{uv}$ and terminate.
2. Output $\perp$ and terminate.

Note that $\mathcal{M}'$ runs in time $O(\ell(T(kn) + kn^2)) = O(\ell T(kn))$. The correctness of $\mathcal{M}'$ folllows from the correctness of the ideal algorithm $\mathcal{N}'$, which succeeds on a $(1-2\delta)$-fraction of $G \sim \mathcal{G}_{n,p}$. □

7.4 Proof of Theorem 1.4

We first invoke the following worst-case to average-case reduction.

Lemma 7.7 (Theorem 2.9 of [BBB21])**.** *Let $0 < p < 1$ be any constant. There exists a universal constant $c > 0$ that satisfies the following: If there exists a $T(n)$-time algorithm $\mathcal{M}$ that solves $(\oplus\mathsf{Triangle}_n, \mathcal{G}_{n,p})$ with success probability $1-c$, then, there exists a randomized $O(T(n)\log n)$-time randomized algorithm $\mathcal{M}'$ that solves $\oplus\mathsf{Triangle}_n$.*

Remark 5. *In the special case of $p = 1/2$, Goldreich [Gol20, Theorem 2] gave a linear time reduction from $\oplus\mathsf{Triangle}$ to $(\oplus\mathsf{Triangle}_n, \mathcal{G}_{n,1/2})$ with a constant error tolerance.*

Lemma 7.8. *Let $0 < p < 1$ be a constant. Let $k \in \mathbb{N}, \epsilon > 0, \delta > 0$ be such that*

$$16 \exp\left(-\frac{k\epsilon^2\delta}{128}\right) \leq \epsilon. \tag{3}$$

Suppose there exists a $T(kn)$-time algorithm $\mathcal{M}$ that solves $(\oplus\mathsf{Triangle}_{kn}, \mathcal{G}_{kn,p})$ with success probability $1/2 + \epsilon$. Then, there exists an $O(T(kn)\epsilon^{-2}\log(1/\epsilon))$-time nonuniform algorithm $\mathcal{M}'$ that solves $(\oplus\mathsf{Triangle}_n, \mathcal{G}_{n,p})$ with success probability $1 - \delta$.

Proof of Theorem 1.4. Set $\delta = c$, where c is the constant from Lemma 7.7. Then, (3) holds for some $k = O(\epsilon^{-2}\log(1/\epsilon))$. From Lemmas 7.7 and 7.8, we obtain the claim. □

Proof of Lemma 7.8. Consider the following randomized reduction:

Randomized Reduction $\mathcal{R}^{\mathcal{O}}(G)$

For $i \in [k]$, let $V_i = \{(i-1)n + 1, \ldots, in\} \subseteq [kn]$.

1. Sample $i \sim [k]$ and k independent random graphs $(G_1, \ldots, G_k) \sim (\mathcal{G}_{n,p})^k$ where each G_i is supposed to have vertex set V_i.
2. Let $\overline{G} = \langle G_1, \ldots, G_{i-1}, G, G_{i+1}, \ldots, G_k\rangle$, where G is supposed to have vertex set V_i. This graph is the query.
3. let G' be the graph obtained from $\overline{G}$ by removing all edges inside V_i. Let A' be the adjacency matrix of G'.
4. Output $\mathcal{O}(\overline{G}) - \oplus\mathsf{Triangle}(G') - \sum_{uv\in E(G)} A'^2_{uv}$ (over $\mathbb{F}_2$).

The correctness of the reduction follows from Lemma 7.4. By Lemma 7.6 and (3), the query graph is a $(\delta, \epsilon/4)$-sampler for density $1/2+\epsilon$. By Corollary 4.2, there is an $O(T(kn)\epsilon^{-2}\log(1/\epsilon) + \epsilon^{-2}(kn)^\omega)$-time algorithm that solves $(\oplus\mathsf{Triangle}(G), \mathcal{G}_{n,p})$ with success probability $1 - \delta$. We can shave off the term $(kn)^\omega$ in the running time using nonuniform advice by the same way as in the previous subsection. Specifically, by Lemma 7.1, we give the adjacency matrix A' of G' as advice (instead of the random seed). We also give $\oplus\mathsf{Triangle}(G')$ and A'^2 as advice to shave off the $O((kn)^\omega)$ term. Then, we obtain a nonuniform $O(T(kn)\epsilon^{-2}\log(1/\epsilon))$-time algorithm. □

8 Planted Clique

For a pair (H, C) of a graph $H = (V, E)$ and a subset $C \subseteq V$, let $H_C = (V, E \cup \binom{C}{2})$. For parameters $n, k \in \mathbb{N}$, let $\mathcal{G}_{n,1/2,k}$ be the distribution of H_C for $H \sim \mathcal{G}_{n,1/2}$ and $C \sim \binom{[n]}{k}$. In this section, n denotes the input size, $N = N(n)$ denotes the size of a query of the reduction that we consider[5], $k = k(n)$ denotes the clique size, and $\epsilon = \epsilon(N)$ denotes the success probability of an average-case solver $\mathcal{M}$ (over the input distribution $\mathcal{G}_{N,1/2,k}$). Our goal is to present an algorithm $\mathcal{M}'$ that solves the planted clique problem over $\mathcal{G}_{n,1/2,k}$ with success probability $1 - \delta(n)$. Note that ϵ is a function on N. For example, an $O(T(n)/\epsilon(N))$-time algorithm runs in time $O(T(n)/\epsilon(N(n)))$ on an n-vertex graph. We write $V := [n]$ and $\overline{V} := [N]$ to denote the vertex sets of the input and

[5]We always assume $N = \mathrm{poly}(n)$ because we are interested in polynomial-time reductions.

query, respectively. It is widely known that $\mathcal{G}_{n,1/2,k}$ contains a unique clique with high probability.

Lemma 8.1. *For any $n, k \in \mathbb{N}$,*

$$\Pr_{G \sim \mathcal{G}_{n,1/2,k}}[G \text{ contains a unique } k\text{-clique}] \geq 1 - 2kn2^{-k/2}.$$

This result directly follows from the proof of [Jer92, Theorem 4] but for completeness we prove it in Appendix A. We encourage the readers to think of k as being $k \gg \log n$ in which case the $2kn2^{-k/n}$ term is negligible everywhere. We consider the following reduction.

Reduction $\mathcal{R}^{\mathcal{O}}(G)$

1. Pick up a uniformly random injection $\phi\colon V \to \overline{V}$.
2. Initialize $\overline{G} = (\overline{V}, \emptyset)$.
3. For every edge $\{u, v\} \in E$, add the edge $\{\phi(u), \phi(v)\}$ to $E(\overline{G})$.
4. For each $\{u, v\} \in \binom{V}{2} \setminus \binom{\phi(V)}{2}$, add the edge $\{u, v\}$ to $\overline{G}$ with probability $1/2$ independent to any other pairs.
5. Output $\mathcal{O}(\overline{G})$.

Note that, if $G \sim \mathcal{G}_{n,1/2,k}$, then the distribution of the graph $\overline{G}$ generated by $\mathcal{R}^{\mathcal{O}}(G)$ is $\mathcal{G}_{N,1/2,k}$.

Lemma 8.2. *For a $(1 - 2\sqrt{kN}2^{-k/4})$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$, the N-vertex graph $\overline{G}$ produced by Steps 2–4 of $\mathcal{R}^{\mathcal{O}}(G)$ contains a unique k-clique with probability $1 - \sqrt{2kN}2^{-k/4}$.*

Proof. For $G \sim \mathcal{G}_{n,1/2,k}$, let $Z = \Pr_{\mathcal{R}}[\overline{G} \text{ has a unique } k\text{-clique}]$ be a random variable. Note that $\mathbf{E}[Z] \geq 1 - \gamma$ for $\gamma = 2kN2^{-k/2}$ by Lemma 8.1. Then, by the Markov inequality, we have $\Pr[Z \leq 1 - \sqrt{\gamma}] \leq \mathbf{E}[1 - Z]/\sqrt{\gamma} \leq \sqrt{\gamma} = \sqrt{2kN}2^{-k/4}$. □

Suppose that $\overline{G}$ is very likely to contain a unique k-clique. By Lemma 8.2, if $\mathcal{O}(\overline{G})$ detects some k-clique C, then the clique is most likely to be the one planted in G. Therefore, the planted clique can be recovered using $\phi^{-1}(C)$.

8.1 Sampler Property of $\mathcal{R}^{\mathcal{O}}$

Lemma 8.3. *For any $0 < c < 1$, the query graph of $\mathcal{R}^{\mathcal{O}}$ is a $\left(\frac{n}{c^2\epsilon N} - 4kn2^{-k/2}, c\right)$-sampler for density ϵ.*

To prove Lemma 8.3, we need more notations. A *marked graph* is a pair $H = (G, C)$ of a graph G and a vertex subset $C \subseteq V(G)$ such that the induced subgraph $G[C]$ forms a clique. We call G *underlying graph* and C *mark*. Let X (resp. Y) be the set of all n-vertex (resp. N-vertex) marked graphs with a mark of size k. Let $\mu \in [0,1]^X$ (resp. $\nu \in [0,1]^Y$) be the uniform distributions over X (resp. Y). We can sample $x \sim \mu$ by selecting $C \sim \binom{V}{k}$ and $G \sim \mathcal{G}_{n,1/2}$ and thereafter setting $x = (G_C, C)$. Therefore, we can sample $\mathcal{G}_{n,1/2,k}$ by taking the underlying graph of $x \sim X$. We consider the following auxiliary reduction $\mathcal{R}_1^{\mathcal{O}}$.

Auxiliary Reduction $\mathcal{R}_1^{\mathcal{O}}(x)$

Given a marked graph $x = (G, C) \in X$, let $G = (V, E)$ be the underlying graph and do the following:

1. Pick up a uniformly random injection $\phi\colon V \to \overline{V}$.
2. Initialize $\overline{G} = (\overline{V}, \emptyset)$.
3. For every edge $\{u, v\} \in E$, add $\{\phi(u), \phi(v)\}$ to $\overline{G}$.
4. For each $\{u, v\} \in \binom{\overline{V}}{2} \setminus \binom{\phi(V)}{2}$, add the edge $\{u, v\}$ to $\overline{G}$ with probability $1/2$ independent to any other pairs.
5. Let $(\overline{G}, \phi(C)) \in Y$ be the query.

Lemma 8.4. *If the query graph of $\mathcal{R}_1^{\mathcal{O}}$ is a (δ, c)-sampler for density ϵ, then the query graph of $\mathcal{R}^{\mathcal{O}}$ is a $(\delta - 4kn2^{-k/2}, c)$-sampler for density ϵ.*

Proof. For an n-vertex graph G, we denote by $\mathbf{E}_{\overline{G}|G}[\cdot]$ the expectation over the random query $\overline{G}$ generated by $\mathcal{R}^{\mathcal{O}}(G)$. Similarly, for a marked graph $H \in X$, we use $\mathbf{E}_{\overline{H}|H}[\cdot]$ to denote the expectation over the query $\overline{H}$ of $\mathcal{R}_1^{\mathcal{O}}(H)$. Fix any ϵ-dense measure w over the support of $\mathcal{G}_{N,1/2,k}$. We say that an n-vertex graph G is *good* if $\mathbf{E}_{\overline{G}|G}[w(\overline{G})] \geq (1-c)\epsilon$. Our goal is to show that G is good for a $(1 - \delta - 4kn2^{-k/2})$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$.

Define the measure $\nu_1 \in [0,1]^Y$ by $\nu_1(\overline{H}) = \nu(\overline{G})$, where $\overline{G}$ is the underlying graph of $\overline{H}$. Then, ν_1 is ϵ-dense since $\mathbf{E}_{\overline{G}}[\nu(\overline{G})] = \mathbf{E}_{\overline{H}\sim Y}[\nu_1(\overline{H})]$. We say that H is *good* if $\mathbf{E}_{\overline{H}|H}[\nu_1(\overline{H})] \geq (1-c)\epsilon$. Since Q_1 is a sampler, a $(1-\delta)$-fraction of $H \sim X$ is good.

Let U be the set of n-vertex graphs G containing a unique k-clique. By Lemma 8.1, we have $\Pr_{G\sim\mathcal{G}_{n,1/2,k}}[G \in U] \geq 1 - 2kn2^{-k/2}$. Let U_1 be the set of marked graphs $H \in X$ such that the underlying graph contains a unique k-clique. Then, there is a one-to-one correspondence between U and U_1: A graph $G \in U$ corresponds to a marked graph $(G, C) \in U_1$, where C is the unique k-clique of G. Conversely, the marked graph $(G, C) \in U_1$ corresponds to the underlying graph $G \in U$. From this correspondence, $G \in U$ is good if and only if $H \in U_1$ is good, and we have

$$\begin{aligned}
\Pr_{G\sim\mathcal{G}_{n,1/2,k}}[G \text{ is good}] &\geq \Pr_{G\sim\mathcal{G}_{n,1/2,k}}[G \text{ is good} | G \in U] \Pr_{G\sim\mathcal{G}_{n,1/2,k}}[G \in U] \\
&\geq \Pr_{H\sim X}[H \text{ is good} | H \in U_1] - \Pr_{G}[G \notin U] \\
&\geq \Pr_{H\sim X}[H \text{ is good}] - \Pr_{H\sim X}[H \notin U_1] - \Pr_{G\sim\mathcal{G}_{n,1/2,k}}[G \notin U] \\
&\geq 1 - \delta - 4kn2^{-k/2}.
\end{aligned}$$

□

In this paper, we present two results that demonstrate the sampler property of the query graph of $\mathcal{R}_1^{\mathcal{O}}$.

Proposition 8.5. *For any $0 < c < 1$, $0 < \epsilon < 1$, the query graph of $\mathcal{R}_1^{\mathcal{O}}$ is a $\left(\frac{4n}{\sqrt{c^3 N\epsilon}}, c\right)$-sampler for density ϵ.*

Lemma 8.6. *For any $0 < c < 1$ and $0 < \epsilon < 1$, the query graph of $\mathcal{R}_1^{\mathcal{O}}$ is a $\left(\frac{n}{c^2\epsilon N}, c\right)$-sampler for density ϵ.*

For example, if δ, ϵ are constants, then Lemma 8.6 implies that the query graph of $\mathcal{R}_1^{\mathcal{O}}$ is a (δ, c)-sampler for some $N = O(n)$, while Proposition 8.5 implies the same for some $N = O(n^2)$. Although there is a cost of a quadratic blow-up in N, there is a simple proof of Proposition 8.5 (we defer this to Appendix A).

The key tool to prove Lemma 8.6 is the spectral bound from a shrinking coupling [Che98]. Recall that a *coupling* of two distributions $\mathcal{D}_1, \mathcal{D}_2$ is a pair of random variables (X_1, X_2) where the marginal distribution of X_i is $\mathcal{D}_i$ for $i = 1, 2$.

Lemma 8.7 (Theorem 13.1 of [LP17])**.** *Let $(V, \mathrm{dist}(\cdot,\cdot))$ be a metric space and $P \in [0,1]^{V \times V}$ be a reversible Markov operator. If for every $x_1, x_2 \in V$ there exists a coupling (x_1', x_2') of $P(x_1, \cdot)$ and $P(x_2, \cdot)$ that satisfies $\mathbf{E}[\mathrm{dist}(x_1', x_2')] \le \theta \cdot \mathrm{dist}(x_1, x_2)$, then, $\lambda(P) \le \theta$.*

Proof of Lemma 8.6. Let $Q = (X, Y, W)$ be the query graph of $\mathcal{R}_1^{\mathcal{O}}$ associated with the Markov operator P. We apply Lemma 8.7 to obtain an upper bound on $\lambda(PP^*)$ and thereafter combine Lemmas 3.5 and 3.6, where P^* denotes the adjoint of P. Consider the following sampling procedure $\mathcal{S}(\overline{H})$ that is given a marked graph $\overline{H} \in Y$.

Sampling Procedure $\mathcal{S}(\overline{H})$

Given a marked graph $\overline{H} = (\overline{G}, \overline{C})$, select an injection $\phi' \colon V \to \overline{V}$ uniformly at random conditioned on $\phi'(V) \supseteq \overline{C}$. Then, output $H' = (G', C') \in X$ for $G' = \phi'^{-1}(\overline{G}[\phi'(V)])$ and $C' = \phi'^{-1}(\overline{C})$.

Claim 8.8. *For the sampling procedure $\mathcal{S}$ above, $\Pr_{\mathcal{S}}[\mathcal{S}(\overline{H}) \text{ outputs } H] = P^*(\overline{H}, H)$.*

Proof. Let $\mathrm{Inj}(V, \overline{V})$ be the set of all injections from V to $\overline{V}$. An *embedding* from G to $\overline{G}$ is an injection $\phi \colon V \to \overline{V}$ that preserves the adjacency (i.e., $\{u, v\} \in E(G)$ if and only if $\{\phi(u), \phi(v)\} \in E(\overline{G})$). For $H = (G, C) \in X$ and $\overline{H} = (\overline{G}, \overline{C}) \in Y$, let $\mathrm{EMB}(H, \overline{H})$ be the set of embeddings ϕ from G to $\overline{G}$ such that $\phi(C) = \overline{C}$. Let $\mathrm{emb}(G, \overline{G}) = \frac{|\mathrm{EMB}(G, \overline{G})|}{|\mathrm{Inj}(V, \overline{V})|}$.

Fix $H = (G, C) \in X$ and $\overline{H} = (\overline{G}, \overline{C}) \in Y$. By the definition of $\mathcal{R}^{\mathcal{O}}$, we have

$$P(H, \overline{H}) = \mathrm{emb}(H, \overline{H}) \cdot (1/2)^{\binom{N}{2} - \binom{n}{2}} = \mathrm{emb}(H, \overline{H}) \cdot \frac{\binom{N}{k}}{\binom{n}{k}} \cdot \frac{\nu(\overline{H})}{\mu(H)}.$$

Thus $P^*(\overline{H}, H) = \mathrm{emb}(H, \overline{H}) \cdot \frac{\binom{N}{k}}{\binom{n}{k}} = \mathrm{emb}(H, \overline{H}) \cdot \frac{\binom{N}{n}}{\binom{N-k}{n-k}}$. By the definition of $\mathcal{S}$, we have

$$\begin{aligned}
\Pr[\mathcal{S}(\overline{H}) = H] &= \Pr_{\phi' \sim \mathrm{Inj}(V, \overline{V})}[\phi' \in \mathrm{EMB}(H, \overline{H}) | \phi'(V) \supseteq \overline{C}] \\
&= \frac{\mathrm{emb}(H, \overline{H})}{\Pr_{\phi' \sim \mathrm{Inj}(V, \overline{V})}[\phi'(V) \supseteq \overline{C}]} \\
&= \mathrm{emb}(H, \overline{H}) \cdot \frac{\binom{N}{n}}{\binom{N-k}{n-k}} \\
&= P^*(\overline{H}, H).
\end{aligned}$$

□

To apply Lemma 8.7, we define a metric $\mathrm{dist}\colon X \times X \to \mathbb{R}$ by

$$\mathrm{dist}(H_1, H_2) = \begin{cases} d_{\mathrm{ham}}(H_1, H_2) & \text{if } H_1 \text{ and } H_2 \text{ have the same mark,} \\ \infty & \text{otherwise.} \end{cases}$$

where $d_{\mathrm{ham}}(\cdot,\cdot)$ denotes the Hamming distance.[6] We construct a coupling satisfying the condition of Lemma 8.7 for the metric $\mathrm{dist}(\cdot,\cdot)$.

Let $\mathcal{S}'$ be the composition of $\mathcal{R}_1^{\mathcal{O}}$ and $\mathcal{S}$ (see Figure 4). Namely, $\mathcal{S}'(H)$ runs the query-making part of $\mathcal{R}_1^{\mathcal{O}}(H)$ to produce a query $\overline{H} \in Y$ and then output $H' = \mathcal{S}(\overline{H})$. By Claim 8.8, $\mathcal{S}'(x)$ samples the distribution $PP^*(x,\cdot)$. For $H = (G, C) \in X$ and a permutation π over V, let $\pi(H) = (\pi(G), \pi(C)) \in X$. Note that, for any fixed π, $\mathcal{S}'(\pi(x))$ also samples the distribution $PP^*(x,\cdot)$. Consider the following coupling:

Coupling

Let $H_1 = (G_1, C_1)$ and $H_2 = (G_2, C_2)$ be inputs.

1. If $C_1 = C_2$, let π be the identity permutation. Otherwise, take any permutation π over V such that $\pi(C_1) = C_2$.
2. Sample a random seed r for $\mathcal{S}'$.
3. Output $(H_1', H_2') = (\mathcal{S}'(\pi(H_1); r), \mathcal{S}'(H_2; r))$. Here, we use the common random seed r for $\mathcal{S}'$.

Claim 8.9. *For the output (H_1', H_2') of the coupling above, $\mathbf{E}[\mathrm{dist}(H_1', H_2')] \le \frac{n}{N}\mathrm{dist}(H_1, H_2)$.*

Proof. Write $H_i = (G_i, C_i)$ and $H_i' = (G_i', C_i')$. Consider two cases:

Case 1: $C_1 \neq C_2$**.** Note that $C_1' = C_2'$ regardless of $C_1 \neq C_2$ or not; thus, we have $\mathrm{dist}(x_1, x_2) = \infty$ and $\mathrm{dist}(x_1', x_2') \le \binom{n}{2}$. Therefore, the statement holds.

Case 2: $C_1 = C_2$**.** Let $\Delta(H_1, H_2) := (E(G_1) \setminus E(G_2)) \cup (E(G_2) \setminus E(G_1))$ be the symmetric difference of the underlying graphs. We aim to show $\mathbf{E}[|\Delta(H_1', H_2')|] \le \frac{n}{N}|\Delta(H_1, H_2)|$. Let $\overline{H}_i = (\overline{G}_i, \overline{C}_i)$ be the random query generated by $\mathcal{R}_1^{\mathcal{O}}(H_i)$. By assumption, the permutation π of Step 1 is the identity and thus $\overline{C}_1 = \overline{C}_2$.

Since $\mathcal{R}_1^{\mathcal{O}}(x_1)$ and $\mathcal{R}_1^{\mathcal{O}}(x_2)$ share the random seed, we have $\Delta(\overline{H}_1, \overline{H}_2) = \{\{\phi(u), \phi(v)\} \colon \{u, v\} \in \Delta(H_1, H_2)\}$, where $\phi \in \mathrm{Inj}(V, \overline{V})$ is the random injection used in $\mathcal{R}_1^{\mathcal{O}}$. Fix $\{a, b\} \in \Delta(\overline{H}_1, \overline{H}_2)$ and consider $\Pr[\{a, b\} \in \Delta(H_1', H_2')]$. Let $\phi' \in \mathrm{Inj}(V, \overline{V})$ be a random injection used in $\mathcal{S}$ to construct H_1' and H_2'. Note that $\{a, b\} \not\subseteq \overline{C}_1$ (since $\overline{C}_1 = \overline{C}_2$); thus we may assume $a \notin \overline{C}_1$. Then,

$$\Pr[\{a, b\} \in \Delta(H_1', H_2')] \le \Pr_{\phi'}[\{a, b\} \subseteq \phi'(V)] \le \Pr_{\phi'}[a \in \phi'(V)] \le \frac{n}{N}$$

and we obtain $\mathbf{E}[|\Delta(H_1', H_2')|] \le \frac{n}{N}|\Delta(H_1, H_2)|$. □

Now, we return to the proof of Lemma 8.6. By Lemma 8.7 and Claim 8.9, $\lambda(PP^*) \le \frac{n}{N}$. By Lemma 3.5, the query graph Q_1 is a $\sqrt{\frac{n}{N}}$-expander. Then, by Lemma 3.6, for any $c, \epsilon > 0$, Q_1 is a $\left(\frac{n}{c^2\epsilon N}, c\right)$-sampler for density ϵ. □

Proof of Lemma 8.3. Combine Lemmas 8.4 and 8.6. □

[6] The proof works even if we replace ∞ with a sufficiently large number, say, 2^n.

8.2 Search Problem

For a randomized algorithm $\mathcal{M}$, we say that $\mathcal{M}(G)$ *finds a k-clique* if $\mathcal{M}(G)$ outputs a k-clique in G with probability at least $2/3$ over the internal randomness of $\mathcal{M}$. In this subsection, we assume that $\mathcal{M}$ knows the clique size k.

Theorem 8.10. *Let $k = k(n), N = N(n), \delta = \delta(n), \epsilon = \epsilon(N)$ be functions that satisfy $N\epsilon \geq 4n\delta^{-2}$ and $\epsilon \geq 6\sqrt{2kN}2^{-k/4}$. Suppose there exists a $T(N)$-time algorithm $\mathcal{M}$ that satisfies*

$$\Pr_{G\sim\mathcal{G}_{N,1/2,k}}[\mathcal{M}(G) \text{ finds a } k\text{-clique}] \geq \epsilon.$$

Then, there exists a randomized algorithm $\mathcal{M}'$ than runs in time $O(T(N)/\epsilon)$ on an n-vertex graph G and satisfies

$$\Pr_{G\sim\mathcal{G}_{n,1/2,k}}\left[\mathcal{M}'(G) \text{ finds a } k\text{-clique}\right] \geq 1 - \delta - \sqrt{2kN}2^{-k/4}.$$

Proof. Let $\mathcal{R}^{\mathcal{O}}$ be the reduction given at the beginning of Section 8. Our algorithm $\mathcal{M}'(G)$ runs $\mathcal{R}^{\mathcal{M}}(G)$ for $\ell := \lceil 10\epsilon^{-1}\rceil$ times. During the iteration, once $\mathcal{M}(\overline{G})$ outputs a k-clique $\overline{C} \subseteq \phi(V)$, then output $\phi^{-1}(\overline{C})$ and terminate. If the algorithm does not terminate during the reduction, then output $\perp$. Let $\mathcal{U}$ be the event that the large graph $\overline{G}$ contains a unique k-clique. If $\mathcal{U}$ occurs and $\mathcal{M}'(G)$ outputs a k-clique, then the clique must be the planted one in G; thus $\mathcal{M}'(G)$ succeeds. By Lemmas 8.2 and 8.3 (with setting $c = 1/2$), for a $(1 - \delta - \sqrt{2kN}2^{-k/4}))$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$, $\Pr_{\overline{G}}[\mathcal{U}$ occurs and $\mathcal{M}(\overline{G})$ finds a k-clique$] \geq \epsilon/2 - \sqrt{2kN}2^{-k/4} \geq \epsilon/3$. Therefore, for such G, $\mathcal{M}'(G)$ outputs a k-clique of G with probability $1 - (1 - \epsilon/6)^{10\ell} \geq 2/3$. □

8.3 Decision Problem

In the decision version of the planted clique, we consider the following two notions:

Definition 8.1. *Let $\mathcal{M}$ be a randomized algorithm that outputs either 0 or 1 and $\mathcal{D}_1, \mathcal{D}_2$ be two input distributions.*

- *We say that $\mathcal{M}$ has a* predicting advantage θ *on $\mathcal{D}_1$ and $\mathcal{D}_2$ if*

$$\Pr_{\substack{x\sim\mathcal{D}_1\\ \mathcal{M}}}[\mathcal{M}(x) = 1] \geq \frac{1+\theta}{2}, \qquad \Pr_{\substack{x\sim\mathcal{D}_2\\ \mathcal{M}}}[\mathcal{M}(x) = 1] \leq \frac{1-\theta}{2}.$$

- *We say that $\mathcal{M}$ has a* distinguishing advantage η *on $\mathcal{D}_1$ and $\mathcal{D}_2$ if*

$$\Pr_{\substack{x\sim\mathcal{D}_1\\ \mathcal{M}}}[\mathcal{M}(x) = 1] - \Pr_{\substack{x\sim\mathcal{D}_2\\ \mathcal{M}}}[\mathcal{M}(x) = 1] \geq \eta.$$

Clearly, the predicting advantage implies the same distinguishing advantage. Indeed, the converse holds at a cost of a small overhead in running time. We prove this in Appendix A.

Lemma 8.11. *Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ with a distinguishing advantage γ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$. Then, there exists an $O\left(\frac{\log^3 n}{\gamma^2}\cdot T(n)\right)$-time randomized algorithm $\mathcal{B}$ that has a predicting advantage $\gamma/4 - n^{-\omega(\log n)}$ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$ without knowing the clique size k.*

The main aim of this subsection is to boost the distinguishing advantage on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,k}$.

Theorem 8.12. *Let $k = k(n), N = N(n), \delta = \delta(n), \epsilon = \epsilon(N)$ be functions satisfying $N\epsilon^2 \geq \frac{36n}{\delta^2}$ and $\epsilon \geq 6\sqrt{2kN}2^{-k/4}$. Suppose there exists a $T(N)$-time randomized algorithm $\mathcal{M}$ with a distinguishing advantage ϵ on $\mathcal{G}_{N,1/2,k}$ and $\mathcal{G}_{N,1/2}$. Then, there exists a randomized algorithm $\mathcal{M}'$ with a predicting advantage $1-2\delta-8kN2^{-k/2}$ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$ that runs on an n-vertex graph in time $O\left(\frac{\log^3(n)\log(1/\delta)}{\epsilon^4} \cdot T(N)\right)$.*

Remark 6. *By inspecting the proof, we may assume that the algorithm $\mathcal{M}'$ is not necessarily given the clique size k as input. We will use this feature in Section 8.4.*

Proof. Since the planted clique can be solved in time $n^{O(\log n)}$, we may assume that $\epsilon \geq n^{-O(\log n)}$. Consider the reduction $\mathcal{R}^{\mathcal{O}}$ given at the beginning of Section 8. By Lemma 8.11, there exists a randomized $O\left(\frac{\log^3 n}{\epsilon^2} \cdot T(n)\right)$-time algorithm $\mathcal{B}$ with a predicting advantage $\epsilon/4 - n^{-\omega(\log n)} \geq \epsilon/5$ (for sufficiently large n). Our algorithm $\mathcal{M}'$ repeats $\mathcal{R}^{\mathcal{B}}(G)$ for $O(\epsilon^{-2}\log(1/\delta))$ times and then outputs the majority.

To see the correctness of $\mathcal{M}'$, we consider two cases: $G \sim \mathcal{G}_{n,1/2,k}$ and $G \sim \mathcal{G}_{n,1/2}$. Our plan is to show that the query graphs of these two cases are samplers. We then apply the Chernoff bound which implies the claim.

Case 1: $G \sim \mathcal{G}_{n,1/2,k}$**.** By Lemma 8.3, the query graph of $\mathcal{R}^{\mathcal{O}}$ is a $\left(\frac{2n}{c^2N} - 4kN2^{-k/2}, c\right)$-sampler for density $\frac{1}{2}+\epsilon$. Note that $\mathcal{M}$ outputs 1 for a $(1/2+\epsilon/2)$-fraction of $\mathcal{G}_{N,1/2,k}$. Set $c = \epsilon/3$ and thus $\frac{2n}{c^2N} \leq \frac{\delta}{2}$. For a $(1-\delta/2-4kN2^{-k/2})$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$, we have $\mathcal{R}^{\mathcal{B}}(G) = 1$ with probability $\frac{(1-c)(1+\epsilon)}{2} \geq \frac{1+\epsilon/3}{2}$ over the internal randomness of $\mathcal{R}^{\mathcal{B}}$. By the Chernoff bound (Lemma 3.2), $\Pr_{\mathcal{M}'}[\mathcal{M}'(G) = 1] \geq 1 - \delta/2$ for a $(1 - \delta/2 - 4kN2^{-k/2})$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$. Therefore, we have $\Pr_{G,\mathcal{M}'}[\mathcal{M}'(G) = 1] \geq 1 - \delta - 4kN2^{-k/2}$.

Case 2: $G \sim \mathcal{G}_{n,1/2}$**.** Consider the auxiliary reduction $\mathcal{R}_1^{\mathcal{O}}$ (in Section 8.1) for $k = 0$. Then the query graphs of $\mathcal{R}^{\mathcal{O}}$ and $\mathcal{R}_1^{\mathcal{O}}$ are isomorphic. Therefore, from Lemma 8.6 (note that we do not make any assumption on k in Lemma 8.6), we have that the query graph of $\mathcal{R}^{\mathcal{O}}$ is a $\left(\frac{2n}{c^2N}, c\right)$-sampler for density $\frac{1}{2}+\epsilon$. (Note that we can identify a graph G with the marked graph $(G, \emptyset)$ with the empty mark and thus the query graphs of $\mathcal{R}^{\mathcal{O}}$ and $\mathcal{R}_1^{\mathcal{O}}$ are isomorphic.) Using the same argument as in Case 1, we have $\Pr_{G,\mathcal{M}'}[\mathcal{M}'(G) = 0] \geq 1 - \delta - 4kN2^{-k/2}$ for a sufficiently large n.

From Cases 1 and 2, we can conclude that $\mathcal{M}'$ has a distinguishing advantage $1-\delta$ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$. □

Proof of Theorem 1.1. Let δ, ϵ be constants and apply Theorem 8.12. Note that $N = O(n)$ and thus $k \geq 3\log N = 3\log n + O(1)$. In this case, $kn2^{-k/2} = n^{-\Omega(1)}$ is negligible. □

8.4 Search-to-Decision

We consider a slightly different setting in which our average-case solver $\mathcal{M}$ is supposed to have a distinguishing advantage on $\mathcal{G}_{n,1/2}$ and $\mathcal{G}_{n,1/2,k'}$ for *all* $k' \geq k$ without knowing the clique size k'. In this setting, Alon et al. [AAKMRX07] presented the following search-to-decision reduction for the planted clique. For completeness, we present a proof sketch in Appendix A.

Lemma 8.13. *Let $k \geq 108\log_2 n$. Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ with a predicting advantage $1 - 1/n^2$ on $\mathcal{G}_{n,1/2,k'}$ and $\mathcal{G}_{n,1/2}$ for all $k' \geq k/3$ without knowing k'. Then, there exists an $O(nT(n))$-time algorithm $\mathcal{M}'$ that finds a k-clique for a $(1-O(1/n))$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$.*

By combining Lemma 8.13 with Theorem 8.12, we obtain the following search-to-decision reduction:

Theorem 8.14. *Let $k = k(n), N = N(n), \delta = \delta(n), \epsilon = \epsilon(N)$ be functions that satisfy $N\epsilon^2 \geq \frac{36n}{\delta}$ and $k \geq 108 \log_2 n$. Suppose there exists a $T(n)$-time randomized algorithm $\mathcal{M}$ with a distinguishing advantage ϵ on $\mathcal{G}_{N,1/2,k'}$ and $\mathcal{G}_{N,1/2}$ for all $k' \geq k$. Then, there exists an $O\left(\frac{n \log^3(n)}{\epsilon^4} \cdot T(N)\right)$-time randomized algorithm $\mathcal{M}'$ that finds a k-clique for a $(1 - O(1/n))$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$.*

Proof. By Lemma 8.11, there exists an algorithm $\mathcal{B}$ with a predicting advantage $\epsilon/4$ on $\mathcal{G}_{n,1/2,k'}$ and $\mathcal{G}_{n,1/2}$ for all $k' \geq k$ (note that $\mathcal{B}$ of Lemma 8.11 works without knowing k'). Let $\mathcal{R}^{\mathcal{O}}$ be the reduction provided at the beginning of this section and let $\mathcal{M}'$ be the algorithm that runs $\mathcal{R}^{\mathcal{B}}(G)$ for $\ell = O(\log(1/\delta)/\epsilon^2)$ times and then outputs the majority. Since the query graph of $\mathcal{R}^{\mathcal{O}}$ is a sampler in the both cases of $G \sim \mathcal{G}_{n,1/2}$ and $G \sim \mathcal{G}_{n,1/2,k'}$ (from Cases 1 and 2 in the proof of Theorem 8.12), we have that $\mathcal{M}'$ has a predicting advantage $1 - \delta$ in distinguishing between $\mathcal{G}_{n,1/2,k'}$ and $\mathcal{G}_{n,1/2}$. Set $\delta = 1/n^2$. By Lemma 8.13, we obtain the desired algorithm. □

Proof of Theorem 1.2. Apply Theorem 8.14. Note that the condition on N in Theorem 8.10 can be rewritten as $N^{2c_0} \geq 36n^3$; thus, $N = n^c$ for $c > \frac{5}{4c_0}$ satisfies this condition. □

Acknowledgement

Shuichi Hirahara is supported by JST, PRESTO Grant Number JPMJPR2024, Japan. Part of this work was completed while Shuichi Hirahara was a Research Fellow at the University of Warwick supported by the EPSRC New Horizons Grant EP/V048201/1. Nobutaka Shimizu is supported by JSPS KAKENHI Grant Number 21K21282, Japan.

References

[AAKMRX07] Noga Alon, Alexandr Andoni, Tali Kaufman, Kevin Matulef, Ronitt Rubinfeld, and Ning Xie. "Testing k-wise and almost k-wise independence". In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2007, pp. 496–505. DOI: 10.1145/1250790.1250863 (cit. on pp. 2, 3, 32, 40).

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. "Public-key cryptography from different assumptions". In: *Proceedings of Symposium on Theory of Computing (STOC)*. STOC '10. Cambridge, Massachusetts, USA: Association for Computing Machinery, 2010, pp. 171–180. DOI: 10.1145/1806689.1806715 (cit. on p. 2).

[AGGS22] Vahid R. Asadi, Alexander Golovnev, Tom Gur, and Igor Shinkar. "Worst-case to average-case reductions via additive combinatorics". In: *Proceedings of ACM Symposium on Theory of Computing (STOC)* (2022), pp. 1566–1574. DOI: 10.1145/3519935.3520041 (cit. on pp. 1, 2, 5, 7).

[AW21] Josh Alman and Virginia Vassilevska Williams. "A Refined Laser Method and Faster Matrix Multiplication". In: *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2021), pp. 522–539. DOI: 10.1137/1.9781611976465.32 (cit. on p. 5).

[BBB21] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. "The Average-Case Complexity of Counting Cliques in Erdös–Rényi Hypergraphs". In: *SIAM Journal on Computing* (2021), FOCS19-39-FOCS19–80. DOI: 10.1137/20M1316044 (cit. on pp. 1, 4, 11, 12, 23, 25).

[BBH18] Matthew Brennan, Guy Bresler, and Wasim Huleihel. "Reducibility and Computational Lower Bounds for Problems with Planted Sparse Structure". In: *Proceedings of the 31st Conference On Learning Theory*. Ed. by Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet. Vol. 75. Proceedings of Machine Learning Research. PMLR, 2018, pp. 48–166 (cit. on pp. 2, 12).

[BGG93] Mihir Bellare, Oded Goldreich, and Shafi Goldwasser. "Randomness in interactive proofs". In: *Comput. Complexity* 3.4 (1993), pp. 319–354. DOI: 10.1007/BF01275487 (cit. on p. 12).

[BKS17] Christoph Berkholz, Jens Keppeler, and Nicole Schweikardt. "Answering Conjunctive Queries under Updates". In: *Proceedings of the 36th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. PODS '17. Chicago, Illinois, USA: Association for Computing Machinery, 2017, pp. 303–318. DOI: 10.1145/3034786.3034789 (cit. on p. 5).

[BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. "Self-testing/correcting with applications to numerical problems". In: *Journal of Computer and System Sciences* 47 (3 1993), pp. 549–595. DOI: 10.1016/0022-0000(93)90044-W (cit. on pp. 1, 5, 7, 17, 19).

[BM84] Manuel Blum and Silvio Micali. "How to Generate Cryptographically Strong Sequences of Pseudorandom Bits". In: *SIAM Journal on Computing* 13 (4 1984), pp. 850–864. DOI: 10.1137/0213053 (cit. on p. 11).

[BR13] Quentin Berthet and Philippe Rigollet. "Complexity Theoretic Lower Bounds for Sparse Principal Component Detection". In: *Proceedings of Conference on Learning Theory (COLT)*. Ed. by Shai Shalev-Shwartz and Ingo Steinwart. Vol. 30. Proceedings of Machine Learning Research. Princeton, NJ, USA: PMLR, 2013, pp. 1046–1066 (cit. on pp. 2, 3).

[BR94] M Bellare and J Rompel. "Randomness-efficient oblivious sampling". In: *Proceedings Symposium on Foundations of Computer Science (FOCS)*. 1994, pp. 276–287. DOI: 10.1109/SFCS.1994.365687 (cit. on p. 12).

[BRSV17] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. "Average-case fine-grained hardness". In: *Proceedings of ACM Symposium on Theory of Computing* (2017), pp. 483–496. DOI: 10.1145/3055399.3055466 (cit. on p. 11).

[BT06a] Andrej Bogdanov and Luca Trevisan. "Average-Case Complexity". In: *Foundations and Trends in Theoretical Computer Science* 2.1 (2006). DOI: 10.1561/0400000004 (cit. on p. 4).

[BT06b] Andrej Bogdanov and Luca Trevisan. "On Worst-Case to Average-Case Reductions for NP Problems". In: *SIAM Journal on Computing* 36 (4 2006), pp. 1119–1159. DOI: 10.1137/S0097539705446974 (cit. on p. 12).

[Che98] Mu-Fa Chen. "Trilogy of Couplings and General Formulas for Lower Bound of Spectral Gap". In: *Probability Towards 2000*. Ed. by L Accardi and C C Heyde. New York, NY: Springer New York, 1998, pp. 123–136. DOI: 10.1007/978-1-4612-2224-8_7 (cit. on pp. 10, 29).

[CPS99] Jin-Yi Cai, A. Pavan, and D. Sivakumar. "On the Hardness of Permanent". In: *Proceedings of Symposium on Theoretical Aspects of Computer Science (STACS)* (1999), pp. 90–99. DOI: 10.1007/3-540-49116-3_8 (cit. on p. 11).

[DHKNT21] Irit Dinur, Prahladh Harsha, Tali Kaufman, Inbal Livni Navon, and Amnon Ta-Shma. "List-Decoding with Double Samplers". In: *SIAM J. Comput.* 50.2 (2021), pp. 301–349. DOI: 10.1137/19M1276650 (cit. on p. 12).

[DK17] Irit Dinur and Tali Kaufman. "High Dimensional Expanders Imply Agreement Expanders". In: *Proceedings of Symposium on Foundations of Computer Science (FOCS)*. 2017, pp. 974–985. DOI: 10.1109/FOCS.2017.94 (cit. on p. 14).

[DLW20] Mina Dalirrooyfard, Andrea Lincoln, and Virginia Vassilevska Williams. "New Techniques for Proving Fine-Grained Average-Case Hardness". In: IEEE, 2020, pp. 774–785. DOI: 10.1109/FOCS46700.2020.00077 (cit. on pp. 4, 11).

[DR06] Irit Dinur and Omer Reingold. "Assignment Testers: Towards a Combinatorial Proof of the PCP Theorem". In: *SIAM J. Comput.* 36.4 (2006), pp. 975–1024. DOI: 10.1137/S0097539705446962 (cit. on p. 12).

[FL92] Uriel Feige and Carsten Lund. "On the hardness of computing the permanent of random matrices (extended abstract)". In: *Proceedings of ACM Symposium on Theory of Computing (STOC)* (1992), pp. 643–654. DOI: 10.1145/129712.129775 (cit. on p. 11).

[Fre79] Rūsiņš Freivalds. "Fast probabilistic algorithms". In: *Proceedings of International Symposium on Mathematical Foundations of Computer Science (MFCS)* (1979), pp. 57–69. DOI: 10.1007/3-540-09526-8_5 (cit. on pp. 7, 8, 18).

[GC20] Elazar Goldenberg and Karthik C. S. "Hardness Amplification of Optimization Problems". In: *Proceedings of Innovations in Theoretical Computer Science Conference (ITCS)* (2020). DOI: 10.4230/LIPIcs.ITCS.2020.1 (cit. on p. 7).

[GMZ17] Chao Gao, Zongming Ma, and Harrison H Zhou. "SPARSE CCA: ADAPTIVE ESTIMATION AND COMPUTATIONAL BARRIERS". In: *Annals of Statistics* 45.5 (2017), pp. 2074–2101 (cit. on p. 3).

[GNW11] Oded Goldreich, Noam Nisan, and Avi Wigderson. "On Yao's XOR-Lemma". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation* (2011), pp. 273–301. DOI: 10.1007/978-3-642-22670-0_23 (cit. on pp. 1, 17).

[Gol11] Oded Goldreich. "A Sample of Samplers: A Computational Perspective on Sampling". In: *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation* (2011), pp. 302–332. DOI: 10.1007/978-3-642-22670-0_24 (cit. on p. 12).

[Gol20] Oded Goldreich. "On Counting t-Cliques Mod 2". In: *ECCC TR20-104* (2020) (cit. on pp. 1, 4, 11, 12, 26).

[GR18] Oded Goldreich and Guy Rothblum. "Counting t-Cliques: Worst-Case to Average-Case Reductions and Direct Interactive Proof Systems". In: *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)* (2018), pp. 77–88. DOI: 10.1109/FOCS.2018.00017 (cit. on pp. 4, 11).

[GS00] Oded Goldreich and Shmuel Safra. "A Combinatorial Consistency Lemma with Application to Proving the PCP Theorem". In: *SIAM J. Comput.* 29.4 (2000), pp. 1132–1154. DOI: 10.1137/S0097539797315744 (cit. on p. 12).

[GS92] Peter Gemmell and Madhu Sudan. "Highly resilient correctors for polynomials". In: *Information Processing Letters* 43 (4 1992), pp. 169–174. DOI: 10.1016/0020-0190(92)90195-2 (cit. on p. 11).

[HK11] Elad Hazan and Robert Krauthgamer. "How Hard Is It to Approximate the Best Nash Equilibrium?" In: *SIAM J. Comput.* 40.1 (2011), pp. 79–91. DOI: 10.1137/090766991 (cit. on p. 2).

[HKNS15] Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. "Unifying and Strengthening Hardness for Dynamic Problems via the Online Matrix-Vector Multiplication Conjecture". In: *Proceedings of ACM Symposium on Theory of Computing (STOC)* (2015), pp. 21–30. DOI: 10.1145/2746539.2746609 (cit. on p. 5).

[HLS22] Monika Henzinger, Andrea Lincoln, and Barna Saha. "The Complexity of Average-Case Dynamic Subgraph Counting". In: *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2022), pp. 459–498. DOI: 10.1137/1.9781611977073.23 (cit. on p. 5).

[HS21] Shuichi Hirahara and Nobutaka Shimizu. "Nearly Optimal Average-Case Complexity of Counting Bicliques Under SETH". In: *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)* (2021), pp. 2346–2365. DOI: 10.1137/1.9781611976465.140 (cit. on pp. 4, 11).

[HS22] Shuichi Hirahara and Nobutaka Shimizu. "Hardness Self-Amplification from Feasible Hard-Core Sets". In: *Proceedings of Symposium on Foundations of Computer Science (FOCS), to appear* (2022) (cit. on pp. 1, 2, 4, 5, 8, 11).

[HVV06] Alexander Healy, Salil Vadhan, and Emanuele Viola. "Using Nondeterminism to Amplify Hardness". In: *SIAM Journal on Computing* 35 (4 2006), pp. 903–931. DOI: 10.1137/S0097539705447281 (cit. on p. 1).

[HWX15] Bruce Hajek, Yihong Wu, and Jiaming Xu. "Computational Lower Bounds for Community Detection on Random Graphs". In: *Proceedings of Conference on Learning Theory (COLT)*. Ed. by Peter Grünwald, Elad Hazan, and Satyen Kale. Vol. 40. Proceedings of Machine Learning Research. Paris, France: PMLR, 2015, pp. 899–928 (cit. on p. 2).

[IJK09a] Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. "Approximate List-Decoding of Direct Product Codes and Uniform Hardness Amplification". In: *SIAM Journal on Computing* 39 (2 2009), pp. 564–605. DOI: 10.1137/070683994 (cit. on p. 1).

[IJK09b] Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. "Chernoff-Type Direct Product Theorems". In: *Journal of Cryptology* 22 (1 2009), pp. 75–92. DOI: 10.1007/s00145-008-9029-7 (cit. on pp. 1, 8, 9, 12, 17).

[IJKW10] Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. "Uniform Direct Product Theorems: Simplified, Optimized, and Derandomized". In: *SIAM Journal on Computing* 39 (4 2010), pp. 1637–1665. DOI: 10.1137/080734030 (cit. on pp. 1, 8, 9, 12).

[Imp95] R. Impagliazzo. "Hard-core distributions for somewhat hard problems". In: *Proceedings of IEEE Symposium on Foundations of Computer Science (FOCS)* (1995), pp. 538–545. DOI: 10.1109/SFCS.1995.492584 (cit. on pp. 1, 2, 17).

[IW97] Russell Impagliazzo and Avi Wigderson. "P = BPP if E requires exponential circuits". In: *Proceedings of ACM Symposium on Theory of Computing (STOC)* (1997), pp. 220–229. DOI: 10.1145/258533.258590 (cit. on pp. 1, 17).

[Jer92] Mark Jerrum. "Large Cliques Elude the Metropolis Process". In: *Random Struct. Algorithms* 3.4 (1992), pp. 347–360. DOI: 10.1002/rsa.3240030402 (cit. on pp. 2, 27).

[JP00] Ari Juels and Marcus Peinado. "Hiding Cliques for Cryptographic Security". In: *Designs, Codes and Cryptography* 20.3 (2000), pp. 269–280. DOI: 10.1023/A:1008374125234 (cit. on pp. 2, 3).

[Kuč95] Luděk Kučera. "Expected Complexity of Graph Partitioning Problems". In: *Discrete Applied Mathematics* 57.2-3 (1995), pp. 193–212. DOI: 10.1016/0166-218X(94)00103-K (cit. on p. 2).

[KZ14] Pascal Koiran and Anastasios Zouzias. "Hidden Cliques and the Certification of the Restricted Isometry Property". In: *IEEE Transactions on Information Theory* 60.8 (2014), pp. 4999–5006. DOI: 10.1109/TIT.2014.2331341 (cit. on p. 2).

[Lip91] Richard Lipton. "New directions in testing". In: *Distributed Computing and Cryptography* 2 (1991). Ed. by Joan Feigenbaum and Michael Merritt, pp. 191–202. DOI: 10.1090/dimacs/002 (cit. on p. 11).

[LMNT15] Kasper Green Larsen, J Ian Munro, Jesper Sindahl Nielsen, and Sharma V Thankachan. "On hardness of several string indexing problems". In: *Theor. Comput. Sci.* 582 (2015), pp. 74–82. DOI: 10.1016/j.tcs.2015.03.026 (cit. on p. 5).

[LP17] David Levin and Yuval Peres. *Markov Chains and Mixing Times*. 2nd. American Mathematical Society, 2017. DOI: 10.1090/mbk/107 (cit. on pp. 10, 29).

[MRS21] Pasin Manurangsi, Aviad Rubinstein, and Tselil Schramm. "The Strongish Planted Clique Hypothesis and Its Consequences". In: *Proceedings of Innovations in Theoretical Computer Science Conference (ITCS)* (2021). DOI: 10.4230/LIPIcs.ITCS.2021.10 (cit. on p. 2).

[MW15] Zongming Ma and Yihong Wu. "COMPUTATIONAL BARRIERS IN MINIMAX SUBMATRIX DETECTION". In: *Annals of Statistics* 43.3 (2015), pp. 1089–1116 (cit. on pp. 2, 3).

[ODo04] Ryan O'Donnell. "Hardness amplification within NP". In: *J. Comput. Syst. Sci.* 69.1 (2004), pp. 68–94. DOI: 10.1016/j.jcss.2004.01.001 (cit. on p. 1).

[San12] Tom Sanders. "On the Bogolyubov–Ruzsa lemma". In: *Analysis & PDE* 5 (3 2012), pp. 627–655. DOI: 10.2140/apde.2012.5.627 (cit. on p. 1).

[SV10] Ronen Shaltiel and Emanuele Viola. "Hardness Amplification Proofs Require Majority". In: *SIAM J. Comput.* 39.7 (2010), pp. 3122–3154. DOI: 10.1137/080735096 (cit. on p. 4).

[Yao81] Andrew Chi-Chih Yao. "Should Tables Be Sorted?" In: *J. ACM* 28.3 (1981), pp. 615–628. DOI: 10.1145/322261.322274 (cit. on p. 19).

[Yao82] Andrew C. Yao. "Theory and application of trapdoor functions". In: *Proceedings of IEEE Symposium on Foundations of Computer Science (SFCS))* (1982), pp. 80–91. DOI: 10.1109/SFCS.1982.45 (cit. on p. 1).

[Zuc97] David Zuckerman. "Randomness-optimal oblivious sampling". In: *Random Struct. Algorithms* 11.4 (1997), pp. 345–367. DOI: 10.1002/(sici)1098-2418(199712)11:4<345::aid-rsa4>3.0.co;2-z (cit. on p. 12).

A Deferred Proof

Lemma 7.3. *Suppose there exists a $T(n)$-time algorithm that solves $\#\mathsf{Triangle}_n$. Then, for any $\ell \in \mathbb{N}$, there exists an $O(T(\lceil n/\ell \rceil) \cdot \ell^3 \log \ell)$-time algorithm that solves $\#\mathsf{Triangle}_n$. The same holds for $\oplus\mathsf{Triangle}_n$.*

Proof. For simplicity, we assume that 3ℓ divides n. Let $G = (V, E)$ be a given n-vertex graph and write $V = \{v_1, \ldots, v_n\}$. Let $G' = (V_1 \cup V_2 \cup V_3, E')$ be the $3n$-vertex tripartite graph defined as:

- Each $V_i = \{v_1^{(i)}, \ldots, v_n^{(i)}\}$ is a copy of V.
- The edge set is given by $E' = E_{12} \cup E_{23} \cup E_{13}$, where

$$E_{ij} = \{\{v_k^{(i)}, v_\ell^{(j)}\} \colon k < \ell, \{v_k, v_\ell\} \in E\}.$$

Note that $\#\mathsf{Triangle}_n(G) = \#\mathsf{Triangle}_{3n}(G')$ (and thus $\oplus\mathsf{Triangle}_n(G) = \oplus\mathsf{Triangle}_{3n}(G')$) because a triangle $v_i v_j v_k$ in G with $i < j < k$ corresponds to a triangle $v_i^{(1)} v_j^{(2)} v_k^{(3)}$ in G' (and vice versa).

For each $i \in [3]$, let $U_1^{(i)}, \ldots, U_{3\ell}^{(i)} \subseteq V_i$ be a partition of V_i such that $|U_1^{(i)}| = \cdots = |U_{3\ell}^{(i)}|$. For $i, j, k \in [3\ell]$, let $G_{ijk} = G'[U_i^{(1)} \cup U_j^{(2)} \cup U_k^{(3)}]$ be the induced subgraph. Then, each G_{ijk} has n/ℓ vertices and

$$\#\mathsf{Triangle}_n(G) = \sum_{i,j,k \in [\ell]} \#\mathsf{Triangle}_{n/\ell}(G_{ijk}).$$

Since we can compute each $\#\mathsf{Triangle}_{n/\ell}(G_{ijk})$ using an $O(T(n)\log \ell)$-time randomized algorithm $\mathcal{M}$ with probability $1 - O(1/\ell^3)$ (over the random seed of $\mathcal{M}$), we can compute $\#\mathsf{Triangle}_n(G)$ in time $O(\ell^3 T(n) \log \ell)$. □

Lemma 8.1. *For any $n, k \in \mathbb{N}$,*

$$\Pr_{G \sim \mathcal{G}_{n,1/2,k}} [G \text{ contains a unique } k\text{-clique}] \geq 1 - 2kn2^{-k/2}.$$

Proof. We may assume $kn2^{-k/2} \leq \frac{1}{2}$, since otherwise, the RHS of the expression would be negative. Let C be the k-clique planted in $\mathcal{G}_{n,1/2,k}$. The expected number of k-cliques $S \subseteq V$ such that $|S \cap C| = t$ is $\binom{n-k}{k-t}\binom{k}{t}2^{-\binom{k}{2}+\binom{t}{2}}$. The summation of this term over $0 \leq t \leq k-1$ implies the claim. Specifically, the probability that $G \sim \mathcal{G}_{n,1/2,k}$ contains a k-clique other than C is at most

$$\begin{aligned}
\sum_{i=0}^{k-1} \binom{n-k}{k-t}\binom{k}{t}2^{-\binom{k}{2}+\binom{t}{2}} &\leq \sum_{t=0}^{k-1} \left(kn2^{-k/2}\right)^{k-t} \\
&\leq kn2^{-k/2} \cdot \sum_{t=0}^{\infty} (1/2)^t \\
&\leq kn2^{-k/2+1}.
\end{aligned}$$

□

Proposition 8.5. *For any $0 < c < 1$, $0 < \epsilon < 1$, the query graph of $\mathcal{R}_1^{\mathcal{O}}$ is a $\left(\frac{4n}{\sqrt{c^3 N\epsilon}}, c\right)$-sampler for density ϵ.*

Proof. Let $Q = (X, Y, W)$ be the query graph of $\mathcal{R}_1^{\mathcal{O}}$ associated with the Markov operator P. By Lemma 3.7, it suffices to show that $Q^* = (Y, X, W^\top)$ is an $\left(\frac{8n^2}{c^2\delta^2 N}, \frac{c}{2}\right)$-sampler for density δ. Let $v \in [0,1]^X$ be a δ-dense measure over X. For a random $\overline{H} = (\overline{G}, \overline{C}) \sim Y$, consider the random variable $Z := \mathbf{E}_{H|\overline{H}}[v(H)] = \mathbf{E}_\phi[v(\overline{H}_\phi)]$, where ϕ denotes a uniformly random injection from V to $\overline{V}$ conditioned on $\phi(V) \supseteq \overline{C}$, and $\overline{H}_\phi = (H, C) \in X$ denotes the marked graph given by $H = \phi^{-1}(\overline{H}[\phi(V)])$ and $C = \phi^{-1}(\overline{C})$ (i.e., $\overline{H}_\phi$ is drawn from the distribution $P^*(\overline{H}, \cdot)$).

We show $\Pr[Z \le (1 - 0.5c)\,\mathbf{E}[v]] \le \frac{8n^2}{c^2\delta^2 N}$ by the Chebyshev inequality (Lemma 3.1). Fix a mark $\overline{C}$ and take ϕ, ϕ' randomly conditioned on $\phi(V) \supseteq \overline{C}$ and $\phi'(V) \supseteq \overline{C}$. We denote by $\mathbf{E}_{\phi,\phi'|\overline{C}}[\cdot]$ the expectation over such ϕ, ϕ'. Conditioned on $\phi(V) \cap \phi'(V) = \overline{C}$, we observe that the random variables $\overline{H}_\phi$ and $\overline{H}_{\phi'}$ are independent of $\overline{G}$ (here, $\overline{G}$ is selected by making the fixed mark $\overline{C}$ clique and then drawing random edges outside $\overline{C}$). This implies $\mathbf{E}_{\overline{G}}[v(\overline{H}_\phi)v(\overline{H}_{\phi'})] = \mathbf{E}_{\overline{G}}[v(\overline{H}_\phi)]\,\mathbf{E}_{\overline{G}}[v(\overline{H}_{\phi'})]$. Then, we have

$$\begin{aligned}
\mathbf{E}[Z^2] &= \mathop{\mathbf{E}}_{\overline{H},\phi,\phi'}\left[v(\overline{H}_\phi)v(\overline{H}_{\phi'})\right] \\
&= \mathop{\mathbf{E}}_{\overline{C}}\left[\mathop{\mathbf{E}}_{\phi,\phi'|\overline{C}}\left[\mathop{\mathbf{E}}_{\overline{G}}[v(\overline{H}_\phi)v(\overline{H}_{\phi'})]\right]\right] \\
&\le \mathop{\mathbf{E}}_{\overline{C}}\left[\mathop{\mathbf{E}}_{\phi,\phi'|\overline{C}}\left[\mathop{\mathbf{E}}_{\overline{G}}[v(\overline{H}_\phi)]\mathop{\mathbf{E}}_{\overline{G}}[v(\overline{H}_{\phi'})]\right]\right] + \mathop{\mathbf{E}}_{\overline{C}}\left[\Pr_{\phi,\phi'|\overline{C}}[\phi(V)\cap\phi'(V) \ne \overline{C}]\right] \\
&\le (\mathbf{E}[Z])^2 + \frac{2n^2}{N}.
\end{aligned}$$

Here, note that $\phi(V) \setminus \overline{C}$ and $\phi'(V) \setminus \overline{C}$ are independent random subset; thus, for any fixed $\overline{C}$,

$$\Pr_{\phi,\phi'|\overline{C}}[\phi(V)\cap\phi'(V) \ne \overline{C}] = 1 - \frac{\binom{N-n}{n-k}}{\binom{N-k}{n-k}} \le \frac{2n^2}{N}.$$

Therefore, we have $\mathbf{Var}[Z] \le \frac{2n^2}{N}$ and thus

$$\Pr[Z \le (1 - c/2)\,\mathbf{E}[v]] \le \frac{4\,\mathbf{Var}[Z]}{c^2\,\mathbf{E}[v]^2} \le \frac{8n^2}{c^2\delta^2 N}.$$

This completes the proof. □

Lemma 8.11. *Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ with a distinguishing advantage γ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$. Then, there exists an $O\left(\frac{\log^3 n}{\gamma^2} \cdot T(n)\right)$-time randomized algorithm $\mathcal{B}$ that has a predicting advantage $\gamma/4 - n^{-\omega(\log n)}$ on $\mathcal{G}_{n,1/2,k}$ and $\mathcal{G}_{n,1/2}$ without knowing the clique size k.*

Proof. Let

$$p = \Pr_{x\sim\mathcal{D}_1,\mathcal{M}}[\mathcal{M}(x) = 1], \qquad q = \Pr_{x\sim\mathcal{D}_2,\mathcal{M}}[\mathcal{M}(x) = 0].$$

For $\ell = O\left(\frac{\log^3 n}{\gamma^2}\right)$, let $G_1, \ldots, G_\ell \sim \mathcal{G}_{n,1/2}$ and $\hat{q} = \frac{1}{\ell}\sum_{i\in[\ell]} \mathcal{M}(G_i)$. We use $\hat{q}$ as an estimate of q. By the Chernoff bound, $|q - \hat{q}| \le 0.01\gamma$ with probability $1 - n^{-\Omega(\log^2 n)}$. Let $\hat{p} = 1 + \gamma - \hat{q}$.

By the assumption, $p+q \ge 1+\gamma$ and thus $p \ge \hat{p} - 0.01\gamma$. Consider three cases: (i) If $|\hat{p}-\hat{q}| \le 0.2\gamma$, then $\mathcal{M}$ has a predicting advantage 0.39γ. (ii) Suppose $\hat{p} > \hat{q} + 0.2\gamma$. For a parameter $\xi \in (0,1)$, consider the algorithm $\mathcal{B}_\xi(x)$ that samples the binary random variable $c \in \{0,1\}$ of $\Pr[c=1] = \xi$ and thereafter returns $\min(c, \mathcal{M}(x))$. If we set $\xi = \frac{1}{1+\hat{p}-\hat{q}}$, a straightforward calculation shows that

$$\Pr_{x\sim\mathcal{D}_1,\mathcal{M}}[\mathcal{B}_\xi(x) = 1] = p\xi \ge \frac{1}{2} + \frac{0.49\gamma}{2+\gamma-2\hat{q}} \ge \frac{1}{2} + 0.16\gamma,$$
$$\Pr_{x\sim\mathcal{D}_2,\mathcal{M}}[\mathcal{B}_\xi(G) = 0] = q + (1-q)(1-\xi) \ge \xi\hat{p} - 0.02\gamma \ge \frac{1}{2} + 0.48\gamma.$$

(iii) The case in which $p < q - 0.2\gamma$ is symmetrical. Consider the algorithm $\mathcal{B}'_{\xi'}(x)$ that outputs $\max(c', \mathcal{M}(x))$ for a binary random variable $c' \in \{0,1\}$ with $\Pr[c'=1] = 1-\xi'$. If we set $\xi' = \frac{1}{1+\hat{q}-\hat{p}}$, then the same argument implies that $\mathcal{B}'_{\xi'}$ has an advantage 0.16γ.

Our algorithm $\mathcal{B}(G)$ runs as follows: Compute $\hat{p}$ and $\hat{q}$. If $|\hat{p} - \hat{q}| \le 0.2\gamma$, output $\mathcal{M}(G)$. If $\hat{p} - \hat{q} > 0.2\gamma$, output $\mathcal{B}_\xi(G)$ for $\xi = \frac{1}{1+\hat{p}-\hat{q}}$. If $\hat{p} - \hat{q} < -0.2\gamma$, output $\mathcal{B}'_{\xi'}(G)$ for $\xi' = \frac{1}{1+\hat{q}-\hat{p}}$. Then, $\mathcal{B}$ has a predicting advantage at least $\gamma/10$. Note that $\mathcal{B}$ does not use the value k. □

Lemma 8.13. *Let $k \ge 108\log_2 n$. Suppose there exists a $T(n)$-time algorithm $\mathcal{M}$ with a predicting advantage $1 - 1/n^2$ on $\mathcal{G}_{n,1/2,k'}$ and $\mathcal{G}_{n,1/2}$ for all $k' \ge k/3$ without knowing k'. Then, there exists an $O(nT(n))$-time algorithm $\mathcal{M}'$ that finds a k-clique for a $(1 - O(1/n))$-fraction of $G \sim \mathcal{G}_{n,1/2,k}$.*

We refer to [AAKMRX07, Section 4.3.3] for the detailed proof (the algorithm we present here is slightly different).

Proof Sketch. Let $G = (V, E)$ be an input graph. We assume that G contains at most one k-clique, which occurs with probability $1 - n^{-\omega(1)}$ in both cases of $G \sim \mathcal{G}_{n,1/2}$ and $G \sim \mathcal{G}_{n,1/2,k}$. For a vertex $v \in V$, let $\Gamma(v)$ be the set of vertices adjacent to v. Let H_v be the graph obtained by resampling random edges inside $v \cup \Gamma(v)$ (i.e., the graph obtained by replacing $G[v \cup \Gamma(v)]$ with an independent sample of an Erdős–Rényi random graph of the same size). Our algorithm $\mathcal{M}'$ outputs $S := \{v \in V : \mathcal{M}(H_v) = 0\}$.

Suppose $G \sim \mathcal{G}_{n,1/2,k}$. If v is in the unique k-clique, then $v \cup \Gamma(v)$ contains the clique and thus $H_v \sim \mathcal{G}_{n,1/2}$. Therefore, $v \in S$ with probability $1 - O(1/n^2)$. If v is not in the unique k-clique, then $H_v \sim \mathcal{G}_{n,1/2,k'}$, where k' is a random variable such that $k' \ge k/3$ with probability $1 - \exp(-k/54) > 1 - n^{-2}$ over the choice of G. For such G, with probability $1 - O(1/n^2)$, we have $v \notin S$. By the union bound over $v \in V$, with probability $1 - O(1/n)$, S is the planted clique. □

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il