Matrix Multiplication Reductions

Ashish Gola
Simon Fraser University, Burnaby, Canada

Igor Shinkar
Simon Fraser University, Burnaby, Canada

Harsimran Singh
Simon Fraser University, Burnaby, Canada

Abstract

In this paper we study a worst case to average case reduction for the problem of matrix multiplication over finite fields. Suppose we have an efficient *average case* algorithm, that given two random matrices A, B outputs a matrix that has a non-trivial correlation with their product $A \cdot B$. Can we transform it into a *worst case* algorithm, that outputs the correct answer for all inputs without incurring a significant overhead in the running time? We present two results in this direction.

Two-sided error in the high agreement regime. We begin with a brief remark about a reduction for high agreement algorithms, i.e., an algorithm which agrees with the correct output on a large (say > 0.9) fraction of entries, and show that the standard self-correction of linearity allows us to transform such algorithms into algorithms that work in worst case.

One-sided error in the low agreement regime. Focusing on average case algorithms with one-sided error, we show that over $\mathbb{F}_2$ there is a reduction that gets an $O(T)$ time *average case* algorithm that given a random input A, B outputs a matrix that agrees with $A \cdot B$ on at least 51% of the entries (i.e., has only a slight advantage over the trivial algorithm), and transforms it into an $\widetilde{O}(T)$ time *worst case* algorithm, that outputs the correct answer for *all inputs* with high probability.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms

Keywords and phrases Matrix Multiplication, Reductions, Worst case to average case reductions

Digital Object Identifier 10.4230/LIPIcs.APPROX/RANDOM.2024.34

Category RANDOM

Related Version *arXiv*: `https://arxiv.org/pdf/2404.08085`

Acknowledgements We are grateful to the anonymous referees for their helpful comments. We also thank Sasha Golovnev and Tom Gur for their valuable feedback.

1 Introduction

The problem of efficiently multiplying two matrices has been extensively studied for decades. Improving on the straightforward $O(n^3)$ time algorithm, Strassen's algorithm [24] computes the product of two matrices in time $O(n^{\log_2 7} = n^{2.807})$, and it is perhaps the most widely used in practice. Since then, a long and exciting line of research ([19, 5, 21, 20, 23, 9, 22, 25, 17, 1]) has led to a significant improvement of the value of the optimal exponent of the running time for matrix multiplication problem. The fastest algorithm known today is due to Duan, Wu, and Zhou [10], and its running time is $O(n^{2.371866})$.

Worst-case to average-case reductions serve as a means to convert algorithms that output correct answers on a fraction of inputs into algorithms with correct outputs on all possible inputs. These reductions can be viewed from two different perspectives. From the hardness point of view, they can be used to show that a problem maintains its hardness even in the

average case. From the algorithmic side, they provide a framework for developing worst-case algorithms, by first designing weak algorithms with average case guarantees, and then transforming them into algorithms which work on all outputs.

In this paper, we study the following variant of a worst-case to average-case reduction for the matrix multiplication problem. Suppose we have an efficient algorithm that given two random matrices $A, B \in \mathbb{F}^{n\times n}$, computes a matrix $C \in \mathbb{F}^{n\times n}$ that agrees with the product $A \cdot B$ on a large fraction of the entries of the matrix. Can we transform such an algorithm into one that computes $A \cdot B$ correctly for all entries of the output matrix without incurring a significant overhead in the running time?

More formally, we define the *agreement* between two matrices as the fraction of entries on which the two matrices agree.

▶ **Definition 1.** *Let $\mathbb{F}$ be a field, and let $A, B \in \mathbb{F}^{n\times n}$ be two matrices. We define agreement between A and B, denoted by* $\mathrm{agr}(A, B)$*, as the fraction of entries (i, j) on which $A_{i,j} = B_{i,j}$, i.e.,*

$$\mathrm{agr}(A, B) = \frac{|\{(i, j) : A_{i,j} = B_{i,j}\}|}{n^2} \ .$$

Then, our goal can be stated as the task of transforming an algorithm that on a random input A, B outputs a matrix C such that $\mathrm{agr}(C, AB) \geq \alpha$ for some parameter $\alpha \in [0, 1]$ into an algorithm that solves the matrix multiplication problem *correctly on all inputs*.

We present two results in this direction. Both results consider the matrix multiplication problem over finite fields.

High agreement regime with two-sided error

We show that any algorithm that solves the matrix multiplication problem correctly on a high fraction of the coordinates, can be converted into a worst case algorithm. Specifically, we prove the following theorem.

▶ **Theorem 2.** *Fix a finite field $\mathbb{F}$. Let $\alpha \in (0, 1/8)$. Let* ALG *be an algorithm that gets as input two matrices $A, B \in \mathbb{F}^{n\times n}$, runs in time $T(n)$, and outputs a matrix* $\mathrm{ALG}(A, B) \in \mathbb{F}^{n\times n}$*. Suppose that*

$$\mathbb{E}_{A,B\in\mathbb{F}^{n\times n}}[\mathrm{agr}(\mathrm{ALG}(A, B), A \cdot B)] > 1 - \alpha \ .$$

Then, there is an algorithm ALG^* *that gets as input two matrices $A, B \in \mathbb{F}^{n\times n}$, runs in time $O(T(n) \cdot \log(n))$, and outputs a matrix* $\mathrm{ALG}^*(A, B) \in \mathbb{F}^{n\times n}$ *such that* for all *A, B it holds that*

$$\Pr[\mathrm{ALG}^*(A, B) = A \cdot B] > 1 - 1/n \ ,$$

where the randomness is only over the internal coins of ALG^**.*

The proof of this result relies on rather standard ideas, and essentially uses the self-correction of linear functions [6].

Low agreement with one-sided error

For this result, we restrict our discussion to the finite field $\mathbb{F}_2$. Note that it is trivial to design an $O(n^2)$ time algorithm such that $\mathbb{E}_{A,B\in\mathbb{F}_2^{n\times n}}[\mathrm{agr}(\mathrm{ALG}(A, B), A \cdot B)] \geq 1/2$. Indeed, the algorithm can simply output 0 in all entries irrespective of the input. Alternatively, the algorithm can output a random 0/1 matrix. Hence, it is natural to ask whether it is possible to obtain a better-than-1/2 algorithm for the matrix multiplication over $\mathbb{F}_2$.

Below we show that in the special case of *one-sided error* approximation, any better-than-1/2 approximation $O(T)$ time algorithm can be transformed into a worst case algorithm with running time $\widetilde{O}(T)$. Formally, we prove the following theorem.

▶ **Theorem 3.** *Let* ALG *be an algorithm that gets input two matrices* $A, B \in \mathbb{F}_2^{n\times n}$*, runs in time* $T(n)$*, and outputs a matrix* $\mathrm{ALG}(A,B) \in \mathbb{F}_2^{n\times n}$*. Let* $\delta > 0$*, and suppose that*

- $\mathbb{E}_{A,B\in\mathbb{F}_2^{n\times n}}[\mathrm{agr}(\mathrm{ALG}(A,B), A\cdot B)] \geq 1/2+\delta$*.*
- *If* $(AB)_{i,j} = 0$*, then* $\mathrm{ALG}(A,B)_{i,j} = 0$*.*

Then, there is an algorithm ALG^* *that gets as input two matrices* $A, B \in \mathbb{F}_2^{n\times n}$*, runs in time* $\widetilde{O}(T(n))$*, and outputs a matrix* $\mathrm{ALG}^*(A,B) \in \mathbb{F}_2^{n\times n}$ *such that for all* A, B *it holds that*

$$\Pr[\mathrm{ALG}^*(A,B) = A\cdot B] > 1 - 1/n,$$

where the randomness is only over the internal coins of ALG^**.*

▶ Remark 4. Below we make several comments about Theorem 3.

1. Note that the conditions of the theorem can be written equivalently as follows.
   - $\Pr_{\substack{A,B\in\mathbb{F}_2^{n\times n}\\ i,j\in[n]}}[\mathrm{ALG}(A,B)_{i,j} = 1] \geq \delta$.
   - If $(AB)_{i,j} = 0$, then $\mathrm{ALG}(A,B)_{i,j} = 0$.
2. The notion of algorithms with one-sided error is typically studied in the context of randomized algorithms, e.g., relating to the classes $\mathcal{RP}$ (and $co\mathcal{RP}$), where the guarantee is that for *every NO input* the algorithm outputs the correct answer with probability 1, and for *every YES input* it is correct with probability at least 2/3. The error model in Theorem 3 is different, as we consider algorithms that are correct on *random inputs* on all output 0-bits, and on at least some α-fraction of 1-bits.
3. Alternatively, we can view the one-side error condition of Theorem 3 as an *errorless heuristic*, where in each entry of the matrix ALG outputs either 1 representing the correct answer, or says "I don't know" and outputs 0.
4. We remark that the standard methods of self-correcting linear functions work in the high agreement regime, but fail when the average case guarantee is low. We apply the techniques from additive combinatorics developed in [2], particularly a version of the probabilistic Bogolyubov-Ruzsa Lemma, to perform a self-correction procedure which helps in this regime.
5. Our proof of Theorem 3 assumes that ALG is deterministic. It is rather straightforward to extend the proof and allow it to be randomized, by appropriately modifying the sets of good inputs ($X_{i,j}$ and $Y^A_{i,j}$) to account for the randomness of the algorithm.

1.1 Related work

The study of average-case complexity began with Levin's work [18], followed by subsequent works like [4]. A substantial body of research (e.g., [16], [15] and related references) identified numerous barriers in formulating worst-case to average-case reductions for NP-complete problems. For a comprehensive overview of this subject, see the classical surveys by Impagliazzo [14], Bogdanov and Trevisan [7] and Goldreich [12].

Asadi et al. [2, 3] presented a new framework for carrying out efficient worst-case to average case reductions for various fundamental problems. Particularly, for the problem of matrix multiplication, they proved that if there exists an $O(T(n))$ time algorithm M for matrix multiplication which computes the correct output on an ϵ fraction of inputs, then there exists a randomized algorithm M' which computes the correct output on all inputs, running in time $O(\exp(O(\log^5(1/\epsilon)) \cdot T(n))$. The proof relied on additive combinatorial techniques and used the probabilistic Bogolyubov-Ruzsa Lemma.

Hirahara and Shimizu [13] improved the $\exp(O(\log^5(1/\epsilon)))$ overhead to an $\widetilde{O}(1/\epsilon)$ factor. Their idea involved dividing the output matrix into smaller blocks and using the Direct-Product Theorem in a black-box manner.

The aforementioned papers assume that we have access to an algorithm which gives a fully correct output on some fraction of the inputs, i.e., for these inputs *all entries* in the output matrix are correct. The setting presented in this paper, where the output of the given algorithm is not fully correct, seems to differ significantly from the works mentioned above. In particular, we do not see how to apply the Direct-Product theorem to our setting of the problem.

A related problem was studied by [11]. Specifically, they provided an $\widetilde{O}(n^2 + kn)$ time randomized algorithm and an $\widetilde{O}(kn^2)$ time deterministic algorithm for correcting the product of two matrices over a ring, where the product has at most k incorrect entries. Theorem 2 improves upon their work for a certain range of k, e.g., $n^2/20 < k < n^2/8$, and Theorem 3 gives a new result for k closer to $n^2/2$ in a very specific error model.

1.2 Open problems

We mention the following two problems that are left open in this work.

Low agreement with two-sided error

Is it possible to transform a two-sided error algorithm over $\mathbb{F}_2$ with a low agreement guarantee into a worst case algorithm. That is, given an $O(T(n))$ time algorithm ALG with the guarantee $\mathbb{E}_{A,B\in\mathbb{F}_2^{n\times n}}[\mathrm{agr}(\mathrm{ALG}(A,B), A\cdot B)] > 1/2 + \delta$, can we convert it into an algorithm that correctly outputs the correct answer on all inputs and has running time $\widetilde{O}(T(n))$?

Generalizing over finite fields

Extend Theorem 3 in a meaningful way to work over any finite field.

2 Preliminaries

For a positive integer n we define $[n] = \{0, 1, \ldots, n-1\}$. We index the coordinates of our matrices starting from 0 rather than 1, which is typically more standard. We refer to the element in the row i and column j of the matrix A as $A_{i,j}$.

We define a notion of *row-shift* (or *row-rotation*) and *column-shift* as follows.

▶ **Definition 5.** *Given a matrix $A \in \mathbb{F}^{n\times m}$, $0 \le \pi \le n-1$, and $0 \le \sigma \le m-1$, define $A^{\pi,\sigma}$ to be the matrix obtained from A by cyclically rotating all its rows downwards by π units and all its columns rightwards by σ units, that is,*

$$(A^{\pi,\sigma})_{i,j} = A_{(i-\pi) \bmod n,(j-\sigma) \bmod m}$$

The following proposition is immediate from the definition above.

▶ **Proposition 6.** *For any $A, B, C \in \mathbb{F}^{n\times n}$ and any π, σ we have $AB = C$ if and only if $A^{\pi,0} \cdot B^{0,\sigma} = C^{\pi,\sigma}$.*

2.1 Additive Combinatorics Tools

We now present the additive combinatorics toolkit which will be useful in the worst-case to average-case reduction for the low agreement regime with one-sided error.

For a set $A \subseteq \mathbb{F}_2^n$, let $1_A \colon \mathbb{F}_2^n \to \{0,1\}$ denote the indicator function of A. The Fourier expansion of a function $f \colon \mathbb{F}_2^n \to \mathbb{C}$ is given by $f(x) = \sum_{r \in \mathbb{F}_2^n} \hat{f}(r) \cdot \chi_r(x)$, where $\chi_r(x) = (-1)^{\langle x,r \rangle}$, and the Fourier coefficients of f are defined as $\hat{f}(r) = \langle f, \chi_r \rangle = \mathbb{E}_x[f(x) \cdot \chi_r(x)]$. *Parseval's identity* says that $\sum_{r \in \mathbb{F}_2{}^n} \widehat{1_A}(r)^2 = \langle 1_A, 1_A \rangle = \alpha$, where α is the density of A.

Define $\mathrm{Spec}_\gamma(A) = \{r \in \mathbb{F}_2^n : \left|\widehat{1_A}(r)\right| \geq \gamma\}$. Below we state Chang's lemma, which describes a certain structure of $\mathrm{Spec}_\gamma(A)$.

▶ **Lemma 7** (Chang's Theorem [8]). *Let $A \subseteq \mathbb{F}^n$ be a set of size $|A| = \alpha \cdot |\mathbb{F}|^n$, and let $\gamma > 0$. Then*

$$\dim(\mathsf{span}(\mathrm{Spec}_{\gamma\alpha}(A))) \leq O\left(\frac{\log(1/\alpha)}{\gamma^2}\right) \ .$$

Recall that the subset sum of two sets A and B is defined as $A+B = \{a+b : a \in A\ , b \in B\}$. Analogously, we define $tA = A+A+\cdots+A$ (t times) as $tA = \{a_1+a_2\cdots+a_t : a_1, a_2, \ldots, a_t \in A\}$. The following lemma says that for an arbitrary set $A \subseteq \mathbb{F}^n$, the sumset tA contains a large affine subspace.

▶ **Lemma 8** (Probabilistic Bogolyubov-Ruzsa lemma). *Let $A \subseteq \mathbb{F}_2^n$ be a set of size $|A| = \alpha \cdot 2^n$, for some $\alpha \in (0,1]$, and let $t \geq 3$ be an integer. Then, tA contains an* affine *subspace $V \subseteq \mathbb{F}_2^n$ of dimension $\dim(V) \geq n - O(\log(1/\alpha))$ such that for all $v \in V$ it holds that*

$$\Pr_{a_1,a_2,..,a_{t-1} \in \mathbb{F}_2^n}[a_1, a_2, a_3, .., a_t \in A] \geq \alpha^t \left(1 + \frac{1}{2^{t-2}}\right) - \frac{\alpha^{t-1}}{2^{t-2}} \ ,$$

where $a_t = v - a_1 - a_2 - .. - a_{t-1}$.

In particular, if $t > \log_2(1/\alpha) + 2$, then tA contains an affine *subspace $V \subseteq \mathbb{F}_2^n$ of dimension $\dim(V) \geq n - k$, for $k = O(\log(1/\alpha))$, such that for all $v \in V$ it holds that*

$$\Pr_{\substack{a_1,a_2,..,a_t \in \mathbb{F}_2^n \\ v = \sum_{i=1}^t a_i}}[a_1, a_2, a_3, .., a_t \in A] \geq (\alpha/2)^t \ .$$

Below we prove Lemma 8 only for odd values of t, which is slightly more complicated than the case of even t. After the proof, we remark how to modify the proof to work for even t's.

Proof. Let $A \subseteq \mathbb{F}_2^n$ be a set of size $|A| = \alpha \cdot |\mathbb{F}|^n$, for some $\alpha \in (0,1]$. Consider the set

$$R = \mathrm{Spec}_{\alpha/2} \setminus \{0\} = \{r \in F_2^n \setminus \{0\} : \left|\widehat{1_A}(r)\right| > \frac{\alpha}{2}\}$$

Next we define an affine subspace $V = \{v \in \mathbb{F}_2^n : \langle v, r \rangle = s_r \ \forall r \in R\}$ for some $s_r \in \{0,1\}$ to be defined later, and claim that V satisfies the conclusions of Lemma 8. We will need the following two claims.

▷ Claim 9. For all $r \in R$ there exists $s_r \in \{0,1\}$ such that (1) $\sum_{r \in R} \widehat{1_A}(r)^t \cdot (-1)^{s_r} \geq 0$ and (2) if $r^* \in R$ is a linear combination $r^* = \sum_{r \in R} c_r \cdot r$ of vectors in R (with $c_r \in \mathbb{F}_2$), then $s_{r^*} = \sum_{r \in R} c_r \cdot s_r \pmod 2$.

Proof. Let R' be a maximal subset of R of linearly independent vectors. Choose $s_{r'} \in \{0,1\}$ independently with probability 0.5 each for every $r' \in R'$. Now any $r \in R \setminus R'$, can be expressed as a linear combination $r = \sum_{r'} c_{r'} \cdot r'$ of vectors in R' with $c_{r'} \in \{0,1\}$ define $s_r = \sum_{r'} c_{r'} \cdot s_{r'}$. It is immediate to verify that condition (2) is satisfied.

In order to satisfy condition (1) note that by linearity of expectation $\mathbb{E}[\sum_{r\in R} \widehat{1_A}(r)^t \cdot (-1)^{s_r}] = 0$, and hence there exists a choice of $(s_r)_{r\in R}$ such that $\sum_{r\in R} \widehat{1_A}(r)^t \cdot (-1)^{s_r} \geq 0$, as required. ◁

▷ Claim 10. We have $\sum_{r\notin R, r\neq 0} \left|\widehat{1_A}(r)\right|^t \leq (\alpha/2)^{t-2}(\alpha - \alpha^2)$.

Proof. For $t \geq 3$, it holds that

$$\begin{aligned}\sum_{r\notin R, r\neq 0} \left|\widehat{1_A}(r)\right|^t &\leq \max_{r\notin R, r\neq 0} \left|\widehat{1_A}(r)\right|^{t-2} \sum_{r\notin R, r\neq 0} \left|\widehat{1_A}(r)\right|^2 \\ &\leq (\alpha/2)^{t-2} \sum_{r\in \mathbb{F_2}^n \setminus \{0\}} \widehat{1_A}(r)^2 \\ &< (\alpha/2)^{t-2}(\alpha - \alpha^2) \ .\end{aligned}$$ ◁

Define an affine subspace $V = \{v \in \mathbb{F}_2^n : \langle v, r\rangle = s_r \ \forall r \in R\}$, where $s_r \in \{0,1\}$ is from Claim 9. Note that if the vectors in R are linearly dependent, then the second condition of Claim 9 guarantees that we can define $V = \{v \in \mathbb{F}^n : \langle v, r'\rangle = s_{r'} \ \forall r' \in R'\}$ for a maximal set $R' \subset R$ of linearly independent vectors in R, and the remaining constraints will be satisfied by linearity. Then, according to Lemma 7 we have

$$\dim(V) \geq n - O\left(\log(1/\alpha)\right) \ .$$

Using the two claims above, and noting that $\Pr_{a_1,a_2,..,a_{t-1}\in\mathbb{F}^n}[a_1, a_2, a_3, .., a_t \in A] = 1_A * 1_A * .. * 1_A(v)$ (t times), for any $v \in V$ we have

$$\begin{aligned}\Pr_{a_1,a_2,..,a_{t-1}\in\mathbb{F}^n}[a_1, a_2, a_3, .., a_t \in A] &= 1_A * 1_A * .. * 1_A(v) \\ &= \sum_{r\in\mathbb{F}^n} \widehat{1_A}(r)^t \chi_r(v) \\ &= \widehat{1_A}(0)^t + \sum_{r\in R} \widehat{1_A}(r)^t \chi_r(v) + \sum_{r\notin R, r\neq 0} \widehat{1_A}(r)^t \chi_r(v) \\ &\geq \alpha^t + \sum_{r\in R} \widehat{1_A}(r)^t \cdot (-1)^{s_r} - (\alpha/2)^{t-2}(\alpha - \alpha^2) \\ &\geq \alpha^t + 0 - (\alpha/2)^{t-2}(\alpha - \alpha^2) \\ &= \alpha^t \left(1 + \frac{1}{2^{t-2}}\right) - \frac{\alpha^{t-1}}{2^{t-2}} \ .\end{aligned}$$

In particular, for $t > \log_2(1/\alpha) + 2$, we have

$$\begin{aligned}\Pr_{a_1,a_2,..,a_{t-1}\in\mathbb{F}^n}[a_1, a_2, a_3, .., a_t \in A] &\geq \alpha^t \left(1 + \frac{1}{2^{t-2}}\right) - \frac{\alpha^{t-1}}{2^{t-2}} \\ &\geq \alpha^t \left(1 + \frac{1}{2^{t-2}}\right) - \alpha^t \\ &\geq (\alpha/2)^t \ ,\end{aligned}$$

as required. ◀

▶ Remark 11. For even values of t the lemma is slightly easier. Specifically, since $\widehat{1_A}(r)^t$ is always non-negative, we can take $s_r = 0$ in Claim 9, and the rest of the proof works the same.

3 High Agreement with Two-Sided Error

In this section, we prove Theorem 2. Specifically, we show that if there exists an algorithm which, given two matrices $A, B \in \mathbb{F}^{n\times n}$, runs in time $T(n)$ and correctly computes their product on a large fraction of all entries of output on average, then there exists another algorithm that runs in $\widetilde{O}(T(n))$ time and correctly computes their product on all entries of output. The proof essentially uses the self-correction of linearity [6].

▶ **Theorem 2.** *Fix a finite field $\mathbb{F}$. Let $\alpha \in (0, 1/8)$. Let* ALG *be an algorithm that gets as input two matrices $A, B \in \mathbb{F}^{n\times n}$, runs in time $T(n)$, and outputs a matrix* $\mathrm{ALG}(A, B) \in \mathbb{F}^{n\times n}$. *Suppose that*

$$\mathbb{E}_{A,B\in\mathbb{F}^{n\times n}}[\mathrm{agr}(\mathrm{ALG}(A, B), A \cdot B)] > 1 - \alpha \ .$$

Then, there is an algorithm ALG^* *that gets as input two matrices $A, B \in \mathbb{F}^{n\times n}$, runs in time $O(T(n) \cdot \log(n))$, and outputs a matrix* $\mathrm{ALG}^*(A, B) \in \mathbb{F}^{n\times n}$ *such that* for all A, B *it holds that*

$$\Pr[\mathrm{ALG}^*(A, B) = A \cdot B] > 1 - 1/n \ ,$$

where the randomness is only over the internal coins of ALG^*.

Proof. Given the algorithm ALG as in the assumption of the theorem, we design ALG^* as follows.

Algorithm 1 Approximation for High Agreement Matrix Multiplication Algorithms.

Input: $A, B \in \mathbb{F}^{n\times n}$, ALG
Output: $A \cdot B$
Let $k = O(\log(n))$
for $r = 0$ *to* k **do**
  Generate two random matrices $R, S \in \mathbb{F}^{n\times n}$
  Select two random variables $\pi, \sigma \in [n]$ independently
  $M = \mathrm{ALG}((A + R)^{\pi,0}, (B + S)^{0,\sigma}) - \mathrm{ALG}(R^{\pi,0}, (B + S)^{0,\sigma}) - \mathrm{ALG}((A + R)^{\pi,0}, S^{0,\sigma}) + \mathrm{ALG}(R^{\pi,0}, S^{0,\sigma})$
  Let $C_r = M^{n-\pi,n-\sigma}$
Define the matrix $C \in \mathbb{F}^{n\times n}$ by taking the majority vote of all C_r in each coordinate.

Correctness

Consider an entry (i, j) in the output matrix $A \cdot B$. In each iteration we call ALG four times, and in each of the calls the input is distributed uniformly in $\mathbb{F}^{n\times n}$. Furthermore, since π, σ are chosen uniformly, it follows that $(i + \pi, j + \sigma)$ are distributed uniformly. Therefore, the probability that that all the four calls of ALG produce the correct answer in this entry is at least $1 - 4\alpha$. Therefore, for each repetition r, we have

$$\Pr[(C_r)_{i,j} = (A \cdot B)_{i,j}] \geq 1 - 4\alpha \ .$$

By Chernoff bound, the probability that the majority vote of the k repetition will produce an incorrect answer is upper bounded by

$$\Pr[(C_r)_{i,j} = (A \cdot B)_{i,j}] \geq \exp\left(-\Omega((1 - 4\alpha - 1/2) \cdot k)\right) < 1/n^3 \ ,$$

Here we make the assumption that α is bounded below $1/8$.

Hence, the probability that a particular entry (i, j) is incorrect after k iterations is at most n^{-3}. By union bound over all entries, the probability that at least one entry is incorrect in the output matrix is at most $n^2 \cdot n^{-3} = 1/n$.

Running time

The total running time is dominated by $O(\log(n))$ invocations of ALG, and hence, the runtime of ALG* is $O(T(n) \cdot \log(n))$. ◀

4 Low Agreement with One-Sided Error

In this section we prove Theorem 3. We restate the theorem here for convenience.

▶ **Theorem 3.** *Let* ALG *be an algorithm that gets input two matrices $A, B \in \mathbb{F}_2^{n \times n}$, runs in time $T(n)$, and outputs a matrix* $\mathrm{ALG}(A, B) \in \mathbb{F}_2^{n \times n}$. *Let $\delta > 0$, and suppose that*

- $\mathbb{E}_{A,B \in \mathbb{F}_2^{n \times n}}[\mathrm{agr}(\mathrm{ALG}(A, B), A \cdot B)] \geq 1/2 + \delta$.
- *If $(AB)_{i,j} = 0$, then* $\mathrm{ALG}(A, B)_{i,j} = 0$.

Then, there is an algorithm ALG* *that gets as input two matrices $A, B \in \mathbb{F}_2^{n \times n}$, runs in time $\widetilde{O}(T(n))$, and outputs a matrix* $\mathrm{ALG}^*(A, B) \in \mathbb{F}_2^{n \times n}$ *such that for all A, B it holds that*

$$\Pr[\mathrm{ALG}^*(A, B) = A \cdot B] > 1 - 1/n,$$

where the randomness is only over the internal coins of ALG*.

Before proving the theorem, we need some definitions. We start by defining the notion of a *good* coordinate. We say a coordinate $(i, j) \in [n] \times [n]$ is *good*, if ALG returns 1 at the entry (i, j) for more than $\delta/2$ fraction of possible inputs.

▶ **Definition 12.** *Denote by G the set of* good *coordinates, defined as*

$$G = \{(i, j) \in [n] \times [n] : \Pr_{A,B \in \mathbb{F}_2^{n \times n}}[\mathrm{ALG}(A, B)_{i,j} = 1] > \delta/2\} \ .$$

The following claim is immediate from the definition and the assumptions of the theorem.

▷ Claim 13. $|G| \geq (\delta/2) \cdot n^2$.

Proof. Let $p_{i,j} = \Pr_{A,B \in \mathbb{F}_2^{n \times n}}[\mathrm{ALG}(A, B)_{i,j} = 1]$. Note that by the assumptions of Theorem 3, we have $\mathbb{E}_{i,j}[p_{i,j}] \geq \delta$. Note that

$$\delta \leq \mathbb{E}_{i,j \in [n] \times [n]}[p_{i,j}] \leq \Pr_{i,j}[(i, j) \in G] \cdot 1 + \Pr_{i,j}[(i, j) \notin G] \cdot (\delta/2) \leq \Pr_{i,j}[(i, j) \in G] \cdot 1 + 1 \cdot (\delta/2) \ ,$$

and hence $\Pr[(i, j) \in G] \geq \delta/2$, as required. ◁

Next, we define the notion of good input matrices with respect to a good coordinate.

▶ **Definition 14.** *For a coordinate (i,j), define $X_{i,j}$ as follows.*

$$X_{i,j} = \{A : \Pr_B[\mathrm{ALG}(A,B)_{i,j} = 1] \geq \delta/4\} \ .$$

Given a coordinate (i,j) and a matrix A, define $Y^A_{i,j}$ to be the set of matrices B for which ALG *returns 1 at the entry (i,j). That is,*

$$Y^A_{i,j} = \{B : \mathrm{ALG}(A,B)_{i,j} = 1\} \ .$$

We make the following claim about the densities of $X_{i,j}$ and $Y^A_{i,j}$.

▷ Claim 15. For any $(i,j) \in G$ it holds that $\Pr_{A\in\mathbb{F}^{n\times n}}[A \in X_{i,j}] \geq \delta/4$. Furthermore, if $A \in X_{i,j}$ then $\Pr_{B\in\mathbb{F}^{n\times n}}[B \in Y^A_{i,j}] \geq \delta/4$.

Proof. Fix a good coordinate $(i,j) \in G$, and for each $A \in \mathbb{F}^{n\times n}$,
Let $p_A = \Pr_{B\in\mathbb{F}^{n\times n}}[\mathrm{ALG}(A,B)_{(i,j)} = 1]$. From the definition of G we have $\mathbb{E}_A[p_A] \geq \delta/2$.

$$\delta/2 \leq \mathbb{E}_A[p_A] = \Pr_{A\in\mathbb{F}^{n\times n}}[A \in X_{i,j}] \cdot 1 + \Pr_{A\in\mathbb{F}^{n\times n}}[A \notin X_{i,j}] \cdot \delta/4 \leq \Pr_{A\in\mathbb{F}^{n\times n}}[A \in X_{i,j}] + \delta/4 \ ,$$

and hence, $\Pr[A \in X_{i,j}] \geq \delta/4$.

The furthermore part is by definition of $X_{i,j}$. ◁

▶ **Definition 16.** *Denote by $L^k \in \mathbb{F}^{n\times n}$ a random matrix of rank at most k, constructed by sampling the first k columns independently uniformly at random from $\mathbb{F}^n$, and then taking the remaining $n-k$ columns to be uniformly random linear combinations of the first k vectors.*

The following lemma is from [2]. It shows that if L^{2k}_A is a random matrix of rank at most $2k$ sampled as in Definition 16, then $M_A = A - (L^{2k}_A)$ belongs to any subspace of matrices of co-dimension k with a non-negligible probability. We provide the proof of the lemma here for completeness.

▶ **Lemma 17** (Lemma 4.8 from [2])**.** *Fix a matrix $A \in \mathbb{F}^{n\times n}$, let k be a parameter, and let $\ell \geq 2k$. Let L^ℓ_A be a random matrix of rank at most ℓ sampled as in Definition 16, and let $M_A = A - (L^\ell_A)$. Then, for any subspace $V \subseteq \mathbb{F}^{n\times n}$ of $\dim(V) \geq n^2 - k$ it holds that*

$$\Pr[M_A \in V] \geq \frac{1}{2|\mathbb{F}|^k} \ .$$

Proof. Since V has co-dimension at most k, a matrix in V must be orthogonal to all the basis vectors of its orthogonal complement. Since there are up to k such basis vectors, the membership condition of M_A in V can be written down in the form of k linear constraints. Viewing M_A as a vector in $\mathbb{F}^{n^2}$, we can write the k linear constraints on the elements of the matrix M^{2k}_A in the form

$$\alpha_1 \cdot (M_A)_{i_1,j_1} + \alpha_2 \cdot (M_A)_{i_2,j_2} + \cdots + \alpha_r (M_A)_{i_t,j_t} = 0 \ .$$

Here, α_i's are constants and $t \in [n^2]$ is the number of elements upon which the constraints depend. Writing M_A as $m \in \mathbb{F}^{n^2}$, we can represent these linear constraints as a system of equations of the form $G \cdot m = 0$, where G is a $k \times n^2$ matrix. Now, we perform Gaussian elimination on G, which gives us a matrix G', where each row has a 1 entry such that all the other entries in the column containing this 1 are 0. We refer to such 1's as *leading 1's*. That is, by permuting the columns of G', we may think of it as being of the form $G' = [I_k|G^*]$.

Consider the set of k coordinates of m corresponding to the k leading 1s in G', one from each row. These k coordinates of m in turn correspond to k pairs of coordinates $\{(i_1, j_1), (i_2, j_2) \dots (i_k, j_k)\}$ in M_A. These k pairs of coordinates in M_A can belong to at most k rows in M_A. We now bound the probability of none of these k rows in L_A^ℓ being a linear combination of the other rows. Let us denote this event as Ω. Then

$$\begin{aligned}
\Pr[\Omega] &= \left(1 - \frac{1}{2^\ell}\right)\left(1 - \frac{2}{2^\ell}\right)\left(1 - \frac{4}{2^\ell}\right)\cdots\left(1 - \frac{2^{k-1}}{2^\ell}\right) \\
&\geq \left(1 - \frac{2^{k-1}}{2^\ell}\right)^k \\
&\geq \left(1 - \frac{1}{2^{k+1}}\right)^k \\
&\geq 1 - \frac{k}{2^{k+1}} \geq \frac{1}{2}
\end{aligned}$$

If Ω happens, then we get a coordinate (i_r, j_r) in M_A corresponding to the r^{th} linear constraint, for all $r \in [k]$, such that no other constraint depends upon it (as it corresponds to a leading 1) and it is chosen uniformly at random (since the rows containing these coordinates are linearly independent). Therefore, the probability that this random value satisfies the i^{th} constraint is $1/|\mathbb{F}|$. To see this, assume that the values of all other coordinates involved in the i^{th} constraint are fixed, then we are left with only one choice for the value of the coordinate (c_i, c_i') which satisfies the constraint. Therefore, we have

$$\begin{aligned}
\Pr[M_A \in V] &= \Pr[\text{All } k \text{ linear constraints are satisfied}] \\
&= \Pr[\Omega] \cdot \frac{1}{|\mathbb{F}|^k} \geq \frac{1}{2|\mathbb{F}|^k} \ .
\end{aligned}$$

This completes the proof of Lemma 17. ◀

4.1 Computing the good coordinates

Next, we start describing the reduction guaranteed by Theorem 3. As a first step we design Algorithm 2, that gets two matrices A, B and outputs a matrix C with values in $\mathbb{F}_2 \cup \{\perp\}$, satisfying the following guarantees.

1. If $C_{i,j} \neq \perp$, then $C_{i,j}$ contains the correct values, i.e., $C_{i^*,j^*} = (A \cdot B)_{i^*,j^*}$.
2. For any good coordinate $(i^*, j^*) \in G$ we have $\Pr[C_{i^*,j^*} = (A \cdot B)_{i^*,j^*}] \geq \delta_0$, where δ_0 is some constant that depends only on δ. That is, with non-negligible probability C_{i^*,j^*} contains the correct answer, and not $\perp$.

Then, in Section 4.2 we use Algorithm 2 as a subroutine, in order to compute the entire matrix $A \cdot B$ correctly.

In lines 3-4 we decompose $A = L_A^{2k} + M_A$, and $B = L_B^{2tk} + M_B$ with the intention of computing $A \cdot B$ by writing

$$\begin{aligned}
AB &= (M_A + L_A^{2k}) \cdot (M_B + L_B^{2tk}) \\
&= M_A \cdot M_B + M_A \cdot L_B^{2tk} + L_A^{2k} \cdot M_B + L_A^{2k} \cdot L_B^{2tk} \ .
\end{aligned}$$

Lines 5-13 try to compute $C = M_A \cdot M_B$. Then, in line 14, we sum up the 4 terms. Using the fact that multiplication of matrices of rank k takes $O(kn^2)$ time, the last three terms can be computed in $O(tkn^2)$ time, and hence, it remains to compute $M_A \cdot M_B$. The remainder of this subsection is dedicated to analyzing lines 5-13, which contain the most involved part of the algorithm.

Algorithm 2 Approximating good coordinates for one-sided error algorithms.

Input: $A, B \in \mathbb{F}_2^{n\times n}$, ALG
Output: An $n \times n$ matrix C with values $\mathbb{F}_2 \cup \{\perp\}$
1 Let $t > \log(4/\delta) + 2$
2 Let $k = O(\log(1/\delta))$ from the "in particular" part of Lemma 8 with $\alpha = \delta/4$ and t chosen above
3 Sample two random matrices L_A^{2k} and L_B^{2tk} of rank at most $2k$ and $2tk$ respectively, as in Definition 16
4 Define $M_A = A - L_A^{2k}$ and $M_B = B - L_B^{2tk}$
5 Let C be the $n \times n$ matrix initialized with all $\perp$
6 Sample $t-1$ random matrices $R_1, R_2, .., R_{t-1} \in \mathbb{F}_2^{n\times n}$ and set $R_t = M_A - (R_1 + R_2 + .. + R_{t-1})$
7 **for** $r = 1, \ldots, t$ **do**
8 Sample $t-1$ random matrices $S_1^{(r)}, S_2^{(r)}, .., S_{t-1}^{(r)} \in \mathbb{F}_2^{n\times n}$ and set $S_t^{(r)} = M_B - (S_1^{(r)} + S_2^{(r)} + .. + S_{t-1}^{(r)})$
9 **for** $r, s = 1, \ldots, t$ **do**
10 Compute $\mathrm{ALG}(R_r, S_s^{(r)})$
11 **for** $(i, j) \in [n] \times [n]$ **do**
12 **if** $\mathrm{ALG}(R_r, S_s^{(r)})_{i,j} = 1$ *for all* $r, s \in \{1, \ldots, t\}$ **then**
13 Set $C_{i,j} = \sum_{r,s} \mathrm{ALG}(R_r, S_s^{(r)})_{i,j} \pmod 2$
14 **return** $C + M_A \cdot L_B^{2tk} + L_A^{2k} \cdot M_B + L_A^{2k} \cdot L_B^{2tk}$ // if $C_{i,j} = \perp$, then we return $\perp$ in the coordinate (i, j)

We would like to compute $M_A \cdot M_B$ by writing $M_A = R_1 + R_2 + \cdots + R_t$, and $M_B = S_1^{(r)} + S_2^{(r)} + \cdots + S_t^{(r)}$ for $r = 1 \ldots t$, and then computing $\mathrm{ALG}(R_r, S_s^{(r)})$ for all r, s. Note that if we could guarantee that $\mathrm{ALG}(R_r, S_s^{(r)})$ returns $R_r \cdot S_s^{(r)}$, then, we would have

$$M_A \cdot M_B = \sum_{r,s} R_r \cdot S_s^{(r)} = \sum_{r,s} \mathrm{ALG}(R_r, S_s^{(r)}) \ . \tag{1}$$

However, ALG is not guaranteed to return the product of the inputs correctly. Instead, we claim that (1) for *some* good coordinates (i^*, j^*) it holds $C_{i^*,j^*} = (M_A \cdot M_B)_{i^*,j^*}$, and (2) the remaining coordinates in C remain $\perp$. This is summarized formally in the next two claims.

▷ Claim 18. For any $(i, j) \in [n] \times [n]$ if $C_{i,j} \in \{0, 1\}$ (i.e., $C_{i,j} \neq \perp$), then $C_{i,j} = (M_A \cdot M_B)_{i,j}$.

Proof. Fix any coordinate (i, j). Note that in line 13 we set $C_{i,j} = \sum_{r,s} \mathrm{ALG}(R_r, S_s^{(r)})_{i,j} \pmod 2$ only if $\mathrm{ALG}(R_r, S_s^{(r)})_{i,j} = 1$ for all r, s. Recall that by the assumption of the algorithm if $\mathrm{ALG}(R_r, S_s^{(r)})_{i,j} = 1$, then $\mathrm{ALG}(R_r, S_s^{(r)})_{i,j} = (R_r \cdot S_s^{(r)})_{i,j}$. The claim follows by Equation (1) restricted to the coordinate (i, j), as

$$C_{i,j} = \sum_{r,s} \mathrm{ALG}(R_r, S_s^{(r)})_{i,j} = \sum_{r,s} (R_r \cdot S_s^{(r)})_{i,j} = (M_A \cdot M_B)_{i,j} \ ,$$

as required. ◁

▷ Claim 19. Fix a good coordinate $(i^*, j^*) \in G$. Then $\Pr[C_{i^*,j^*} = (M_A \cdot M_B)_{i^*,j^*}] \geq \delta_0 = 0.5^{O(\log^3(1/\delta))}$.

Proof. Consider the set X_{i^*,j^*} from Definition 14 for a good entry $(i^*, j^*) \in G$. By Claim 15, the density of $X_{i,j}$ is at least $\delta/4$, and hence, Lemma 8 guarantees the existence of an affine subspace V_{i^*,j^*} of dimension $\dim(V_{i^*,j^*}) \geq n-k$. Then, using Lemma 17 with V_{i^*,j^*} we have

$$\Pr[M_A \in V_{X_{i,j}}] \geq \frac{1}{2 \cdot 2^k} \ . \tag{2}$$

Let us condition on the event that $M^A \in V_{X_{i^*,j^*}}$. Then by Lemma 8,

$$\Pr_{\substack{R_1,R_2,..,R_t \in \mathbb{F}_n^{n \times n} \\ \sum_r R_r = M_A}} [R_1, R_2, .., R_t \in X_{i,j}] \geq (\delta/8)^t \ . \tag{3}$$

For each of $R_1, \ldots, R_t$ define the sets $Y^{R_1}, Y^{R_2}, \ldots, Y^{R_t}$ as in Definition 14. (Recall Y^R is the set of are all matrices S such that $(R \cdot S)_{i^*,j^*} = 1$. We omit the subscript (i^*, j^*) for readability.)

From Claim 15, we know each of $Y^{R_1}, \ldots, Y^{R_t}$ has density at least $\delta/4$. Hence, by applying Lemma 8 on each of them, we obtain subspaces $V_{Y^{R_1}}, .., V_{Y^{R_t}}$ of co-dimension at most k. Define $V_Y = V_{Y^{R_1}} \cap V_{Y^{R_2}} \cap \cdots \cap V_{Y^{R_t}}$ to be their intersection, and note that $\dim(V_Y) \geq n - tk$. Therefore, by applying Lemma 17 on the matrix B with the subspace V_Y, we get [1]

$$\Pr[M_B \in V_Y] \geq \frac{1}{2 \cdot 2^{tk}} \ . \tag{4}$$

Conditioning further on the event that $M_B \in V_Y$, we apply Lemma 8, and for each $r = 1, \ldots, t$ we get

$$\Pr_{\substack{S_1^{(r)},\ldots,S_t^{(r)} \\ \sum_s S_s^{(r)} = M_B}} [S_1^{(r)}, \ldots, S_t^{(r)} \in Y^{R_r}] \geq (\delta/8)^t \ .$$

Since the events above are independent between different r's, the probability that the algorithm returns correct output on the entry (i^*, j^*) is lower bounded by the product of the probabilities in Equations (2)–(4), and hence

$$\Pr[C_{i^*,j^*} = (M_A \cdot M_B)_{i^*,j^*}] \geq \frac{1}{2^{k+1}} \times (\delta/8)^t \times \frac{1}{2 \cdot 2^{tk}} \times \left((\delta/8)^t\right)^t \geq \frac{1}{2^{O(\log^3(1/\delta))}} \ .$$

This completes the proof of Claim 19. ◁

4.2 Proof of Theorem 3

We are now ready to prove Theorem 3. Algorithm 3 uses Algorithm 2 as a subroutine, by running it several times. We claim that Algorithm 3 correctly computes the correct answer with high probability for any input A, B. Note that Algorithm 2 is guaranteed to be correct only for good coordinates, although it does not get the good coordinates as an input, and the guarantee about the good coordinates only appears in the analysis.

[1] Note that although the algorithm samples M_B before $R_1, \ldots, R_t$, in fact they are sampled independently of each other, and hence Lemma 17 is applicable here.

Algorithm 3 Approximation for one-sided Agreement Matrix Multiplication Algorithms.

Input: $A, B \in \mathbb{F}_2^{n \times n}$
Output: $A \cdot B$
Let C be the $n \times n$ matrix initialized with all $\perp$.
Let δ_0 be the constant from Claim 19
repeat $O\left(\frac{\log(n)}{\delta \cdot \delta_0}\right)$ **times**
    Sample uniformly random $\pi, \sigma \in [n]$.
    Run Algorithm 2 with the inputs as $A^{\pi,0}, B^{0,\sigma}$, ALG.
    Let C^* be the resulting matrix
    for $(i,j) \in [n] \times [n]$ **do**
        if $C^*_{i+\pi \pmod n, j+\sigma \pmod n} \neq \perp$ **then**
            Set $C_{i,j} = C^*_{i+\pi, j+\sigma}$
return C

The following claim completes the proof of Theorem 3.

▷ Claim 20. Fix a coordinate $(i,j) \in [n] \times [n]$. Algorithm 3 returns the matrix C such that $\Pr[C_{i,j} = (A \cdot B)_{i,j}] \geq 1 - 1/n^3$.

In particular, by taking the union bound over all coordinates (i,j) it follows that for any input A, B Algorithm 3 returns their product with probability at least $1 - 1/n$.

Proof. Fix a coordinate $(i,j) \in [n] \times [n]$. The algorithm chooses random π and σ, and runs Algorithm 2 on the shifted matrices $A^{\pi,0}$ and $B^{0,\sigma}$.

Note that since π and σ are chosen uniformly at random, it follows that $\Pr[(i + \pi \pmod n), j + \sigma \pmod n)) \in G] = |G|/n^2 \geq \delta/2$.

Suppose that $(i + \pi \pmod n), j + \sigma \pmod n))$ is indeed a good coordinate. Then by Claim 20 with probability δ_0 we obtain the correct answer in the coordinate $(i+\pi \pmod n), j+\sigma \pmod n))$, in which case we set $C_{i,j}$ to be that answer $(A \cdot B)_{i,j}$. Otherwise, $C_{i,j}$ remains $\perp$.

Therefore, with probability at least $(\delta/2) \cdot \delta_0$ in each iteration $C_{i,j}$ changes from $\perp$ to $(A \cdot B)_{i,j}$, and once it changes, it never changes its value again.

By repeating the procedure $R = O(\frac{\log(n)}{\delta \cdot \delta_0})$ times, the probability that in the end of the algorithm $C_{i,j} = \perp$ is upper bounded by $\Pr[C_{i,j} = \perp] \leq (1 - \delta_0)^R < 1/n^3$. This completes the proof of the claim. ◁

We conclude the proof with the analysis of the running time of the algorithm.

Running Time

The total running time of Algorithm 3 is essentially dominated by the running time of Algorithm 2 multiplied by $O\left(\frac{\log(n)}{\delta \cdot \delta_0}\right)$. Each iteration of Algorithm 2 involves $O(t^2)$ calls to ALG plus additional $O(tn^2)$ time. Therefore, the running time is $O(t^2 \log(n) T(n)/\delta \cdot \delta_0)$. Since $t = O(\log 1/\delta)$, and $\delta_0 = 0.5^{O(\log^3(1/\delta))}$ the total running time of the algorithm is is $2^{O(\log^3(1/\delta))} \cdot T(n) \log(n)$.

In particular, even for a slightly sub-constant $\delta \geq \exp(-\log^{0.33}(n))$, our algorithm runs in time $T(n) \cdot n^{o(1)}$.

This completes the proof of Theorem 3.

References

1 Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In *SODA 2021*, pages 522–539. SIAM, 2021.

2 Vahid R. Asadi, Alexander Golovnev, Tom Gur, and Igor Shinkar. Worst-case to average-case reductions via additive combinatorics. In *STOC 2022*, pages 00–00. ACM, 2022.

3 Vahid R. Asadi, Alexander Golovnev, Tom Gur, Igor Shinkar, and Sathyawageeswar Subramanian. *Quantum Worst-Case to Average-Case Reductions for All Linear Problems*, pages 2535–2567. Society for Industrial and Applied Mathematics, 2024. doi:10.1137/1.9781611977912.90.

4 Shai Ben-David, Benny Chor, Oded Goldreich, and Michel Luby. On the theory of average case complexity. *Journal of Computer and System Sciences*, 44(2):193–219, 1992. doi:10.1016/0022-0000(92)90019-F.

5 Dario Bini, Milvio Capovani, Francesco Romani, and Grazia Lotti. O(n2.7799) complexity for n × n approximate matrix multiplication. *Information Processing Letters*, 8(5):234–235, 1979. doi:10.1016/0020-0190(79)90113-3.

6 Manuel Blum, Michael Luby, and Ronitt Rubinfeld. Self-testing/correcting with applications to numerical problems. In *STOC 1990*, pages 73–83. ACM, 1990.

7 Andrej Bogdanov and Luca Trevisan. Average-case complexity. *Found. Trends Theor. Comput. Sci.*, 2(1):1–106, October 2006. doi:10.1561/0400000004.

8 Mei-Chu Chang. A polynomial bound in Freiman's theorem. *Duke Mathematical Journal*, 113(3):399–419, 2002. doi:10.1215/S0012-7094-02-11331-3.

9 Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990. Computational algebraic complexity editorial. doi:10.1016/S0747-7171(08)80013-2.

10 Ran Duan, Hongxun Wu, and Renfei Zhou. Faster matrix multiplication via asymmetric hashing. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 2129–2138. IEEE, 2023. doi:10.1109/FOCS57990.2023.00130.

11 Leszek Gąsieniec, Christos Levcopoulos, Andrzej Lingas, Rasmus Pagh, and Takeshi Tokuyama. Efficiently correcting matrix products. *Algorithmica*, 79(2):428–443, October 2017. doi:10.1007/s00453-016-0202-3.

12 Oded Goldreich, editor. *Studies in complexity and cryptography: miscellanea on the interplay between randomness and computation*. Springer-Verlag, Berlin, Heidelberg, 2011.

13 Shuichi Hirahara and Nobutaka Shimizu. Hardness self-amplification: Simplified, optimized, and unified. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, pages 70–83. Association for Computing Machinery, 2023. doi:10.1145/3564246.3585189.

14 Russell Impagliazzo. A personal view of average-case complexity. *Proceedings of Structure in Complexity Theory. Tenth Annual IEEE Conference*, pages 134–147, 1995. URL: https://api.semanticscholar.org/CorpusID:2154064.

15 Russell Impagliazzo. Relativized separations of worst-case and average-case complexities for NP. In *Proceedings of the 26th Annual IEEE Conference on Computational Complexity, CCC 2011, San Jose, California, USA, June 8-10, 2011*, pages 104–114. IEEE Computer Society, 2011. doi:10.1109/CCC.2011.34.

16 Russell Impagliazzo and Leonid A. Levin. No better ways to generate hard NP instances than picking uniformly at random. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 812–821. IEEE Computer Society, 1990. doi:10.1109/FSCS.1990.89604.

17 François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, ISSAC '14, pages 296–303, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2608628.2608664.

18 Leonid A. Levin. Average case complete problems. *SIAM Journal on Computing*, 15(1):285–286, 1986. doi:10.1137/0215020.

19 V. Ya. Pan. Strassen's algorithm is not optimal trilinear technique of aggregating, uniting and canceling for constructing fast algorithms for matrix operations. In *19th Annual Symposium on Foundations of Computer Science (SFCS 1978)*, pages 166–176, 1978. doi:10.1109/SFCS.1978.34.

20 Francesco Romani. Some properties of disjoint sums of tensors related to matrix multiplication. *SIAM Journal on Computing*, 11(2):263–267, 1982. doi:10.1137/0211020.

21 A. Schönhage. Partial and total matrix multiplication. *SIAM Journal on Computing*, 10(3):434–455, 1981. doi:10.1137/0210032.

22 Andrew James Stothers. On the complexity of matrix multiplication. In *University of Edinburgh*, 2010. URL: https://api.semanticscholar.org/CorpusID:262795811.

23 V. Strassen. The asymptotic spectrum of tensors and the exponent of matrix multiplication. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*, pages 49–54, 1986. doi:10.1109/SFCS.1986.52.

24 Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

25 Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 887–898, New York, NY, USA, 2012. Association for Computing Machinery. doi:10.1145/2213977.2214056.