

Error-Correction of Matrix Multiplication Algorithms over Integers

Shuichi Hirahara
National Institute of Informatics
s.hirahara@nii.ac.jp

Nobutaka Shimizu
Institute of Science Tokyo
shimizu.n.ah@m.titech.ac.jp

Abstract

Suppose there is an oracle $\mathcal{O}$ that computes a tiny fraction of the entries of the product of two uniformly random binary matrices over integers. We prove that there is a nearly linear-size randomized $\mathcal{O}$ -oracle circuit that, with high probability, computes all the entries of the product of every pair of binary matrices. This extends the previous “worst-case exact to average-case approximate” reduction of Hirahara and Shimizu (STOC 2025), which assumes the average-case distribution to be uniformly random matrices over a *finite field*, to uniformly random *binary* matrices.

The technical core of our reduction is an approximate list-decoding procedure for a signed sum encoding of the matrix product over the integers based on expander walks. To decode this encoding, we combine the near-linear-time approximation algorithm for MAX k -CSP supported on splittable tuples by Jeronimo (RANDOM 2023) with the XOR lemma for multi-output functions by Hirahara and Shimizu (STOC 2025).

Contents

1	Introduction	2
2	Proof Overview	4
2.1	Distance Amplification via Signed Sum Encoding	5
2.2	Approximate List Decoding for Signed Sum Encoding	7
2.3	Self-Correction over Boolean Cubes	10
2.4	Organization	11
3	Preliminaries	11
3.1	Graphs	12
3.2	Valued CSP over Expanding Structures	13
3.3	Dimension Adjustment	14
3.4	Other Auxiliary Results	17
4	Entry-wise List Recovery	18
5	Distance Amplification	22
5.1	Signed Sum Encoding over Expander Walks	22
5.2	Simulating Encoding by Matrix Multiplication	25
5.3	Approximate List-Decoding for Signed Sum Encoding	26
5.4	Proof of Distance Amplification	30

6	Hardness Amplification	32
6.1	Amplification via Direct Product Lemma	32
6.2	Amplification via Entry-Wise Majority	35
7	Chinese Remainder Theorem and Binary Expansion	35
8	Proof of the Main Theorem	40
A	Simulating Encoding by Matrix Multiplication	47
B	Near-linear Time Algorithm for MAX k-VCSP	50
C	From Binary to Bounded-Integer Matrix Multiplication	50
D	From Per-entry L_p Error to Hamming Agreement	51

1 Introduction

Matrix multiplication is a fundamental primitive in algorithms and scientific computing, with applications ranging from numerical linear algebra and graph algorithms to modern machine learning. Its computational complexity has been studied for more than fifty years: letting ω denote the exponent of matrix multiplication, Strassen’s breakthrough showed that $\omega < 3$ [Str69], and subsequent work further lowered the upper bound to below 2.372 [DWZ23; VXXZ24; ADWXXZ25]. Despite these advances, asymptotically fast algorithms are far from practical for realistic input sizes, and even in the Boolean setting, a more combinatorial line of work focuses on improving the cubic running time by lower-order factors [BW12; AFKLM24]. Consequently, improving the efficiency of matrix multiplication—both theoretically and in practice—remains a central challenge.

Beyond exact computation, there has also been extensive work on approximation algorithms for matrix multiplication. Broadly speaking, this line of research splits into two distinct paradigms.

- The first paradigm, which we refer to as *approximate matrix multiplication*, studies algorithms that, given arbitrary real matrices $A, B \in \mathbb{R}^{n \times n}$, compute a matrix C that approximates AB in a numerical norm such as the Frobenius norm [CL99; DKM06; Sar06; Pag13; CW13; CNW16; UW25]. Typically, these algorithms achieve worst-case guarantees for real-valued inputs by utilizing tools from randomized linear algebra, and have found practical applications in modern machine learning systems.
- The second paradigm, which we call *error-correcting reductions*, takes a fundamentally different approach [SS25; GSS24; HS25a; HS25b; VZ26]. This line of work is motivated in part by the possibility of performing matrix multiplication using low-energy physical systems, where computations are inherently noisy. In this setting, one assumes access to an oracle that computes the product of random matrices with some nontrivial accuracy in Hamming distance, and uses this oracle to design an algorithm that computes the product of arbitrary matrices exactly. The techniques developed in this paradigm rely on error-correcting codes, additive combinatorics, and average-case complexity.

There is a substantial gap between these two paradigms: Approximate matrix multiplication algorithms work over *real-valued* inputs and come with *worst-case* numerical guarantees. On the

other hand, existing error-correcting reductions show that even a very weak *average-case* guarantee suffices to recover exact computation but they crucially exploit algebraic structure and are restricted to *finite fields*.

This naturally raises the following question. Can one combine the strengths of these two paradigms to design an algorithm for matrix multiplication that is both efficient and exact? That is, can we obtain worst-case exact computation while retaining the efficiency and practical relevance of approximate matrix multiplication?

This paper takes a step toward answering this question. We study error-correcting reductions in a setting that is both discrete and physically natural, where the inputs are binary matrices and the computation is performed over the integers $\mathbb{Z}$. This model retains the discreteness of finite fields while avoiding their algebraic structure, and is therefore significantly closer to physically implemented computation. Indeed, working on binary matrices is especially important for error-correcting physical implementation of matrix multiplication, as the method suggested by Valiant [Val24] performs the multiplication of binary matrices over integers, and there is no modulo operation built into the computation.

To state it more formally, for two matrices X, Y of the same dimensions, define their normalized Hamming agreement by

$$\text{agree}(X, Y) := \Pr_{i,j}[X_{i,j} = Y_{i,j}].$$

Our main result is the following.

Theorem 1.1. *Let $\varepsilon > 0$ be a constant. For all sufficiently large $n \in \mathbb{N}$, for every oracle $\mathcal{O}$ such that*

$$\mathbb{E}_{A,B \sim \{0,1\}^{n \times n}}[\text{agree}(\mathcal{O}(A, B), AB)] \geq \varepsilon, \quad (1)$$

where A and B are chosen uniformly at random from $\{0,1\}^{n \times n}$, and the multiplication AB is performed over $\mathbb{Z}$, there exists a randomized $\tilde{O}((\log n)^2)$ -query oracle circuit $C^{\mathcal{O}}$ of size $\tilde{O}(n^2)$ such that, for all $A, B \in \{0,1\}^{n \times n}$,

$$\Pr_{C^{\mathcal{O}}} [C^{\mathcal{O}}(A, B) = AB] \geq \frac{2}{3},$$

where the probability is over the internal randomness of $C^{\mathcal{O}}$.

Thus, even a very weak average-case approximation guarantee—correctness on merely an ε -fraction of entries of the product of random binary matrices—can be bootstrapped into exact worst-case computation with a small overhead.

A couple of remarks about the generality of Theorem 1.1 are in order. First, the constant $\frac{2}{3}$ can be amplified to $1 - \delta$ by repeating the computation of $C^{\mathcal{O}}$ independently $O(\log(1/\delta))$ times and taking the majority vote. Second, although the circuit $C^{\mathcal{O}}$ only computes the product of matrices over $\{0,1\}^{n \times n}$, it can be used to compute the product of matrices whose entries lie in $\mathbb{Z} \cap [-\text{poly}(n), \text{poly}(n)]$ using binary expansion; see Section C for details. Lastly and most importantly, although the approximation error of $\mathcal{O}$ is measured by normalized Hamming distance, our reduction can be applied to oracles whose errors are measured in the L_p norm. For $p \geq 1$ and $\alpha \geq 1$, consider an oracle $\mathcal{O}$ such that

$$\mathbb{E}_{\substack{A,B \sim \{0,1\}^{n \times n} \\ i,j \sim [n]}} [|\mathcal{O}(A, B)_{i,j} - (AB)_{i,j}|^p] \leq \alpha^p.$$

We observe in Section D that such an oracle $\mathcal{O}$ can be converted into a randomized oracle $\mathcal{O}'$ that makes one call to $\mathcal{O}$ and satisfies the assumption (1) with $\varepsilon = 1/(10\alpha)$. Thus, our main theorem is applicable to any oracle whose per-entry L_p error α is at most $O(1)$. This connects our result to the paradigm of approximate matrix multiplication [CL99; DKM06; Sar06; Pag13; CW13; CNW16; UW25], which typically measures approximation errors in Frobenius norm (i.e., in L_2 norm). Optimistically, our reduction can be seen as an approach towards converting these approximation algorithms into exact algorithms. Pessimistically, it can also be regarded as giving a lower bound on approximation errors.

Our result is far from practical, even though our non-uniform reduction makes only $\tilde{O}((\log n)^2)$ oracle queries and runs in time $\tilde{O}(n^2)$ for every fixed constant $\varepsilon > 0$. Tracking the hidden constants in the running time through the current proof reveals at least an

$$\exp(\exp(\exp(\text{poly}(1/\varepsilon)))) \quad (2)$$

dependence on ε , while the hidden constant in the query complexity is at most $\text{poly}(1/\varepsilon)$; see Section 2.2 for details. Thus the present reduction is purely qualitative and completely unsuitable for realistic implementations. Obtaining a *uniform* error-correcting reduction with reasonable running time is left as an important open problem.

Error-Correction over Finite Fields. There has been an exciting line of recent work on error-correcting reductions for matrix multiplication over *finite fields* [AGGS22; HS23; GSS24; HS25b; HS25a; SS25; VZ26]. The reductions developed in this line of work assume an oracle $\mathcal{O}$ such that, for uniformly random matrices $A, B \sim \mathbb{F}_p^{n \times n}$ over a finite field $\mathbb{F}_p$, the average agreement satisfies

$$\mathbb{E}_{A, B \sim \mathbb{F}_p^{n \times n}} [\text{agree}(\mathcal{O}(A, B), AB)] \geq \frac{1}{p} + \varepsilon$$

for a constant $\varepsilon > 0$, and convert it into an efficient oracle algorithm $M^{\mathcal{O}}$ that computes the product AB for every input $A, B \in \mathbb{F}_p^{n \times n}$ [HS25b; HS25a; SS25; VZ26]. Such results are optimal because an oracle that always outputs the zero matrix has the expected agreement at least $1/p$.

The primary difference between these previous results and ours is that we restrict the input distribution to be a pair of random binary matrices, i.e., $(A, B) \sim \{0, 1\}^{n \times n} \times \{0, 1\}^{n \times n}$. In the literature of error-correcting codes, the corresponding task is called “error-correction over the Boolean cube” [BBB21; SS18; BHKS21; ABPSS24; ABPSS25], which is technically more challenging than the standard error-correction. In this sense, our theorem can be viewed as transporting the error-correction framework of [GSS24; HS25b; HS25a; SS25] into the Boolean-cube regime.

Most of the techniques developed in the literature [AGGS22; HS23; GSS24; HS25b; HS25a; SS25; VZ26] crucially rely on performing matrix multiplication modulo p . Our technical contribution is to develop a set of new techniques applicable to the error-correction of matrix multiplication algorithms over the Boolean cube, which we explain in the next section.

2 Proof Overview

Suppose we are given an oracle $\mathcal{O}$ that satisfies

$$\mathbb{E}_{A, B \sim \{0, 1\}^{n \times n}} [\text{dist}(\mathcal{O}(A, B), AB)] \leq 1 - \varepsilon,$$

where $\text{dist}(X, Y) := 1 - \text{agree}(X, Y)$ denotes the normalized Hamming distance between two matrices X and Y . A matrix X is said to be α -close to Y if $\text{dist}(X, Y) \leq \alpha$.

Our goal is to design a randomized oracle circuit $C^\mathcal{O}$ that computes AB for any instance $A, B \in \{0, 1\}^{n \times n}$. The worst-case exact solver $C^\mathcal{O}$ proceeds through the following steps:

1. Using the oracle $\mathcal{O}$ whose output is $(1 - \varepsilon)$ -close to AB on average, we compute a matrix that is δ -close to AB for any small constant $\delta > 0$. This procedure succeeds for an $\Omega(\varepsilon)$ -fraction of instances $A, B \sim \{0, 1\}^{n \times n}$ (Lemma 5.1).
2. Next, we convert this high-agreement algorithm with constant success probability to an algorithm with the same high-agreement guarantee with success probability $1 - 1/\text{poly}(\log \log n)$ (Lemmas 6.1 and 6.2).
3. Finally, we convert this average-case guarantee into a worst-case exact algorithm using self-correction over integers (Lemma 7.2).

The second step can be obtained by a straightforward modification of existing arguments [HS25b]. Therefore, in what follows, we focus on outlining the proofs of Steps 1 and 3.

2.1 Distance Amplification via Signed Sum Encoding

Our starting point is a reduction based on the expander-walk direct sum encoding [Ta17], as used in [HS25a] in the literature of error-correction for matrix multiplication over finite fields. Let $W \subseteq [n]^k$ denote the set of all walks of length $k - 1$ on a constant-degree expander graph G with n vertices. For a matrix $X \in \mathbb{F}_p^{n \times n}$, define the encoding $\text{Enc}_W(X) \in \mathbb{F}_p^{W \times W}$ by

$$\text{Enc}_W(X)_{(v_1, \dots, v_k), (w_1, \dots, w_k)} = \sum_{i=1}^k X_{v_i, w_i}, \quad (3)$$

where $(v_1, \dots, v_k), (w_1, \dots, w_k) \in W$ and $\mathbb{F}_p$ denotes the finite field with p elements. Here, the k -tuple $((v_1, w_1), \dots, (v_k, w_k)) \in ([n]^2)^k$ is a walk of length $k - 1$ on the tensor product graph $G^{\otimes 2}$, which is known to be an expander graph if the original graph G is an expander graph.

This encoding admits a near-linear-time *approximate list-decoding* algorithm (implicitly shown in [JST21; Jer23]) for every constant prime p . Specifically, given $Y' \in \mathbb{F}_p^{W \times W}$, the algorithm outputs a set $L \subseteq \mathbb{F}_p^{n \times n}$ with $|L| = O(1)$ such that for any codeword $Y = \text{Enc}_W(X)$ satisfying

$$\text{dist}(Y, Y') \leq 1 - 1/p - \varepsilon,$$

there exists some $X' \in L$ such that

$$\text{dist}(X, X') \leq \delta,$$

where $\varepsilon, \delta > 0$ are any constants (see Fig. 1).

Moreover, [HS25a] shows that, given an oracle that approximately multiplies random matrices over $\mathbb{F}_p^{n \times n}$, one can compute a matrix $C' \in \mathbb{F}_p^{W \times W}$ satisfying

$$\text{dist}(\text{Enc}_W(AB), C') \leq 1 - 1/p - \varepsilon + \frac{2k}{d}$$

for uniformly random $A, B \sim \mathbb{F}_p^{n \times n}$. Their proof is based on a combinatorial decomposition of W into vertex-disjoint path subsets and does not rely on algebraic properties of $\mathbb{F}_p$. Thus, essentially the same reduction applies to uniformly random matrices over $\{0, 1\}^{n \times n}$. Namely, given an oracle $\mathcal{O}$ satisfying

$$\mathbb{E}_{A, B \sim \{0, 1\}^{n \times n}} [\text{dist}(\mathcal{O}(A, B), AB)] \leq 1 - \varepsilon,$$

we can compute a matrix $C' \in \mathbb{F}_p^{W \times W}$ such that

$$\mathbb{E}_{A, B \sim \{0, 1\}^{n \times n}} [\text{dist}(\text{Enc}_W(AB), C')] \leq 1 - \varepsilon + \frac{2k}{d}.$$

This yields distance amplification as follows: Choose $k, d \in \mathbb{N}$ so that $2k/d \leq \varepsilon/10$. From an oracle that outputs a matrix that is $(1 - 0.9\varepsilon)$ -close to AB , we compute C' as above and run the approximate list-decoding algorithm on input C' to obtain a list $L \subseteq \mathbb{F}_p^{n \times n}$ of $|L| = O(1)$ such that there exists a matrix $X' \in L$ that is δ -close to AB . We can find such $X' \in L$ by estimating $\text{dist}(X, AB)$ for every candidate matrix $X \in L$, which can be done repeatedly sampling random entries $i, j \sim [n]$ and checking whether $X_{i,j} = (AB)_{i,j}$.

However, when attempting to transplant this idea from $\mathbb{F}_p$ to integer matrix multiplication, we encounter two fundamental difficulties.

The first issue is that the running time of approximate list decoding for the encoding over $\mathbb{F}_p$ is double exponential in p , namely $\exp(\exp(\text{poly}(p, 1/\varepsilon, 1/\delta)))$ [Jer23]. For integer matrix multiplication, a natural attempt is to use the Chinese Remainder Theorem to reduce the multiplication over integers to the case of prime fields. However, this approach fails since we require an error-correction over $\mathbb{F}_p$ for $p = \Omega(\log n)$, which makes the decoding time exponentially large in n .

The second issue is that, when viewed as an error-correcting code over the integers, the encoding Enc_W defined in (3) fails to achieve distance amplification. In general, for approximate list decoding to be possible, an encoding $\text{Enc}(\cdot)$ must satisfy

$$\text{dist}(\text{Enc}(X), \text{Enc}(X')) \geq 1 - \delta \text{ for any } X, X' \text{ with } \text{dist}(X, X') \geq \varepsilon.$$

The encoding Enc_W defined in (3) satisfies this property over finite fields even for the optimal parameter $\delta \approx 1/p$ when $k = k(\varepsilon, \delta, p)$ is sufficiently large, but over the integers this fails (cf. Remark 5.4): for any k , there exist X, X' such that

$$\text{dist}(X, X') = 1 \quad \text{but} \quad \text{dist}(\text{Enc}_W(X), \text{Enc}_W(X')) \approx 1 - \frac{1}{e}.$$

One of the main technical contributions of this paper is to overcome these issues by introducing a *signed walk encoding* as an error-correcting code over *integers*. Let $W \subseteq [n]^k$ be the set of walks of length $k - 1$ on an n -vertex regular expander graph. Define $\text{Enc}_W^\pm: \mathbb{Z}^{n \times n} \rightarrow \mathbb{Z}^{(\{\pm 1\}^k \times W) \times W}$ by

$$\text{Enc}_W^\pm(X)_{(\mathbf{b}, \mathbf{v}), \mathbf{w}} = \sum_{i=1}^k b_i X_{v_i, w_i}$$

for $\mathbf{b} = (b_1, \dots, b_k) \in \{\pm 1\}^k$ and $\mathbf{v} = (v_1, \dots, v_k), \mathbf{w} = (w_1, \dots, w_k) \in W$. Note that $\text{Enc}_W^\pm$ maps an $n \times n$ matrix to a $2^k |W| \times |W|$ matrix. The key point is that introducing random signs enforces anti-concentration via the Littlewood–Offord inequality, which implies the distance amplification property of $\text{Enc}_W^\pm$ (Lemma 5.3).

In addition, we prove that this encoding satisfies the following two properties.

- First, using an oracle $\mathcal{O}$ for approximate matrix multiplication over integers on binary random inputs, we can compute in time $\tilde{O}(n^2)$ a matrix C' such that

$$\text{dist}(\text{Enc}_W^\pm(AB), C') \leq 1 - \varepsilon.$$

This process succeeds for an $\Omega(\varepsilon)$ fraction of $A, B \sim \{0, 1\}^{n \times n}$.

- Second, using the same oracle $\mathcal{O}$ and given $A, B \in \{0, 1\}^{n \times n}$ and C' as above, we can construct in time $\tilde{O}(n^2)$ a set of matrices $L' \subseteq \mathbb{Z}^{n \times n}$ such that $|L'| = O(1)$ and

$$\min_{X \in L'} \text{dist}(X, AB) \leq \delta. \quad (4)$$

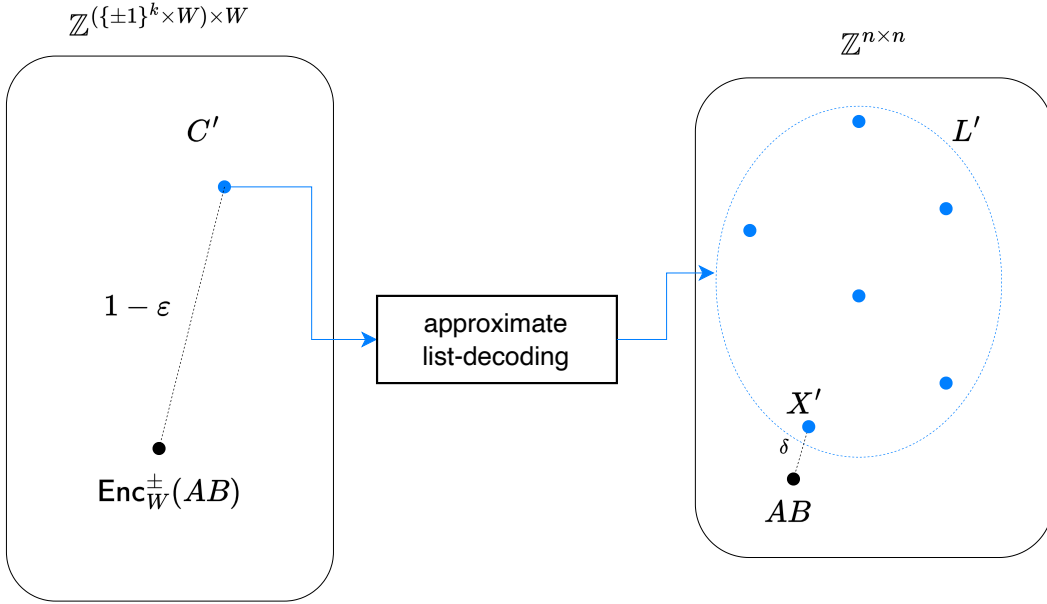


Figure 1: The list obtained by the approximate list-decoding algorithm on input C' that is $(1 - \varepsilon)$ -close to $\text{Enc}_W^\pm(AB)$ contains a matrix X' that is δ -close to the original message AB .

From these properties, we obtain distance amplification as follows. Given C' and L' computed as above, output a matrix $X' \in L'$ with $\text{dist}(X', AB) \leq \delta$ (see Fig. 1). Such $X' \in L'$ can be found by estimating $\text{dist}(X, AB)$ for every candidate matrix $X \in L'$, which can be done repeatedly sampling random entries $i, j \sim [n]$ and checking whether $X_{i,j} = (AB)_{i,j}$.

The first property (i.e., computing C' that is $(1 - \varepsilon)$ -close to $\text{Enc}_W^\pm(AB)$) can be shown using the ideas of [HS25a] (see Lemma 5.7 for details).

In contrast, proving the second property requires additional technical ideas, which we explain in the next subsection.

2.2 Approximate List Decoding for Signed Sum Encoding

The algorithm that computes the list L' of constant size satisfying (4) proceeds in the following two steps:

Step 1. Entry-wise list construction (Lemma 4.1). First, using the oracle $\mathcal{O}$, we construct a collection of *local lists* $(L_{i,j})_{i,j \in [n]}$ with $L_{i,j} \subseteq \mathbb{Z}$ and $|L_{i,j}| = O(1)$ such that, for some

$$X' \in \mathcal{L} := \{X \in \mathbb{Z}^{n \times n} : X_{i,j} \in L_{i,j} \text{ for all } i, j \in [n]\},$$

it holds that $\text{dist}(X', AB) \leq \delta$.

This is done by using the idea based on *XOR Lemma for multi-output functions* [HS25b, Theorem 2.1]. Specifically, let $A, B \in \{0, 1\}^{n \times n}$ be the given matrices. Choose a parameter $K = O(\log(1/\varepsilon)/\delta^4 \varepsilon^2) \in \mathbb{N}$, sample a random index $\ell \sim [K]$ and random matrices $A_t, B_t \sim \{0, 1\}^{n \times n}$ for $t \in [K] \setminus \{\ell\}$. Construct $\bar{A} \in \{0, 1\}^{n \times Kn}$ and $\bar{B} \in \{0, 1\}^{Kn \times n}$ by concatenating these matrices as

$$\bar{A} := [A_1 \mid \cdots \mid A_K], \quad \bar{B} := \begin{bmatrix} B_1 \\ \vdots \\ B_K \end{bmatrix},$$

where we set $A_\ell := A$ and $B_\ell := B$.

Observe that

$$\bar{A}\bar{B} = \sum_{t \in [K] \setminus \{\ell\}} A_t B_t + AB.$$

Thus we can reconstruct AB by computing $\bar{A}\bar{B}$ and then subtracting $\sum_{t \in [K] \setminus \{\ell\}} A_t B_t$. Since A, B, A_t, B_t are uniformly random, the matrices $\bar{A}$ and $\bar{B}$ are uniformly random as well.

Our algorithm samples ℓ and A_t, B_t as above, uses the oracle $\mathcal{O}$ to compute $\bar{A}\bar{B}$ (in fact, we can reduce multiplying $n \times Kn$ and $Kn \times n$ matrices to multiplying $Kn \times Kn$ matrices; see Lemma 3.5), receives $A_t B_t$ for $t \in [K] \setminus \{\ell\}$ as nonuniform advice¹, and then reconstructs AB .

By exploiting the correlation property of the pair of instances $((A, B), (\bar{A}, \bar{B}))$ (see Lemma 3.9), we can show that for a $(1 - \delta)$ -fraction of $A, B \sim \{0, 1\}^{n \times n}$ and for a $(1 - \delta)$ -fraction of $i, j \sim [n]$, we can conclude that

$$\Pr_{\ell, A_1, \dots, A_K, B_1, \dots, B_K} \left[\mathcal{O}(\bar{A}, \bar{B})_{i,j} = (AB)_{i,j} + \sum_{t \in [K] \setminus \{\ell\}} (A_t B_t)_{i,j} \right] \geq \Omega(\varepsilon).$$

Therefore, we construct the local lists $(L_{i,j})$ by repeatedly sampling $\ell, (A_t), (B_t)$ and reconstructing AB and adding its (i, j) -th entry to the list $L_{i,j}$.

Step 2. Approximate list-decoding inside $\mathcal{L}$ (Lemma 5.8). Second, we perform an *approximate list-decoding inside $\mathcal{L}$* for $\text{Enc}_W^\pm$ on input C' that is $(1 - \varepsilon)$ -close to $\text{Enc}_W^\pm(AB)$. Specifically, on input C' , the approximate list-decoding algorithm computes a list $L' \subseteq \mathcal{L}$ such that for any $X^* \in \mathcal{L}$ with $\text{dist}(\text{Enc}_W^\pm(X^*), C') \leq 1 - \varepsilon/2$, there exists a matrix $X' \in L'$ such that $\text{dist}(X^*, X') \leq \delta$.

Let $\delta^* := \min\{\frac{\varepsilon}{2k}, \delta\}$ and construct the local lists $(L_{i,j})$ for the error parameter δ^* . By construction, there exists a matrix $X^* \in \mathcal{L}$ such that $\text{dist}(X^*, AB) \leq \delta^*$. Then, we can prove $\text{dist}(\text{Enc}_W^\pm(X^*), \text{Enc}_W^\pm(AB)) \leq k\delta^* \leq \frac{\varepsilon}{2}$ (Lemma 5.3). Since $\text{dist}(C', \text{Enc}_W^\pm(AB)) \leq 1 - \varepsilon$, the triangle inequality yields that

$$\text{dist}(\text{Enc}_W^\pm(X^*), C') \leq 1 - \frac{\varepsilon}{2}. \tag{5}$$

¹Specifically, A_t and B_t are hardwired and their product $A_t B_t$ is given as nonuniform advice.

Therefore, the list L' obtained by the approximate list-decoding algorithm contains a matrix $X' \in L'$ that is δ -close to X^* . Again, the triangle inequality yields

$$\text{dist}(X', AB) \leq \text{dist}(X', X^*) + \text{dist}(X^*, AB) \leq \delta + \delta^* \leq \frac{3\delta}{2}.$$

Rescaling δ by a factor of $\frac{2}{3}$, we conclude that the list L' satisfies the desired property (4).

Now we describe the approximate list-decoding algorithm. To this end, we formulate the decoding problem as finding an assignment X that maximizes agreement between $\text{Enc}_W^\pm(X)$ and C' . Specifically, consider a MAX k -CSP instance φ whose variables are $(X_{i,j})_{i,j \in [n]}$ for $X_{i,j} \in L_{i,j}$ for all $i, j \in [n]$ and constraints are defined as follows. We associate each row index $(\mathbf{b}, \mathbf{v}) \in \{\pm 1\}^k \times W$ and column index $\mathbf{w} \in W$ of $C' \in \mathbb{Z}^{\{\pm 1\}^k \times W \times W}$ with an equality constraint

$$\text{Enc}_W^\pm(X)_{(\mathbf{b}, \mathbf{v}), \mathbf{w}} = C'_{(\mathbf{b}, \mathbf{v}), \mathbf{w}}.$$

Since the left hand side depends on k entries of X , this constitutes a k -CSP instance having n^2 variables over alphabet of size $\ell := \max_{i,j} |L_{i,j}| = O(1)$ and $2^k |W|^2 = O(n^2)$ constraints.

Let OPT denote the maximum fraction of constraints that can be satisfied by any assignment, i.e.,

$$\text{OPT} = \max_{X \in \mathcal{L}} \{\text{agree}(\text{Enc}_W^\pm(X), C')\}.$$

For a parameter $\gamma > 0$, an assignment $X \in \mathcal{L}$ is called γ -optimal if $\text{agree}(\text{Enc}_W^\pm(X), C') = \text{OPT} - \gamma$.

The key observation is that the fraction of constraints that are satisfied by an assignment $X \in \mathcal{L}$ equals $\text{agree}(\text{Enc}_W^\pm(X), C')$. Therefore, by (5), we have

$$\text{OPT} \geq \text{agree}(\text{Enc}_W^\pm(X^*), C') = 1 - \text{dist}(\text{Enc}_W^\pm(X^*), C') \geq \frac{\varepsilon}{2}.$$

On the other hand, using the expansion property of W , we can apply the approximation algorithm for MAX k -CSP whose constraints are supported on expander walks [Jer23] to find a near-optimal solution to φ in time $\tilde{O}(n^2)$. Specifically, Jeronimo's algorithm [Jer23, Theorem 1.6] is given a parameter $\gamma > 0$ and runs in time $\tilde{O}(n^2)$ and outputs a γ -optimal solution to φ (see Lemma 3.4).

Although Jeronimo's algorithm outputs only *one* nearly optimal solution, combining it with the *peering argument* inspired by [NRW23] and the distance amplification property of $\text{Enc}_W^\pm$, we can construct a list $L' \subseteq \mathcal{L}$ satisfying (4). Specifically, using Jeronimo's algorithm as a subroutine, we construct the desired list $L' \subseteq \mathcal{L}$ as follows:

1. Initialize $L' = \emptyset$.
2. Run Jeronimo's algorithm with parameter $\gamma = \varepsilon/3$ to find a γ -optimal solution $X \in \mathcal{L}$ to φ .
3. Let S denote the set of constraints that are satisfied by X . If $|S| \leq (\varepsilon/3) \cdot 2^k |W|^2$, output L' and terminate. Otherwise, append X to L' and remove constraints in S from φ to obtain a new instance φ' of MAX k -CSP.
4. Update $\varphi \leftarrow \varphi'$ and back to Step 2.

Thanks to the distance amplification property of $\text{Enc}_W^\pm$ (Lemma 5.3), the output L' of this algorithm satisfies the desired property (4). See Section 5.3 for details.

In fact, the triple-exponential dependence on ε in (2) is inherited from the MAX k -CSP approximation routine in Lemma 3.4, which is the main bottleneck in our reduction; thus, improving the hidden constant factors in this algorithm yields a reduction with a better dependence on ε .

2.3 Self-Correction over Boolean Cubes

Next, we explain how to use an oracle $\mathcal{O}_0$ satisfying

$$\mathbb{E}_{A,B \sim \{0,1\}^{n \times n}} [\text{agree}(\mathcal{O}_0(A, B), AB)] \geq 1 - \varepsilon_0$$

for $\varepsilon_0 = \frac{1}{(\log \log n)^3}$ to compute all entries of AB for any input $A, B \in \{0,1\}^{n \times n}$.

At a high level, our reduction follows the framework of [BBB21], which combines the Chinese Remainder Theorem, a binary decomposition of $\mathbb{Z}_q$, and random self-reduction to obtain a worst-case-to-average-case reduction for subgraph counting problems. We apply this framework to integer matrix multiplication over Boolean cubes.

Step 1: From binary inputs to $\mathbb{Z}_q$ inputs via unbiased bits (Lemma 7.5). We first convert the oracle $\mathcal{O}_0$ above into an oracle $\mathcal{O}_1$ that approximately multiplies random matrices uniformly drawn from $\mathbb{Z}_q^{n \times n}$ for small integers q . Using a standard symmetrization argument [GSS24; HS25b], we may assume that $\mathcal{O}_0$ satisfies the following entry-wise agreement guarantee: for every fixed $(i, j) \in [n]^2$,

$$\Pr_{A,B \sim \{0,1\}^{n \times n}} [\mathcal{O}_0(A, B)_{i,j} = (AB)_{i,j}] \geq 1 - \varepsilon_0.$$

We construct an oracle $\mathcal{O}_1$ that, given $X, Y \sim \mathbb{Z}_q^{n \times n}$, satisfies that for every fixed $(i, j) \in [n]^2$,

$$\Pr[\mathcal{O}_1(X, Y)_{i,j} = (XY \bmod q)_{i,j}] \geq 1 - \varepsilon_1,$$

where $\varepsilon_1 = O((\log q)^2 \varepsilon_0)$.

This reduction is based on a *unbiased binary expansion* of elements in $\mathbb{Z}_q$. Let $t = \lceil \log_2 q \rceil + 1$ and $T_q := \{0, \dots, q-1\} \cup \{2^t - q, \dots, 2^t - 1\}$ (note that this is a disjoint union of two intervals by the choice of t). For each $a \in \mathbb{Z}_q$, define

$$T_{q,a} := \{y \in T_q : y \equiv a \pmod{q}\}.$$

Then $T_{q,a}$ is a set of size two. Given $a \sim \mathbb{Z}_q$, we sample $\hat{a} \sim T_{q,a}$ and write

$$\hat{a} = \sum_{s=0}^{t-1} 2^s \hat{a}^{(s)}, \quad \hat{a}^{(s)} \in \{0, 1\}.$$

Then, it holds that each bit $\hat{a}^{(s)}$ is uniformly distributed over $\{0, 1\}$.

This allows us to represent a uniformly random matrix $X \in \mathbb{Z}_q^{n \times n}$ as

$$X \equiv \sum_{a=0}^{t-1} 2^a X^{(a)} \pmod{q},$$

where each $X^{(a)} \in \{0, 1\}^{n \times n}$ is uniformly random, and similarly for Y . Using bilinearity, we obtain

$$XY \equiv \sum_{a,b} 2^{a+b} X^{(a)} Y^{(b)} \pmod{q}.$$

Thus, by querying $\mathcal{O}_0$ on each pair $(X^{(a)}, Y^{(b)})$ and combining the results, we obtain $\mathcal{O}_1$ (see Lemma 7.5).

Step 2: Self-correction over $\mathbb{Z}_q$ (Lemma 7.6). The oracle $\mathcal{O}_1$ computes $(XY)_{i,j} \bmod q$ correctly with high probability for uniformly random inputs for every fixed $(i, j) \in [n]^2$.

Using the standard self-correction technique [BLR93] for bilinear functions (see Lemma 7.6), we convert $\mathcal{O}_1$ into an oracle $\mathcal{O}_2$ that works for arbitrary inputs.

Specifically, assuming that for every fixed (i, j) ,

$$\Pr_{X, Y \sim \mathbb{Z}_q^{n \times n}} [\mathcal{O}_1(X, Y)_{i,j} = (XY)_{i,j} \bmod q] \geq \frac{15}{16},$$

we obtain an oracle $\mathcal{O}_2$ such that for every fixed $A, B \in \mathbb{Z}_q^{n \times n}$ and $(i, j) \in [n]^2$,

$$\Pr[\mathcal{O}_2(A, B)_{i,j} = (AB)_{i,j} \bmod q] \geq 1 - \delta,$$

for any desired $\delta > 0$, using $O(\log(1/\delta))$ oracle calls.

Step 3: Error reduction and Chinese Remainder Theorem (Lemma 7.2). Finally, we combine the above reductions over a set $\mathcal{S}_n$ of primes such that $\prod_{p \in \mathcal{S}_n} p > n$.

For each $p \in \mathcal{S}_n$, we apply the previous steps to obtain an oracle that computes $(AB)_{i,j} \bmod p$ correctly with high probability for all inputs. By repeating the procedure $O(\log n)$ times and taking entry-wise majority, we can compute $(AB)_{i,j} \bmod p$ for all $i, j \in [n]$ and all $p \in \mathcal{S}_n$.

Since each entry $(AB)_{i,j}$ lies in $[0, n]$ and $\prod_{p \in \mathcal{S}_n} p > n$, the Chinese Remainder Theorem implies that the exact value of $(AB)_{i,j}$ is uniquely determined by its residues modulo $p \in \mathcal{S}_n$.

This yields a randomized algorithm that computes AB exactly for every input with high probability.

2.4 Organization

This paper is organized as follows. In Section 3, we collect several auxiliary notions and tools used throughout the reduction. In Section 4, we show how to recover, for a typical input (A, B) , a small candidate list for each entry $(AB)_{i,j}$ using the multi-output XOR lemma. In Section 5, we convert these local lists into a globally consistent matrix that agrees with AB on a $1 - \delta$ fraction of entries. In Section 6, we amplify the success probability of the high-agreement procedure. In Section 7, we apply self-correction and the Chinese Remainder Theorem to obtain an exact worst-case algorithm. Finally, in Section 8, we combine the reductions to prove the main theorem Theorem 1.1.

AI Disclosure. The overall proof strategy of this paper, including the idea of signed sum encoding over expander walks, are due to the authors but some parts are based on an idea suggested by AI. Specifically, GPT-5.4 suggested ideas for the proof of local list decoding (Lemma 5.8) and improved self-correction over Boolean cubes (Lemma 7.6). The authors independently verified, developed, and simplified the argument presented here and take full responsibility for its correctness, exposition, and attribution.

3 Preliminaries

For $n \in \mathbb{N}$, we use $[n] = \{1, \dots, n\}$ throughout this paper. For a finite set S , we write $i \sim S$ to denote that i is chosen uniformly at random from S .

For two vectors $x, y \in \mathbb{Z}^n$, let $\text{dist}(x, y) = \Pr_{i \sim [n]}[x_i \neq y_i]$ be the normalized Hamming distance between x and y , and let $\text{agree}(x, y) = 1 - \text{dist}(x, y)$. We extend this notion to matrices in a natural way by defining $\text{dist}(A, B) = \Pr_{i, j \sim [n]}[A_{i, j} \neq B_{i, j}]$ and $\text{agree}(A, B) = 1 - \text{dist}(A, B)$.

3.1 Graphs

A graph $G = ([n], E)$ is d -regular if every vertex has degree exactly d . Let $1 = \lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq -1$ be the eigenvalues of the normalized adjacency matrix (i.e., the transition matrix of the simple random walk) of G . We say that G is λ -expander if $\max\{|\lambda_2|, |\lambda_n|\} \leq \lambda$.

For a graph $G = ([n], E)$ and $k \in \mathbb{N}$, a k -walk on G is a k -tuple $(v_1, \dots, v_k) \in [n]^k$ such that $(v_i, v_{i+1}) \in E$ for all $i \in [k-1]$. Let $W \subseteq [n]^k$ be the set of all k -walks on G . The *girth* of a graph G is the length of the shortest cycle length at least 3 in G , i.e., the smallest $\ell \in \mathbb{N}$ such that there exists an ℓ -walk $(v_1, \dots, v_\ell)$ such that $v_1 = v_\ell$ and $v_1, \dots, v_{\ell-1}$ are all distinct.

In this section, we will use random regular graphs as our building block due to the following properties:

Lemma 3.1. *For $n, d \in \mathbb{N}$ such that nd is even, let $G_{n,d}$ be a random d -regular graph, a graph chosen uniformly at random from the set of all vertex-labeled simple d -regular n -vertex graphs. Then, for every fixed $d \geq 3$, the following holds:*

- If d is even, then there exists an $O(n)$ -time randomized algorithm that, on input 1^n , samples $G_{n,d}$ with probability $2/3$ for all large n [Wor99].²
- There exists a constant $c > 0$ that depends only on d such that $G_{n,d}$ is $\left(\frac{2\sqrt{d-1}}{d} + c\left(\frac{\log \log n}{\log n}\right)^2\right)$ -expander with probability $1 - n^{-10}$ [Fri03; Bor15].
- For any constant $g \in \mathbb{N}$, with probability $\exp\left(-\sum_{r=3}^g \frac{(d-1)^r}{2^r} + o(1)\right)$ as $n \rightarrow \infty$, the random graph $G_{n,d}$ has girth greater than g [MWW04].

Remark 3.2. *The above properties can alternatively be achieved using explicit Ramanujan graphs, such as the Lubotzky–Phillips–Sarnak construction [LPS88]. However, such graphs are known to exist only for infinitely many values of n , rather than for all sufficiently large n .*

Since our reduction needs to handle matrices of arbitrary dimension n , this restriction is inconvenient. In principle, one could still use such explicit graphs by adjusting the dimension (e.g., via padding), but for simplicity we work with random regular graphs, which exist for all sufficiently large n .

The *tensor product* of two graphs $G_1 = ([n_1], E_1)$ and $G_2 = ([n_2], E_2)$ is the graph $G_1 \otimes G_2 = ([n_1] \times [n_2], E)$ where the vertices are the pairs (i, j) for $i \in [n_1]$ and $j \in [n_2]$, and the edges are the pairs $\{(i, j), (i', j')\}$ for $(i, j), (i', j') \in [n_1] \times [n_2]$ such that $(i, i') \in E_1$ and $(j, j') \in E_2$. In this paper, we will always consider $G_1 = G_2 = G$ and write $G^{\otimes 2}$ as $G \otimes G$. It is known that if G is a d -regular λ -expander, then $G \otimes G$ is a d^2 -regular λ -expander.³

²If the degree d is odd, then we can sample $G_{n,d}$ for every large even n .

³This follows from the well-known fact that the eigenvalues of a Kronecker product $A \otimes B$ are all pairwise products of the eigenvalues of the factors A and B (cf. [HJ91, Theorem 4.2.12]).

3.2 Valued CSP over Expanding Structures

We borrow the framework of k -CSP over expanding structure from [JST21; Jer23] and extend it to Valued CSP (VCSP) by allowing constraint predicates to take values in $[0, 1]$.

Definition 3.3 (MAX k -VCSP(Σ)). *Let Σ be an alphabet and k be a positive integer. An instance of MAX k -VCSP(Σ) is given by $\varphi = (m, W, (P_{\mathbf{w}})_{\mathbf{w} \in W})$, where $W \subseteq [m]^k$ is a collection of k -tuples called a supporting set and each*

$$P_{\mathbf{w}} : \Sigma^k \rightarrow [0, 1]$$

is a valued constraint. When Σ is clear from the context, we simply write MAX k -VCSP instead of MAX k -VCSP(Σ).

The special case where all $P_{\mathbf{w}}$ are $\{0, 1\}$ -valued constraints is called MAX k -CSP.

For an assignment $a \in \Sigma^m$, define

$$\text{val}(a, \varphi) = \mathbb{E}_{\mathbf{w}=(w_1, \dots, w_k) \in W} [P_{\mathbf{w}}(a_{w_1}, \dots, a_{w_k})]$$

and let $\text{val}(\varphi) = \max_{a \in \Sigma^m} \text{val}(a, \varphi)$.

If every $P_{\mathbf{w}}$ is a $\{0, 1\}$ -valued constraint, then MAX k -VCSP coincides with the MAX k -CSP. Jeronimo [Jer23], utilizing a strong framework of efficient splittable regularity from [JST21], showed that for any $\gamma > 0$, we can compute in time $\tilde{O}(|W|)$ an additive γ -approximation of any MAX k -CSP instance over a finite alphabet if $W \subseteq [n]^k$ is the set of walks of length $k - 1$ on a regular λ -expander graph and λ is a sufficiently small constant.⁴ A slight modification of his proof yields the following result for MAX k -VCSP.

Lemma 3.4 ([Jer23, Theorem 6] for VCSP, specialized to expander walks). *Let k, γ, ℓ be constants. There exist an absolute constant $\lambda_0 > 0$ and a randomized algorithm that satisfy the following:*

Given an instance $\varphi = (m, W, (P_{\mathbf{w}})_{\mathbf{w} \in W})$ of MAX k -VCSP(Σ) with $|\Sigma| = \ell$ such that the supporting set W is the set of k -walks on a regular λ_0 -expander graph $G = ([m], E)$, the algorithm runs in time $\tilde{O}(m + |W|)$ and outputs an assignment $a \in \Sigma^m$ such that

$$\text{val}(a, \varphi) \geq \text{val}(\varphi) - \gamma.$$

The hidden constant in the time complexity of the algorithm of Lemma 3.4 is at least triple exponential in $k\ell/\gamma$, i.e.,

$$\exp(\exp(\exp(\text{poly}(k\ell/\gamma)))).$$

In our case, the constants k, γ, ℓ are chosen depending on the agreement ε of the approximate matrix multiplication algorithm.

The proof of Lemma 3.4 follows essentially the same argument as Jeronimo [Jer23, Theorem 6] for MAX k -CSP. We present the proof outline and the necessary modifications in Section B.

⁴Indeed, Jeronimo [Jer23] proved the same result if $W \subseteq [m]^k$ satisfies an expansion property called λ -splittable. But the special case of expander walks is sufficient for our purpose.

3.3 Dimension Adjustment

In the following, we will frequently modify the dimensions of the matrices involved in the computation. The next lemmas justify that such dimension changes do not essentially affect the achievable agreement of the oracle algorithm.

The first lemma shows that an oracle for $N \times N$ matrix multiplication can be used to compute $n \times N$ by $N \times m$ products with the same agreement.

Lemma 3.5 ([HS25b, Lemma 4.24]). *There exists a one-query oracle algorithm $M^\mathcal{O}$ with the following property: For every randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\substack{\bar{A}, \bar{B} \sim \{0,1\}^{N \times N} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A} \cdot \bar{B})] \geq \alpha,$$

for every $n, m \leq N$, it holds that

$$\mathbb{E}_{\substack{A \sim \{0,1\}^{n \times N} \\ B \sim \{0,1\}^{N \times m} \\ M^\mathcal{O}}} [\text{agree}(M^\mathcal{O}(A, B), A \cdot B)] \geq \alpha.$$

Moreover, $M^\mathcal{O}$ runs in time $O(N^2)$.

Proof sketch. Given $A \in \{0,1\}^{n \times N}$ and $B \in \{0,1\}^{N \times m}$, choose independent uniform permutations σ and τ over $[n]$ and $[m]$, respectively. Construct $\bar{A}, \bar{B} \in \{0,1\}^{N \times N}$ by embedding the rows of A into rows $\sigma(1), \dots, \sigma(n)$ of $\bar{A}$ and filling the remaining rows uniformly at random, and embedding the columns of B into columns $\tau(1), \dots, \tau(m)$ of $\bar{B}$ and filling the remaining columns uniformly at random. Then $\bar{A}, \bar{B}$ are uniform random matrices in $\{0,1\}^{N \times N}$.

Query the oracle on $(\bar{A}, \bar{B})$, and let C be its output. Define $M^\mathcal{O}(A, B)$ by

$$(M^\mathcal{O}(A, B))_{i,j} := C_{\sigma(i), \tau(j)} \quad (i \in [n], j \in [m]).$$

Since

$$(\bar{A} \cdot \bar{B})_{\sigma(i), \tau(j)} = (AB)_{i,j},$$

the entry (i, j) of $M^\mathcal{O}(A, B)$ is correct exactly when the entry $(\sigma(i), \tau(j))$ of C agrees with $\bar{A}\bar{B}$. Averaging over the random permutations σ, τ , each extracted position is uniformly distributed over $[n] \times [m]$. Therefore the expected agreement of $M^\mathcal{O}(A, B)$ with AB is exactly the expected agreement of $\mathcal{O}(\bar{A}, \bar{B})$ with $\bar{A}\bar{B}$, which is at least α . The running time outside the oracle call is $O(N^2)$. $\square$

The second lemma shows that an oracle that approximately multiplies $n \times N$ and $N \times n$ matrices can be used to approximately multiply $n \times n$ matrices for any $N \geq n$ using *nonuniform* advice.

Lemma 3.6. *There exists an $O(nN)$ -time one-query nonuniform oracle algorithm $M^\mathcal{O}$ such that for every randomized oracle $\mathcal{O}$ such that*

$$\mathbb{E}_{\substack{\bar{A} \sim \{0,1\}^{n \times N} \\ \bar{B} \sim \{0,1\}^{N \times n} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \alpha,$$

where $N \geq n$, it holds that

$$\mathbb{E}_{\substack{A \sim \{0,1\}^{n \times n} \\ B \sim \{0,1\}^{n \times n} \\ M^{\mathcal{O}}}} [\text{agree}(M^{\mathcal{O}}(A, B), A \cdot B)] \geq \alpha.$$

Proof. We write $N = n + t$ where $t \geq 0$. For $A \in \{0, 1\}^{n \times n}$ and $B \in \{0, 1\}^{n \times n}$, and for auxiliary matrices $X \in \{0, 1\}^{n \times t}$ and $Y \in \{0, 1\}^{t \times n}$, define

$$\bar{A} := (A \mid X) \in \{0, 1\}^{n \times N}, \quad \bar{B} := \begin{pmatrix} B \\ Y \end{pmatrix} \in \{0, 1\}^{N \times n}.$$

Then

$$\bar{A}\bar{B} = AB + XY.$$

Now let $A, B \sim \{0, 1\}^{n \times n}$, $X \sim \{0, 1\}^{n \times t}$ and $Y \sim \{0, 1\}^{t \times n}$ be chosen independently and uniformly at random. Then, the matrices $\bar{A} = (A \mid X)$ and $\bar{B} = \begin{pmatrix} B \\ Y \end{pmatrix}$ are uniformly distributed over $\{0, 1\}^{n \times N}$ and $\{0, 1\}^{N \times n}$, respectively. Therefore, by the assumption on $\mathcal{O}$,

$$\mathbb{E}_{\substack{A, B, X, Y \\ \mathcal{O}}} \left[\text{agree} \left(\mathcal{O} \left((A \mid X), \begin{pmatrix} B \\ Y \end{pmatrix} \right), AB + XY \right) \right] \geq \alpha.$$

Averaging over X and Y , there exist fixed matrices $X^* \in \{0, 1\}^{n \times t}$ and $Y^* \in \{0, 1\}^{t \times n}$ such that

$$\mathbb{E}_{\substack{A, B \\ \mathcal{O}}} \left[\text{agree} \left(\mathcal{O} \left((A \mid X^*), \begin{pmatrix} B \\ Y^* \end{pmatrix} \right), AB + X^*Y^* \right) \right] \geq \alpha.$$

We hardwire X^* , Y^* , and their product $P^* := X^*Y^* \in \mathbb{Z}^{n \times n}$ as nonuniform advice.

Define the one-query nonuniform oracle algorithm $M^{\mathcal{O}}$ as follows. On input $A, B \in \{0, 1\}^{n \times n}$, construct

$$\bar{A} := (A \mid X^*) \in \{0, 1\}^{n \times N}, \quad \bar{B} := \begin{pmatrix} B \\ Y^* \end{pmatrix} \in \{0, 1\}^{N \times n},$$

query the oracle on $(\bar{A}, \bar{B})$, obtaining $C := \mathcal{O}(\bar{A}, \bar{B})$, and output $M^{\mathcal{O}}(A, B) := C - P^*$.

Since $\bar{A}\bar{B} = AB + P^*$, we have for every entry $(i, j) \in [n] \times [n]$,

$$(M^{\mathcal{O}}(A, B))_{i,j} = (AB)_{i,j} \iff C_{i,j} = (\bar{A}\bar{B})_{i,j}.$$

Hence

$$\text{agree}(M^{\mathcal{O}}(A, B), AB) = \text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B}).$$

Taking expectation over uniform A, B and the randomness of $\mathcal{O}$, we obtain

$$\mathbb{E}_{\substack{A \sim \{0,1\}^{n \times n} \\ B \sim \{0,1\}^{n \times n} \\ M^{\mathcal{O}}}} [\text{agree}(M^{\mathcal{O}}(A, B), AB)] \geq \alpha.$$

It remains to bound the running time. Constructing $\bar{A}, \bar{B}$ takes $O(nN)$ time, and subtracting the hardwired matrix P^* from the oracle output takes $O(n^2)$ time. Since $N \geq n$, this is $O(nN)$ time overall outside the oracle call. Thus $M^{\mathcal{O}}$ is an $O(nN)$ -time one-query nonuniform oracle algorithm, as claimed. $\square$

Note that Lemma 3.5 shows a *uniform* reduction, whereas Lemma 3.6 shows a *nonuniform* reduction. The nonuniform advice in this reduction can be computed in time $O(n^2N)$.

The third lemma shows that, as long as $|\Delta|$ is sufficiently small compared to n , we can change the matrix dimension from n to $n + \Delta$ with only negligible overhead and essentially no loss in the agreement guarantee.

Lemma 3.7. *For any parameter $\Delta \in \mathbb{Z}$ such that $|\Delta| \leq \text{polylog}(n)$, there exists a one-query $\tilde{O}(n^2)$ -time randomized oracle algorithm $M^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\Pr_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(\mathcal{O}(A, B), AB) \geq \alpha] \geq \beta,$$

the algorithm $M^\mathcal{O}$ satisfies

$$\Pr_{A', B' \sim \{0,1\}^{(n+\Delta) \times (n+\Delta)}}^{M^\mathcal{O}} \left[\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \alpha - O\left(\frac{|\Delta|}{n}\right) \right] \geq \beta.$$

Moreover, if $\Delta \geq 0$, then the following stronger bound holds:

$$\Pr_{A', B' \sim \{0,1\}^{(n+\Delta) \times (n+\Delta)}}^{M^\mathcal{O}} [\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \alpha] \geq \beta.$$

Proof. We consider the cases $\Delta \geq 0$ and $\Delta < 0$ separately. Let $p = |\Delta|$ and $n' = n + \Delta$.

First suppose $\Delta \geq 0$, so that $n' = n + p$. Write the given matrices $A', B' \in \{0,1\}^{n' \times n'}$ as block matrices

$$A' = \begin{pmatrix} A & X \\ Y & Z \end{pmatrix}, \quad B' = \begin{pmatrix} B & U \\ V & W \end{pmatrix},$$

where $A, B \in \{0,1\}^{n \times n}$ and the other blocks have compatible sizes. Then

$$A'B' = \begin{pmatrix} AB + XV & AU + XW \\ YB + ZV & YU + ZW \end{pmatrix}.$$

Define

$$M^\mathcal{O}(A', B') := \begin{pmatrix} \mathcal{O}(A, B) + XV & AU + XW \\ YB + ZV & YU + ZW \end{pmatrix}.$$

Since $p = |\Delta|$, all correction terms involving the smaller blocks can be computed exactly in time $O(n^2|\Delta|)$. Now $M^\mathcal{O}(A', B')$ and $A'B'$ agree on every entry outside the top-left $n \times n$ block, and on that block they agree exactly where $\mathcal{O}(A, B)$ agrees with AB . Therefore, whenever

$$\text{agree}(\mathcal{O}(A, B), AB) \geq \alpha,$$

we also have

$$\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \alpha.$$

Finally, if A', B' are uniform in $\{0,1\}^{n' \times n'}$, then their top-left blocks A, B are independent uniform in $\{0,1\}^{n \times n}$. Hence

$$\Pr_{A', B'} [\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \alpha] \geq \Pr_{A, B} [\text{agree}(\mathcal{O}(A, B), AB) \geq \alpha] \geq \beta.$$

Next suppose $\Delta < 0$, so that $n' = n - p$. Given $A', B' \in \{0, 1\}^{n' \times n'}$, construct matrices

$$A = \begin{pmatrix} A' & X \\ Y & Z \end{pmatrix}, \quad B = \begin{pmatrix} B' & U \\ V & W \end{pmatrix},$$

where the blocks X, Y, Z, U, V, W are binary matrices chosen independently and uniformly at random. Then A, B are uniform in $\{0, 1\}^{n \times n}$. Compute $C = \mathcal{O}(A, B)$ and let C' be its top-left $n' \times n'$ block. Since

$$AB = \begin{pmatrix} A'B' + XV & * \\ * & * \end{pmatrix},$$

define $M^\mathcal{O}(A', B') := C' - XV$. If $\mathcal{O}(A, B)$ agrees with AB on at least αn^2 entries, then even if all disagreements inside the top-left block are concentrated there as much as possible, the number of agreements within that block is at least $\alpha n^2 - (n^2 - n'^2)$. Therefore

$$\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \frac{\alpha n^2 - (n^2 - n'^2)}{n'^2} = \alpha - O\left(\frac{|\Delta|}{n}\right),$$

where we used that $p = |\Delta|$ and $n' = n - p = n - |\Delta|$. Thus

$$\Pr[\text{agree}(M^\mathcal{O}(A', B'), A'B') \geq \alpha - O(|\Delta|/n)] \geq \Pr[\text{agree}(\mathcal{O}(A, B), AB) \geq \alpha] \geq \beta.$$

This completes the proof. $\square$

3.4 Other Auxiliary Results

We present a randomized procedure for testing whether a given matrix C is close to the product AB .

Lemma 3.8 ([HS25a, Lemma 9]). *There exists an $O(n^2 + n \log n / \beta^2)$ -time randomized algorithm M that, on input $A \in \mathbb{Z}^{n \times m}, B \in \mathbb{Z}^{m \times n}, C \in \mathbb{Z}^{n \times n}$ and parameters $\alpha, \beta > 0$, satisfies the following:*

1. *If $\text{agree}(AB, C) \geq \alpha + \beta$, then M accepts with probability $1 - 1/n^3$.*
2. *If $\text{agree}(AB, C) \leq \alpha$, then M rejects with probability $1 - 1/n^3$.*

Proof sketch. The algorithm repeatedly samples random entries of the product and checks whether they match the corresponding entries of C . Specifically, for $t = O(\log n / \beta^2)$ iterations, it samples independent uniform indices $i, j \sim [n]$ and computes the entry $(AB)_{i,j}$ in time $O(n)$. It then checks whether $(AB)_{i,j} = C_{i,j}$.

Let $X_{i,j}$ be the indicator that $(AB)_{i,j} \neq C_{i,j}$. Then $\mathbb{E}[X_{i,j}] = \text{dist}(AB, C)$. The algorithm estimates this quantity by the empirical average of the sampled indicators and compares it with a suitable threshold between α and $\alpha + \beta$.

By a standard Chernoff bound, using $t = O(\log n / \beta^2)$ samples ensures that the estimate deviates from $\text{dist}(AB, C)$ by at most $\beta/2$ with probability at least $1 - 1/n^3$. Therefore the algorithm correctly distinguishes the two cases with probability at least $1 - 1/n^3$. $\square$

Lemma 3.9 (Direct product lemma; [IJK09; HS23; HS24]). *Let $\mathcal{X}$ be a set. For all sufficiently small $\delta^*, \varepsilon^* > 0$, for every $K \geq O(\log(1/\varepsilon^*)) / (\delta^* \varepsilon^*)^2$, for every function $S: \mathcal{X}^K \rightarrow [0, 1]$, it holds that*

$$\Pr_{x \sim \mathcal{X}} \left[\left| \mathbb{E}_{y' \sim \Gamma(x)} [S(y')] - \mathbb{E}_{y \sim \mathcal{X}^K} [S(y)] \right| \leq \varepsilon^* \right] \geq 1 - \delta^*.$$

Here, $\Gamma(x)$ is the distribution over $y' \in \mathcal{X}^K$ defined by the following sampling procedure: Sample $y \sim \mathcal{X}^K$, $k \sim [K]$, replace the k -th element of y with x to obtain y' , and output y' .

4 Entry-wise List Recovery

Using the techniques developed by Hirahara and Shimizu [HS25b], we prove the following.

Lemma 4.1. *For all sufficiently small $\delta, \varepsilon > 0$ and every randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\substack{\bar{A} \sim \{0,1\}^{n \times Kn} \\ \bar{B} \sim \{0,1\}^{Kn \times n} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon,$$

there exist

$$K = K(\delta, \varepsilon) = O(\log(1/\varepsilon)/\delta^4 \varepsilon^2)$$

and an $O(Kn^2)$ -time randomized nonuniform oracle algorithm $M^\mathcal{O}$ that, on input $A, B \in \{0,1\}^{n \times n}$, makes $O(\log(1/\delta)/\varepsilon)$ oracle calls to $\mathcal{O}$ and, with probability at least $1 - \delta$ over uniform $A, B \sim \{0,1\}^{n \times n}$ and the internal randomness of $M^\mathcal{O}$, outputs a collection of lists $(L_{i,j})_{i,j \in [n]}$ such that $|L_{i,j}| = O(\log(1/\delta)/\varepsilon)$ for all $(i,j) \in [n]^2$ and

$$\Pr_{i,j \sim [n]} [(AB)_{i,j} \in L_{i,j}] \geq 1 - \delta.$$

Moreover, if $\varepsilon > 1/2$ is a fixed constant, then the same algorithm additionally satisfies the following: with probability at least $1 - \delta$ over uniform $A, B \sim \{0,1\}^{n \times n}$ and the internal randomness of $M^\mathcal{O}$, for at least a $(1 - \delta)$ -fraction of pairs $(i,j) \in [n]^2$, the value that appears most frequently in $L_{i,j}$ is exactly $(AB)_{i,j}$.

Proof. Using the reduction of Lemma 7.3 in place of $\mathcal{O}$, we may assume without loss of generality that

$$\Pr_{\substack{\bar{A} \sim \{0,1\}^{n \times Kn} \\ \bar{B} \sim \{0,1\}^{Kn \times n} \\ \mathcal{O}}} [\mathcal{O}(\bar{A}, \bar{B})_{i,j} = (\bar{A}\bar{B})_{i,j}] \geq \varepsilon$$

for every $(i,j) \in [n]^2$.

Define $\mathcal{X} := \{0,1\}^{n \times n} \times \{0,1\}^{n \times n}$. For $y = ((A_1, B_1), \dots, (A_K, B_K)) \in \mathcal{X}^K$, define

$$\bar{A}(y) := [A_1 \mid \dots \mid A_K] \in \{0,1\}^{n \times Kn}, \quad \bar{B}(y) := \begin{bmatrix} B_1 \\ \vdots \\ B_K \end{bmatrix} \in \{0,1\}^{Kn \times n},$$

and

$$F(y) := \bar{A}(y) \cdot \bar{B}(y) = \sum_{t \in [K]} A_t B_t \in \mathbb{Z}^{n \times n}.$$

For $x = (A, B) \in \mathcal{X}$, we define the distribution $\Gamma(x)$ as follows. Consider a random seed r that consists of a uniformly random index $\kappa^{(r)} \sim [K]$ and pairs of uniformly random binary matrices

$$(A_t^{(r)}, B_t^{(r)}) \sim \mathcal{X}$$

for all $t \in [K] \setminus \{\kappa^{(r)}\}$. Given $x = (A, B) \in \mathcal{X}$, let $\Gamma_r(x) \in \mathcal{X}^K$ be the K -tuple obtained by placing x in the $\kappa^{(r)}$ -th coordinate and $(A_t^{(r)}, B_t^{(r)})$ in the t -th coordinate for every $t \in [K] \setminus \{\kappa^{(r)}\}$. Let $\Gamma(x)$ be the distribution of $\Gamma_r(x)$ over a random seed r . Define the matrix

$$\alpha(r) := \sum_{t \in [K] \setminus \{\kappa^{(r)}\}} A_t^{(r)} B_t^{(r)}$$

Then, for every $x = (A, B)$,

$$F(\Gamma_r(x)) = AB + \alpha(r). \quad (6)$$

Note that $\alpha(r)$ does not depend on the input x .

This motivates us to define the following randomized algorithm $M_*^\mathcal{O}$ that takes the advice string α that depends on its internal randomness. The algorithm takes $(A, B) \in \mathcal{X}$ as input and repeats the following for $T = O(\log(1/\delta)/\varepsilon)$ times: For each $t \in [T]$, sample a random seed r_t , compute $y := \Gamma_{r_t}(A, B)$ and $C := \mathcal{O}(y) - \alpha(r_t)$, and add $C_{i,j}$ to the list $L_{i,j}$ for every $(i, j) \in [n]^2$. This completes the description of $M_*^\mathcal{O}$. (The final algorithm $M^\mathcal{O}$ will be obtained by fixing the internal randomness of $M_*^\mathcal{O}$ and the advice string.)

We now analyze the randomized algorithm $M_*^\mathcal{O}$. Fix an arbitrary $(i, j) \in [n]^2$. For each $y \in \mathcal{X}^K$, define

$$S_{i,j}(y) := \Pr_{\mathcal{O}}[\mathcal{O}(\bar{A}(y), \bar{B}(y))_{i,j} = F(y)_{i,j}].$$

By assumption, we have

$$\mathbb{E}_{y \sim \mathcal{X}^K} [S_{i,j}(y)] \geq \varepsilon.$$

Applying Lemma 3.9 to the function $S_{i,j}: \mathcal{X}^K \rightarrow [0, 1]$, $\varepsilon^* := \varepsilon/2$ and $\delta^* := \delta^2/4$, we obtain

$$\Pr_{x \sim \mathcal{X}} \left[\left| \mathbb{E}_{y' \sim \Gamma(x)} [S_{i,j}(y')] - \mathbb{E}_{y \sim \mathcal{X}^K} [S_{i,j}(y)] \right| \leq \varepsilon^* \right] \geq 1 - \delta^*.$$

Since $\mathbb{E}_{y \sim \mathcal{X}^K} [S_{i,j}(y)] \geq \varepsilon$, for at least a $(1 - \delta^*)$ -fraction of inputs $x \in \mathcal{X}$ we have

$$\mathbb{E}_{y' \sim \Gamma(x)} [S_{i,j}(y')] \geq \varepsilon^*.$$

We call such an input x (i, j) -good and define $G_{i,j}$ to be the set of (i, j) -good inputs. Taking the average over $(i, j) \sim [n]^2$ and using Markov's inequality, with probability $1 - \sqrt{\delta^*}$ over $x \sim \mathcal{X}$, we have

$$\Pr_{(i,j) \sim [n]^2} [x \notin G_{i,j}] \leq \sqrt{\delta^*} \leq \delta/2.$$

We call such an input x good.

Fix $(i, j) \in [n]^2$, take any input $x = (A, B) \in G_{i,j}$, and consider the t -th iteration of $M_*^\mathcal{O}$ that uses a random seed r_t . Observe that if $\mathcal{O}(y)_{i,j} = (\bar{A}\bar{B})_{i,j}$, then

$$C_{i,j} = \mathcal{O}(y)_{i,j} - \alpha(r_t)_{i,j} = F(y)_{i,j} - \alpha(r_t)_{i,j} = (AB)_{i,j},$$

where the first equality is by the definition of C in the algorithm of $M_*^\mathcal{O}$, and the second equality is by the assumption and the definition of F , and the last equality is from Equation (6). Thus, we have

$$\Pr_{M_*^\mathcal{O}} [C_{i,j} = (AB)_{i,j}] \geq \Pr_{y \sim \Gamma(x)}_{\mathcal{O}} [\mathcal{O}(y)_{i,j} = (\bar{A}\bar{B})_{i,j}] \geq \varepsilon^*,$$

where the last inequality holds by the definition of good inputs x . Choose the hidden constant in $T = O(\log(1/\delta)/\varepsilon)$ large enough so that

$$(1 - \varepsilon^*)^T \leq \exp(-\varepsilon T/2) \leq \delta^2/4.$$

Then, for every $(i, j) \in [n]^2$ and every (i, j) -good input $x = (A, B) \in G_{i,j}$,

$$\Pr_{M_*^\mathcal{O}}[(AB)_{i,j} \notin L_{i,j}] \leq \delta^2/4.$$

Now fix any good input $x = (A, B)$. Define the random variable

$$Z_x := \Pr_{(i,j) \sim [n]^2}[x \in G_{i,j} \text{ and } (AB)_{i,j} \notin L_{i,j}].$$

By linearity of expectation and the bound above,

$$\mathbb{E}_{M_*^\mathcal{O}}[Z_x] \leq \delta^2/4.$$

Hence, by Markov's inequality,

$$\Pr_{M_*^\mathcal{O}}[Z_x > \delta/2] \leq \frac{\delta^2/4}{\delta/2} = \delta/2.$$

Since x is good, we also have

$$\Pr_{(i,j) \sim [n]^2}[x \notin G_{i,j}] \leq \delta/2.$$

Therefore, for every good input $x = (A, B)$, with probability at least $1 - \delta/2$ over the randomness of $M_*^\mathcal{O}$, we have

$$\Pr_{(i,j) \sim [n]^2}[(AB)_{i,j} \in L_{i,j}] \geq 1 - \delta.$$

Since x is good with probability at least $1 - \delta/2$ over uniform $x \sim \mathcal{X}$, we conclude that

$$\Pr_{\substack{x \sim \mathcal{X} \\ M_*^\mathcal{O}}} \left[\Pr_{(i,j) \sim [n]^2}[(AB)_{i,j} \in L_{i,j}] \geq 1 - \delta \right] \geq 1 - \delta.$$

This proves the existence of a reduction $M_*^\mathcal{O}$ that takes an advice string α that depends on internal randomness with the stated guarantee.

We now fix the seeds $\bar{r} = (r_1, \dots, r_T)$ to obtain a nonuniform reduction $M^\mathcal{O}$. Let $\text{Fail}(\bar{r})$ denote the probability, over uniform $x = (A, B) \sim \mathcal{X}$ and the randomness of $\mathcal{O}$, that the lists produced using $\bar{r}$ fail to satisfy

$$\Pr_{(i,j) \sim [n]^2}[(AB)_{i,j} \in L_{i,j}] \geq 1 - \delta.$$

The proof above shows $\mathbb{E}_{\bar{r}}[\text{Fail}(\bar{r})] \leq \delta$, and hence there exists a fixed seed sequence $\bar{r}^* = (r_1^*, \dots, r_T^*)$ for which $\text{Fail}(\bar{r}^*) \leq \delta$. We hardwire this sequence and the corresponding advice strings $\alpha(r_1^*), \dots, \alpha(r_T^*)$ into the non-uniform advice of the reduction $M^\mathcal{O}$. This yields the randomized nonuniform oracle algorithm $M^\mathcal{O}$ stated in Lemma 4.1.

For the moreover part, assume that $\varepsilon > 1/2$ is a fixed constant, and set

$$\eta := \frac{\varepsilon - 1/2}{2} > 0.$$

Fix $(i, j) \in [n]^2$ and apply Lemma 3.9 with $\delta^* := \delta^2/4$ and $\varepsilon^* := \eta$. Then, for at least a $(1 - \delta^*)$ -fraction of inputs $x \in \mathcal{X}$, we have

$$\mathbb{E}_{y' \sim \Gamma(x)}[S_{i,j}(y')] \geq \mathbb{E}_{y \sim \mathcal{X}^K}[S_{i,j}(y)] - \varepsilon^* \geq \varepsilon - \eta = \frac{1}{2} + \eta.$$

We call such an input x *majority- (i, j) -good*, and define $G_{i,j}^{\text{maj}}$ to be the set of majority- (i, j) -good inputs. Averaging over $(i, j) \sim [n]^2$ and applying Markov's inequality exactly as above, with probability at least $1 - \sqrt{\delta^*} = 1 - \delta/2$ over $x \sim \mathcal{X}$, we have

$$\Pr_{(i,j) \sim [n]^2} [x \notin G_{i,j}^{\text{maj}}] \leq \sqrt{\delta^*} = \delta/2.$$

Fix such an input $x = (A, B)$, and choose the hidden constant in $T = O(\log(1/\delta))$ large enough so that the Hoeffding bound below is at most $\delta^2/4$. For every (i, j) with $x \in G_{i,j}^{\text{maj}}$, each iteration produces the correct value $(AB)_{i,j}$ with probability at least $1/2 + \eta$ independently. Hence, by Hoeffding's inequality, the probability that the majority value in $L_{i,j}$ is different from $(AB)_{i,j}$ is at most $\delta^2/4$. Define the random variable

$$Z_x^{\text{maj}} := \Pr_{(i,j) \sim [n]^2} [x \in G_{i,j}^{\text{maj}} \text{ and the majority value in } L_{i,j} \text{ is not } (AB)_{i,j}].$$

Then

$$\mathbb{E}_{M_*^{\mathcal{O}}} [Z_x^{\text{maj}}] \leq \delta^2/4.$$

By Markov's inequality,

$$\Pr_{M_*^{\mathcal{O}}} [Z_x^{\text{maj}} > \delta/2] \leq \delta/2.$$

Therefore, for every such input $x = (A, B)$, with probability at least $1 - \delta/2$ over the randomness of $M_*^{\mathcal{O}}$, the majority value in $L_{i,j}$ is equal to $(AB)_{i,j}$ for at least a $(1 - \delta)$ -fraction of pairs $(i, j) \in [n]^2$. Since such an input x occurs with probability at least $1 - \delta/2$ over uniform $x \sim \mathcal{X}$, we conclude that

$$\Pr_{\substack{x \sim \mathcal{X} \\ M_*^{\mathcal{O}}}} \left[\Pr_{(i,j) \sim [n]^2} [\text{the majority value in } L_{i,j} \text{ is } (AB)_{i,j}] \geq 1 - \delta \right] \geq 1 - \delta.$$

Finally, in the regime $\varepsilon > 1/2$, let $\text{MajFail}(\bar{r})$ denote the probability, over uniform $x = (A, B) \sim \mathcal{X}$ and the randomness of $\mathcal{O}$, that the lists produced using $\bar{r}$ fail to satisfy

$$\Pr_{(i,j) \sim [n]^2} [\text{the majority value in } L_{i,j} \text{ is } (AB)_{i,j}] \geq 1 - \delta.$$

The argument above shows that

$$\mathbb{E}_{\bar{r}} [\text{MajFail}(\bar{r})] \leq \delta.$$

Hence there exists a fixed seed sequence for which $\text{MajFail}(\bar{r}) \leq \delta$. Hardwiring this seed sequence and the corresponding advice strings yields a non-uniform algorithm satisfying the stronger majority guarantee; since that stronger guarantee implies the first part, the same fixed algorithm satisfies both claims. $\square$

5 Distance Amplification

The goal of this section is to convert the local information obtained in the previous section (Lemma 4.1) into a *globally consistent* matrix.

Specifically, given a collection of small lists $(L_{i,j})_{i,j \in [n]}$ such that $(AB)_{i,j} \in L_{i,j}$ for most entries, we show how to efficiently find a single matrix X that agrees with AB on a $1 - \delta$ fraction of entries.

Lemma 5.1. *For any constants $\varepsilon, \delta > 0$, there exist a constant $k' = k'(\varepsilon, \delta) \in \mathbb{N}$ and an $\tilde{O}(n^2)$ -time randomized nonuniform oracle algorithm $M^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\substack{\bar{A}, \bar{B} \sim \{0,1\}^{k'n \times k'n} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon,$$

the algorithm $M^\mathcal{O}$ is given $A, B \sim \{0,1\}^{n \times n}$ as input and satisfies

$$\Pr_{\substack{A, B \sim \{0,1\}^{n \times n} \\ M^\mathcal{O}}} [\text{agree}(M^\mathcal{O}(A, B), AB) \geq 1 - \delta] \geq \frac{2\varepsilon}{5}.$$

Moreover, $M^\mathcal{O}$ makes at most $O(1)$ oracle calls to $\mathcal{O}$.

The main difficulty is that naively searching over all possible choices from the lists is computationally infeasible. To overcome this, we use a signed sum encoding over expander walks to amplify distances and reduce the problem to an approximate optimization problem over a structured constraint system.

5.1 Signed Sum Encoding over Expander Walks

The key building block is the *signed W -sum encoding* defined as follows.

Definition 5.2 (Signed W -sum encoding). *Let $W \subseteq [n]^k$ be a set of k -tuples of $[n]$ and $W^\pm = \{\pm 1\}^k \times W$. The signed W -sum encoding of a vector $x \in \mathbb{Z}^n$ is the vector $\text{Enc}_W^\pm(x) \in \mathbb{Z}^{W^\pm}$ such that, for $\mathbf{w}^\pm = (\mathbf{b}, \mathbf{w}) \in W^\pm$ with $\mathbf{b} = (b_1, \dots, b_k) \in \{\pm 1\}^k$ and $\mathbf{w} = (w_1, \dots, w_k) \in W$, the $\mathbf{w}^\pm$ -th entry of $\text{Enc}_W^\pm(x)$ is given by*

$$\text{Enc}_W^\pm(x)_{\mathbf{w}^\pm} = b_1 x_{w_1} + b_2 x_{w_2} + \dots + b_k x_{w_k},$$

where the addition is performed over $\mathbb{Z}$.

Note that the signed W -sum encoding is a modification of the W -sum encoding [Ta17; HS25a] that maps a vector $x \in \mathbb{Z}^n$ to $\text{Enc}_W(x) \in \mathbb{Z}^W$ defined by

$$\text{Enc}_W(x)_{\mathbf{w}} = x_{w_1} + x_{w_2} + \dots + x_{w_k} \text{ for } \mathbf{w} = (w_1, \dots, w_k) \in W. \quad (7)$$

The main reason why we use the *signed* summation is that the signed W -sum encoding exhibits the following distance amplification property even for integer vectors.

Lemma 5.3. *Let $W \subseteq [n]^k$ be the set of k -walks on a d -regular λ -expander graph $G = ([n], E)$, where $0 < \lambda < 1$ is a constant. Then,*

1. For any $x, y \in \mathbb{Z}^n$, it holds that

$$\text{dist}(\text{Enc}_W^\pm(x), \text{Enc}_W^\pm(y)) \leq k \cdot \text{dist}(x, y).$$

2. There exists a constant $C = C(\lambda) > 0$ such that, for every $\beta > 0$, every $k \in \mathbb{N}$, and for every $x, y \in \mathbb{Z}^n$ satisfying $\text{dist}(x, y) \geq \beta$, it holds that

$$\text{dist}(\text{Enc}_W^\pm(x), \text{Enc}_W^\pm(y)) \geq 1 - \frac{C}{\sqrt{\beta k}}.$$

Remark 5.4. Consider the encoding $\text{Enc}_W: \mathbb{Z}^n \rightarrow \mathbb{Z}^W$ defined by (7). If we work over a finite field $\mathbb{F}_p$ (that is, the summation is taken modulo p), then the encoding has a strong distance amplification property: for any prime p and any $\delta, \varepsilon > 0$, there exists $k = k(p, \delta, \varepsilon) \in \mathbb{N}$ such that for any $x, y \in \mathbb{F}_p^n$ with $\text{dist}(x, y) \geq \delta$, it holds that

$$\text{dist}(\text{Enc}_W(x), \text{Enc}_W(y)) \geq 1 - \frac{1}{p} - \varepsilon.$$

However, such a distance amplification does not hold in general for vectors over $\mathbb{Z}^n$. To see this, consider the complete graph on n vertices with a self-loop at each vertex. In this case, choosing $\mathbf{w} \sim W$ corresponds to selecting k vertices independently and uniformly from $[n]$.

Let $k \in \mathbb{N}$ be any constant and define $y = 0$ (the all-zero vector) and $x \in \mathbb{Z}^n$ as follows: set $x_i = k - 1$ for a $1/k$ fraction of indices $i \in [n]$ and $x_i = -1$ for the remaining $(1 - 1/k)$ fraction (for simplicity, assume k divides n). Then $\text{dist}(x, y) = 1$.

On the other hand, for a random walk $\mathbf{w} \sim W$, the $\mathbf{w}$ -th entry

$$\text{Enc}_W(x - y)_{\mathbf{w}} = \text{Enc}_W(x)_{\mathbf{w}} = x_{w_1} + \cdots + x_{w_k}$$

is identical to the sum of k i.i.d. random variables $X_1, \dots, X_k$ where

$$X_i = \begin{cases} k - 1 & \text{with probability } 1/k, \\ -1 & \text{with probability } 1 - 1/k. \end{cases}$$

Hence

$$\Pr \left[\sum_{i=1}^k X_i = 0 \right] = \Pr[\text{exactly one } X_i \text{ equals } k - 1] = \binom{k}{1} \frac{1}{k} \left(1 - \frac{1}{k}\right)^{k-1} = \left(1 - \frac{1}{k}\right)^{k-1} \rightarrow e^{-1}$$

as $k \rightarrow \infty$. Thus $\text{Enc}_W(x)$ and $\text{Enc}_W(y)$ still agree on a constant fraction of coordinates, showing that distance amplification fails over $\mathbb{Z}^n$.

To prove Lemma 5.3, we will use the following two inequalities.

Lemma 5.5 (Littlewood–Offord inequality [Erd45]; see, e.g., [TV09, Corollary 7.4]). Let $a_1, \dots, a_n \in \mathbb{R} \setminus \{0\}$ be nonzero real numbers and let $\mathbf{b} = (b_1, \dots, b_n) \sim \{\pm 1\}^n$ be a random sign vector. Then for every $x \in \mathbb{R}$,

$$\Pr_{\mathbf{b}} \left[\sum_{i=1}^n b_i a_i = x \right] \leq \frac{\binom{n}{\lfloor n/2 \rfloor}}{2^n}.$$

Lemma 5.6 (Expander Chernoff bound [Gil98]). *There exists an absolute constant $c > 0$ such that the following holds. Let $G = ([n], E)$ be a d -regular λ -expander graph, and let $(u_1, \dots, u_k)$ be a random walk on G whose initial vertex u_1 is distributed uniformly over $[n]$. Let $B \subseteq [n]$ be any subset, and define $N := |\{a \in [k] : u_a \in B\}|$. Then for every $\kappa \in (0, 1)$,*

$$\Pr\left[N \leq (1 - \kappa) \frac{|B|}{n} k\right] \leq \exp\left(-c(1 - \lambda) \kappa^2 \frac{|B|}{n} k\right).$$

Now we prove Lemma 5.3.

Proof of Item 1. Since $\Pr_{u \sim [n]}[x_u \neq y_u] = \text{dist}(x, y)$, we have

$$\begin{aligned} \text{dist}(\text{Enc}_W^\pm(x), \text{Enc}_W^\pm(y)) &= \Pr_{\substack{(u_1, \dots, u_k) \sim W \\ (\mathbf{b}_1, \dots, \mathbf{b}_k) \sim \{\pm 1\}^k}} \left[\sum_{a=1}^k b_a x_{u_a} \neq \sum_{a=1}^k b_a y_{u_a} \right] \\ &\leq \Pr_{(u_1, \dots, u_k) \sim W} [\exists a \in [k] : x_{u_a} \neq y_{u_a}] \\ &\leq \sum_{a \in [k]} \Pr_{(u_1, \dots, u_k) \sim W} [x_{u_a} \neq y_{u_a}] \\ &\leq k \cdot \text{dist}(x, y). \end{aligned}$$

In the last inequality, note that for every fixed $a \in [k]$, the marginal distribution of u_a for $(u_1, \dots, u_k) \sim W$ is uniform over $[n]$ since G is regular. $\square$

Proof of Item 2. Suppose that $\text{dist}(x, y) \geq \beta$. Let $z := x - y \in \mathbb{Z}^n$ and define $B := \{u \in [n] : z_u \neq 0\}$. Then $|B| \geq \beta n$.

Let $\mathbf{u} = (u_1, \dots, u_k) \sim W$ be a random k -walk on G and $\mathbf{b} \sim \{\pm 1\}^k$. Define $S := \text{Enc}_W^\pm(z)_{\mathbf{b}, \mathbf{u}} = \sum_{a=1}^k b_a z_{u_a}$. Then $\text{dist}(\text{Enc}_W^\pm(x), \text{Enc}_W^\pm(y)) = \Pr[S \neq 0] = 1 - \Pr[S = 0]$.

For the walk $\mathbf{u}$, define $N := |\{a \in [k] : u_a \in B\}|$. Condition on $\mathbf{u}$. If $N = t$, then S can be written as $S = \sum_{j=1}^t \varepsilon_j a_j$, where each $a_j \neq 0$ and $\varepsilon_j \in \{\pm 1\}$ are independent. Thus, by the Littlewood–Offord inequality (Lemma 5.5), we have $\Pr_{\mathbf{b}}[S = 0 \mid \mathbf{u}] \leq \frac{C_0}{\sqrt{t+1}}$ for some absolute constant C_0 . Taking expectation over $\mathbf{u}$, we obtain

$$\Pr[S = 0] = \mathbb{E}_{\mathbf{u}} \left[\Pr_{\mathbf{b}}[S = 0 \mid \mathbf{u}] \right] \leq \mathbb{E}_{\mathbf{u}} \left[\frac{C_0}{\sqrt{N+1}} \right] \quad (8)$$

We now bound N . Since each u_a is marginally uniform over $[n]$, we have $\mathbb{E}[N] \geq \beta k$.

By the expander Chernoff bound (Lemma 5.6), there exists an absolute constant $c > 0$ such that

$$\Pr\left[N \leq \frac{\beta k}{2}\right] \leq \exp(-c(1 - \lambda)\beta k).$$

Using this, we bound the expectation as

$$\begin{aligned} \mathbb{E}[1/\sqrt{N+1}] &\leq \Pr\left[N < \frac{\beta k}{2}\right] + \frac{1}{\sqrt{\frac{\beta k}{2} + 1}} \\ &\leq \exp(-c(1 - \lambda)\beta k) + \frac{\sqrt{2}}{\sqrt{\beta k}} \\ &\leq \frac{C_1}{\sqrt{\beta k}} \end{aligned} \quad (9)$$

for some absolute constant C_1 . Note that this inequality holds for all $k \in \mathbb{N}$ since $\max_{k \in \mathbb{N}} \exp(-c(1-\lambda)\beta k) \cdot \sqrt{\beta k}$ is bounded.

Combining (8) and (9), we obtain

$$\text{dist}(\text{Enc}_W^\pm(x), \text{Enc}_W^\pm(y)) = 1 - \Pr[S = 0] \geq 1 - \frac{C_0 C_1}{\sqrt{\beta k}}.$$

□

5.2 Simulating Encoding by Matrix Multiplication

We naturally extend the notion of the signed W -sum encoding to deal with matrices as follows. For $W \subseteq [n]^k$, define $W^{\otimes 2} \subseteq ([n]^2)^k$ as

$$W^{\otimes 2} = \{((w_1, v_1), \dots, (w_k, v_k)) \in ([n]^2)^k : (w_1, \dots, w_k), (v_1, \dots, v_k) \in W\}.$$

If $W \subseteq [n]^k$ is the set of k -walks on a regular expander graph $G = ([n], E)$, then $W^{\otimes 2}$ is the set of k -walks on the tensor product graph $G \otimes G$. Moreover, if G is λ -expander, then $G \otimes G$ is also λ -expander (see Section 3.1).

While $\text{Enc}_W^\pm(\cdot)$ is used to encode a vector, we can use $W^{\otimes 2}$ to encode a matrix as follows: Any matrix $C \in \mathbb{Z}^{n \times n}$ can be encoded into $\text{Enc}_{W^{\otimes 2}}^\pm(C) \in \mathbb{Z}^{W^\pm \times W}$, where for $\mathbf{w}^\pm = (\mathbf{b}, \mathbf{w})$ with $\mathbf{w} = (w_1, \dots, w_k)$ and $\mathbf{v} = (v_1, \dots, v_k) \in W$, the $(\mathbf{w}^\pm, \mathbf{v})$ -entry of $\text{Enc}_{W^{\otimes 2}}^\pm(C)$ is given by

$$\text{Enc}_{W^{\otimes 2}}^\pm(C)_{\mathbf{w}^\pm, \mathbf{v}} = b_1 C_{w_1, v_1} + b_2 C_{w_2, v_2} + \dots + b_k C_{w_k, v_k},$$

In the following, we show that, using an average-case approximation algorithm for random matrix multiplication, we can compute an approximation of $\text{Enc}_{W^{\otimes 2}}^\pm(AB)$ by following the proof idea of Hirahara and Shimizu [HS25a, Lemma 15] who showed the same result for the W -sum encoding $\text{Enc}_W(\cdot)$ of (7).

Lemma 5.7. *Let $k, d \in \mathbb{N}$ and $\varepsilon > 0$ be any constants. There exists an oracle algorithm $M^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\bar{A}, \bar{B} \sim \{0,1\}^{kn \times kn}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon,$$

the algorithm $M^\mathcal{O}$ is given $A, B \sim \{0,1\}^{n \times n}$ and a d -regular graph $G = ([n], E)$ with girth at least k as input, satisfies

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^\mathcal{O}(A, B, G), \text{Enc}_{W^{\otimes 2}}^\pm(AB))] \geq \varepsilon - \frac{2k}{d},$$

where $W \subseteq [n]^k$ is the set of k -walks on G . Moreover, $M^\mathcal{O}$ makes at most $O(1)$ oracle calls to $\mathcal{O}$.

Note that $\mathcal{O}$ receives two $kn \times kn$ matrices, whereas $M^\mathcal{O}$ receives two $n \times n$ matrices. The proof is deferred to Section A.

Difference from the finite-field setting. The proof of [HS25a, Lemma 15] assumes that the input matrices are uniformly random over a finite field $\mathbb{F}_p$, and works with the unsigned encoding $\text{Enc}_{W^{\otimes 2}}(\cdot)$. In contrast, in our setting the input matrices are drawn from $\{0, 1\}^{n \times n}$ and we consider the signed encoding $\text{Enc}_{W^{\otimes 2}}^{\pm}(\cdot)$.

The main technical difference is that, over $\mathbb{F}_p$, the distribution is invariant under multiplication by nonzero scalars, whereas over $\{0, 1\}$ this is no longer the case. In particular, multiplying a matrix by -1 changes its distribution, which prevents us from directly simulating the signed encoding.

To address this issue, we first apply an affine transformation that maps $\{0, 1\}^{n \times n}$ to $\{\pm 1\}^{n \times n}$, where the distribution becomes invariant under sign flips. We then simulate the signed encoding in this $\{\pm 1\}$ domain. Finally, to invoke the oracle (which expects $\{0, 1\}$ -valued inputs), we convert the resulting matrices back to $\{0, 1\}$ by another affine transformation.

5.3 Approximate List-Decoding for Signed Sum Encoding

In this subsection, we present an approximate list-decoding algorithm for the signed sum encoding $\text{Enc}_W^{\pm}(\cdot): \mathbb{Z}^n \rightarrow \mathbb{Z}^{W^{\pm}}$ under the assumption that each entry of the original message is associated with a list consisting of a small number of possible values.

Lemma 5.8. *Let $d, \ell, k \in \mathbb{N}$ and $\delta, \varepsilon > 0$ be constants, and let λ_0 be the absolute constant in Lemma 3.4. Assume $k \geq \frac{C}{\varepsilon^4 \delta}$ for a sufficiently large absolute constant $C > 0$ (depending only on λ_0).*

Let $G = ([n], E)$ be a d -regular λ_0 -expander graph, and let $W \subseteq [n]^k$ be the set of k -walks on G . For each $i \in [n]$, let $L_i \subseteq \mathbb{Z}$ be a list of size ℓ , and define

$$\mathcal{L} := \{x \in \mathbb{Z}^n : x_i \in L_i \text{ for all } i \in [n]\}.$$

There exists an algorithm with the following property. Given $\tilde{y} \in \mathbb{Z}^{W^{\pm}}$, the algorithm runs in time $\tilde{O}(|W|)$ and outputs a list $\tilde{L} \subseteq \mathcal{L}$ of size at most $\lceil 2/\varepsilon \rceil$. Moreover, for every $x^ \in \mathcal{L}$ satisfying*

$$\text{agree}(\text{Enc}_W^{\pm}(x^*), \tilde{y}) \geq \varepsilon,$$

there exists $\tilde{x} \in \tilde{L}$ such that

$$\text{agree}(\tilde{x}, x^*) \geq 1 - \delta.$$

To prove Lemma 5.8, given $\tilde{y} \in \mathbb{Z}^{W^{\pm}}$ and a collection of lists $(L_i)_{i \in [n]}$ for $L_i \subseteq \mathbb{Z}$ with $|L_i| = \ell$ for all $i \in [n]$, we first construct an instance φ_R of MAX k -VCSP($[\ell]$) for $R \subseteq W^{\pm}$ as follows.

MAX k -CSP instance φ_R

For $a \in [\ell]$, let $L_i[a] \in L_i$ denote the a -th element of L_i (we assume that the elements of L_i are in arbitrary order). Let $\mathcal{L} = \{x \in \mathbb{Z}^n : x_i \in L_i \text{ for all } i \in [n]\}$.

Variables. The variables are $s_1, \dots, s_n \in [\ell]$ for $i \in [n]$ (thus the alphabet is $[\ell]$). For an assignment $s \in [\ell]^n$, let $x[s] = (L_1[s_1], \dots, L_n[s_n]) \in \mathcal{L}$ be the vector whose i -th entry is the s_i -th element of L_i for $i \in [n]$.

Constraints. The constraint $P_{\mathbf{w}^\pm} : [\ell]^k \rightarrow \{0, 1\}$ associated with a signed walk $\mathbf{w}^\pm = (\mathbf{b}, \mathbf{w}) \in W^\pm$ is given by

$$P_{\mathbf{w}^\pm}(a_1, \dots, a_k) = \begin{cases} 1 & \text{if } \mathbf{w}^\pm \in R \text{ and } \tilde{y}_{\mathbf{w}^\pm} = b_1 L_{w_1}[a_1] + b_2 L_{w_2}[a_2] + \dots + b_k L_{w_k}[a_k], \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathbf{b} = (b_1, \dots, b_k) \in \{\pm 1\}^k$ and $\mathbf{w} = (w_1, \dots, w_k) \in W$.

The motivation of the above construction is that if $R = W^\pm$, then the value $\text{val}(s, \varphi_R)$ of an assignment $s \in [\ell]^n$ equals to

$$\Pr_{\mathbf{w}^\pm \sim W^\pm} [\tilde{y}_{\mathbf{w}^\pm} = b_1 L_{w_1}[s_{w_1}] + b_2 L_{w_2}[s_{w_2}] + \dots + b_k L_{w_k}[s_{w_k}]] = \text{agree}(\text{Enc}_W^\pm(L[s]), \tilde{y}).$$

Therefore, finding an optimal solution to the MAX k -CSP instance φ_R with $R = W^\pm$ is equivalent to finding a vector $x \in \mathcal{L}$ such that $\text{Enc}_W^\pm(x)$ is closest to $\tilde{y}$ among all vectors in $\mathcal{L}$.

We can use the near-linear time algorithm for MAX k -VCSP of Lemma 3.4 to find an approximate solution to φ_R .

Lemma 5.9. *Let d, k, ℓ be any constants and λ_0 be the absolute constant in Lemma 3.4. For any constant $\gamma > 0$, there exists an algorithm M_γ that satisfies the following: Given a MAX k -CSP instance φ_R as above with $W \subseteq [n]^k$ being the set of k -walks on a d -regular λ_0 -expander graph $G = ([n], E)$, the algorithm M_γ runs in time $\tilde{O}(|W|)$ and outputs an assignment $s \in [\ell]^n$ such that*

$$\text{val}(s, \varphi_R) \geq \text{val}(\varphi_R) - \gamma.$$

Although the instance φ_R is a MAX k -CSP supported on $W^\pm = \{\pm 1\}^k \times W$, the set $W^\pm$ is a disjoint union of 2^k copies of W and hence does not itself form an expanding structure. Therefore, we cannot directly apply Lemma 3.4 to φ_R .

To circumvent this issue, we aggregate, for each $\mathbf{w} \in W$, the constraints corresponding to all sign vectors $\mathbf{b} \in \{\pm 1\}^k$ into a single valued constraint by taking their average. This yields a MAX k -VCSP instance ψ_R supported on W , which preserves the objective value of every assignment. Consequently, solving ψ_R approximately via Lemma 3.4 yields an approximate solution to the original MAX k -CSP instance φ_R .

Proof of Lemma 5.9. For each $\mathbf{w} = (w_1, \dots, w_k) \in W$, define a valued constraint $Q_{\mathbf{w}} : [\ell]^k \rightarrow [0, 1]$ by

$$Q_{\mathbf{w}}(a_1, \dots, a_k) := \frac{1}{2^k} \sum_{\mathbf{b}=(b_1, \dots, b_k) \in \{\pm 1\}^k} P_{(\mathbf{b}, \mathbf{w})}(a_1, \dots, a_k),$$

where $P_{(\mathbf{b}, \mathbf{w})}$ is the Boolean constraint of the MAX k -CSP instance φ_R above. Consider a MAX k -VCSP instance $\psi_R = (n, W, (Q_{\mathbf{w}})_{\mathbf{w} \in W})$ whose supporting set is W .

Note that $\text{val}(s, \psi_R) = \text{val}(s, \varphi_R)$ for every assignment $s \in [\ell]^n$ since

$$\begin{aligned} \text{val}(s, \psi_R) &= \mathbb{E}_{\mathbf{w} \in W} [Q_{\mathbf{w}}(s_{w_1}, \dots, s_{w_k})] \\ &= \mathbb{E}_{\mathbf{w} \in W, \mathbf{b} \in \{\pm 1\}^k} [P_{(\mathbf{b}, \mathbf{w})}(s_{w_1}, \dots, s_{w_k})] \\ &= \text{val}(s, \varphi_R). \end{aligned}$$

Now apply Lemma 3.4 to the VCSP instance ψ_R . Since W is the set of k -walks on the d -regular λ_0 -expander graph G , there exists a randomized algorithm running in time $\tilde{O}(n + |W|) = \tilde{O}(|W|)$ that outputs an assignment $s \in [\ell]^n$ satisfying

$$\text{val}(s, \psi_R) \geq \text{val}(\psi_R) - \gamma.$$

Since $\text{val}(s, \psi_R) = \text{val}(s, \varphi_R)$, this completes the proof. $\square$

Now, we can prove Lemma 5.8.

Proof of Lemma 5.8. Our algorithm M uses the algorithm $M_{\varepsilon/8}$ of Lemma 5.9 for $\gamma = \varepsilon/8$ as a subroutine. Given a vector $\tilde{y} \in \mathbb{Z}^{W^\pm}$ and a collection of lists $(L_i)_{i \in [n]}$ with $|L_i| = \ell$ for all $i \in [n]$, the algorithm M runs as follows:

1. Initialize $R_0 = W^\pm$ and $\tilde{L} = \emptyset$.
2. For $t = 1, 2, 3, \dots$, do:
 - (a) Compute $s^{(t)} := M_{\varepsilon/8}(\varphi_{R_{t-1}}) \in [\ell]^n$ and let $x^{(t)} = L[s^{(t)}] \in \mathbb{Z}^n$.
 - (b) Let
$$S_t = \{(\mathbf{b}, \mathbf{w}) \in R_{t-1} : \text{Enc}_W^\pm(x^{(t)})_{(\mathbf{b}, \mathbf{w})} = \tilde{y}_{(\mathbf{b}, \mathbf{w})}\}.$$
 - (c) If $|S_t| \leq (\varepsilon/2)|W^\pm|$, output $\tilde{L}$ and terminate.
 - (d) Otherwise, append $x^{(t)}$ to $\tilde{L}$ and set $R_t = R_{t-1} \setminus S_t$.

Call a round t *successful* if $|S_t| > (\varepsilon/2)|W^\pm|$ and *failed* otherwise. Note that M terminates at the first failed round.

Since each successful round removes at least $(\varepsilon/2)|W^\pm|$ elements from the remaining set R_{t-1} , the algorithm terminates after at most $\lceil 2/\varepsilon \rceil$ successful rounds. Hence the size of $\tilde{L}$ is at most $\lceil 2/\varepsilon \rceil$ and the running time of M is $\tilde{O}(|W|) = \tilde{O}(n)$ since each call of $M_{\varepsilon/8}$ takes time $\tilde{O}(|W|)$ and $|W| = nd^{k-1}$ for $k, d = O(1)$.

Correctness. Now we prove the correctness of the algorithm M . Fix any $x^* \in \mathcal{L}$ such that $\text{dist}(\text{Enc}_W^\pm(x^*), \tilde{y}) \leq 1 - \varepsilon$. Define the set of coordinates on which $\text{Enc}_W^\pm(x^*)$ agrees with $\tilde{y}$ by

$$S^* = \{(\mathbf{b}, \mathbf{w}) \in W^\pm : \text{Enc}_W^\pm(x^*)_{(\mathbf{b}, \mathbf{w})} = \tilde{y}_{(\mathbf{b}, \mathbf{w})}\}.$$

Note that $|S^*| \geq \varepsilon|W^\pm|$.

Suppose for contradiction that *none* of the outputs in $\tilde{L}$ is δ -close to x^* , that is,

$$\text{dist}(x^{(t)}, x^*) > \delta$$

for every successful round t . By Lemma 5.3 (Item 2) with $\beta = \delta$, we have

$$\text{agree}(\text{Enc}_W^\pm(x^{(t)}), \text{Enc}_W^\pm(x^*)) < \frac{C'}{\sqrt{\delta k}} \leq \frac{\varepsilon^2}{16}, \quad (10)$$

where $C' > 0$ is the absolute constant from Lemma 5.3 (Item 2). The last inequality is due to the assumption on k .

Now fix a successful round t . Every coordinate $(\mathbf{b}, \mathbf{w})$ in $S_t \cap S^*$ satisfies

$$\text{Enc}_W^\pm(x^{(t)})_{(\mathbf{b}, \mathbf{w})} = \tilde{y}_{(\mathbf{b}, \mathbf{w})} = \text{Enc}_W^\pm(x^*)_{(\mathbf{b}, \mathbf{w})}$$

and thus

$$S_t \cap S^* \subseteq \{(\mathbf{b}, \mathbf{w}) \in W^\pm : \text{Enc}_W^\pm(x^{(t)})_{(\mathbf{b}, \mathbf{w})} = \text{Enc}_W^\pm(x^*)_{(\mathbf{b}, \mathbf{w})}\}.$$

By (10), we have

$$|S_t \cap S^*| \leq \frac{\varepsilon^2}{16} |W^\pm|.$$

Let $m \leq 2/\varepsilon$ be the number of successful rounds. Consider the first unsuccessful round $t = m+1$. At the beginning of this round, we have

$$R_m = W^\pm \setminus (S_1 \cup \dots \cup S_m)$$

and thus

$$\begin{aligned} |S^* \cap R_m| &= |S^*| - \sum_{t=1}^m |S_t \cap S^*| \\ &\geq \varepsilon |W^\pm| - m \cdot \frac{\varepsilon^2}{16} \cdot |W^\pm| \\ &\geq \varepsilon |W^\pm| - \frac{2}{\varepsilon} \cdot \frac{\varepsilon^2}{16} \cdot |W^\pm| \\ &= \frac{7\varepsilon}{8} |W^\pm|. \end{aligned}$$

This means that, even after all successful rounds, the true codeword $\text{Enc}_W^\pm(x^*)$ agrees with $\tilde{y}$ on at least $\frac{7\varepsilon}{8} |W^\pm|$ coordinates in R_m . Therefore

$$\text{val}(\varphi_{R_m}) \geq \frac{7}{8} \varepsilon.$$

By Lemma 5.9, the output $x^{(m+1)}$ of $M_{\varepsilon/8}(\varphi_{R_m})$ satisfies at least

$$\left(\frac{7}{8} \varepsilon - \frac{\varepsilon}{8}\right) |W^\pm| = \frac{3\varepsilon}{4} |W^\pm|$$

constraints of φ_{R_m} . Thus, $|S_{m+1}| \geq \frac{3\varepsilon}{4} |W^\pm|$. This means that round $m+1$ is successful, which contradicts the definition of m . □

5.4 Proof of Distance Amplification

In this subsection, we prove Lemma 5.1.

Proof of Lemma 5.1. Given constants $\varepsilon, \delta > 0$, we choose parameters $\delta_0, k, d, \varepsilon^*, \delta^*$ satisfying

$$\begin{aligned}\delta_0 &:= \frac{\delta}{4}, \\ \varepsilon^* &:= \frac{\varepsilon}{4}, \\ \frac{3}{\sqrt{d}} &\leq \lambda_0, \\ \frac{2k}{d} &\leq \frac{\varepsilon}{10}, \\ k &\geq \frac{C}{\delta_0(\varepsilon^*)^4}, \\ \delta^* &:= \min\left\{\frac{\delta}{4}, \frac{\varepsilon}{8k}\right\},\end{aligned}$$

where $0 < \lambda_0 < 1$ is the absolute constant in Lemma 3.4 and $C > 0$ is the absolute constant from Lemma 5.8.

Since ε, δ are constants, such a choice is possible by first taking k sufficiently large and then taking d sufficiently large, e.g., $k = \Theta\left(\frac{1}{\delta\varepsilon^4}\right)$ and $d = \Theta\left(\frac{k}{\varepsilon}\right) = \Theta\left(\frac{1}{\delta\varepsilon^5}\right)$.

Let $K = K(\delta^*, \varepsilon^*)$ be the constant from Lemma 4.1, and let

$$k' := \max\{K, k\}.$$

By Lemmas 3.5 and 3.7, the assumption on $\mathcal{O}$ that multiplies two $k'n \times k'n$ matrices implies that there exist nonuniform randomized oracles $\hat{\mathcal{O}}_1$ and $\mathcal{O}_2$ such that

$$\mathbb{E}_{\substack{\bar{A}, \bar{B} \sim \{0,1\}^{Kn \times Kn} \\ \hat{\mathcal{O}}_1}} [\text{agree}(\hat{\mathcal{O}}_1(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon \quad \text{and} \quad \mathbb{E}_{\substack{\bar{A}, \bar{B} \sim \{0,1\}^{kn \times kn} \\ \mathcal{O}_2}} [\text{agree}(\mathcal{O}_2(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon.$$

Again, applying Lemma 3.5 to $\hat{\mathcal{O}}_1$, we obtain an oracle $\mathcal{O}_1$ such that

$$\mathbb{E}_{\substack{\bar{A} \sim \{0,1\}^{n \times Kn} \\ \bar{B} \sim \{0,1\}^{Kn \times n} \\ \mathcal{O}_1}} [\text{agree}(\mathcal{O}_1(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon.$$

Let $A, B \in \{0,1\}^{n \times n}$ be the input matrices.

We first apply Lemma 4.1 with parameter $\delta := \delta^*$ and oracle $\mathcal{O}_1$. Then, with probability at least $1 - \delta^*$ over $A, B \sim \{0,1\}^{n \times n}$, we obtain a collection of nonempty multisets $(L_{i,j})_{i,j \in [n]}$ such that $|L_{i,j}| = O(\log(1/\delta^*)/\varepsilon) = O(1)$ for all $i, j \in [n]$ and

$$\Pr_{i,j \sim [n]} [(AB)_{i,j} \in L_{i,j}] \geq 1 - \delta^*.$$

Fix such a successful outcome. Then there exists a matrix $X^* = (x_{i,j}^*)_{i,j \in [n]}$ such that $x_{i,j}^* \in L_{i,j}$ for all $i, j \in [n]$ and

$$\text{dist}(X^*, AB) \leq \delta^*.$$

If necessary, by duplicating elements in each multiset, we may assume that $|L_{i,j}| = \ell'$ for all $i, j \in [n]$, where $\ell' = O(1)$ depends only on δ and ε .

Next, let $G = ([n], E)$ be a d -regular λ_0 -expander graph of girth at least k , and let $W \subseteq [n]^k$ be the set of k -walks on G . Such a graph can be constructed in time $\tilde{O}(n)$ by Lemma 3.1 (specifically, repeat sampling random regular graphs until a graph with girth at least k is obtained for $\tilde{O}(1)$ times). Define

$$\text{Enc}(\cdot) := \text{Enc}_{W^{\otimes 2}}^{\pm}(\cdot).$$

Applying Lemma 5.7 with oracle $\mathcal{O}_2$, we obtain a matrix $\tilde{C} \in \mathbb{Z}^{W^{\otimes 2} \times W}$ such that

$$\mathbb{E}_{A,B,\mathcal{O}_2} \left[\text{agree}(\tilde{C}, \text{Enc}(AB)) \right] \geq \varepsilon - \frac{2k}{d} \geq \frac{9\varepsilon}{10}.$$

Since $0 \leq \text{agree}(\tilde{C}, \text{Enc}(AB)) \leq 1$, the Markov inequality implies that

$$\Pr_{A,B,\mathcal{O}_2} \left[\text{agree}(\tilde{C}, \text{Enc}(AB)) \geq \frac{\varepsilon}{2} \right] \geq \frac{9\varepsilon/10 - \varepsilon/2}{1 - \varepsilon/2} \geq \frac{2\varepsilon}{5}.$$

Fix such an outcome. Then

$$\text{dist}(\tilde{C}, \text{Enc}(AB)) \leq 1 - \frac{\varepsilon}{2}.$$

By Lemma 5.3 (Item 1) applied to the tensor-product graph $G \otimes G$,

$$\text{dist}(\text{Enc}(X^*), \text{Enc}(AB)) \leq k \cdot \text{dist}(X^*, AB) \leq k\delta^* \leq \frac{\varepsilon}{8}.$$

Hence, by the triangle inequality,

$$\text{dist}(\text{Enc}(X^*), \tilde{C}) \leq \text{dist}(\text{Enc}(X^*), \text{Enc}(AB)) + \text{dist}(\text{Enc}(AB), \tilde{C}) \leq \frac{\varepsilon}{8} + 1 - \frac{\varepsilon}{2} = 1 - \frac{3\varepsilon}{8}.$$

Equivalently,

$$\text{agree}(\text{Enc}(X^*), \tilde{C}) \geq \frac{3\varepsilon}{8} > \varepsilon^*.$$

Now we view matrices in $\mathbb{Z}^{n \times n}$ as vectors indexed by $[n]^2$. Since $W^{\otimes 2}$ is the set of k -walks on the tensor-product graph $G \otimes G$, and $G \otimes G$ is a d^2 -regular graph whose second eigenvalue is at most λ , we may apply Lemma 5.8 with parameters $(d^2, \ell', k, \delta_0, \varepsilon^*)$ to the received word $\tilde{C}$ and the collection of lists $(L_{i,j})_{(i,j) \in [n]^2}$. Thus, in time $\tilde{O}(|W^{\otimes 2}|) = \tilde{O}(n^2)$, we obtain a list $\tilde{L}$ of matrices of size at most $\lceil 2/\varepsilon^* \rceil = O(1/\varepsilon)$ such that, since $\text{dist}(\text{Enc}(X^*), \tilde{C}) \leq 1 - \varepsilon^*$, there exists some matrix $\hat{X} \in \tilde{L}$ satisfying

$$\text{dist}(\hat{X}, X^*) \leq \delta_0.$$

Therefore,

$$\text{dist}(\hat{X}, AB) \leq \text{dist}(\hat{X}, X^*) + \text{dist}(X^*, AB) \leq \delta_0 + \delta^* = \frac{\delta}{2}.$$

It remains to identify such a matrix from the list $\tilde{L}$. For each candidate $X \in \tilde{L}$, we run the tester from Lemma 3.8 on input (A, B, X) with parameters

$$\alpha := 1 - \delta \quad \text{and} \quad \beta := \frac{\delta}{4}.$$

If the tester accepts, we output X and halt. If no candidate is accepted, we output an arbitrary matrix.

Since $\text{agree}(\hat{X}, AB) \geq 1 - \delta/2 \geq \alpha + \beta$, the tester accepts $\hat{X}$ with probability at least $1 - 1/n^3$. On the other hand, any candidate X with $\text{agree}(X, AB) \leq 1 - \delta$ is rejected with probability at least $1 - 1/n^3$. Because $|\tilde{L}| = O(1)$, by a union bound, with probability at least $1 - O(1/n^3)$ the tester simultaneously behaves correctly on every candidate in $\tilde{L}$. On this event, at least one candidate is accepted, and every accepted candidate satisfies $\text{agree}(X, AB) > 1 - \delta$.

Combining all success events, we conclude that

$$\Pr_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB) \geq 1 - \delta] \geq \frac{2\varepsilon}{5}.$$

The total running time is $\tilde{O}(n^2)$. Indeed, Lemma 4.1 and Lemma 5.7 use only $O(1)$ oracle calls for constant parameters, Lemma 5.8 runs in time $\tilde{O}(n^2)$, and the final testing step runs in time

$$O(|\tilde{L}| \cdot (n^2 + n \log n / \delta^2)) = \tilde{O}(n^2),$$

since $|\tilde{L}| = O(1)$. This completes the proof. $\square$

6 Hardness Amplification

In Section 5, we obtained an algorithm that produces a highly accurate approximation of AB for a constant fraction of random inputs. In this section, we amplify the success probability of our algorithm from a constant to an arbitrarily high level.

Due to technical constraints, we perform this amplification in two stages:

1. **From constant to 0.99-agreement.** We first transform an algorithm that succeeds with probability at least ε into one that succeeds with probability at least $1 - \gamma$ for any constant $0 < \gamma < 1$.
2. **From 0.99 to $1 - o(1)$ -agreement.** We then further amplify the success probability from $1 - \gamma$ to $1 - \delta$ for an arbitrarily small $\delta = \delta(n) > 0$. This reduction is based on Lemma 4.1 and thus it is nonuniform.

6.1 Amplification via Direct Product Lemma

In this subsection, we prove the following lemma, which asserts that the average agreement can be amplified to

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB)] \geq 1 - \gamma$$

for any small constant $\gamma > 0$.

Lemma 6.1. *For any constants $\gamma, \varepsilon > 0$, there exist a constant $\delta = \delta(\gamma, \varepsilon)$ and an $\tilde{O}(n^2)$ -time $O(1)$ -query randomized oracle algorithm $M^{\mathcal{O}}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\Pr_{A, B \sim \{0,1\}^{n \times n}}_{\mathcal{O}} [\text{agree}(\mathcal{O}(A, B), AB) \geq 1 - \delta] \geq \varepsilon,$$

the algorithm $M^\mathcal{O}$ is given $A, B \sim \{0, 1\}^{n \times n}$ as input and satisfies

$$\mathbb{E}_{\substack{A, B \sim \{0, 1\}^{n \times n} \\ M^\mathcal{O}}} [\text{agree}(M^\mathcal{O}(A, B), AB)] \geq 1 - \gamma.$$

The proof is the same as the proof of [HS25a, Lemma 17] but we present it here for completeness.

Proof. Given $\gamma, \varepsilon > 0$, set

$$K := \frac{c \log(1/\varepsilon)}{\gamma^2 \varepsilon^2}, \quad \delta = \frac{\gamma}{2K^2}.$$

Let $A, B \sim \{0, 1\}^{n \times n}$ be the input to $M^\mathcal{O}$. From Lemma 3.7, we may assume that the size n is divisible by K . Let $I_1 \sqcup \dots \sqcup I_K = [n]$ be a partition of $[n]$ into K subsets defined by

$$I_1 = \{1, \dots, n/K\}, \quad I_2 = \{n/K + 1, \dots, 2n/K\}, \quad \dots \quad I_K = \{(K-1)n/K + 1, \dots, n\}.$$

For every $i, j \in [K]$, let $A_{I_i, *}, B_{*, I_j} \in \{0, 1\}^{(n/K) \times n}$ and $B_{*, I_j} \in \{0, 1\}^{n \times (n/K)}$ be the submatrices of A and B obtained by taking the rows indexed by I_i and the columns indexed by I_j , respectively.

The algorithm $M^\mathcal{O}(A, B)$ runs as follows:

1. Initialize a matrix $D \in \mathbb{Z}^{n \times n}$ to be the all-zero matrix.
2. For every $i, j \in [K]$, repeat the following process for $O(\log(1/\gamma)/\varepsilon)$ times:
 - (a) Sample $\bar{A}, \bar{B} \sim \{0, 1\}^{n \times n}$ and $t \sim [K]$ uniformly at random.
 - (b) Update $\bar{A}_{I_t, *} \leftarrow A_{I_t, *}$ and $\bar{B}_{*, I_t} \leftarrow B_{*, I_t}$.
 - (c) Let $C^{i,j} := \mathcal{O}(\bar{A}, \bar{B})_{I_t, I_t}$.
 - (d) Run the tester of Lemma 3.8 with parameters $\alpha = 1 - 3\gamma/4$ and $\beta = \gamma/4$ on input $A_{I_i, *}, B_{*, I_j}, C^{i,j}$. If the tester accepts, then update $D_{I_i, I_j} \leftarrow C^{i,j}$ and terminate the loop for this pair (i, j) .
3. Return D .

Note that $M^\mathcal{O}$ makes $K^2 \cdot O(\log(1/\gamma)/\varepsilon) = O_{\gamma, \varepsilon}(1)$ oracle calls to $\mathcal{O}$ and runs in time $O(n^2)$.

To prove the correctness of $M^\mathcal{O}$, we recall the Direct Product Lemma (Lemma 3.9). Let

$$\mathcal{X} = \{0, 1\}^{(n/K) \times n} \times \{0, 1\}^{n \times (n/K)}.$$

For $y = ((R_1, S_1), \dots, (R_K, S_K)) \in \mathcal{X}^K$, define $A^y, B^y \in \{0, 1\}^{n \times n}$ by

$$A_{I_t, *}^y := R_t \quad \text{and} \quad B_{*, I_t}^y := S_t \quad (t \in [K]).$$

That is, the row block of A^y indexed by I_t is set to R_t , and the column block of B^y indexed by I_t is set to S_t . Since the sets $I_1, \dots, I_K$ form a partition of $[n]$, this uniquely determines A^y and B^y .

If y is uniform in $\mathcal{X}^K$, then A^y, B^y are uniform random matrices in $\{0, 1\}^{n \times n}$. Define

$$S(y) = \Pr_{\mathcal{O}}[\text{agree}(\mathcal{O}(A^y, B^y), A^y B^y) \geq 1 - \delta].$$

By the assumption on $\mathcal{O}$, we have $\mathbb{E}_{y \sim \mathcal{X}^K}[S(y)] \geq \varepsilon$.

Consider the block $i = j = 1$ (other blocks can be shown similarly) and let

$$x := (A_{I_1,*}, B_{*,I_1}) \in \mathcal{X}.$$

Observe that the sampling procedure in Step 2 of the algorithm $M^\mathcal{O}$ corresponds exactly to sampling $y' \sim \Gamma(x)$ in Lemma 3.9. Applying Lemma 3.9 with

$$\delta^* = \gamma/8, \quad \varepsilon^* = \varepsilon/2,$$

and using the choice of K , we obtain

$$\Pr_{x \sim \mathcal{X}} \left[\mathbb{E}_{y' \sim \Gamma(x)} [S(y')] \geq \varepsilon/2 \right] \geq 1 - \gamma/8.$$

Call such x *nice*. Since (A, B) are uniform random matrices,

$$\Pr_{A,B} [(A_{I_1,*}, B_{*,I_1}) \text{ is nice}] \geq 1 - \gamma/8.$$

Fix a nice pair $x = (A_{I_1,*}, B_{*,I_1})$. Each repetition of Step 2 produces $y' \sim \Gamma(x)$, which satisfies

$$\mathbb{E}_{y'} [S(y')] = \Pr_{y', \mathcal{O}} [\text{agree}(\mathcal{O}(A^y, B^y), A^y B^y) \geq 1 - \delta] \geq \frac{\varepsilon}{2}.$$

Thus after $T = O(\log(1/\gamma)/\varepsilon)$ repetitions, with probability $\geq 1 - \gamma/8$, in some repetition we have $\text{dist}(C, \bar{A}\bar{B}) \leq \delta$, where $C = \mathcal{O}(\bar{A}, \bar{B})$. In particular, C and $\bar{A}\bar{B}$ disagree on at most δn^2 entries. Since the block size is $(n/K)^2$, the matrix $C^{1,1} = C_{I_1,I_1}$ satisfies

$$\text{dist}(C^{1,1}, A_{I_1,*} B_{*,I_1}) \leq \frac{\delta n^2}{(n/K)^2} = \delta K^2 = \gamma/2,$$

that is, $\text{agree}(C^{1,1}, A_{I_1,*} B_{*,I_1}) \geq 1 - \gamma/2$.

Running the tester of Lemma 3.8 with parameters $\alpha = 1 - \frac{3\gamma}{4}$ and $\beta = \frac{\gamma}{4}$ ensures that accepted candidates satisfy

$$\text{agree}(D_{I_1,I_1}, A_{I_1,*} B_{*,I_1}) \geq 1 - 3\gamma/4$$

with probability $1 - o(1)$ (over the internal randomness of the tester).

Hence

$$\Pr_{M^\mathcal{O}} [\text{agree}(D_{I_1,I_1}, A_{I_1,*} B_{*,I_1}) \geq 1 - 3\gamma/4] \geq 1 - \gamma/8 - o(1).$$

Averaging over the event that x is nice yields

$$\mathbb{E}_{A,B,M^\mathcal{O}} [\text{agree}(D_{I_1,I_1}, A_{I_1,*} B_{*,I_1})] \geq 1 - \gamma.$$

By symmetry the same holds for every block (i, j) .

Therefore

$$\mathbb{E}_{A,B,M^\mathcal{O}} [\text{agree}(M^\mathcal{O}(A, B), AB)] = \mathbb{E}_{i,j} [\text{agree}(D_{I_i,I_j}, A_{I_i,*} B_{*,I_j})] \geq 1 - \gamma.$$

□

6.2 Amplification via Entry-Wise Majority

In this subsection, we prove that the average agreement can be amplified to

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB)] \geq 1 - \delta$$

for any small parameter $\delta = \delta(n) > 0$. Unlike the previous lemma (Lemma 6.1), this lemma amplifies the agreement to $1 - o(1)$. The proof is straightforward from the moreover part of Lemma 4.1.

Lemma 6.2. *For any parameter $\delta > 0$, there exist a parameter $k = k(\delta) = O(1/\delta^4)$ and an $\tilde{O}(n^2 \log(1/\delta))$ -time randomized nonuniform oracle algorithm $M^{\mathcal{O}}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\bar{A}, \bar{B} \sim \{0,1\}^{kn \times kn}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \frac{3}{4},$$

the algorithm $M^{\mathcal{O}}$ satisfies

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB)] \geq 1 - \delta.$$

Moreover, $M^{\mathcal{O}}$ makes $O(\log(1/\delta))$ oracle calls.

Proof. By Lemma 3.5 with $N = kn$ and $\alpha = 3/4$, we can construct an oracle $\mathcal{O}'$ satisfying

$$\mathbb{E}_{\substack{\bar{A} \sim \{0,1\}^{n \times kn} \\ \bar{B} \sim \{0,1\}^{kn \times n} \\ \mathcal{O}'}} [\text{agree}(\mathcal{O}'(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \frac{3}{4}.$$

Then we apply the moreover part of Lemma 4.1 for parameters $\delta/2$ and $\varepsilon = \frac{3}{4}$, using $\mathcal{O}'$ as oracle, and set

$$k := K(\delta/2, 3/4).$$

Then, with probability at least $1 - \delta/2$ over $A, B \sim \{0,1\}^{n \times n}$, the oracle algorithm of Lemma 4.1 yields a collection of lists $(L_{i,j})_{i,j \in [n]}$ such that $|L_{i,j}| = O(\log(1/\delta))$ for all $(i,j) \in [n]^2$. Moreover, the majority element of $L_{i,j}$ is exactly $(AB)_{i,j}$ for at least a $(1 - \delta)$ -fraction of pairs $(i,j) \in [n]^2$. Therefore,

$$\Pr_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB) \geq 1 - \delta/2] \geq 1 - \delta/2.$$

In particular,

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(M^{\mathcal{O}}(A, B), AB)] \geq (1 - \delta/2)^2 \geq 1 - \delta.$$

□

7 Chinese Remainder Theorem and Binary Expansion

The goal of this section is to convert an average-case approximation algorithm with high agreement into a worst-case exact algorithm.

Our approach is to compute the product AB modulo several small primes and then reconstruct the exact result efficiently by the Chinese Remainder Theorem.

Lemma 7.1 (Chinese Remainder Theorem; [Sho12, Theorem 4.6]). *Let $p_1, \dots, p_m$ be distinct primes and $a_1, \dots, a_m$ be integers such that $0 \leq a_i < p_i$ for all $i \in [m]$. Then, there exists a unique integer $0 \leq x < P := \prod_{i \in [m]} p_i$ such that*

$$x \equiv a_i \pmod{p_i} \quad \text{for all } i \in [m].$$

Moreover, given $p_1, \dots, p_m$ and $a_1, \dots, a_m$ as input, one can compute x in time $O((\log P)^2)$.

For each prime p , we reduce computing $AB \bmod p$ to computing UV for uniformly random $U, V \sim \mathbb{F}_p^{n \times n}$ based on the standard self-correction technique of [BLR93] for bilinear functions.

Next we reduce multiplication for matrices uniformly drawn from $\mathbb{F}_p^{n \times n}$ to multiplication for matrices uniformly drawn from $\{0, 1\}^{n \times n}$ using the binary expansion technique.

Lemma 7.2. *There exists a randomized $O((\log n)^2 \cdot \log \log n)$ -query oracle algorithm $M^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\substack{A, B \sim \{0, 1\}^{n \times n} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(A, B), AB)] \geq 1 - \frac{1}{(\log \log n)^3},$$

for any input $(A, B) \in (\{0, 1\}^{n \times n})^2$, the algorithm $M^\mathcal{O}$ satisfies

$$\Pr_{M^\mathcal{O}} [M^\mathcal{O}(A, B) = AB] \geq 1 - \frac{1}{n}.$$

In several places (e.g., Lemma 7.5) we will assume that an oracle computes each fixed entry of the matrix product correctly with high probability. However, the guarantees we obtain are typically stated only in terms of *expected agreement* with the true product. Hirahara and Shimizu [HS25b, Lemma 3.2] (and implicitly in Gola, Shinkar, and Singh [GSS24]) proved that these two guarantees are essentially equivalent.

Lemma 7.3 ([HS25b, Lemma 3.2]). *There exists an $O((n+m)^2)$ -time one-query randomized oracle algorithm $N^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ satisfying*

$$\mathbb{E}_{\substack{A \sim \{0, 1\}^{n \times m} \\ B \sim \{0, 1\}^{m \times n} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(A, B), AB)] \geq \alpha,$$

the following holds: for every fixed $(i, j) \in [n]^2$,

$$\Pr_{\substack{A \sim \{0, 1\}^{n \times m} \\ B \sim \{0, 1\}^{m \times n} \\ N^\mathcal{O}}} [N^\mathcal{O}(A, B)_{i,j} = (AB)_{i,j}] \geq \alpha.$$

Proof sketch. The algorithm $N^\mathcal{O}$ simply “symmetrizes” the oracle by randomly permuting the rows of A and the columns of B , and then undoing these permutations on the output. Specifically, $N^\mathcal{O}$ chooses uniformly random permutation matrices P and Q and outputs $P^{-1} \cdot \mathcal{O}(PA, BQ) \cdot Q^{-1}$ on input (A, B) . $\square$

In the following, we show how to represent a uniformly random element of $\mathbb{Z}_q$ as a collection of independent unbiased binary variables. This allows us to reduce matrix multiplication over $\mathbb{Z}_q$ to multiple binary instances.

Lemma 7.4. *Let q be an odd integer and let $t = \lceil \log_2 q \rceil + 1$, and define $N := 2^t$. Let*

$$T_q := \{0, \dots, q-1\} \cup \{N-q, \dots, N-1\}.$$

Let Z be uniformly distributed over T_q and denote its binary expansion as

$$Z = \sum_{a=0}^{t-1} 2^a Z^{(a)} \quad \text{with} \quad Z^{(a)} \in \{0, 1\}.$$

Then for every $a \in \{0, \dots, t-1\}$, the bit $Z^{(a)}$ is uniformly distributed over $\{0, 1\}$.

Proof. Since q is odd and $t = \lceil \log_2 q \rceil + 1$, we have $N = 2^t > 2q$, and thus the two intervals in the definition of T_q are disjoint.

Define $\phi(z) := N - 1 - z$. Since T_q is the disjoint union of the lowest q integers and the highest q integers in $\{0, \dots, N-1\}$, the map ϕ preserves T_q and swaps these two intervals. Thus, if Z is uniform over T_q , then so is $\phi(Z)$.

Moreover, in t -bit binary representation, $\phi(Z)$ is obtained by flipping every bit of Z . Therefore, for every fixed a , the a -th bit of Z takes values 0 and 1 with equal probability. $\square$

Lemma 7.4 allows us to reduce matrix multiplication over $\mathbb{Z}_q$ to multiple random binary instances as follows.

Lemma 7.5. *Let $q \geq 3$ be an odd integer and let $t := \lceil \log_2 q \rceil + 1$. There exists a randomized oracle algorithm $N^\mathcal{O}$ such that, for any randomized oracle $\mathcal{O}$ such that, for every fixed $(i, j) \in [n]^2$,*

$$\Pr_{A, B \sim \{0, 1\}^{n \times n}} [\mathcal{O}(A, B)_{i,j} = (AB)_{i,j}] \geq \alpha,$$

the algorithm $N^\mathcal{O}$ is given $X, Y \sim \mathbb{Z}_q^{n \times n}$ as input, makes exactly t^2 oracle calls to $\mathcal{O}$, and satisfies that, for every fixed $(i, j) \in [n]^2$,

$$\Pr_{\substack{X, Y \sim \mathbb{Z}_q^{n \times n} \\ N^\mathcal{O}}} [N^\mathcal{O}(X, Y)_{i,j} = (XY \bmod q)_{i,j}] \geq 1 - t^2(1 - \alpha).$$

Moreover, if one oracle call to $\mathcal{O}$ takes time $T(n)$, then $N^\mathcal{O}$ runs in time $O(t^2 T(n) + t^2 n^2)$.

Proof. Let $N := 2^t$ and $T_q := \{0, \dots, q-1\} \cup \{N-q, \dots, N-1\}$. For each $x \in \mathbb{Z}_q$, let $T_{q,x} := \{y \in T_q : y \equiv x \pmod{q}\}$.

Given $X, Y \in \mathbb{Z}_q^{n \times n}$, the algorithm $N^\mathcal{O}$ proceeds as follows. For each entry $X_{u,v}$, sample $\hat{X}_{u,v} \sim T_{q, X_{u,v}}$ uniformly and write

$$\hat{X}_{u,v} = \sum_{a=0}^{t-1} 2^a X_{u,v}^{(a)}$$

with $X_{u,v}^{(a)} \in \{0, 1\}$. This defines binary matrices $X^{(0)}, \dots, X^{(t-1)} \in \{0, 1\}^{n \times n}$. Define $Y^{(0)}, \dots, Y^{(t-1)}$ analogously.

For each $(a, b) \in \{0, \dots, t-1\}^2$, query

$$C^{(a,b)} := \mathcal{O}(X^{(a)}, Y^{(b)}).$$

The output is

$$N^{\mathcal{O}}(X, Y) = \left(\sum_{a,b} 2^{a+b} C^{(a,b)} \right) \bmod q.$$

By construction, $X \equiv \sum_a 2^a X^{(a)} \pmod{q}$ and $Y \equiv \sum_b 2^b Y^{(b)} \pmod{q}$. By Lemma 7.4, for each a , the matrix $X^{(a)}$ is uniformly distributed over $\{0, 1\}^{n \times n}$, and similarly for $Y^{(b)}$. Moreover, the randomness is independent, so each pair $(X^{(a)}, Y^{(b)})$ is distributed as independent uniform binary matrices.

Fix (i, j) . If $C_{i,j}^{(a,b)} = (X^{(a)} Y^{(b)})_{i,j}$ holds for all a, b , then

$$N^{\mathcal{O}}(X, Y)_{i,j} = \sum_{a,b} 2^{a+b} (X^{(a)} Y^{(b)})_{i,j} \equiv (XY)_{i,j} \pmod{q}.$$

Thus an error occurs only if some oracle call is incorrect at (i, j) . By a union bound,

$$\Pr[N^{\mathcal{O}}(X, Y)_{i,j} \neq (XY)_{i,j}] \leq t^2(1 - \alpha).$$

The algorithm makes t^2 oracle calls and runs in time $O(t^2 T(n) + t^2 n^2)$. $\square$

The previous lemma allows us to simulate matrix multiplication over $\mathbb{Z}_q$ on *uniformly random* inputs using the binary-input oracle. In the next step we remove the randomness requirement on the inputs.

The following lemma shows that if an oracle computes the product correctly with sufficiently high probability on uniformly random matrices over $\mathbb{Z}_q$, then it can be converted into an algorithm that computes the product for *arbitrary* inputs over $\mathbb{Z}_q$. The key tool is the standard *self-correction* technique of Blum, Luby, and Rubinfeld [BLR93] for bilinear functions: by adding random matrices to the inputs and combining several oracle evaluations, we recover the correct value of the product with high probability.

Lemma 7.6 (Self-correction over $\mathbb{Z}_q$). *Let $q = q(n) \geq 2$ be an integer with $q \leq \text{poly}(n)$ and let $\delta = \delta(n) > 0$ be any parameter. There exists an $\tilde{O}(n^2 \log(1/\delta))$ -time randomized oracle algorithm $S^{\mathcal{O}}$ such that, for any randomized oracle $\mathcal{O}$ satisfying that, for every fixed $(i, j) \in [n]^2$,*

$$\Pr_{\substack{X, Y \sim \mathbb{Z}_q^{n \times n} \\ \mathcal{O}}}[\mathcal{O}(X, Y)_{i,j} = (XY)_{i,j} \bmod q] \geq \frac{15}{16},$$

the following holds: for every fixed $A, B \in \mathbb{Z}_q^{n \times n}$ and every index $i, j \in [n]$,

$$\Pr_{S^{\mathcal{O}}} [S^{\mathcal{O}}(A, B)_{i,j} = (AB)_{i,j} \bmod q] \geq 1 - \delta.$$

Moreover, $S^{\mathcal{O}}$ makes $O(\log(1/\delta))$ oracle calls.

Proof. The algorithm $S^{\mathcal{O}}$ given $A, B \in \mathbb{Z}_q^{n \times n}$ runs as follows.

1. Let $r := C \log(1/\delta)$, where $C > 0$ is a sufficiently large absolute constant.
2. For each $s \in [r]$ do the following:

- (a) Sample $R^{(s)}, S^{(s)} \sim \mathbb{Z}_q^{n \times n}$ independently and uniformly.
- (b) Use the oracle to compute

$$Z^{(s)} := \mathcal{O}(A+R^{(s)}, B+S^{(s)}) - \mathcal{O}(A+R^{(s)}, S^{(s)}) - \mathcal{O}(R^{(s)}, B+S^{(s)}) + \mathcal{O}(R^{(s)}, S^{(s)}) \pmod{q}.$$

- 3. For every entry $(i, j) \in [n]^2$, output the majority value among $Z_{i,j}^{(1)}, \dots, Z_{i,j}^{(r)}$. Denote the resulting matrix by $S^{\mathcal{O}}(A, B)$.

Since $R^{(s)}, S^{(s)}$ are uniform over $\mathbb{Z}_q^{n \times n}$, each of the four pairs

$$(A + R^{(s)}, B + S^{(s)}), \quad (A + R^{(s)}, S^{(s)}), \quad (R^{(s)}, B + S^{(s)}), \quad (R^{(s)}, S^{(s)})$$

is distributed as two independent uniform matrices in $\mathbb{Z}_q^{n \times n}$. Hence, for every fixed (i, j) , each oracle call is correct at entry (i, j) with probability at least $\frac{15}{16}$.

If all four oracle calls in iteration s are correct at entry (i, j) , then

$$\begin{aligned} Z_{i,j}^{(s)} &= ((A + R^{(s)})(B + S^{(s)}))_{i,j} - ((A + R^{(s)})S^{(s)})_{i,j} \\ &\quad - (R^{(s)}(B + S^{(s)}))_{i,j} + (R^{(s)}S^{(s)})_{i,j} \pmod{q} \\ &= (AB)_{i,j} \pmod{q}, \end{aligned}$$

by bilinearity of matrix multiplication over $\mathbb{Z}_q$.

Therefore, for every (i, j) ,

$$\Pr[Z_{i,j}^{(s)} \neq (AB)_{i,j} \pmod{q}] \leq 4 \left(1 - \frac{15}{16}\right) \leq \frac{1}{4}$$

and thus each trial outputs the correct value at (i, j) with probability at least $3/4$.

For $s \in [r]$, let

$$W_s := \mathbf{1}[Z_{i,j}^{(s)} = (AB)_{i,j} \pmod{q}].$$

Then the variables $W_1, \dots, W_r$ are independent and satisfy $\Pr[W_s = 1] \geq 3/4$. By the Chernoff bound, since $r = C \log(1/\delta)$ for a sufficiently large constant C , we have

$$\Pr\left[\sum_{s=1}^r W_s \leq \frac{r}{2}\right] \leq \delta.$$

Thus, for every entry $i, j \in [n]$, it holds that $\Pr[S^{\mathcal{O}}(A, B)_{i,j} \neq (AB)_{i,j} \pmod{q}] \leq \delta$.

Finally, the number of oracle calls is $4r = O(\log(1/\delta))$, and the running time bound follows immediately. $\square$

Proof of Lemma 7.2. Let $\mathcal{S}_n$ be a set of odd primes such that

$$|\mathcal{S}_n| = O\left(\frac{\log n}{\log \log n}\right), \quad \prod_{p \in \mathcal{S}_n} p > n, \quad \text{and} \quad \max_{p \in \mathcal{S}_n} p = O(\log n).$$

Such a set exists for all sufficiently large n and can be efficiently constructed in time $\tilde{O}(1)$. For each $p \in \mathcal{S}_n$, let

$$t_p := \lceil \log_2 p \rceil + 1.$$

Since $p = O(\log n)$ for every $p \in \mathcal{S}_n$, we have $t_p = O(\log \log n)$.

Fix any $p \in \mathcal{S}_n$. Apply Lemma 7.5 with $q = p$ and $\alpha := 1 - \frac{1}{16t_p^2}$. Note that the oracle $\mathcal{O}$ satisfies $\mathbb{E}_{A,B}[\text{agree}(\mathcal{O}(A, B), AB)] \geq 1 - 1/(\log \log n)^3 \geq \alpha$ for every sufficiently large n . Moreover, by Lemma 7.3, we can assume without loss of generality that $\mathcal{O}$ satisfies the same guarantee for every fixed $(i, j) \in [n]^2$.

Then, we obtain a randomized $O(t_p^2)$ -query oracle algorithm $N_p^\mathcal{O}$ such that, for every fixed $(i, j) \in [n]^2$,

$$\Pr_{\substack{X, Y \sim \mathbb{Z}_p^{n \times n} \\ N_p^\mathcal{O}}} [N_p^\mathcal{O}(X, Y)_{i,j} = (XY \bmod p)_{i,j}] \geq \frac{15}{16}.$$

Now apply Lemma 7.6 to the oracle $N_p^\mathcal{O}$ with $\delta := 1/3$. Then we obtain a randomized oracle algorithm $S_p^\mathcal{O}$ such that, for any $A, B \in \mathbb{Z}_p^{n \times n}$ and every fixed $i, j \in [n]$,

$$\Pr_{S_p^\mathcal{O}} [S_p^\mathcal{O}(A, B)_{i,j} = (AB)_{i,j} \bmod p] \geq 2/3.$$

Note that $S_p^\mathcal{O}$ makes at most $O(t_p^2)$ queries to $\mathcal{O}$ in total.

We now define the final algorithm $M^\mathcal{O}$. On input $A, B \in \{0, 1\}^{n \times n}$, for every $p \in \mathcal{S}_n$ and $t = 1, 2, \dots, O(\log n)$, compute

$$D_t^{(p)} := S_p^\mathcal{O}(A, B) \in \mathbb{Z}_p^{n \times n}.$$

and let $C^{(p)} \in \mathbb{Z}^{n \times n}$ be the entry-wise majority vote among $D_t^{(p)}$ over $t = 1, 2, \dots, O(\log n)$. Since $(D_t^{(p)})_{i,j} = (AB)_{i,j} \bmod p$ with probability $2/3$ for every $i, j \in [n]$, the Chernoff bound yields that $(C^{(p)})_{i,j} = (AB)_{i,j} \bmod p$ with probability $1 - 1/n^4$. By a union bound over $i, j \in [n]$ and $p \in \mathcal{S}_n$, we have

$$\Pr[\forall p \in \mathcal{S}_n, C^{(p)} = AB \bmod p] \geq 1 - \frac{1}{n}.$$

If this event occurs, we know the residues $AB \bmod p$ for all $p \in \mathcal{S}_n$. Since

$$0 \leq (AB)_{i,j} \leq n \text{ and } \prod_{p \in \mathcal{S}_n} p > n,$$

the Chinese Remainder Theorem (Lemma 7.1) efficiently and uniquely determines $(AB)_{i,j}$ from these residues. Therefore, $M^\mathcal{O}$ reconstructs AB exactly.

Finally, we bound the query complexity. Note that $M^\mathcal{O}$ makes $O(\log n)$ oracle calls to $S_p^\mathcal{O}$ for each $p \in \mathcal{S}_n$ and $S_p^\mathcal{O}$ makes at most $O(t_p^2)$ oracle calls to $\mathcal{O}$. Since $t_p = O(\log p)$ and $\max_{p \in \mathcal{S}_n} p = O(\log n)$, the total number of oracle calls to $\mathcal{O}$ is at most $O(|\mathcal{S}_n| \cdot \log n \cdot (\log \log n)^2) = O((\log n)^2 \log \log n)$. $\square$

8 Proof of the Main Theorem

In this section, we prove Theorem 1.1. We do so by proving the following oracle version, which implies Theorem 1.1 by instantiating the oracle with the given algorithm and dimension adjustment (Lemma 3.7).

Lemma 8.1. *For every constant $\varepsilon > 0$, there exists a parameter $k = k(n) = \text{poly}(\log \log n) \in \mathbb{N}$ and a nonuniform randomized $O((\log n)^2 \cdot \text{polylog}(\log \log n))$ -query oracle algorithm $M^\mathcal{O}$ with the following property.*

Suppose that $\mathcal{O}$ is a randomized oracle such that

$$\mathbb{E}_{\substack{\bar{A}, \bar{B} \sim \{0,1\}^{kn \times kn} \\ \mathcal{O}}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon.$$

Then, for every input $A, B \in \{0,1\}^{n \times n}$,

$$\Pr_{M^\mathcal{O}} [M^\mathcal{O}(A, B) = AB] \geq 1 - \frac{1}{n}.$$

Proof. Fix a constant $\varepsilon > 0$. The proof consists of four reductions from previous sections.

Choice of parameters. Let $\gamma := \frac{1}{4}$. Apply Lemma 6.1 with parameters γ and $\varepsilon' := \frac{2\varepsilon}{5}$, and let $\delta_0 := \delta(\gamma, \varepsilon') > 0$ be the resulting constant. Next, let $k_1 := k'(\varepsilon, \delta_0)$ be the parameter from Lemma 5.1 corresponding to (ε, δ_0) . Set $\delta_1 := \frac{1}{(\log \log n)^3}$ and let $k_2 := k_2(\delta_1) = O((\log \log n)^{12})$ be the parameter from Lemma 6.2.

Finally define

$$k := k_1 k_2.$$

Since k_1 depends only on ε and δ_0 , and $k_2 = O((\log \log n)^{12})$, we have $k = O(k_2) = O((\log \log n)^{12})$.

Reduction. First, we apply the distance amplification lemma (Lemma 5.1). Let $m := k_2 n$. From the assumption on $\mathcal{O}$,

$$\mathbb{E}_{\bar{A}, \bar{B} \sim \{0,1\}^{k_1 m \times k_1 m}} [\text{agree}(\mathcal{O}(\bar{A}, \bar{B}), \bar{A}\bar{B})] \geq \varepsilon.$$

Applying Lemma 5.1 at dimension m , we obtain a nonuniform randomized oracle algorithm $N_1^\mathcal{O}$ such that

$$\Pr_{U, V \sim \{0,1\}^{m \times m}} [\text{agree}(N_1^\mathcal{O}(U, V), UV) \geq 1 - \delta_0] \geq \frac{2\varepsilon}{5}.$$

Second, we apply the first hardness amplification lemma (Lemma 6.1). Apply Lemma 6.1 (with parameters $\gamma = \frac{1}{4}$ and $\varepsilon' = \frac{2\varepsilon}{5}$) to $N_1^\mathcal{O}$ at dimension $m = k_2 n$. We obtain an oracle algorithm $N_2^\mathcal{O}$ such that

$$\mathbb{E}_{U, V \sim \{0,1\}^{m \times m}} [\text{agree}(N_2^\mathcal{O}(U, V), UV)] \geq \frac{3}{4}.$$

Third, we apply the second hardness amplification lemma (Lemma 6.2) to obtain an oracle with subconstant error probability. Apply Lemma 6.2 with parameter δ_1 to $N_2^\mathcal{O}$. Since $m = k_2 n$, we obtain an oracle algorithm $N_3^\mathcal{O}$ such that

$$\mathbb{E}_{A, B \sim \{0,1\}^{n \times n}} [\text{agree}(N_3^\mathcal{O}(A, B), AB)] \geq 1 - \delta_1.$$

Finally, we apply the self-correction lemma (Lemma 7.2) to obtain a worst-case exact algorithm. Apply Lemma 7.2 to $N_3^\mathcal{O}$ as oracle. This yields a randomized oracle algorithm $M^\mathcal{O}$ such that, for every fixed $A, B \in \{0,1\}^{n \times n}$,

$$\Pr_{M^\mathcal{O}} [M^\mathcal{O}(A, B) = AB] \geq 1 - \frac{1}{n}.$$

Running time. The algorithm $N_1^\mathcal{O}$ makes $O(1)$ calls to $\mathcal{O}$, $N_2^\mathcal{O}$ makes $O(1)$ calls to $N_1^\mathcal{O}$, and $N_3^\mathcal{O}$ makes $O(\log \log \log n)$ calls to $N_2^\mathcal{O}$. Finally, $M^\mathcal{O}$ makes $O((\log n)^2 \cdot \log \log n)$ calls to $N_3^\mathcal{O}$. Thus the total number of calls to $\mathcal{O}$ is

$$O((\log n)^2 \cdot (\log \log n) \cdot (\log \log \log n)),$$

and each call is on matrices of dimension kn .

All additional work is near-quadratic and absorbed in $\tilde{O}(n^2)$; thus the running time of $M^\mathcal{O}$ is $\tilde{O}(n^2)$. $\square$

Finally, we prove Theorem 1.1. Note that Lemma 8.1 reduces exact multiplication of any two $n \times n$ matrices to approximate multiplication of random two $kn \times kn$ matrices. Thus, it suffices to multiply two $kn \times kn$ matrices using an exact multiplication algorithm on dimension n .

Proof of Theorem 1.1. Let $k = \text{poly}(\log \log(n)) \in \mathbb{N}$ be the parameter from Lemma 8.1. Let $A, B \in \{0, 1\}^{n \times n}$ be the given matrices and suppose $\mathcal{O}$ approximately multiplies two random binary $n \times n$ matrices. By Lemma 3.7, we may assume k divides n . Indeed, if k does not divide n , then by Lemma 3.7, we can construct another oracle $\mathcal{O}'$ that approximately multiplies two random $(k \lceil n/k \rceil) \times (k \lceil n/k \rceil)$ matrices with increasing the running time by a factor of at most $\text{poly}(k) = \text{poly}(\log \log n)$, which can be then converted into an algorithm $N^{\mathcal{O}'}$ that multiplies two arbitrary $(k \lceil n/k \rceil) \times (k \lceil n/k \rceil)$ matrices. To compute AB for given $A, B \in \mathbb{Z}^{n \times n}$, let $A', B' \in \{0, 1\}^{(k \lceil n/k \rceil) \times (k \lceil n/k \rceil)}$ be matrices obtained by padding zeros to A and B respectively, and then run $N^{\mathcal{O}'}$ to compute $A'B'$. Therefore, we may assume k divides n .

For $k, n \in \mathbb{N}$ such that k divides n , define

$$I_1 = \{1, \dots, n/k\}, \quad I_2 = \{n/k + 1, \dots, 2n/k\}, \dots, I_k = \{(k-1)n/k + 1, \dots, n\}.$$

Given matrices $A, B \in \{0, 1\}^{n \times n}$, we divide them into k^2 blocks as

$$A = \begin{bmatrix} A_{11} & \dots & A_{1k} \\ \vdots & \ddots & \vdots \\ A_{k1} & \dots & A_{kk} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & \dots & B_{1k} \\ \vdots & \ddots & \vdots \\ B_{k1} & \dots & B_{kk} \end{bmatrix},$$

where $A_{i,j} := A|_{I_i, I_j} \in \{0, 1\}^{(n/k) \times (n/k)}$ and $B_{i,j} := B|_{I_i, I_j} \in \{0, 1\}^{(n/k) \times (n/k)}$.

Since the given oracle $\mathcal{O}$ approximately multiplies two $n \times n$ matrices, by Lemma 8.1 on input dimension n/k , there exists an $O((\log n)^2 \cdot \text{poly}(\log \log n))$ -query nonuniform randomized oracle algorithm $P^\mathcal{O}$ that, for every pair of matrices $A', B' \in \{0, 1\}^{(n/k) \times (n/k)}$, it holds that

$$\Pr_{P^\mathcal{O}}[P^\mathcal{O}(A', B') = A'B'] \geq 1 - \frac{1}{n}.$$

Our final oracle algorithm $M^\mathcal{O}(A, B)$ outputs the matrix $C \in \mathbb{Z}^{n \times n}$ defined by

$$C_{I_i, I_j} := \sum_{\ell \in [k]} P^\mathcal{O}(A_{i\ell}, B_{\ell j})$$

for all $i, j \in [k]$.

Since $P^{\mathcal{O}}$ runs in time $\tilde{O}(n^2)$ and $k = \text{poly}(\log \log n)$, the running time of $M^{\mathcal{O}}$ is $\tilde{O}(n^2)$. Moreover, $M^{\mathcal{O}}$ makes k^3 oracle calls to $P^{\mathcal{O}}$ and $P^{\mathcal{O}}$ makes at most $O((\log n)^2 \cdot \text{poly}(\log \log n))$ queries to $\mathcal{O}$, the total number of oracle calls to $\mathcal{O}$ made by $M^{\mathcal{O}}$ is

$$O(k^3(\log n)^2 \cdot \text{poly}(\log \log n)) = O((\log n)^2 \cdot \text{poly}(\log \log n)).$$

Since $P^{\mathcal{O}}(A_{i\ell}, B_{\ell j}) = A_{i\ell}B_{\ell j}$ with probability $1 - 1/n$ for every $i, j, \ell \in [k]$, a union bound over $i, j, \ell \in [k]$ yields

$$\Pr[\forall i, j, \ell \in [k], \quad P^{\mathcal{O}}(A_{i\ell}, B_{\ell j}) = A_{i\ell}B_{\ell j}] \geq 1 - \frac{k^3}{n} \geq \frac{2}{3},$$

which immediately implies $M^{\mathcal{O}}(A, B)$ computes AB with probability $2/3$. $\square$

Acknowledgements

We thank anonymous reviewers for helpful comments. Shuichi Hirahara is supported by JSPS KAKENHI Grant Number 24K21317 and JST FOREST Program Grant Number JPMJFR226Y, Japan. Nobutaka Shimizu is supported by JSPS KAKENHI Grant Number 24K21317, 23K16837 and JST PRESTO Grant Number JPMJPR26K4, Japan.

References

- [ABPSS24] Prashanth Amireddy, Amik Raj Behera, Manaswi Paraashar, Srikanth Srinivasan, and Madhu Sudan. “Local Correction of Linear Functions over the Boolean Cube”. In: *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*. 2024, pp. 764–775. DOI: [10.1145/3618260.3649746](#) (cit. on p. 4).
- [ABPSS25] Prashanth Amireddy, Amik Raj Behera, Manaswi Paraashar, Srikanth Srinivasan, and Madhu Sudan. “Low Degree Local Correction Over the Boolean Cube”. In: *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2025, pp. 5504–5511. DOI: [10.1137/1.9781611978322.187](#) (cit. on p. 4).
- [ADWXXZ25] Josh Alman, Ran Duan, Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. “More asymmetry yields faster matrix multiplication”. en. In: *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2025, pp. 2005–2039. DOI: [10.1137/1.9781611978322.63](#) (cit. on p. 2).
- [AFKLM24] Amir Abboud, Nick Fischer, Zander Kelley, Shachar Lovett, and Raghu Meka. “New Graph Decompositions and Combinatorial Boolean Matrix Multiplication Algorithms”. In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2024, pp. 935–943. DOI: [10.1145/3618260.3649696](#) (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 the Symposium on Theory of Computing (STOC)*. 2022, pp. 1566–1574. DOI: [10.1145/3519935.3520041](#) (cit. on p. 4).

- [BBB21] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. “The Average-Case Complexity of Counting Cliques in Erdős–Rényi Hypergraphs”. In: *SIAM Journal on Computing* (2021), FOCS19–39–FOCS19–80. DOI: [10.1137/20M1316044](#) (cit. on pp. 4, 10).
- [BHKS21] Siddharth Bhandari, Prahladh Harsha, Mrinal Kumar, and Madhu Sudan. “Decoding Multivariate Multiplicity Codes on Product Sets”. In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. 2021, pp. 1489–1501. DOI: [10.1145/3406325.3451027](#) (cit. on p. 4).
- [BLR93] Manuel Blum, Michael Luby, and Ronitt Rubinfeld. “Self-Testing/Correcting with Applications to Numerical Problems”. In: *J. Comput. Syst. Sci.* 47.3 (1993), pp. 549–595. DOI: [10.1016/0022-0000\(93\)90044-W](#) (cit. on pp. 11, 36, 38).
- [Bor15] Charles Bordenave. “A new proof of Friedman’s second eigenvalue Theorem and its extension to random lifts”. In: *arXiv [math.CO]* (2015). eprint: [1502.04482](#) (math.CO) (cit. on p. 12).
- [BW12] Nikhil Bansal and Ryan Williams. “Regularity Lemmas and Combinatorial Algorithms”. In: *Theory Comput.* 8.1 (2012), pp. 69–94. DOI: [10.4086/TOC.2012.V008A004](#) (cit. on p. 2).
- [CL99] Edith Cohen and David D. Lewis. “Approximating Matrix Multiplication for Pattern Recognition Tasks”. In: *Journal of Algorithms* 30.2 (1999), pp. 211–252. DOI: [10.1006/jagm.1998.0989](#) (cit. on pp. 2, 4).
- [CNW16] Michael B. Cohen, Jelani Nelson, and David P. Woodruff. “Optimal Approximate Matrix Product in Terms of Stable Rank”. In: *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016)*. Leibniz International Proceedings in Informatics (LIPIcs). 2016, 11:1–11:14. DOI: [10.4230/LIPIcs.ICALP.2016.11](#) (cit. on pp. 2, 4).
- [CW13] Kenneth L. Clarkson and David P. Woodruff. “Low Rank Approximation and Regression in Input Sparsity Time”. In: *Proceedings of the 45th Annual ACM Symposium on Theory of Computing*. 2013, pp. 81–90. DOI: [10.1145/2488608.2488620](#) (cit. on pp. 2, 4).
- [DKM06] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. “Fast Monte Carlo Algorithms for Matrices I: Approximating Matrix Multiplication”. In: *SIAM J. Comput.* 36.1 (2006), pp. 132–157. DOI: [10.1137/S0097539704442684](#) (cit. on pp. 2, 4).
- [DWZ23] Ran Duan, Hongxun Wu, and Renfei Zhou. “Faster Matrix Multiplication via Asymmetric Hashing”. In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2023, pp. 2129–2138. DOI: [10.1109/FOCS57990.2023.00130](#) (cit. on p. 2).
- [Erd45] Paus Erdős. “On a lemma of Littlewood and Offord”. en. In: *Bulletin of the American Mathematical Society* 51 (12 1945), pp. 898–902 (cit. on p. 23).
- [Fri03] Joel Friedman. “A proof of alon’s second eigenvalue conjecture”. In: *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing* (San Diego, CA, USA). STOC ’03. 2003, pp. 720–724. DOI: [10.1145/780542.780646](#) (cit. on p. 12).

- [Gil98] David Gillman. “A Chernoff Bound for Random Walks on Expander Graphs”. In: *SIAM Journal on Computing* 27 (4 1998), pp. 1203–1220. DOI: [10.1137/S0097539794268765](#) (cit. on p. 24).
- [GSS24] Ashish Gola, Igor Shinkar, and Harsimran Singh. “Matrix Multiplication Reductions”. In: *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*. 2024, 34:1–34:15. DOI: [10.4230/LIPICS.APPROX/RANDOM.2024.34](#) (cit. on pp. 2, 4, 10, 36).
- [HJ91] Roger A Horn and Charles R Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991. DOI: [10.1017/CB09780511840371](#) (cit. on p. 12).
- [HS23] Shuichi Hirahara and Nobutaka Shimizu. “Hardness Self-Amplification: Simplified, Optimized, and Unified”. In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2023, pp. 70–83. DOI: [10.1145/3564246.3585189](#) (cit. on pp. 4, 17).
- [HS24] Shuichi Hirahara and Nobutaka Shimizu. “Planted Clique Conjectures Are Equivalent”. In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2024, pp. 358–366. DOI: [10.1145/3618260.3649751](#) (cit. on p. 17).
- [HS25a] Shuichi Hirahara and Nobutaka Shimizu. *An optimal error-correcting reduction for matrix multiplication*. en. 2025. DOI: [10.4230/LIPICS.ICALP.2025.97](#) (cit. on pp. 2, 4, 5, 7, 17, 22, 25, 26, 33, 47, 48).
- [HS25b] Shuichi Hirahara and Nobutaka Shimizu. “Error-correction of matrix multiplication algorithms”. en. In: *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*. STOC ’25: 57th Annual ACM Symposium on Theory of Computing (Prague Czechia). 2025, pp. 785–794. DOI: [10.1145/3717823.3718244](#) (cit. on pp. 2, 4, 5, 8, 10, 14, 18, 36).
- [IJK09] Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. “Chernoff-Type Direct Product Theorems”. In: *J. Cryptol.* 22.1 (2009), pp. 75–92. DOI: [10.1007/s00145-008-9029-7](#) (cit. on p. 17).
- [Jer23] Fernando Granha Jeronimo. *Fast decoding of explicit almost optimal ε -balanced q -Ary codes and fast approximation of expanding k -CSPs*. en. 2023. DOI: [10.4230/LIPICS.APPROX/RANDOM.2023.60](#) (cit. on pp. 5, 6, 9, 13, 50).
- [JST21] Fernando Granha Jeronimo, Shashank Srivastava, and Madhur Tulsiani. “Near-linear time decoding of Ta-Shma’s codes via splittable regularity”. In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing* (Virtual, Italy). STOC 2021. 2021, pp. 1527–1536. DOI: [10.1145/3406325.3451126](#) (cit. on pp. 5, 13, 50).
- [LPS88] A Lubotzky, R Phillips, and P Sarnak. “Ramanujan graphs”. In: *Combinatorica. An International Journal on Combinatorics and the Theory of Computing* 8 (3 1988), pp. 261–277. DOI: [10.1007/BF02126799](#) (cit. on p. 12).
- [MWW04] Brendan D. McKay, Nicholas C. Wormald, and Beata Wysocka. “Short cycles in random regular graphs”. en. In: *Electronic journal of combinatorics* 11 (1 2004), R66. DOI: [10.37236/1819](#) (cit. on p. 12).

- [NRW23] Parker Newton, Silas Richelson, and Chase Wilson. “A High Dimensional Goldreich-Levin Theorem”. In: *Proceedings of the Symposium on Theory of Computing (STOC)*. 2023, pp. 1463–1474. DOI: [10.1145/3564246.3585224](#) (cit. on p. 9).
- [Pag13] Rasmus Pagh. “Compressed Matrix Multiplication”. In: *ACM Transactions on Computation Theory* 5.3 (2013), 9:1–9:17. DOI: [10.1145/2493252.2493254](#) (cit. on pp. 2, 4).
- [Sar06] Tamás Sarlós. “Improved Approximation Algorithms for Large Matrices via Random Projections”. In: *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*. 2006, pp. 143–152. DOI: [10.1109/FOCS.2006.37](#) (cit. on pp. 2, 4).
- [Sho12] Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, 2012. DOI: [10.1017/cbo9781139165464](#) (cit. on p. 36).
- [SS18] Srikanth Srinivasan and Madhu Sudan. “Local Decoding and Testing of Polynomials over Grids”. In: *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*. Ed. by Anna R. Karlin. Leibniz International Proceedings in Informatics (LIPIcs). 2018, 26:1–26:14. DOI: [10.4230/LIPIcs.ITCS.2018.26](#) (cit. on p. 4).
- [SS25] Igor Shinkar and Harsimran Singh. *A simplified reduction for error correcting matrix multiplication algorithms*. 2025. DOI: [10.4230/LIPICS.APPROX/RANDOM.2025.29](#) (cit. on pp. 2, 4).
- [Str69] Volker Strassen. “Gaussian elimination is not optimal”. In: *Numerische mathematik* 13.4 (1969), pp. 354–356 (cit. on p. 2).
- [Ta17] Amnon Ta-Shma. “Explicit, almost optimal, epsilon-balanced codes”. In: *Proceedings of Symposium on Theory of Computing (STOC)* (Montreal, Canada). STOC 2017. 2017, pp. 238–251. DOI: [10.1145/3055399.3055408](#) (cit. on pp. 5, 22).
- [TV09] Terence Tao and Vu Van H. *Cambridge studies in advanced mathematics: Additive combinatorics series number 105*. en. Cambridge studies in advanced mathematics. Cambridge, England: Cambridge University Press, 2009. 532 pp. DOI: [10.1017/cbo9780511755149](#) (cit. on p. 23).
- [UW25] Yahel Uffenheimer and Omri Weinstein. *(Approximate) Matrix Multiplication via Convolutions*. 2025. DOI: [10.48550/arXiv.2510.22193](#). arXiv: [2510.22193](#) (cit. on pp. 2, 4).
- [Val24] Gregory Valiant. “Matrix Multiplication in Quadratic Time and Energy? Towards a Fine-Grained Energy-Centric Church-Turing Thesis”. In: *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*. 2024, 96:1–96:13. DOI: [10.4230/LIPICS.ITCS.2024.96](#) (cit. on p. 3).
- [VXXZ24] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. “New Bounds for Matrix Multiplication: from Alpha to Omega”. In: *Proceedings of the Symposium on Discrete Algorithms (SODA)*. 2024, pp. 3792–3835. DOI: [10.1137/1.9781611977912.134](#) (cit. on p. 2).

- [VZ26] Vinod Vaikuntanathan and Or Zamir. “Improving algorithmic efficiency using cryptography: Trapdoored matrices and applications”. en. In: *Proceedings of the 2026 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2026, pp. 2554–2574. DOI: [10.1137/1.9781611978971.92](https://doi.org/10.1137/1.9781611978971.92) (cit. on pp. 2, 4).
- [Wor99] Nicholas C. Wormald. “Models of Random Regular Graphs”. In: *Surveys in Combinatorics, 1999* (1999), pp. 239–298. DOI: [10.1017/CB09780511721335.010](https://doi.org/10.1017/CB09780511721335.010) (cit. on p. 12).

A Simulating Encoding by Matrix Multiplication

In this section, we prove Lemma 5.7.

The proof follows the same high-level structure as Hirahara and Shimizu [HS25a, Lemma 15], with additional care to handle signed encodings and $\{0, 1\}$ -valued inputs by applying affine transformations.

Proof of Lemma 5.7. For a sign vector $\mathbf{b} = (b_1, \dots, b_k) \in \{\pm 1\}^k$ and a matrix $X \in \mathbb{Z}^{n \times n}$, define $\text{Lift}_W^{\mathbf{b}}(X) \in \mathbb{Z}^{W \times [kn]}$ as follows: for each $\mathbf{w} = (w_1, \dots, w_k) \in W$, the $\mathbf{w}$ -th row of $\text{Lift}_W^{\mathbf{b}}(X)$ is

$$(b_1 X_{w_1, *}, b_2 X_{w_2, *}, \dots, b_k X_{w_k, *}) \in \mathbb{Z}^{kn}.$$

We also define $\text{Lift}_W^{\pm}(X) \in \mathbb{Z}^{W^{\pm} \times [kn]}$ by indexing rows by $(\mathbf{b}, \mathbf{w}) \in W^{\pm}$ and setting

$$\text{Lift}_W^{\pm}(X)_{(\mathbf{b}, \mathbf{w}), *} := \text{Lift}_W^{\mathbf{b}}(X)_{\mathbf{w}, *}.$$

Similarly, define the unsigned lift $\text{Lift}_W(X) \in \mathbb{Z}^{W \times [kn]}$ by

$$\text{Lift}_W(X) := \text{Lift}_W^{(1, \dots, 1)}(X).$$

A direct calculation shows that, for every $X, Y \in \mathbb{Z}^{n \times n}$,

$$\text{Lift}_W^{\pm}(X) \cdot \text{Lift}_W(Y^{\top})^{\top} = \text{Enc}_{W \otimes 2}^{\pm}(XY). \quad (11)$$

Indeed, for $(\mathbf{b}, \mathbf{w}) \in W^{\pm}$ with $\mathbf{w} = (w_1, \dots, w_k)$ and $\mathbf{v} = (v_1, \dots, v_k) \in W$, the $((\mathbf{b}, \mathbf{w}), \mathbf{v})$ -entry of the left-hand side is

$$\sum_{t=1}^k b_t \langle X_{w_t, *}, Y_{*, v_t} \rangle = \sum_{t=1}^k b_t (XY)_{w_t, v_t} = \text{Enc}_{W \otimes 2}^{\pm}(XY)_{(\mathbf{b}, \mathbf{w}), \mathbf{v}}.$$

First, we transform the input matrices to random $\{\pm 1\}$ -valued matrices by a standard affine transformation. Let $J_{r,s}$ denote the all-one matrix in $\mathbb{Z}^{r \times s}$, and write $J_n := J_{n,n}$.

Given $A, B \in \{0, 1\}^{n \times n}$, define

$$U := 2A - J_n \in \{\pm 1\}^{n \times n}, \quad V := 2B - J_n \in \{\pm 1\}^{n \times n}.$$

If $A, B \sim \{0, 1\}^{n \times n}$, then U, V are independent uniform matrices over $\{\pm 1\}^{n \times n}$.

Since $2A = U + J_n$ and $2B = V + J_n$, we have

$$4AB = (U + J_n)(V + J_n) = UV + UJ_n + J_nV + nJ_n.$$

Therefore,

$$4 \text{Enc}_{W^{\otimes 2}}^{\pm}(AB) = \text{Enc}_{W^{\otimes 2}}^{\pm}(UV) + \text{Enc}_{W^{\otimes 2}}^{\pm}(UJ_n) + \text{Enc}_{W^{\otimes 2}}^{\pm}(J_nV) + n \text{Enc}_{W^{\otimes 2}}^{\pm}(J_n). \quad (12)$$

The last three terms can be computed exactly from A, B, G , so the remaining part of the proof focuses on approximating $\text{Enc}_{W^{\otimes 2}}^{\pm}(UV)$.

Second, we partition the set of k -walks W into a small number of vertex-disjoint sets. Specifically, we invoke the following lemma.

Lemma A.1 ([HS25a, Corollary 12]). *Let $k, d \in \mathbb{N}$ be constants, $G = ([n], E)$ be a d -regular graph with girth at least k , and $W \subseteq [n]^k$ be the set of k -walks on G . Let $a = k^2 d^{k-1} + 1$. Then, we can compute in time $O(|W|)$ a partition $W = P_1 \sqcup \dots \sqcup P_a \sqcup P'$ such that*

- *The walks in P_i are pairwise vertex-disjoint for each $i = 1, \dots, a$,*
- *$|P_i| \leq n/k$ for $i = 1, \dots, a$,*
- *Every walk in each P_i visits exactly k distinct vertices,*
- *$|P'| \leq \frac{k}{d} \cdot |W|$.*

Let $P_1 \sqcup \dots \sqcup P_a \sqcup P'$ be the partition of W into vertex-disjoint sets, given by Lemma A.1. For each $\mathbf{b} \in \{\pm 1\}^k$ and $i \in [a]$, let

$$Q_{\mathbf{b},i} := \{(\mathbf{b}, \mathbf{w}) \in W^{\pm} : \mathbf{w} \in P_i\} \subseteq W^{\pm}.$$

Next, we will approximate the block of $\text{Enc}_{W^{\otimes 2}}^{\pm}(UV)$ indexed by $Q_{\mathbf{b},i} \times P_j$. Fix $\mathbf{b} \in \{\pm 1\}^k$ and $i, j \in [a]$, and write

$$r := |P_i|, \quad s := |P_j|.$$

Set

$$X := \text{Lift}_W^{\mathbf{b}}(U)|_{P_i,*} \in \{\pm 1\}^{r \times kn}, \quad Y := \text{Lift}_W(V^{\top})^{\top}|_{*,P_j} \in \{\pm 1\}^{kn \times s}.$$

By construction and (11),

$$XY = \text{Enc}_{W^{\otimes 2}}^{\pm}(UV)|_{Q_{\mathbf{b},i} \times P_j}.$$

Since the walks in P_i are pairwise vertex-disjoint, the rows of X are independent and each is uniformly distributed over $\{\pm 1\}^{kn}$. Likewise, since the walks in P_j are pairwise vertex-disjoint, the columns of Y are independent and each is uniformly distributed over $\{\pm 1\}^{kn}$. Hence X and Y are independent and uniformly distributed over $\{\pm 1\}^{r \times kn} \times \{\pm 1\}^{kn \times s}$.

Next, we convert the matrices X and Y to random $\{0, 1\}$ -valued matrices. Define

$$\bar{X} := \frac{X + J_{r,kn}}{2} \in \{0, 1\}^{r \times kn}, \quad \bar{Y} := \frac{Y + J_{kn,s}}{2} \in \{0, 1\}^{kn \times s}.$$

Then $\bar{X}, \bar{Y}$ are independent uniform binary matrices, and

$$XY = 4\bar{X}\bar{Y} - 2\bar{X}J_{kn,s} - 2J_{r,kn}\bar{Y} + kn \cdot J_{r,s}. \quad (13)$$

Thus, if $\tilde{Z} \in \mathbb{Z}^{r \times s}$ approximates $\bar{X}\bar{Y}$, then

$$\tilde{T} := 4\tilde{Z} - 2\bar{X}J_{kn,s} - 2J_{r,kn}\bar{Y} + kn \cdot J_{r,s}$$

approximates XY with the same agreement, i.e.,

$$\text{agree}(\tilde{T}, XY) = \text{agree}(\tilde{Z}, \tilde{X}\tilde{Y}). \quad (14)$$

Since $\mathcal{O}$ satisfies the hypothesis of Lemma 3.5 with $N = kn$ and $\alpha = \varepsilon$, we may apply that lemma to $\tilde{X} \in \{0, 1\}^{r \times kn}$ and $\tilde{Y} \in \{0, 1\}^{kn \times s}$, where $r, s \leq kn$. Thus, there exists a one-query oracle algorithm that outputs a matrix $\tilde{Z} \in \mathbb{Z}^{r \times s}$ such that

$$\mathbb{E}[\text{agree}(\tilde{Z}, \tilde{X}\tilde{Y})] \geq \varepsilon.$$

Combining this with (14), we obtain a matrix $\tilde{T}^{(\mathbf{b}, i, j)} \in \mathbb{Z}^{r \times s}$ satisfying

$$\mathbb{E}_{U, V, \mathcal{O}} \left[\text{agree} \left(\tilde{T}^{(\mathbf{b}, i, j)}, \text{Enc}_{W^{\otimes 2}}^{\pm}(UV) \Big|_{Q_{\mathbf{b}, i} \times P_j} \right) \right] \geq \varepsilon. \quad (15)$$

For every $\mathbf{b} \in \{\pm 1\}^k$ and $i, j \in [a]$, run the above procedure and place the resulting matrix $\tilde{T}^{(\mathbf{b}, i, j)}$ into the block $Q_{\mathbf{b}, i} \times P_j$.

For entries whose row index lies in

$$(W^{\pm})' := \{(\mathbf{b}, \mathbf{w}) \in W^{\pm} : \mathbf{w} \in P'\}$$

or whose column index lies in P' , fill them arbitrarily. Let the resulting matrix be $\tilde{E} \in \mathbb{Z}^{W^{\pm} \times W}$.

Averaging (15) over all $\mathbf{b}, i, j$, we get

$$\mathbb{E}_{U, V, \mathcal{O}} \left[\text{agree}(\tilde{E}, \text{Enc}_{W^{\otimes 2}}^{\pm}(UV)) \right] \geq \varepsilon - \frac{|(W^{\pm})'|}{|W^{\pm}|} - \frac{|P'|}{|W|}.$$

Since $|(W^{\pm})'|/|W^{\pm}| = |P'|/|W| \leq k/d$ (by Lemma A.1), it follows that

$$\mathbb{E}_{U, V, \mathcal{O}} \left[\text{agree}(\tilde{E}, \text{Enc}_{W^{\otimes 2}}^{\pm}(UV)) \right] \geq \varepsilon - \frac{2k}{d}. \quad (16)$$

Finally, we recover $\text{Enc}_{W^{\otimes 2}}^{\pm}(AB)$ from $\tilde{E}$ by an affine transformation again. Using (12), define

$$M^{\mathcal{O}}(A, B, G) := \frac{1}{4} \left(\tilde{E} + \text{Enc}_{W^{\otimes 2}}^{\pm}(UJ_n) + \text{Enc}_{W^{\otimes 2}}^{\pm}(J_nV) + n \text{Enc}_{W^{\otimes 2}}^{\pm}(J_n) \right).$$

By (12), for each entry the above affine transformation preserves equality, and hence

$$\text{agree}(M^{\mathcal{O}}(A, B, G), \text{Enc}_{W^{\otimes 2}}^{\pm}(AB)) = \text{agree}(\tilde{E}, \text{Enc}_{W^{\otimes 2}}^{\pm}(UV)).$$

Taking expectation and applying (16), we conclude that

$$\mathbb{E}_{\substack{A, B \sim \{0, 1\}^{n \times n} \\ M^{\mathcal{O}}}} \left[\text{agree}(M^{\mathcal{O}}(A, B, G), \text{Enc}_{W^{\otimes 2}}^{\pm}(AB)) \right] \geq \varepsilon - \frac{2k}{d}.$$

□

B Near-linear Time Algorithm for MAX k -VCSP

We sketch the argument for Lemma 3.4 by extending the approximation algorithm of Jeronimo [Jer23] for MAX k -CSP to MAX k -VCSP. Since the proof is nearly identical, we only highlight the necessary modifications.

Let $\varphi = (m, W, (P_{\mathbf{w}})_{\mathbf{w} \in W})$ be an instance of MAX k -VCSP(Σ). For each $\alpha \in \Sigma^k$, define

$$g_{\alpha}(\mathbf{w}) = \begin{cases} P_{\mathbf{w}}(\alpha) & \text{if } \mathbf{w} \in W, \\ 0 & \text{otherwise.} \end{cases}$$

In Jeronimo [Jer23], g_{α} is the indicator of a subset of W since $P_{\mathbf{w}}$ is a Boolean constraint. The only place where Booleanity is used is in defining g_{α} , which we replace by a bounded function in $[0, 1]$. All subsequent arguments apply without change.

Applying the efficient weak regularity decomposition of Jeronimo, Srivastava, and Tulsiani [JST21] to each g_{α} , we obtain functions

$$h_{\alpha} = \sum_{\ell} c_{\alpha, \ell} \cdot 1_{S_1^{\alpha, \ell}} \otimes \cdots \otimes 1_{S_k^{\alpha, \ell}}$$

with the same guarantees as in Jeronimo [Jer23].

For any assignment $a \in \Sigma^m$ and $\sigma \in \Sigma$, let

$$T_{\sigma} = \{i : a_i = \sigma\}.$$

Then, we obtain

$$\text{val}(a, \varphi) = \sum_{\alpha \in \Sigma^k} \sum_{\ell} c_{\alpha, \ell} \prod_{j=1}^k \frac{|T_{\alpha_j} \cap S_j^{\alpha, \ell}|}{m} \pm \gamma,$$

exactly as in the Boolean case.

The remainder of the proof in Jeronimo [Jer23]—namely, the reduction to a constant-size optimization problem, discretization, and enumeration—applies verbatim. This yields a γ -approximation in time $\tilde{O}(m + |W|)$.

C From Binary to Bounded-Integer Matrix Multiplication

This section records a standard reduction showing that exact matrix multiplication for binary matrices implies exact matrix multiplication for matrices with polynomially bounded integer entries.

For an integer $x \in \mathbb{Z}$, define its positive and negative parts by

$$x^+ := \max\{x, 0\}, \quad x^- := \max\{-x, 0\}.$$

Then $x = x^+ - x^-$ and $x^+, x^- \in \mathbb{Z}_{\geq 0}$. For a matrix $A \in \mathbb{Z}^{n \times n}$, define $A^+, A^- \in \mathbb{Z}_{\geq 0}^{n \times n}$ entrywise by

$$(A^+)_{i,j} := (A_{i,j})^+, \quad (A^-)_{i,j} := (A_{i,j})^-.$$

Proposition C.1. *Let $M \in \mathbb{N}$, and let $\ell := \lceil \log_2(M + 1) \rceil$. There is a deterministic algorithm that, on input $A, B \in [-M, M]^{n \times n}$, makes $4\ell^2$ oracle calls to exact matrix multiplication on binary $n \times n$ matrices and reconstructs AB over $\mathbb{Z}$. The additional running time is $O(n^2 \ell^2)$.*

Proof. Since entries of A^+, A^-, B^+, B^- belong to $[0, M] \cap \mathbb{Z}$, each such entry has a binary expansion of length ℓ . Therefore, for every $s, t \in \{0, \dots, \ell - 1\}$, there exist binary matrices

$$A^{+,s}, A^{-,s}, B^{+,t}, B^{-,t} \in \{0, 1\}^{n \times n}$$

with

$$\begin{aligned} A^+ &= \sum_{s=0}^{\ell-1} 2^s A^{+,s}, & A^- &= \sum_{s=0}^{\ell-1} 2^s A^{-,s}, \\ B^+ &= \sum_{t=0}^{\ell-1} 2^t B^{+,t}, & B^- &= \sum_{t=0}^{\ell-1} 2^t B^{-,t}. \end{aligned}$$

Since $A = A^+ - A^-$ and $B = B^+ - B^-$, we have

$$AB = A^+ B^+ - A^+ B^- - A^- B^+ + A^- B^-.$$

Expanding each term in binary gives

$$AB = \sum_{s,t=0}^{\ell-1} 2^{s+t} (A^{+,s} B^{+,t} - A^{+,s} B^{-,t} - A^{-,s} B^{+,t} + A^{-,s} B^{-,t}).$$

Thus AB is a fixed integer linear combination of $4\ell^2$ products of binary matrices. Each of those products can be obtained from one oracle call, and summing them with the displayed coefficients reconstructs AB . The reconstruction cost is $O(n^2 \ell^2)$. $\square$

Corollary C.2. *If exact binary matrix multiplication on $n \times n$ matrices can be computed in time $T(n)$, then exact matrix multiplication on $n \times n$ matrices with entries in $[-M, M]$ can be computed in time*

$$O(\ell^2 \cdot (T(n) + n^2)), \quad \ell = \lceil \log_2(M + 1) \rceil.$$

In particular, when $M = \text{poly}(n)$, the overhead is polylogarithmic in n .

D From Per-entry L_p Error to Hamming Agreement

We next present a simple reduction showing that a bound on the L_p -norm error implies the Hamming-agreement hypothesis used in the main theorem. Throughout this section, the expectation is over uniformly random $A, B \in \{0, 1\}^{n \times n}$, uniformly random $(i, j) \in [n]^2$, and the internal randomness of the oracle.

Proposition D.1. *Let $p \geq 1$ and let $\varepsilon \in (0, 1]$. Suppose that $\mathcal{O}$ is a randomized oracle which, on every input $(A, B) \in \{0, 1\}^{n \times n} \times \{0, 1\}^{n \times n}$, outputs a matrix in $\mathbb{Z}^{n \times n}$ and satisfies*

$$\mathbb{E}[|\mathcal{O}(A, B)_{i,j} - (AB)_{i,j}|^p] \leq \frac{1}{\varepsilon^p}.$$

Then there exists a randomized oracle $\tilde{\mathcal{O}}$, obtained from one call to $\mathcal{O}$, such that

$$\mathbb{E}_{A, B \sim \{0, 1\}^{n \times n}} \left[\text{agree}(\tilde{\mathcal{O}}(A, B), AB) \right] \geq \frac{\varepsilon}{10}.$$

Proof. Set $T := \lceil 2/\varepsilon \rceil$. Given (A, B) , run $\mathcal{O}$ once to obtain a matrix $C := \mathcal{O}(A, B)$. Then sample a random matrix

$$R \sim \{-T, -T+1, \dots, T-1, T\}^{n \times n}$$

whose entries are independent and uniformly distributed on the integer interval $[-T, T]$. Define

$$\tilde{\mathcal{O}}(A, B) := C + R.$$

Fix an input pair (A, B) and an entry (i, j) . Let

$$D := C_{i,j} - (AB)_{i,j} \in \mathbb{Z}.$$

Conditioned on the value of D , the event

$$\tilde{\mathcal{O}}(A, B)_{i,j} = (AB)_{i,j}$$

is equivalent to the event $R_{i,j} = -D$. Hence

$$\Pr\left[\tilde{\mathcal{O}}(A, B)_{i,j} = (AB)_{i,j} \mid D\right] = \begin{cases} \frac{1}{2T+1} & \text{if } |D| \leq T, \\ 0 & \text{if } |D| > T. \end{cases}$$

Therefore

$$\Pr\left[\tilde{\mathcal{O}}(A, B)_{i,j} = (AB)_{i,j} \mid D\right] \geq \frac{\mathbf{1}_{|D| \leq T}}{2T+1}.$$

Taking expectation over all randomness gives

$$\mathbb{E}\left[\text{agree}(\tilde{\mathcal{O}}(A, B), AB)\right] \geq \frac{\Pr[|D| \leq T]}{2T+1}.$$

By Markov's inequality,

$$\Pr[|D| > T] = \Pr[|D|^p > T^p] \leq \frac{\mathbb{E}[|D|^p]}{T^p} \leq \frac{1/\varepsilon^p}{(2/\varepsilon)^p} \leq \frac{1}{2}.$$

Hence $\Pr[|D| \leq T] \geq 1/2$. Also, since $T = \lceil 2/\varepsilon \rceil$ and $\varepsilon \leq 1$, we have $2T+1 \leq 5/\varepsilon$. Combining the last two bounds yields

$$\mathbb{E}\left[\text{agree}(\tilde{\mathcal{O}}(A, B), AB)\right] \geq \frac{1/2}{2T+1} \geq \frac{\varepsilon}{10}.$$

This proves the claim. □