Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 173 (2020)

Block Rigidity: Strong Multiplayer Parallel Repetition implies Super-Linear Lower Bounds for Turing Machines

Kunal Mittal[*] Ran Raz[*]

Abstract

We prove that a sufficiently strong parallel repetition theorem for a special case of multiplayer (multiprover) games implies super-linear lower bounds for multi-tape Turing machines with advice. To the best of our knowledge, this is the first connection between parallel repetition and lower bounds for time complexity and the first major potential implication of a parallel repetition theorem with more than two players.

Along the way to proving this result, we define and initiate a study of *block rigidity*, a weakening of Valiant's notion of *rigidity* [Val77]. While rigidity was originally defined for matrices, or, equivalently, for (multi-output) linear functions, we extend and study both rigidity and block rigidity for general (multi-output) functions. Using techniques of Paul, Pippenger, Szemerédi and Trotter [PPST83], we show that a block-rigid function cannot be computed by multi-tape Turing machines that run in linear (or slightly super-linear) time, even in the non-uniform setting, where the machine gets an arbitrary advice tape.

We then describe a class of multiplayer games, such that, a sufficiently strong parallel repetition theorem for that class of games implies an explicit block-rigid function. The games in that class have the following property that may be of independent interest: for every random string for the verifier (which, in particular, determines the vector of queries to the players), there is a unique correct answer for each of the players, and the verifier accepts if and only if all answers are correct. We refer to such games as *independent games*. The theorem that we need is that parallel repetition reduces the value of games in this class from v to $v^{\Omega(n)}$, where n is the number of repetitions.

As another application of block rigidity, we show conditional size-depth tradeoffs for boolean circuits, where the gates compute arbitrary functions over large sets.

[*] Department of Computer Science, Princeton University. Research supported by the Simons Collaboration on Algorithms and Geometry, by a Simons Investigator Award and by the National Science Foundation grants No. CCF-1714779, CCF-2007462.

ISSN 1433-8092

1 Introduction

We study relations between three seemingly unrelated topics: parallel repetition of multi-player games, a variant of Valiant's notion of rigidity, that we refer to as block rigidity, and proving super-linear lower bounds for Turing machines with advice.

1.1 Super-Linear Lower Bounds for Turing Machines

Deterministic multi-tape Turing machines are the standard model of computation for defining time-complexity classes. Lower bounds for the running time of such machines are known by the time-hierarchy theorem [HS65], using a diagonalization argument. Moreover, the seminal work of Paul, Pippenger, Szemerédi and Trotter gives a separation of non-deterministic linear time from deterministic linear time, for multi-tape Turing machines [PPST83] (using ideas from [HPV77] and [PR80]). That is, it shows that $\mathrm{DTIME}(n) \subsetneq \mathrm{NTIME}(n)$. This result has been slightly improved to give $\mathrm{DTIME}(n\sqrt{\log^* n}) \subsetneq \mathrm{NTIME}(n\sqrt{\log^* n})$ [San01].

However, the above mentioned lower bounds do not hold in the non-uniform setting, where the machines are allowed to use arbitrary advice depending on the length of the input. In the non-uniform setting, no super-linear lower bound is known for the running time of deterministic multi-tape Turing machines. Moreover, such bounds are not known even for a multi-output function.

1.2 Block Rigidity

The concept of matrix rigidity was introduced by Valiant as a means to prove super-linear lower bounds against circuits of logarithmic depth [Val77]. Since then, it has also found applications in communication complexity [Raz89] (see also [Wun12, Lok01]). We extend Valiant's notion of rigid matrices to the concept of *rigid functions*, and further to *block-rigid functions*. We believe that these notions are of independent interest. We note that block-rigidity is a weaker condition than rigidity and hence it may be easier to find explicit block-rigid functions. Further, our result gives a new application of rigidity.

Over a field $\mathbb{F}$, a matrix $A \in \mathbb{F}^{n\times n}$ is said to be an (r,s)-rigid matrix if it is not possible to reduce the rank of A to at most r, by changing at most s entries in each row of A. Valiant showed that if $A \in \mathbb{F}^{n\times n}$ is $(\epsilon n, n^{\epsilon})$-rigid for some constant $\epsilon > 0$, then A is not computable by a linear-circuit of logarithmic depth and linear size. As in many problems in complexity, the challenge is to find explicit rigid matrices. By explicit, we mean that a polynomial time deterministic Turing machine should be able to output a rigid matrix $A \in \mathbb{F}^{n\times n}$ on input 1^n. The best known bounds on explicit rigid matrices are far from what is needed to get super-linear circuit lower bounds (see [Fri93, SS97, SSS97, Lok00, Lok06, KLPS14, AW17, GT18, AC19, BHPT20]).

We extend the above definition to functions $f : \{0,1\}^n \to \{0,1\}^n$, by saying that f is *not* an (r,s)-rigid function if there exists a subset $X \subseteq \{0,1\}^n$ of size at least 2^{n-r}, such that over X, each output bit of f can be written as a function of some s input bits (see Definition 7). By a simple counting argument (see Proposition 8), it follows that random functions are rigid with good probability.

We further extend this definition to what we call block-rigid functions (see Definition 11). For this, we'll consider vectors $x \in \{0,1\}^{nk}$, which are thought of as composed of k blocks, each of size n. We say that a function $f : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is *not* an (r,s)-block-rigid function, if there exists a subset $X \subseteq \{0,1\}^{nk}$ of size at least 2^{nk-r}, such that over X, each *output block* of f can be written as a function of some s *input blocks*.

We conjecture that it is possible to obtain large block-rigid functions, using smaller rigid functions. For a function $f : \{0,1\}^k \to \{0,1\}^k$, we define the function $f^{\otimes n} : \{0,1\}^{nk} \to \{0,1\}^{nk}$ as follows. For each $x = (x_{ij})_{i\in[n],j\in[k]} \in \{0,1\}^{nk}$, we define $f^{\otimes n}(x)$ to be the vector obtained by applying f to $(x_{i1}, \dots, x_{ik})$, in place for each $i \in [n]$ (see Definition 13).

Conjecture 1. *There exists a universal constant $c > 0$ such that the following is true. Let $f : \{0,1\}^k \to \{0,1\}^k$ be an (r, s)-rigid function, and $n \in \mathbb{N}$. Then, $f^{\otimes n} : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is a (cnr, cs)-block-rigid function.*

We prove the following theorem. It is restated and proved as Theorem 28 in Section 5.

Theorem 2. *Let $t : \mathbb{N} \to \mathbb{N}$ be any function such that $t(n) = \omega(n)$. Assuming Conjecture 1, there exists an (explicitly given) function $f : \{0,1\}^* \to \{0,1\}^*$ such that*

1. *On inputs x of length n bits, the output $f(x)$ is of length at most n bits.*
2. *The function f is computable by a multi-tape deterministic Turing machine that runs in time $O(t(n))$ on inputs of length n.*
3. *The function f is not computable by any multi-tape deterministic Turing machine that takes advice and runs in time $O(n)$ on inputs of length n.*

More generally, we show that families of block-rigid functions cannot be computed by non-uniform Turing machines running in linear-time. This makes it interesting to find such families that are computable in polynomial time. The following theorem is restated as Theorem 29.

Theorem 3. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$ and $k(n) = 2^{o(n)}$, and $f_n : \{0,1\}^{nk(n)} \to \{0,1\}^{nk(n)}$ be a family of $(\epsilon nk(n), \epsilon k(n))$-block-rigid functions, for some constant $\epsilon > 0$. Let M be any multi-tape deterministic linear-time Turing machine that takes advice. Then, there exists $n \in \mathbb{N}$, and $x \in \{0,1\}^{nk(n)}$, such that $M(x) \neq f_n(x)$.*

As another application, based on Conjecture 1, we show size-depth tradeoffs for boolean circuits, where the gates compute arbitrary functions over large (with respect to the input size) sets (see Section 6).

1.3 Parallel Repetition

In a k-player game $\mathcal{G}$, questions $(x_1, \dots, x_k)$ are chosen from some joint distribution μ. For each $j \in [k]$, player j is given x_j and gives an answer a_j that depends only on x_j. The players are said to win if their answers satisfy a fixed predicate $V(x_1, \dots, x_k, a_1, \dots, a_k)$. We note that V might be randomized, that is, it might depend on some random string that is sampled independently of $(x_1, \dots, x_k)$. The value of the game $\text{val}(\mathcal{G})$ is defined to be the maximum winning probability over the possible strategies of the players.

It is natural to consider the parallel repetition $\mathcal{G}^{\otimes n}$ of such a game $\mathcal{G}$. Now, the questions $(x_1^{(i)}, \dots, x_k^{(i)})$ are chosen from μ, independently for each $i \in [n]$. For each $j \in [k]$, player j is given $(x_j^{(1)}, \dots, x_j^{(n)})$ and gives answers $(a_j^{(1)}, \dots, a_j^{(n)})$. The players are said to win if the answers satisfy $V(x_1^{(i)}, \dots, x_k^{(i)}, a_1^{(i)}, \dots, a_k^{(i)})$ for every $i \in [n]$. The value of the game $\text{val}(\mathcal{G}^{\otimes n})$ is defined to be the maximum winning probability over the possible strategies of the players. Note that the players are allowed to correlate their answers to different repetitions of the game.

Parallel repetition of games was first studied in [FRS94], owing to its relation with multiprover interactive proofs [BOGKW88]. It was hoped that the value $\mathrm{val}(\mathcal{G}^{\otimes n})$ of the repeated game goes down as $\mathrm{val}(\mathcal{G})^n$. However, this is not the case, as shown in [For89, Fei91, FV02, Raz11].

A lot is known about parallel repetition of 2-player games. The, so called, parallel repetition theorem, first proved by Raz [Raz98] and further simplified and improved by Holenstein [Hol09], shows that if $\mathrm{val}(\mathcal{G}) < 1$, then $\mathrm{val}(\mathcal{G}^{\otimes n}) \leq 2^{-\Omega(n/s)}$, where s is the length of the answers given by the players. The bounds in this theorem were later made tight even for the case when the initial game has small value (see [DS14] and [BG15]).

Much less is known for k-player games with $k \geq 3$. Verbitsky [Ver96] showed that if $\mathrm{val}(G) < 1$, then the value of the the repeated game goes down to zero as n grows larger. The result shows a very weak rate of decay, approximately equal to $\frac{1}{\alpha(n)}$, where α is the inverse-Ackermann function, owing to the use of the density Hales-Jewett theorem (see [FK91] and [Pol12]). A recent result by Dinur, Harsha, Venkat and Yuen [DHVY17] shows exponential decay, but only in the special case of what they call *expanding games*. This approach fails when the input to the players have strong correlations.

In this paper (see Section 4), we show that a sufficiently strong parallel repetition theorem for multiplayer games implies Conjecture 1. The following theorem is proved formally as Theorem 19 in Section 4.

Theorem 4. *There exists a family $\{\mathcal{G}_{\mathcal{S},k}\}$ of k-player games (where $\mathcal{S}$ is some parameter), such that a strong parallel repetition theorem for all games in $\{\mathcal{G}_{\mathcal{S},k}\}$ implies Conjecture 1.*

Although the games in this family do not fit into the framework of [DHVY17], they satisfy some very special properties. Every k-player game in the family satisfies the following:

1. The questions to the k-players are chosen as follows: First, k bits, $x_1, \ldots, x_k \in \{0, 1\}$, are drawn uniformly and independently. Each of the k-players sees some subset of these k-bits.

2. The predicate V satisfies the condition that on fixing the bits $x_1, \ldots, x_k$, there is a unique accepting answer for each player (independently of all other answers) and the verifier accepts if every player answers with the accepting answer. We refer to games that satisfy this property as *independent games*.

We believe that these properties may allow us to prove strong upper bounds on the value of parallel repetition of such games, despite our lack of understanding of multiplayer parallel repetition. The bounds that we need are that parallel repetition reduces the value of such games from v to $v^{\Omega(n)}$, where n is the number of repetitions (as is proved in [DS14] and [BG15] for 2-player games).

1.4 Open Problems

1. The main open problem is to make progress towards proving Conjecture 1, possibly using the framework of parallel repetition. The remarks after Theorem 28 mention some weaker statements that suffice for our applications. The examples of matrix-transpose and matrix-product in Section 8 also serve as interesting problems.

2. Our techniques, which are based on [PPST83], heavily exploit the fact that the Turing machines have one-dimensional tapes. Time-space lower bounds for satisfiability in

the case of multi-tape Turing machines with random access [FLvMV05], and Turing machines with one d-dimensional tape [vMR05], are known. Extending such results to the non-uniform setting is an interesting open problem.

3. The question of whether a rigid-matrix $A \in \mathbb{F}_2^{n\times n}$ is rigid when seen as a function $A : \{0,1\}^n \to \{0,1\}^n$ is very interesting (see Section 7). This question is closely related to a Conjecture of Jukna and Schnitger [JS11] on the linearization of depth-2 circuits. This is also related to the question of whether data structures for linear problems can be optimally linearized (see [DGW19]). We note that there are known examples of linear problems for which the best known data-structures are non-linear, without any known linear data-structure achieving the same bounds (see [KU11]).

2 Preliminaries

Let $\mathbb{N} = \{1,2,3,\ldots\}$ be the set of all natural numbers. For any $k \in \mathbb{N}$, we use $[k]$ to denote the set $\{1,2,\ldots,k\}$.

We use $\mathbb{F}$ to denote an arbitrary finite field, and $\mathbb{F}_2$ to denote the finite field on two elements.

Let $x \in \{0,1\}^k$. For $i \in [k]$, we use x_i to denote the i^{th} coordinate of x. For $S \subseteq [k]$, we denote by $x|_S$ the vector $(x_i)_{i\in S}$, which is the restriction of x to coordinates in S.

We also consider vectors $x \in \{0,1\}^{nk}$, for some $n \in \mathbb{N}$. We think of these as composed of k blocks, each consisting of a vector in $\{0,1\}^n$. That is, $x = (x_{ij})_{i\in[n],j\in[k]}$. By abuse of notation, for $S \subseteq [k]$, we denote by $x|_S$ the vector $(x_{ij})_{i\in[n],j\in S}$, which is the restriction of x to the blocks indexed by S.

Let $A \in \mathbb{F}^{nk\times nk}$ be an $nk \times nk$ matrix. We think of A as a block-matrix consisting of k^2 blocks, each block being an $n \times n$ matrix. That is, $A = (A_{ij})_{i,j\in[k]}$, where for all $i,j \in [k]$, $A_{ij} \in \mathbb{F}^{n\times n}$. For each $i \in [k]$, we call $(A_{ij})_{j\in[k]}$ the i^{th} block-row of A.

For every $n \in \mathbb{N}$, we define $\log^* n = \min\{\ell \in \mathbb{N} \cup \{0\} : \underbrace{\log_2 \log_2 \ldots \log_2}_{\ell \text{ times}} n \le 1\}$.

3 Rigidity and Block Rigidity

3.1 Rigidity

The concept of matrix rigidity was introduced by Valiant [Val77]. It is defined as follows.

Definition 5. *A matrix $A \in \mathbb{F}^{n\times n}$ is said to be an (r,s)-rigid matrix if it cannot be written as $A = B + C$, where B has rank at most r, and C has at most s non-zero entries in each row.*

Valiant [Val77] showed the existence of rigid matrices by a simple counting argument. For the sake of completeness, we include this proof.

Proposition 6. *For any constant $0 < \epsilon \le \frac{1}{8}$, and any $n \in \mathbb{N}$, there exists a matrix $A \in \mathbb{F}^{n\times n}$ that is an $(\epsilon n, \epsilon n)$-rigid matrix.*

Proof. Fix any $0 < \epsilon \le \frac{1}{8}$. We bound the number of $n \times n$ matrices that are not $(\epsilon n, \epsilon n)$-rigid matrices.

1. Any $n \times n$ matrix with rank at most r can be written as the product of an $n \times r$ and an $r \times n$ matrix. Hence, the number of matrices of rank at most ϵn is at most $|\mathbb{F}|^{2\epsilon n^2} \le |\mathbb{F}|^{\frac{n^2}{4}}$.

2. The number of matrices that have at most ϵn non-zero entries in each row is at most
$$\left(\binom{n}{\epsilon n}|\mathbb{F}|^{\epsilon n}\right)^n \le \left(\frac{e}{\epsilon}\right)^{\epsilon n^2}|\mathbb{F}|^{\epsilon n^2} = |\mathbb{F}|^{\epsilon n^2(1+\log_{|\mathbb{F}|}\frac{e}{\epsilon})} < |\mathbb{F}|^{\frac{3n^2}{4}}.$$
We used the binomial estimate $\binom{n}{r} \le \left(\frac{en}{r}\right)^r$.

Since each matrix that is not an (r, s)-rigid matrix can be written as the sum of a matrix with rank at most r, and a matrix with at most s non-zero entries in each row, the number of matrices that are not $(\epsilon n, \epsilon n)$-rigid matrices is strictly less than $|\mathbb{F}|^{\frac{n^2}{4}} \cdot |\mathbb{F}|^{\frac{3n^2}{4}} = |\mathbb{F}|^{n^2}$, which is the total number of $n \times n$ matrices. □

Observe that a matrix $A \in \mathbb{F}_2^{n\times n}$ is not an (r, s)-rigid matrix if and only if there is a subspace $X \subseteq \mathbb{F}_2^n$ of dimension at least $n-r$, and a matrix C with at most s non-zero entries in each row, such that $Ax = Cx$ for all $x \in X$.

We use this formulation to extend the concept of rigidity to general functions.

Definition 7. *A function $f : \{0,1\}^n \to \{0,1\}^n$ is said to be an (r, s)-rigid function if for every subset $X \subseteq \{0,1\}^n$ of size at least 2^{n-r}, and subsets $S_1, \ldots, S_n \subseteq [n]$ of size s, and functions $g_1, \ldots, g_n : \{0,1\}^s \to \{0,1\}$, there exists $x \in X$ such that $f(x) \neq (g_1(x|_{S_1}), g_2(x|_{S_2}), \ldots, g_n(x|_{S_n}))$.*

Using a similar counting argument as in Proposition 6, we show the existence of rigid functions.

Proposition 8. *For any constant $0 < \epsilon \le \frac{1}{8}$, and any (large enough) integer n, there exists a function $f : \{0,1\}^n \to \{0,1\}^n$ that is an $(\epsilon n, \epsilon n)$-rigid function.*

Proof. Fix any constant $0 < \epsilon \le \frac{1}{8}$, and any integer $n \ge \frac{2}{\epsilon}$. We count the number of functions $f : \{0,1\}^n \to \{0,1\}^n$ that are not $(\epsilon n, \epsilon n)$-rigid functions.

1. Note that in Definition 7, it is without loss of generality to assume that $|X| = 2^{n-\epsilon n}$. The number subsets $X \subseteq \{0,1\}^n$ of size $2^{n-\epsilon n}$ is $\binom{2^n}{2^{n-\epsilon n}} \le (e2^{\epsilon n})^{2^{n-\epsilon n}} < 2^{2\epsilon n 2^{n-\epsilon n}} \le 2^{\frac{n}{4}2^{n-\epsilon n}}$.

2. The number of subsets $S_1, \ldots, S_n \subseteq [n]$ of size ϵn is at most $\binom{n}{\epsilon n}^n < n^{\epsilon n^2} < 2^{\frac{n}{4}2^{n-\epsilon n}}$.

3. The number of functions $g_1, \ldots, g_n : \{0,1\}^{\epsilon n} \to \{0,1\}$ is at most $2^{n2^{\epsilon n}} < 2^{\frac{n}{4}2^{n-\epsilon n}}$.

4. The number of choices for values of f on $\{0,1\}^n \setminus X$ is at most $2^{n(2^n-|X|)} \le 2^{n(2^n-2^{n-\epsilon n})}$.

Hence, the total number of functions $f : \{0,1\}^n \to \{0,1\}^n$ that are not $(\epsilon n, \epsilon n)$-rigid functions is strictly less than $\left(2^{\frac{n}{4}2^{n-\epsilon n}}\right)^3 2^{n2^n-n2^{n-\epsilon n}} < 2^{n2^n}$. □

3.2 Block Rigidity

In this section, we introduce the notion of block rigidity.

Definition 9. *A matrix $A \in \mathbb{F}^{nk \times nk}$ is said to be an (r,s)-block-rigid matrix if it cannot be written as $A = B + C$, where B has rank at most r, and C has at most s non-zero matrices in each block-row.*

Observe that if $A \in \mathbb{F}^{nk \times nk}$ is an (r, ns)-rigid matrix, then it is also (r,s)-block-rigid matrix. Combining this with Proposition 6, we get the following.

Observation 10. *For any constant $0 < \epsilon \leq \frac{1}{8}$, and positive integers n, k, there exists an $(\epsilon nk, \epsilon k)$-block-rigid matrix $A \in \mathbb{F}^{nk \times nk}$.*

Following the definition of rigid-functions in Section 3.1, we define block-rigid functions as follows.

Definition 11. *A function $f : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is said to be an (r,s)-block-rigid function if for every subset $X \subseteq \{0,1\}^{nk}$ of size at least 2^{nk-r}, and subsets $S_1, \ldots, S_k \subseteq [k]$ of size s, and functions $g_1, \ldots, g_k : \{0,1\}^{ns} \to \{0,1\}^n$, there exists $x \in X$ such that $f(x) \neq (g_1(x|_{S_1}), g_2(x|_{S_2}), \ldots, g_k(x|_{S_k}))$.*

Observe that if $f : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is an (r, ns)-rigid function, then it is also (r,s)-block-rigid function. Combining this with Proposition 8, we get the following.

Observation 12. *For any constant $0 < \epsilon \leq \frac{1}{8}$, and (large enough) integers n, k, there exists an $(\epsilon nk, \epsilon k)$-block-rigid function $f : \{0,1\}^{nk} \to \{0,1\}^{nk}$.*

Note that $n = 1$ in the definition of block-rigid matrices (functions) gives the usual definition of rigid matrices (functions). For our applications, we will mostly be interested in the case when n is much larger than k.

3.3 Rigidity Amplification

A natural question is whether there is a way to amplify rigidity. That is, given a rigid matrix (function), is there a way to obtain a larger matrix (function) which is rigid, or even block-rigid.

Definition 13. *Let $f : \{0,1\}^k \to \{0,1\}^k$ be any function. Define $f^{\otimes n} : \{0,1\}^{nk} \to \{0,1\}^{nk}$ as following. Let $x = (x_{ij})_{i \in [n], j \in [k]} \in \{0,1\}^{nk}$ and $i \in [n], j \in [k]$. The $(i,j)^{th}$ coordinate of $f^{\otimes n}(x)$ is defined to be the j^{th} coordinate of $f(x_{i1}, x_{i2}, \ldots, x_{ik})$.*

Basically, applying $f^{\otimes n}$ on $x \in \{0,1\}^{nk}$ is the same as applying f on $(x_{i1}, x_{i2}, \ldots, x_{ik})$, in place for each $i \in [n]$. For a linear function given by matrix $A \in \mathbb{F}_2^{k \times k}$, this operation corresponds to $A \otimes I_n$, where I_n is the $n \times n$ identity matrix, and $\otimes$ denotes the Kronecker product of matrices.

It is easy to see that if f is not rigid, then $f^{\otimes n}$ is not block-rigid.

Observation 14. *Suppose $f : \{0,1\}^k \to \{0,1\}^k$ is not an (r,s)-rigid function. Then $f^{\otimes n}$ is not an (nr, s)-block-rigid function.*

The converse of Observation 14 is more interesting. We believe that it is true, and restate Conjecture 1 below.

Conjecture 15. *There exists a universal constant $c > 0$ such that the following is true. Let $f : \{0,1\}^k \to \{0,1\}^k$ be an (r,s)-rigid function, and $n \in \mathbb{N}$. Then, $f^{\otimes n} : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is a (cnr, cs)-block-rigid function.*

4 Parallel Repetition

In this section, we show an approach to prove Conjecture 15 regarding rigidity amplification. This is based on proving a strong parallel repetition theorem for a k-player game.

Fix some $k \in \mathbb{N}$, a function $f : \{0,1\}^k \to \{0,1\}^k$, an integer $1 \leq s < k$, and $\mathcal{S} = (S_1, \dots, S_k)$, where each $S_i \subseteq [k]$ is of size s. We define a k-player game $\mathcal{G}_{\mathcal{S}}$ as follows:

The k-players choose functions $g_1, \dots, g_k : \{0,1\}^s \to \{0,1\}$, which we call a strategy. A verifier chooses $x_1, \dots, x_k \in \{0,1\}$ uniformly and independently. Let $x = (x_1, \dots, x_k) \in \{0,1\}^k$. For each $j \in [k]$, Player j is given the input $x|_{S_j}$, and they answer $a_j = g_j(x|_{S_j}) \in \{0,1\}$. The verifier accepts if and only if $f(x) = (a_1, \dots, a_k)$. The goal of the players is to maximize the winning probability. Formally, the value of the game is defined as

$$\mathrm{val}(\mathcal{G}_{\mathcal{S}}) := \max_{g_1,\dots,g_k} \Pr_{x_1,\dots,x_k \in \{0,1\}} \left[f(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k})) \right].$$

The n-fold repetition of $\mathcal{G}_{\mathcal{S}}$, denoted by $\mathcal{G}_{\mathcal{S}}^{\otimes n}$ is defined as follows. The players choose a strategy $g_1, \dots, g_k : \{0,1\}^{ns} \to \{0,1\}^n$. The verifier chooses $x_1, \dots, x_k \in \{0,1\}^n$ uniformly and independently. Let $x = (x_1, \dots, x_k) \in \{0,1\}^{nk}$. Player j is given the input $x|_{S_j}$, and they answer $a_j = g_j(x|_{S_j}) \in \{0,1\}^n$. The verifier accepts if and only if $f^{\otimes n}(x) = (a_1, \dots, a_k)$. That is, for each $i \in [n]$, $j \in [k]$, the j^{th} bit of $f(x_{i1}, \dots, x_{ik})$ equals the i^{th} bit of a_j. The value of this repeated game is

$$\mathrm{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) := \max_{g_1,\dots,g_k} \Pr_{x_1,\dots,x_k \in \{0,1\}^n} \left[f^{\otimes n}(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k})) \right].$$

From Definition 11, we get the following:

Observation 16. *Let $f : \{0,1\}^k \to \{0,1\}^k$ be a function, and $n \in \mathbb{N}$. Then, $f^{\otimes n}$ is an (r,s)-block-rigid function if and only if for every $\mathcal{S} = (S_1, \dots, S_k)$ with set sizes as s, val$(\mathcal{G}_{\mathcal{S}}^{\otimes n}) < 2^{-r}$.*

Proof. Let $f^{\otimes n}$ be an (r,s)-block-rigid function. Suppose, for the sake of contradiction, that $\mathcal{S} = (S_1, \dots, S_k)$ is such that $\mathrm{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) \geq 2^{-r}$. Let the functions $g_1, \dots, g_k : \{0,1\}^{ns} \to \{0,1\}^n$ be an optimal strategy for the players. Define $X := \left\{ x \in \{0,1\}^{nk} \mid f^{\otimes n}(x) = (g_1(x|_{S_1}), \dots, g_k(x|_{S_k})) \right\}$. Then, $|X| = \mathrm{val}(\mathcal{G}_{\mathcal{S}}) \cdot 2^{nk} \geq 2^{nk-r}$, which contradicts the block rigidity of $f^{\otimes n}$.

Conversely, suppose that $f^{\otimes n}$ is not (r,s)-block-rigid. Then, there exists $X \subseteq \{0,1\}^{nk}$ with $|X| \geq 2^{nk-r}$, subsets $S_1, \dots, S_k \subseteq [k]$ of size s, and functions $g_1, \dots, g_k : \{0,1\}^{ns} \to \{0,1\}^n$, such that for all $x \in X$, $f^{\otimes n}(x) = (g(x|_{S_1}), \dots, g(x|_{S_k}))$. Let $\mathcal{S} = (S_1, \dots, S_k)$, and suppose the players use strategy $g_1, \dots, g_k$. Then, $\mathrm{val}(\mathcal{G}_{\mathcal{S}}^{\otimes n}) \geq |X| \cdot 2^{-nk} \geq 2^{-r}$. □

In particular, for $n = 1$, Observation 16 gives the following:

Observation 17. *A function $f : \{0,1\}^k \to \{0,1\}^k$ is an (r,s)-rigid-function if and only if for every $\mathcal{S} = (S_1, \dots, S_k)$ with set sizes as s, val$(\mathcal{G}_{\mathcal{S}}) < 2^{-r}$.*

We conjecture the following strong parallel repetition theorem.

Conjecture 18. *There exists a constant $c > 0$ such that the following is true. Let $f : \{0,1\}^k \to \{0,1\}^k$ be any function, and $\mathcal{S} = (S_1, \dots, S_k)$ be such that for each $i \in [k]$, $S_i \subseteq [k]$ is of size s. Then, for all $n \in \mathbb{N}$, val$(\mathcal{G}_{\mathcal{S}}^{\otimes n}) \leq (val(\mathcal{G}_{\mathcal{S}}))^{cn}$.*

Combining Observation 16 and 17, we get the following:

Theorem 19. *Conjecture 18 $\implies$ Conjecture 15.*

Remarks. *(i) By looking only at some particular player, it can be shown that if $val(\mathcal{G}_{\mathcal{S}}) < 1$, then $val(\mathcal{G}_{\mathcal{S}}^{\otimes n}) \leq 2^{-\Omega(n)}$. In fact, such a result holds for all* independent games*. The harder part seems to be showing strong parallel repetition when the initial game has small value.*

(ii) Observe that the game $\mathcal{G}_{\mathcal{S}}$ has a randomized predicate in the case $\cup_{j=1}^{k} S_j \neq [k]$. This condition can be removed (even for general independent games) by introducing a new player. This player is given the random string used by the verifier, and is always required to answer a single bit equal to zero. This maintains the independent game property, and ensures that the predicate used by the verifier is a deterministic function of the vector of input queries to the players.

5 Turing Machine Lower Bounds

In this section, we show a conditional super-linear lower bound for multi-tape deterministic Turing machines that can take advice.

Without loss of generality, we only consider machines that have a separate read-only input tape. We assume that the advice string, which is a function of the input length, is written on a separate advice tape at the beginning of computation. We are interested in machines that compute multi-output functions. For this, we assume that at the end of computation, the machine writes the entire output on a separate write-only output tape, and then halts.

We consider the following problem.

Definition 20. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function. We define the problem Tensor$_k$ as follows:*

- Input: *(f, x), where $f : \{0,1\}^k \to \{0,1\}^k$ is a function, and $x \in \{0,1\}^{nk}$, for some $n \in \mathbb{N}$ and $k = k(n)$.*
- Output: $f^{\otimes n}(x) \in \{0,1\}^{nk}$.

The function f is given as input in the form of its entire truth table. The input $x = (x_{ij})_{i\in[n],j\in[k]}$ is given in the order $(x_{11}, \ldots, x_{n1}, x_{12}, \ldots, x_{n2}, \ldots, x_{1k}, \ldots, x_{nk})$. The total length of the input is $m(n) := 2^k k + nk$.

We observe that if the function $k : \mathbb{N} \to \mathbb{N}$ grows very slowly with n, the problem $\mathrm{Tensor_k}$ can be solved by a deterministic Turing machine in slightly super-linear time.

Observation 21. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function. There exists a deterministic Turing machine that solves the problem Tensor$_k$ in time $O(nk2^k)$, on input (f, x) of length $nk + 2^k k$, where $k = k(n)$.*

Proof. We note that applying $f^{\otimes n}$ on $x = (x_{ij})_{i\in[n],j\in[k]}$ is the same as applying f on $(x_{i1}, x_{i2}, \ldots, x_{ik})$, in place for each $i \in [n]$. A Turing machine can do the following:

1. Find n and k, using the fact that the description of f is of length $2^k k$, and that of x is of length nk.
2. Rearrange the input so that for each $i \in [n]$, the part $(x_{i1}, x_{i2}, \ldots, x_{ik})$ is written consecutively on the tape.

3. Using the truth table of f, compute the output for each such part.

4. Rearrange the entire output back to the desired form.

The total time taken is $O(nk + nk^2 + nk2^k + nk^2) = O(nk2^k)$. □

We now state and prove the main technical theorem of this section.

Theorem 22. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$ and $k(n) = o(\log_2 n)$. Let $m = m(n) := 2^k k + nk$, where $k = k(n)$.*

Suppose M is a deterministic multi-tape Turing machine that takes advice, and runs in linear time in the length of its input. Assuming Conjecture 15, the machine M does not solve the problem Tensor$_k$ correctly for all inputs. That is, there exists $n \in \mathbb{N}$, and $y = (f, x) \in \{0,1\}^m$ such that $M(y) \neq f^{\otimes n}(x)$.

There are two main technical ideas that will be useful to us. The first is the notion of block-respecting Turing machines, defined by Hopcroft, Paul and Valiant [HPV77]. The second is a graph theoretic result, which was proven by Paul, Pippenger, Szemerédi and Trotter [PPST83], and was used to show a separation between deterministic and non-deterministic linear time.

Definition 23. *Let M be a Turing machine, and let $b : \mathbb{N} \to \mathbb{N}$ be a function. Partition the computation of M, on any input y of length m, into time* segments *of length $b(m)$, with the last segment having length at most $b(m)$. Also, partition each of the tapes of M into* blocks, *each consisting of $b(m)$ contiguous cells.*

We say that M is block-respecting with respect to block size b, if on inputs of length m, the tape heads of M cross blocks only at times that are integer multiples of $b(m)$.

Lemma 24. *[HPV77] Let $t : \mathbb{N} \to \mathbb{N}$ be a function, and M be a multi-tape deterministic Turing machine running in time $t(m)$ on inputs of length m. Let $b : \mathbb{N} \to \mathbb{N}$ be a function such that $b(m)$ is computable from 1^m by a multi-tape deterministic Turing machine running in time $O(t(m))$. Then, the language recognized by M is also recognized by a multi-tape deterministic Turing Machine M', which runs in time $O(t(m))$, and is block-respecting with respect to b.*

The rest of this section is devoted to the proof of Theorem 22. Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$ and $k(n) = o(\log_2 n)$. Let $m = m(n) := 2^k k + nk$, where $k = k(n)$.

Suppose that M is a deterministic multi-tape Turing Machine, which on input $y = (f, x) \in \{0,1\}^m$, takes advice, runs in time $O(m)$, and outputs $f^{\otimes n}(x)$.

Let $b : \mathbb{N} \to \mathbb{N}$ be a function such that $b(m) = n$. By our assumption that $k(n) = o(\log_2 n)$, we can assume that the input $y = (f, x) \in \{0,1\}^m$ consists of $k + 1$ blocks, where the first block contains f (possibly padded by blank symbols to the left), and the remaining k blocks contain x. By Lemma 24, we can further assume that M is block-respecting with respect to b. Note that we can assume n to be a part of the advice, and hence we don't need to care about the computability of b.

For an input $y = (f, x) \in \{0,1\}^m$, the number of time segments for which M runs on x is at most $\frac{O(m)}{b(m)} = \frac{O(nk)}{n} = O(k) := a(n)$.

We define the computation graph $G_M(y)$, for input $y \in \{0,1\}^m$, as follows.

Definition 25. *The vertex set of $G_M(y)$ is defined to be $V_M(y) = \{1, \ldots, a(n)\}$. For each $1 \le i < j \le a(n)$, the edge set $E_M(y)$ has the edge (i, j), if either* *(i) $j = i + 1$, or*

(ii) there is some tape, such that, during the computation on y, M visits the same block on that tape in both time-segments i and j, and never visits that block during any time-segment strictly between i and j.

In a directed acyclic graph G, we say that a vertex u is a predecessor of a vertex v, if there exists a directed path from u to v.

Lemma 26. *[PPST83] For every y, the graph $G_M(y)$ satisfies the following:*

1. *Each vertex in $G_M(y)$ has degree $O(1)$.*

2. *There exists a set of vertices $J \subset V_M(y)$ in $G_M(y)$, of size $O\left(\frac{a(n)}{\log^* a(n)}\right)$ such that every vertex of $G_M(y)$ has at most $O\left(\frac{a(n)}{\log^* a(n)}\right)$ many predecessors in the induced subgraph on the vertex set $V_M(y) \setminus J$.*

We note that the constants here might depend on the number of tapes of M.

Lemma 27. *Let $\epsilon > 0$ be any constant and $\mathcal{Y} \subseteq \{0,1\}^m$ be any subset of the inputs. For all (large enough) n, there exists a subset $Y \subseteq \mathcal{Y}$ of size $|Y| \geq |\mathcal{Y}| \cdot 2^{-\epsilon nk}$, and subsets $S_1, \ldots, S_k \subseteq [k]$ of size ϵk, such that for each $y = (f, x) \in Y$, and each $i \in [k]$, the i^{th} block (of length n) of $f^{\otimes n}(x)$ can be written as a function of $x|_{S_i}$ and the truth-table of f.*

Proof. For input $y = (f, x) \in \{0,1\}^m$, let $J(y) \subset V_M(y)$ be a set as in Lemma 26.

Let $C(y)$ denote the following information about the computation of M:

(i) The internal state of M at the end of each time-segment.

(ii) The position of all tape heads at the end of each time-segment.

(iii) For each time segment in $J(y)$, and for each tape of M, the final transcription (of length n) of the block that was visited on this tape during this segment.

Let $g : \mathcal{Y} \to \{0,1\}^*$ be the function given by $g(y) = (G_M(y), J(y), C(y))$. Observe that the output of g can be described using $O\left(k \log_2 k + \frac{nk}{\log^* k}\right)$ bits. By our assumption that $k(n) = \omega(1)$ and $k(n) = o(\log_2 n)$, we have that for large n, this is at most ϵnk bits. Hence, there exists a set $Y \subseteq \mathcal{Y}$ of size $|Y| \geq |\mathcal{Y}| \cdot 2^{-\epsilon nk}$, such that for each $y \in Y$, $g(y)$ takes on some fixed value $(G = (V, E), J, C)$.

Now, consider any $y = (f, x) \in Y$. The machine writes the k blocks of the output $f^{\otimes n}(x)$ on the output tape in the last k time segments before halting. For each of these time segments, the corresponding vertex in G has at most $O\left(\frac{k}{\log^* k}\right) \leq \epsilon k$ predecessors in the induced subgraph on $V \setminus J$. These further correspond to at most ϵk distinct blocks of y that are visited (on the input tape) during these predecessor time segments. Since the relevant block transcriptions at the end of time segments for vertices in J are fixed in C, each output block can be written as a function of at most ϵk blocks of y. For the i^{th} block of output, without loss of generality, this includes the first block of y, which contains the truth table of f, and blocks of x which indexed by some subset $S_i \subseteq [k]$ of size ϵk.

□

Proof of Theorem 22. Let $\delta = \frac{1}{8}$. Fix some sufficiently large n, and a $(\delta k, \delta k)$-rigid function $f_0 : \{0,1\}^k \to \{0,1\}^k$. The existence of such a function is guaranteed by Proposition 8. By Conjecture 15, the function $f_0^{\otimes n}$ is an $(\epsilon nk, \epsilon k)$-block-rigid function for some constant $\epsilon > 0$. On the other hand, Lemma 27, with $\mathcal{Y} = \left\{(f_0, x) : x \in \{0,1\}^{nk}\right\}$, shows that $f_0^{\otimes n}$ is not an $(\epsilon nk, \epsilon k)$-block-rigid function for any constant $\epsilon > 0$. □

We now restate and prove Theorem 2.

Theorem 28. *Let $t : \mathbb{N} \to \mathbb{N}$ be any function such that $t(n) = \omega(n)$. Assuming Conjecture 15, there exists a function $f : \{0,1\}^* \to \{0,1\}^*$ such that*

1. *On inputs x of length n bits, the output $f(x)$ is of length at most n bits.*

2. *The function f is computable by a multi-tape deterministic Turing machine that runs in time $O(t(n))$ on inputs of length n.*

3. *The function f is not computable by any multi-tape deterministic Turing machine that takes advice and runs in time $O(n)$ on inputs of length n.*

Proof. Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$, $k(n) = o(\log_2 n)$, and $nk2^k \leq t(2^k k + nk)$. The theorem then follows from Observation 21 and Theorem 22. □

Remarks. *(i) We note that for the proof of Theorem 22, it suffices to find, for infinitely many n, a single function $f : \{0,1\}^{k(n)} \to \{0,1\}^{k(n)}$ such that $f^{\otimes n}$ is an $(\epsilon nk, \epsilon k)$-block-rigid function, where $\epsilon > 0$ is a constant. This would show that M cannot give the correct answer to Tensor_k for inputs of the form (f, x), where $x \in \{0,1\}^{nk}$.*

(ii) For the proof of Theorem 22, it can be shown that it suffices for the following condition to hold for infinitely many n, and some constant $\epsilon > 0$. Let $S_1, \ldots, S_k \subseteq [k]$ be fixed sets of size ϵk, and $f : \{0,1\}^k \to \{0,1\}^k$ be a function chosen uniformly at random. Then, with probability at least $1 - 2^{-\omega(k \log_2 k)}$, the function $f^{\otimes n}$ is an $(\epsilon nk, \epsilon k)$-block-rigid function against the fixed sets $S_1, \ldots, S_k$. We note that the probability here is not good enough to be able to union bound over $S_1, \ldots, S_k$ and get a single function as mentioned in the previous remark.

Essentially the same argument as that of Theorem 22 also proves Theorem 3, which we restate below.

Theorem 29. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$ and $k(n) = 2^{o(n)}$, and $f_n : \{0,1\}^{nk(n)} \to \{0,1\}^{nk(n)}$ be a family of $(\epsilon nk(n), \epsilon k(n))$-block-rigid functions, for some constant $\epsilon > 0$. Let M be any multi-tape deterministic linear-time Turing machine that takes advice. Then, there exists $n \in \mathbb{N}$, and $x \in \{0,1\}^{nk(n)}$, such that $M(x) \neq f_n(x)$.*

The above theorem makes it interesting to find families of block-rigid functions that are computable in polynomial time.

6 Size-Depth Tradeoffs

In this section, we will consider boolean circuits over a set F. These are directed acyclic graphs with each node v labelled either as an input node or by an arbitrary function $g_v : F \times F \to F$. The input nodes have in-degree 0 and all other nodes have in-degree 2. Some nodes are further labelled as output nodes, and they compute the outputs (in the usual manner), when the inputs are from the set F. The size of the circuit is defined to be the number of edges in the graph. The depth of the circuit is defined to be the length of a longest directed path from an input node to an output node.

Valiant [Val77] showed that if $A \in \mathbb{F}^{n\times n}$ is an $(\epsilon n, n^{\epsilon})$-rigid matrix for some constant $\epsilon > 0$, then the corresponding function cannot be computed by an $O(n)$-size and $O(\log_2 n)$-depth linear circuit over $\mathbb{F}$. By a linear circuit, we mean that each gate computes a linear function (over $\mathbb{F}$) of its inputs. A similar argument can be used to prove the following.

Lemma 30. *[Val77] Suppose $f : \{0,1\}^{nk} \to \{0,1\}^{nk}$ is an $(\epsilon nk, k^{\epsilon})$-block-rigid function, for some constant $\epsilon > 0$. Then, the function $g : (\{0,1\}^n)^k \to (\{0,1\}^n)^k$ corresponding to f cannot be computed by an $O(k)$-size and $O(\log_2 k)$-depth circuit over the set $F = \{0,1\}^n$.*

Theorem 31. *Let $k : \mathbb{N} \to \mathbb{N}$ be a function such that $k(n) = \omega(1)$ and $k(n) = o(\log_2 n)$. Let $m = m(n) := 2^k k + nk$, where $k = k(n)$.*

Assuming Conjecture 15, the problem Tensor_k is not solvable by $O(k)$-size and $O(\log_2 k)$-depth circuits over the set $F = \{0,1\}^n$. Here, the input (f, x) to the circuit is given in the form of $k+1$ elements in $\{0,1\}^n$, the first one being the truth table of f, and the remaining k being the blocks of x.

Proof. Let $\delta = \frac{1}{8}$. Fix some large n, and a $(\delta k, \delta k)$-rigid function $f_0 : \{0,1\}^k \to \{0,1\}^k$, where $k = k(n)$. The existence of such a function is guaranteed by Proposition 8. Assuming Conjecture 15, the function $f_0^{\otimes n}$ is an $(\epsilon nk, \epsilon k)$-block-rigid function, for some universal constant $\epsilon > 0$. By Lemma 30, the corresponding function on $(\{0,1\}^n)^k$ cannot be computed by an $O(k)$-size and $O(\log_2 k)$-depth circuit over $F = \{0,1\}^n$. Since f_0 can be hard-wired in any circuit solving $\mathrm{Tensor_k}$, we have the desired result. □

7 Rigid Matrices and Rigid Functions

A natural question to ask is whether the functions corresponding to rigid matrices are rigid functions or not.

Conjecture 32. *There exists a universal constant $c > 0$ such that whenever $A \in \mathbb{F}_2^{n\times n}$ is an (r, s)-rigid matrix, the corresponding function $A : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a (cr, cs)-rigid function.*

We show that a positive answer to the above resolves a closely related conjecture by Jukna and Schnitger [JS11].

Definition 33. *Consider a depth-2 circuit, with $x = (x_1, \ldots, x_n)$ as the input variables, w gates in the middle layer, computing boolean functions $h_1, \ldots, h_w$ and m output gates, computing boolean functions $g_1, \ldots, g_m$. The circuit computes a function $f = (f_1, \ldots, f_m) : \mathbb{F}_2^n \to \mathbb{F}_2^m$ satisfying $f_i(x_1 \ldots, x_n) = g_i(x, h_1(x), \ldots, h_w(x))$, for each $i \in [m]$. The width of the circuit is defined to be w. The degree of the circuit is defined to be the maximum over all gates g_i, of the number of wires going directly from the inputs $x_1, \ldots, x_n$ to g_i.*

We remark that Lemma 27 essentially shows that any function computable by a deterministic linear-time Turing Machine has a depth-2 circuit of small width and small 'block-degree'.

Conjecture 34. *[JS11] Suppose $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a linear function computable by a depth-2 circuit with width w and degree d. Then, f is computable by a depth-2 circuit, with width $O(w)$, and degree $O(d)$, each of whose gates compute a linear function.*

Observation 35. *Conjecture 32 $\implies$ Conjecture 34.*

Proof. Suppose $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ is a linear function computable by a depth-2 circuit with width w and degree d. Then, there exists a set $X \subseteq \mathbb{F}_2^n$ of size at least 2^{n-w}, such that for each $x \in X$, the value of the functions computed by the gates in the middle layer is the same. Hence, for each $x \in X$, each element of $f(x)$ can be written as a function of at most d elements of x. This shows that f is not an (w, d)-rigid function. Assuming Conjecture 32, the matrix $A \in \mathbb{F}_2^{n\times n}$ for the function f is not a (cw, cd)-rigid matrix, for some constant $c > 0$. Then, $A = B + C$, where the rank of B is at most cw, and C has at most cd non-zero entries in each row. Now, there exist matrices $B_1 \in \mathbb{F}_2^{n\times cw}$ and $B_2 \in \mathbb{F}_2^{cw\times n}$, such that $B = B_1B_2$. Then, f is computable by a linear depth-2 circuit with width cw and degree cd, where the middle layer computes output of the function corresponding to B_2. □

8 Future Directions

In this section, we state some well known problems related to matrices. It seems interesting to study the block-rigidity of the these functions.

8.1 Matrix Transpose

The matrix-transpose problem is described as follows:

- *Input:* A matrix $X \in \mathbb{F}_2^{n\times n}$ as a vector of length n^2 bits, in row-major order, for some $n \in \mathbb{N}$.

- *Output:* The matrix X column-major order (or equivalently, the transpose of X in row-major order).

It is well known (see [DMS91] for a short proof) that the above problem can be solved on a 2-tape Turing machine in time $O(N \log N)$, on inputs of length $N = n^2$. We believe that this cannot be solved by Turing machines in linear-time, and that the notion of block-rigidity might be a viable approach to prove this. Next, we observe some structural details about the problem.

The matrix-transpose problem computes a linear function, whose $N \times N$ matrix A on inputs of length $N = n^2$ is described as follows. For each $i, j \in [n]$, let $e_{ij} \in \mathbb{F}_2^{n\times n}$ denote the matrix whose $(i, j)^{\text{th}}$ entry is 1 and rest of the entries are zero. The matrix A is an $N \times N$ matrix made up of n^2 blocks, with the $(i, j)^{\text{th}}$ block equal to e_{ji}.

Using a similar argument as in Observation 16, one can show that the value of the following game captures the block-rigidity of the matrix-transpose function. Fix integers $n \in \mathbb{N}$, $1 \leq s < n$, and a collection $\mathcal{S} = (S_1, \ldots, S_n)$, where each $S_i \subseteq [n]$ is of size s. We define an n-player game $\mathcal{G}_{\mathcal{S}}$ as follows: A verifier chooses a matrix $X \in \mathbb{F}_2^{n\times n}$, with each entry being chosen uniformly and independently. For each $j \in [n]$, player j is given the rows of the matrix indexed by S_j, and they answer $y_j \in \mathbb{F}_2^n$. The verifier accepts if and only if for each $j \in [n]$, y_j equals the j^{th} column of X.

Conjecture 36. *There exists a constant $c > 0$ such that the function given by the matrix A is a (cn^2, cn)-block-rigid function. Equivalently, for each collection $\mathcal{S}$ with set sizes as cn, the value of the game $\mathcal{G}_{\mathcal{S}}$ is at most 2^{-cn^2}.*

We note that the above game is of independent interest from a combinatorial point of view as well. Basically, it asks whether there exists a large family of $n \times n$ matrices, in which each column can be represented as some function of a small fraction of the rows. The problem of whether the matrix A is a block-rigid matrix is also interesting. This corresponds to the players in the above game using strategies which are linear functions.

8.2 Matrix Product

The matrix-product problem is described as follows:

- *Input:* Matrices $X, Y \in \mathbb{F}_2^{n\times n}$ as vectors of length n^2 bits, in row-major order, for some $n \in \mathbb{N}$.
- *Output:* The matrix $Z = XY$ in row-major order.

The block-rigidity of the matrix-product function is captured by the following game: Fix integers $n \in \mathbb{N}$, $1 \leq s < n$, and collections $\mathcal{S} = (S_1, \ldots, S_n)$, $\mathcal{T} = (T_1, \ldots, T_n)$ where each $S_i, T_i \subseteq [n]$ is of size s. We define a n-player game $\mathcal{G}_{\mathcal{S},\mathcal{T}}$ as follows: A verifier chooses matrices $X, Y \in \mathbb{F}_2^{n\times n}$, with each entry being chosen uniformly and independently. For each $j \in [n]$, player j is given the rows of the matrices X and Y indexed by S_j and T_j respectively, and they answer $y_j \in \mathbb{F}_2^n$. The verifier accepts if and only if for each $j \in [n]$, y_j equals the j^{th} row of XY.

Conjecture 37. *There exists a constant $c > 0$ such that for each $\mathcal{S}, \mathcal{T}$ with set sizes as cn, the value of the game $\mathcal{G}_{\mathcal{S},\mathcal{T}}$ is at most 2^{-cn^2}.*

One may change the row-major order for some (or all) of the matrices to column-major order. It is easy to modify the above game in such a case.

References

[AC19] Josh Alman and Lijie Chen. Efficient construction of rigid matrices using an NP oracle. In *FOCS*, pages 1034–1055, 2019. 2

[AW17] Josh Alman and Ryan Williams. Probabilistic rank and matrix rigidity. In *STOC*, pages 641–652, 2017. 2

[BG15] Mark Braverman and Ankit Garg. Small value parallel repetition for general games. In *STOC*, pages 335–340, 2015. 4

[BHPT20] Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular PCPs. In *FOCS*, 2020. accepted. 2

[BOGKW88] Michael Ben-Or, Shafi Goldwasser, Joe Kilian, and Avi Wigderson. Multi-prover interactive proofs: How to remove intractability assumptions. In *STOC*, pages 113–131, 1988. 4

[DGW19] Zeev Dvir, Alexander Golovnev, and Omri Weinstein. Static data structure lower bounds imply rigidity. In *STOC*, pages 967–978, 2019. 5

[DHVY17] Irit Dinur, Prahladh Harsha, Rakesh Venkat, and Henry Yuen. Multiplayer parallel repetition for expanding games. In *ITCS*, volume 67 of *LIPIcs*, pages Art. No. 37, 16. 2017. 4

[DMS91] Martin Dietzfelbinger, Wolfgang Maass, and Georg Schnitger. The complexity of matrix transposition on one-tape off-line Turing machines. *Theoret. Comput. Sci.*, 82:113–129, 1991. 14

[DS14] Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *STOC*, pages 624–633, 2014. 4

[Fei91] Uriel Feige. On the success probability of the two provers in one-round proof systems. In *Structure in Complexity Theory Conference*, pages 116–123, 1991. 4

[FK91] H. Furstenberg and Y. Katznelson. A density version of the Hales-Jewett theorem. *J. Anal. Math.*, 57:64–119, 1991. 4

[FLvMV05] Lance Fortnow, Richard Lipton, Dieter van Melkebeek, and Anastasios Viglas. Time-space lower bounds for satisfiability. *J. ACM*, 52(6):835–865, 2005. 4

[For89] Lance Jeremy Fortnow. *Complexity theoretic aspects of interactive proof systems.* 1989. Ph.D. Thesis, MIT. 4

[Fri93] Joel Friedman. A note on matrix rigidity. *Combinatorica*, 13(2):235–239, 1993. 2

[FRS94] Lance Fortnow, John Rompel, and Michael Sipser. On the power of multi-prover interactive protocols. *Theoret. Comput. Sci.*, 134(2):545–557, 1994. 4

[FV02] Uriel Feige and Oleg Verbitsky. Error reduction by parallel repetition – a negative result. *Combinatorica*, 22(4):461–478, 2002. 4

[GT18] Oded Goldreich and Avishay Tal. Matrix rigidity of random Toeplitz matrices. *Comput. Complexity*, 27(2):305–350, 2018. (also in STOC 2016). 2

[Hol09] Thomas Holenstein. Parallel repetition: simplifications and the no-signaling case. *Theory Comput.*, 5:141–172, 2009. (also in STOC 2007). 4

[HPV77] John Hopcroft, Wolfgang Paul, and Leslie Valiant. On time versus space. *J. Assoc. Comput. Mach.*, 24(2):332–337, 1977. (also in FOCS 1975). 2, 10

[HS65] J. Hartmanis and R. E. Stearns. On the computational complexity of algorithms. *Trans. Amer. Math. Soc.*, 117:285–306, 1965. 2

[JS11] Stasys Jukna and Georg Schnitger. Min-rank conjecture for log-depth circuits. *J. Comput. System Sci.*, 77(6):1023–1038, 2011. 5, 13

[KLPS14] Abhinav Kumar, Satyanarayana V. Lokam, Vijay M. Patankar, and M. N. Jayalal Sarma. Using elimination theory to construct rigid matrices. *Comput. Complexity*, 23(4):531–563, 2014. (also in FSTTCS 2009). 2

[KU11] Kiran S. Kedlaya and Christopher Umans. Fast polynomial factorization and modular composition. *SIAM J. Comput.*, 40(6):1767–1802, 2011. (also in STOC 2008). 5

[Lok00] Satyanarayana V. Lokam. On the rigidity of Vandermonde matrices. *Theoret. Comput. Sci.*, 237(1-2):477–483, 2000. 2

[Lok01] Satyanarayana V. Lokam. Spectral methods for matrix rigidity with applications to size-depth trade-offs and communication complexity. *J. Comput. System Sci.*, 63(3):449–473, 2001. 2

[Lok06] Satyanarayana V. Lokam. Quadratic lower bounds on matrix rigidity. In *Theory and Applications of Models of Computation*, pages 295–307, 2006. 2

[Pol12] D. H. J. Polymath. A new proof of the density Hales-Jewett theorem. *Ann. of Math. (2)*, 175(3):1283–1327, 2012. 4

[PPST83] Wolfgang J. Paul, Nicholas Pippenger, Endre Szemeredi, and William T. Trotter. On determinism versus non-determinism and related problems. In *FOCS*, pages 429–438, 1983. 1, 2, 4, 10, 11

[PR80] Wolfgang Paul and Rüdiger Reischuk. On alternation. II. A graph-theoretic approach to determinism versus nondeterminism. *Acta Inform.*, 14(4):391–403, 1980. 2

[Raz89] Alexander Razborov. On rigid matrices (in russian). Unpublished Manuscript, 1989. 2

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM J. Comput.*, 27(3):763–803, 1998. (also in STOC 1995). 4

[Raz11] Ran Raz. A counterexample to strong parallel repetition. *SIAM J. Comput.*, 40(3):771–777, 2011. (also in FOCS 2008). 4

[San01] Rahul Santhanam. On separators, segregators and time versus space. In *CCC*, pages 286–294, 2001. 2

[SS97] Victor Shoup and Roman Smolensky. Lower bounds for polynomial evaluation and interpolation problems. *Comput. Complexity*, 6(4):301–311, 1996/97. 2

[SSS97] M. A. Shokrollahi, D. A. Spielman, and V. Stemann. A remark on matrix rigidity. *Inform. Process. Lett.*, 64(6):283–285, 1997. 2

[Val77] Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In *MFCS*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176, 1977. 1, 2, 5, 13

[Ver96] Oleg Verbitsky. Towards the parallel repetition conjecture. *Theoret. Comput. Sci.*, 157(2):277–282, 1996. 4

[vMR05] Dieter van Melkebeek and Ran Raz. A time lower bound for satisfiability. *Theoret. Comput. Sci.*, 348(2-3):311–320, 2005. (also in ICALP 2004). 4

[Wun12] Henning Wunderlich. On a theorem of Razborov. *Comput. Complexity*, 21(3):431–477, 2012. 2

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il