Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 38 (2025)

Conditional Complexity Hardness: Monotone Circuit Size, Matrix Rigidity, and Tensor Rank Under NSETH and Beyond

Nikolai Chukhin [*] Alexander S. Kulikov [†] Ivan Mihajlin [‡] Arina Smirnova [§]

April 4, 2025

Abstract

Proving complexity lower bounds remains a challenging task: currently, we only know how to prove conditional uniform (algorithm) lower bounds and nonuniform (circuit) lower bounds in restricted circuit models. About a decade ago, Williams (STOC 2010) showed how to derive nonuniform lower bounds from uniform upper bounds: roughly, by designing a fast algorithm for checking satisfiability of circuits, one gets a lower bound for this circuit class. Since then, a number of results of this kind have been proved. For example, Jahanjou et al. (ICALP 2015) and Carmosino et al. (ITCS 2016) proved that if NSETH fails, then E^{NP} has series-parallel circuit size $\omega(n)$.

One can also derive nonuniform lower bounds from nondeterministic uniform lower bounds. Perhaps the most well-known example is the Karp–Lipton theorem (STOC 1980): if $\Sigma_2 \neq \Pi_2$, then $\mathsf{NP} \not\subset \mathsf{P/poly}$. Some recent examples include the following. Nederlof (STOC 2020) proved a lower bound on the the matrix multiplication tensor rank under an assumption that TSP cannot be solved faster than in 2^n time. Belova et al. (SODA 2024) proved that there exists an explicit polynomial family of arithmetic circuit size $\Omega(n^\delta)$, for any $\delta > 0$, assuming that MAX-3-SAT cannot be solved faster that in 2^n nondeterministic time. Williams (FOCS 2024) proved an exponential lower bound for $\mathsf{ETHR} \circ \mathsf{ETHR}$ circuits under the *Orthogonal Vectors* conjecture. Under an assumption that *Set Cover* problem cannot be solved faster than in 2^n, Björklund and Kaski (STOC 2024) and Pratt (STOC 2024) constructed an explicit tensor with superlinear rank. Whereas all of the lower bounds above are proved under strong assumptions that might eventually be refuted, the revealed connections are of great interest and may still give further insights: one may be able to weaken the used assumptions or to construct generators from other fine-grained reductions.

In this paper, we continue developing this line of research and show how uniform nondeterministic lower bounds can be used to construct generators of various types of combinatorial objects that are notoriously hard to analyze: Boolean functions of high circuit size, matrices of high rigidity, and tensors of high rank. Specifically, we prove the following.

- If, for some ε and k, k-SAT cannot be solved in input-oblivious co-nondeterministic time $O(2^{(1/2+\varepsilon)n})$, then there exists a monotone Boolean function family in coNP of monotone circuit size $2^{\Omega(n/\log n)}$. Combining this with the result above, we get win-win circuit lower bounds: either E^{NP} requires series-parallel circuits of size $omega(n)$ or coNP requires monotone circuits of size $2^{\Omega(n/\log n)}$.
- If, for all $\varepsilon > 0$, MAX-3-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, then there exist small families of matrices with rigidity exceeding the best known constructions as well as small families of three-dimensional tensors of rank $n^{1+\Delta}$, for some $\Delta > 0$.

[*] Neapolis University Pafos and JetBrains Research. Email: buyolitsez1951@gmail.com
[†] JetBrains Research. Email: alexander.s.kulikov@gmail.com
[‡] JetBrains Research. Email: ivmihajlin@gmail.com
[§] Neapolis University Pafos. Email: arina.00.ds@gmail.com

ISSN 1433-8092

Contents

1 Complexity Lower Bounds

Finding the minimum time required to solve a given computational problem is a central question in computational complexity. Answering such a question for a particular problem involves proving a complexity lower bound, that is, showing that no fast algorithm can solve this problem. While the Time Hierarchy Theorem [HS65, HS66] guarantees that there are problems in P that cannot be solved in time $O(n^k)$, for any $k > 1$, we have no superlinear lower bounds for specific problems. For example, for SAT, one of the most important NP-complete problems, we have no algorithms working significantly faster than a brute force approach and at the same time have no methods of excluding a possibility that it can be solved in linear time.

Conditional Lower Bounds

As unconditional complexity lower bounds remain elusive, the classical complexity theory allows one to prove *conditional* lower bounds of the following form: if a problem A cannot be solved in polynomial time, then B cannot be solved in polynomial time too. Such results are proved via reductions that are essentially algorithms: one shows how to transform an instance of A into an instance of B. Nowadays, hundreds of such reductions between various NP-hard problems are known. For instance, if SAT cannot be solved in polynomial time, then the *Hamiltonian Cycle* problem also cannot be solved in polynomial time.

In a recently emerged area of *fine-grained complexity*, one aims to construct tighter reductions between problems showing that even a tiny improvement of an algorithm for one of them automatically leads to improved algorithms for the other one. For example, as proved by Williams [Wil05], if SAT cannot be solved in time $O(2^{(1-\varepsilon)n})$, for any $\varepsilon > 0$, then the *Orthogonal Vectors* problem cannot be solved in time $O(n^{2-\varepsilon})$, for any $\varepsilon > 0$. Again, many reductions of this form have been developed in recent years. We refer the reader to a recent survey by Vassilevska Williams [Vas18].

Circuit Lower Bounds

One of the reasons why proving complexity lower bounds is challenging is that an algorithm (viewed as a Turing machine or a RAM machine) is a relatively complex object: it has a memory, may contain loops, function calls (that may in turn be recursive). A related computational model of *Boolean circuits* has a much simpler structure (a straight-line program) and the same time is powerful enough to model algorithms: if a problem can be solved by algorithms in time $T(n)$, then it can also be solved by circuits in time $O(T(n) \log T(n))$ [PF79]. It turns out that proving circuit lower bounds is also challenging: while it is not difficult to show that almost all Boolean functions can be computed by circuits of exponential size only (this was proved by Shannon [Sha49] back in 1949), for no function from NP, we can currently exclude a possibility that it can be computed by circuits of linear size [LY22, FGHK16]. Strong lower bounds are only known for restricted models such as monotone circuits, constant-depth circuits, and formulas. Various such unconditional lower bounds can be found in the book by Jukna [Juk12].

An important difference between algorithms and circuits is that algorithms is a *uniform* model of computation (an algorithm is a program that needs to process instances of all possible lengths), whereas circuits are *nonuniform*: when saying that a problem can be solved by circuits, one usually means that there is an infinite collection of circuits, one circuit for every possible input length, and different circuits in this collection can, in principle, implement different programs. This makes the circuit model strictly more powerful than algorithms: on the one hand, every problem solved by algorithms can be solved by circuits of roughly the same size; on the other hand, it is not difficult to come up with a problem of small circuit size that cannot be solved by algorithms.

Connections Between Lower and Upper Bounds

Intuitively, it seems that proving complexity upper bounds should be easier than proving lower bounds. This intuition is well supported by a much higher number of results on algorithms compared to the number of results on lower bounds. Indeed, to prove an upper bound on the complexity of a problem, one designs an algorithm for the problem and analyzes it. Whereas to prove a complexity lower bound, one needs to reason about a wide range of fast algorithms (or small circuits) and to argue that none of them is able to solve the problem at hand. Perhaps surprisingly, the tasks of proving lower and upper complexity bounds are connected to each other. A classical example is Karp–Lipton theorem [KL80] stating that if $\mathsf{P} = \mathsf{NP}$, then EXP requires circuits of size $\Omega(2^n/n)$. More recently, Williams [Wil13] established a deep connection between upper bound for *Circuit Sat* and circuit lower bounds. Extending his results, Jahanjou, Miles and Viola [JMV18] proved that if NSETH is false (meaning that UNSAT can be solved fast with nondeterminism), then E^{NP} requires series-parallel Boolean circuits of size $\omega(n)$. Such results show how to derive nonuniform lower bounds (that is, circuit lower bounds) from uniform *upper* bounds (algorithm upper bounds).

Even though one can simulate an algorithm using circuits with slight overhead, the converse is not true as there are undecidable languages of low circuit complexity. Recently, [BKM$^+$24] showed results analogous to those presented herein, particularly on deriving a nonuniform lower bound from a non-randomized uniform lower bound. Specifically, they proved that if MAX-k-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, for any $\varepsilon > 0$, then, for any $\delta > 0$, there exists an explicit polynomial family that cannot be computed by arithmetic circuits of size $O(n^\delta)$. Also, Williams [Wil24] proved that if the *Orthogonal Vectors* conjecture (OVC) holds, then *Boolean Inner Product* on n-bit vectors cannot be computed by $\mathsf{ETHR} \circ \mathsf{ETHR}$ circuits of size $2^{\varepsilon n}$, for some $\varepsilon > 0$. Combined with the result above (since OVC is weaker than NSETH), it immediately leads to win-win circuit lower bounds: if NSETH fails, we have a lower bound for series-parallel circuits, otherwise we have a strong lower bound for $\mathsf{ETHR} \circ \mathsf{ETHR}$ circuits.

1.1 Our Contribution

In this paper, we derive a number of nonuniform lower bounds from uniform nondeterministic lower bounds. Our lower bounds apply for various objects that are notoriously hard to analyze: Boolean functions of high monotone circuit size, high rigidity matrices, and high rank tensors. For circuits, we get a win-win situation similar to the one by Williams.

Our first result shows how to get $2^{\Omega(n/\log n)}$ monotone circuit lower bounds (improving best known bounds of the form $2^{\Omega(\sqrt{n})}$) under an assumption that SAT requires co-nondeterministic time $O(2^{(1/2+\varepsilon)n})$ if the verifier is given a proof that depends on the length of the input only.

Theorem 1. *If, for some* $\varepsilon > 0$ *and* $k \in \mathbb{Z}_{\geqslant 3}$*,* k-SAT *cannot be solved in input-oblivious co-nondeterministic time* $O(2^{(1/2+\varepsilon)n})$*, then there exists a monotone Boolean function family in* coNP *of monotone circuit size* $2^{\Omega(n/\log n)}$.

We then extend this result further to get a monotone circuit lower bound $2^{\Omega(n)}$ for a monotone function from P/poly that can be computed in linear time (and hence has almost linear circuit size).

Theorem 2. *If, for some $\varepsilon > 0$ and $k \in \mathbb{Z}_{\geqslant 3}$, k-SAT cannot be solved in co-nondeterministic time $O(2^{(1/2+\varepsilon)n})$, then there exists a monotone Boolean function family in* P/poly *that requires monotone circuits of size $2^{\Omega(n)}$ and can be computed in linear time.*

As both previous theorems use weaker assumptions than NSETH, we get the following corollary.

Corollary 1. *If* NSETH *holds, there exists a monotone Boolean function family in* coNP *of monotone circuit size $2^{\Omega(n/\log n)}$ and a monotone Boolean function family in* P/poly *that requires monotone circuits of size $2^{\Omega(n)}$ and can be computed in linear time.*

Combining this with circuit lower bounds from the negation of NSETH due to [JMV18, CGI+16], leads to win-win circuit lower bounds. It should be noted that proving any of these two circuit lower bounds is a challenging long-standing open problem.

Corollary 2. *At least one of the following two circuit lower bounds holds:*

1. E^{NP} *requires series-parallel circuits of size $\omega(n)$;*
2. *There exists a monotone Boolean function family in* coNP *of monotone circuit size $2^{\Omega(n/\log n)}$ and a monotone Boolean function family in* P/poly *that requires monotone circuits of size $2^{\Omega(n)}$ and can be computed in linear time.*

Another result in this direction demonstrates how to derive circuit lower bounds from NETH (which asserts that 3-SAT cannot be solved in co-nondeterministic time $2^{o(n)}$). Specifically, we establish lower bounds for the classes Q_t^n.

Definition 1. *Let Q_t^n denote the set of all Boolean functions f over n variables such that for any $x^1, \ldots, x^t \in f^{-1}(0)$, there exists an $i \in [n]$ for which $x_i^1 = \ldots = x_i^t = 0$.*

These sets have been studied in secure multiparty computation; see [HM97, FM98, HM00] for references. They have also been viewed from the complexity-theoretic perspective [CDI+13, KP22]. We denote by $\mathsf{THR}_a^b \colon \{0,1\}^b \to \{0,1\}$ the function that evaluates to one if and only if the input contains at least a ones. It turns out the class Q_t^n is exactly the class of functions computable by $\mathsf{THR}_{l+1}^{lt+1}$ gates for arbitrary $l \geqslant 1$ where the negations are not permitted (even at the leaves). The following lemma is a combination of Lemmas 5.2 and 5.3 from [CDI+13] and Theorem 3 from [KP22]. In the statement, by $C \leqslant f$ we mean that $C(x) \leqslant f(x)$, for all x.

Lemma 1. *The set Q_t^n is equal to the set of functions f for which there exists a circuit C, composed only of $\mathsf{THR}_{l+1}^{lt+1}$ gates for arbitrary $l \geqslant 1$, such that $C \leqslant f$.*

Our result indicates that Q_t^n contains functions that are exponentially hard to compute using $\mathsf{THR}_{l+1}^{lt+1}$ gates only while being easy to compute by regular circuits.

Theorem 3. *If* NETH *holds, then for any $t = \omega(1)$ and infinitely many $n = \omega(t)$, there exists a function $f \in Q_t^n$ such that any circuit $C \leqslant f$, composed of $\mathsf{THR}_{l+1}^{lt+1}$ gates, satisfies $\mathrm{size}(C) = 2^{\Omega(n)}$. Moreover, $f \in$ P/poly and can be computed in linear time.*

Thus, although the class Q_t^n may appear relatively simple, it contains functions that are exceedingly difficult to compute using only $\mathsf{THR}_{l+1}^{lt+1}$ gates but which can be computed by almost linear circuits. Theorem 3 consequently yields another win-win lower bound: if NETH fails, then E^{NP} cannot be computed by linear-size circuits [CRTY23].

Corollary 3. *At least one of the following two circuit lower bounds must hold:*

1. E^{NP} *requires circuits of size* $\omega(n)$*;*

2. *For any* $t = \omega(1)$*, there exists a function* $f \in Q_t^n \cap \mathsf{P/poly}$ *such that any circuit* C*, composed only of* $\mathsf{THR}_{l+1}^{lt+1}$ *gates for arbitrary* $l \geqslant 1$ *over variables, has size* $2^{\Omega(n)}$*. Moreover,* f *can be computed in linear time.*

Our second result shows how to construct small families of matrices with rigidity exceeding the best known constructions under an assumption that MAX-3-SAT requires co-nondeterministic time 2^n.

Theorem 4. *If, for every* $\varepsilon > 0$*,* MAX-*3*-SAT *cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$*, then, for all* $\delta > 0$*, there is a generator* $g \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k}$ *computable in time polynomial in* k *such that, for infinitely many* k*, there exist a seed* s *for which* $g(s)$ *has* $k^{\frac{1}{2}-\delta}$*-rigidity* $k^{2-\delta}$*.*

Our third result extends the second result by including high-rank tensors.

Theorem 5. *If, for any* $\varepsilon > 0$ MAX-*3*-SAT*, cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$*, then, for all* $\delta > 0$ *and some* $\Delta > 0$*, there are two generators* $g_1 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k}$ *and* $g_2 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k\times k}$ *computable in time polynomial in* k *such that, for infinitely many* k*, at least one of the following is satisfied:*

- $g_1(s)$ *has* $k^{1-\delta}$*-rigidity* $k^{2-\delta}$*, for some* s*;*

- $\operatorname{rank}(g_2(s))$ *is at least* $k^{1+\Delta}$*, for some* s*.*

It is worth noting that [BKM$^+$24] showed circuit lower bounds under the same assumption: if MAX-k-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$ for any $\varepsilon > 0$, then for any $\delta > 0$, there exists an explicit polynomial family that cannot be computed by arithmetic circuits of size $O(n^\delta)$.

The best known lower bounds for the size of depth-three circuits computing an explicit Boolean function is $2^{\Omega(\sqrt{n})}$ [Hås89, PPSZ05]. Proving a $2^{\omega(\sqrt{n})}$ lower bound for this restricted circuit model remains a challenging open problem and it is known that a lower bound as strong as $2^{\omega(n/\log\log n)}$ would give an $w(n)$ lower bound for unrestricted circuits via Valiant's reduction [Val77]. One way of proving better depth-three circuit lower bounds is via *canonical circuits* introduced by Goldreich and Wigderson [GW20]. They are closely related to rigid matrices: if T is an $n \times n$ matrix of r-rigidity r^3, then the corresponding bilinear function requires canonical circuits of size $2^{\Omega(r)}$ [GW20]. Goldreich and Tal [GT18] showed that a random Toeplitz matrix has r-rigidity $\frac{n^3}{r^2 \log n}$, which implies a $2^{\Omega(n^{3/5})}$ lower bound on canonical depth-three circuits for an explicit function. By substituting $n^{2/3-\delta}$-rigidity for some $\delta > 0$ in Theorem 5, one gets the following result.

Corollary 4. *If, for every* $\varepsilon > 0$*,* MAX-*3*-SAT *cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$*, then, for any* $\delta > 0$*, one can construct an explicit family of* $2^{\log^{O(1)} n}$ *functions such that, for infinitely many* n*, at least one of them is either bilinear and requires canonical circuits of size* $2^{\Omega(n^{2/3-\delta})}$ *or trilinear and requires arithmetic circuits of size* $\Omega(n^{1.25})$*.*

This conditionally improves the recent result of Goldreich [Gol22], who presented an $O(1)$-linear function that requires canonical depth-*two* circuits of size $2^{\Omega(n^{1-\varepsilon})}$, for every $\varepsilon > 0$. Moreover, every bilinear function can be computed by canonical circuits of size $2^{O(n^{2/3})}$, so the lower bound is almost optimal and conditionally addresses Open Problem 6.5 from [GT18].

1.2 Known Explicit Constructions

In this section, we review known constructions of combinatorial objects (functions of high monotone circuit size, matrices of high rigidity, and tensors of high rank).

Monotone Functions

For monotone NP-problems (like *Clique*, *Matching*, *Hamiltonian Cycle*), it is natural to ask what is their monotone circuit size. A celebrated result by Razborov [Raz85] is a lower bound of $n^{\Omega(\log n)}$ on monotone circuit size obtained by the approximation method. Subsequently, Andreev [And85] proved a $2^{n^{1/8-o(1)}}$ lower bound for another explicit monotone function. Following the work of [AB87, And87, Juk99, HR00], in 2020, Cavalar, Kumar, and Rossman [CKR22] achieved the best-known lower bound of $2^{n^{1/2-o(1)}}$. Recently, another approach for proving monotone circuit lower bounds was developed using lower bounds from Resolution proofs and lifting theorem [GGKS20]. Specifically, if an unsatisfiable formula F is hard to refute in the resolution proof system, then a monotone function associated with F has large monotone circuit complexity. In this manner, following the work of [GGKS20, GKRS19, LMM$^+$22], the lower bound of $2^{\Omega(\sqrt{n})}$ was also achieved.

Proving a $2^{\omega(n^{1/2})}$ lower bound remains a challenging open problem (whereas a lower bound $2^{\Omega(n)}$ was recently proved by Pitassi and Robere [PR17] for monotone *formulas*). Our Corollary 1 establishes a stronger lower bound under an assumption that NSETH holds.

Matrix Rigidity

A matrix M over a field $\mathbb{F}$ has *r-rigidity s* if for any matrices R, S over a field $\mathbb{F}$ such that $M = R + S$ and $\text{rank}(R) \leqslant r$, S has at least s nonzero entries. That is, one needs to change at least s elements in M to change its rank down to at most r. The concept of rigidity was introduced by Valiant [Val77] and Grigoriev [Gri80]. It has striking connections to areas such as computational complexity [Lok09, AW17, AC19, GKW21], communication complexity [Wun12], data structure lower bounds [DGW19, RR20], and error-correcting codes [Dvi11].

Valiant [Val77] proved that if a matrix M has εn-rigidity $n^{1+\delta}$ for some $\varepsilon, \delta > 0$, then the bilinear form of M cannot be computed by arithmetic circuits of size $O(n)$ and depth $O(\log n)$. Following Razborov [Raz89], Wunderlich [Wun12] proved that the existence of strongly-explicit matrices with $2^{(\log\log n)^{\omega(1)}}$-rigidity δn^2, for some $\delta > 0$, implies the existence of a language that does not belong to the communication complexity analog of PH. Although it is known [Val77] that for any r almost every $n \times n$ matrix has r-rigidity $\Omega(\frac{(n-r)^2}{\log n})$ over algebraically closed fields, obtaining an explicit constructions of rigid matrices remains a long-standing open question. Many works have aimed at finding explicit or semi-explicit rigid matrices [Fri93, PV91, SSS97, AW17, DE19, AC19, DL20, VK22, BGKM23, BHPT24]. Also, a recent line of works establishes a connection between the *Range Avoidance* problem and the construction of matrices with high rigidity [Kor21, GLW22, GGNS23, CHR24, Li24].

Small explicit[1] families of rigid matrices can be rused to prove arithmetic circuits lower bounds [Val77]. The best known polynomial-time constructible matrices have r-rigidity $\frac{n^2}{r}\log(n/r)$ for any r, which was proved by Shokrollahi, Spielman and Stemann [SSS97]. Goldreich and Tal [GT18]

[1] Matrix or family of matrices is called explicit if it is polynomial time constructible.

proved that a random $n \times n$ Toeplitz matrix over $\mathbb{F}_2$ (i.e., a matrix of the form $A_{i,j} = a_{i-j}$ for random bits $a_{-(n-1)}, \dots, a_{n-1}$) has r-rigidity $\frac{n^3}{r^2 \log n}$ for $r \geqslant \sqrt{n}$. However, the size of that family is exponential in n. Our Theorem 4 demonstrates that, under the assumption that $\mathsf{MAX\text{-}3\text{-}SAT}$ is hard, for any $\delta > 0$, for infinitely many n one can construct a $2^{\log^{O(1)} n}$-sized family of $n \times n$ matrices with at least one having $n^{1/2-\delta}$-rigidity $n^{2-\delta}$.

This result is still far from the regime where circuit lower bounds can be derived via Valiant's result, but it strictly improves the polynomial-time construction [SSS97] for any $r < \sqrt{n}$ and improves the result of Goldreich and Tal [GT18] by substantially reducing the family size while maintaining the same rigidity for $r \approx \sqrt{n}$. An open question remains as to whether explicit constructions of rigid matrices exist in the class P^{NP} [Ram20]. The construction provided by Goldreich and Tal [GT18] lies in E^{NP}. Following the work of Alman and Chen [AC19], [BHPT24] established that there exists a constant $\delta > 0$ such that one can construct $n \times n$ matrices with $2^{\log n/\Omega(\log \log n)}$-rigidity δn^2 in FNP. Subsequently, Chen and Lyu [CL21] demonstrated a method for constructing highly rigid matrices, proving that there exists a constant $\delta > 0$ such that one can construct $n \times n$ matrices with $2^{\log^{1-\delta} n}$-rigidity $(1/2 - \exp(-\log^{2/3 \cdot \delta} n)) \cdot n^2$ in P^{NP}. More recently, Alman and Liang [AL25] showed that the Walsh-Hadamard matrix H_n has $c_1 \log n$-rigidity $n^2 \left(\frac{1}{2} - n^{c_2}\right)$ for some constants $c_1, c_2 > 0$. Our construction, in the class $\mathsf{DTIME}[2^{\log^{O(1)} n}]^{\mathsf{NP}}$, produces matrices with $n^{\frac{1}{2}-\delta}$-rigidity $n^{2-\delta}$ for any $\delta > 0$, under the condition that $\mathsf{MAX\text{-}3\text{-}SAT}$ is hard.

Tensor Rank and Arithmetic Circuits

Proving arithmetic circuit lower bounds is another important challenge in complexity theory. An arithmetic circuit over a field $\mathbb{F}$ uses as inputs formal variables and field elements and computes in every gate either a sum or a product. As proved by Strassen [Str73, Str75] and Baur and Strassen [BS83], computing $\sum_{i=1}^{n} x_i^n$ requires arithmetic circuits of size $\Omega(n \log n)$, provided n does not divide the characteristic of $\mathbb{F}$. Raz [Raz03] further established that arithmetic circuits with bounded coefficients require $\Omega(n^2 \log n)$ gates to perform matrix multiplication over $\mathbb{R}$ or $\mathbb{C}$, following the work in [RS03]. However, no superlinear lower bounds are known for polynomials of constant degree. For constant-depth arithmetic circuits over fields of characteristic 2, exponential lower bounds are known [Raz87, Smo87]. For other finite characteristics, exponential lower bounds are known only for depth 3 [GK98, GR98]. For characteristic 0, the best lower bound for depth 3 is $\Omega(n^{2-\varepsilon})$ [SW99].

Matrix multiplication is one of the fundamental problems whose arithmetic circuit size is of great interest. While many highly nontrivial algorithms for it are known (starting from Strassen [Str69]), we still do not have superlinear lower bounds on its arithmetic circuit complexity. Proving such lower bounds is closely related to the problem of determining the rank of tensors. A d-dimensional tensor is said to have rank q if it can be expressed as a sum of q rank-one tensors. Here, a rank-one d-dimensional tensor is a tensor of the form $u_1 \otimes \cdots \otimes u_d$, where $\otimes$ stands for a tensor product. By a multiplication tensor, we mean a tensor of size $n^2 \times n^2 \times n^2$ (formally defined in Section 2.4). Establishing an upper bound for the rank of the multiplication tensor provides a means of proving upper bounds for matrix multiplication via the laser method [Str86]. Moreover, proving a lower bound for the tensor rank would yield superlinear lower bounds for arithmetic circuits computing the polynomial defined by that tensor.

Therefore, proving lower bounds on the tensor rank provides a path to proving lower bounds for arithmetic circuits. For the rank of the matrix multiplication tensor, Bshouty [Bsh89] and

Bläser [Blä99] proved a lower bound $2.5n^2 - \Theta(n)$. Subsequently, Shpilka [Shp03] improved the bound to $3n^2 - o(n^2)$ over $\mathbb{F}_2$. The bound $3n^2 - o(n)$ was later achieved by Landsberg [Lan14] over arbitrary fields and further slightly improved by Massarenti and Raviolo [MR13, MR14]. Alexeev, Forbes and Tsimerman [AFT11] constructed explicit d-dimensional tensors with rank $2n^{\lfloor \frac{d}{2} \rfloor} + n - \Theta(d \log n)$, thus improving the lower bounds on high-dimensional tensors. Nevertheless, superlinear size lower bounds for constant-degree polynomials remain unknown. Additionally, Håstad [Hås90] established that determining the rank of a d-dimensional tensor is NP-hard for any $d \geqslant 3$. Consequently, a major open problem is to construct an explicit family of d-dimensional tensors with rank at least $n^{\lfloor \frac{d}{2} \rfloor + \varepsilon}$ for some $\varepsilon > 0$ and $d \geqslant 3$.

Our Theorem 5 shows that, under an assumption that MAX-3-SAT cannot be solved fast co-nondeterministically, one gets an explicit $2^{\log^{O(1)} n}$-size family of $n \times n$-matrices and $n \times n \times n$-tensors, such that, for any $\delta > 0$ and some $\Delta > 0$, at least one of the matrices has $n^{1-\delta}$-rigidity $n^{2-\delta}$ or one of the tensors has rank $n^{1+\Delta}$. Furthermore, we establish a trade-off between matrix rigidity and tensor rank, see Theorem 11. Other results for proving lower bounds on tensor rank under certain assumptions are known. Nederlof [Ned20] proved that, if for any $\varepsilon > 0$, the bipartite *Traveling Salesman* problem cannot be solved in time $2^{(1-\varepsilon)n}$, then the matrix multiplication tensor has superlinear rank. Additionally, Björklund and Kaski [BK24] recently proved that if, for any $\varepsilon > 0$, there exists a k such that the k-*Set Cover* problem cannot be solved in time $O(2^{(1-\varepsilon)n}|\mathcal{F}|)$, then there is an explicit tensor with superlinear rank, where $\mathcal{F}$ is a family of subsets of $[n]$, each of size at most k. Pratt [Pra24] improved this result, showing that under the same conjecture there exists an explicit tensor of shape $n \times n \times n$ and rank at least $n^{1.08}$. [BCH$^+$24] showed that if for every $\varepsilon > 0$ *Chromatic Number* problem cannot be solved in time $2^{(1-\varepsilon)n}$, then there exists an explicit tensor of superlinear rank.

1.3 Discussion and Open Problems

Many of the lower bounds mentioned above are proved under various strong assumptions (on the complexity of SAT, MAX-SAT, *Set Cover*, *Chromatic Number*, *Traveling Salesman*). They seem much stronger than merely P $\neq$ NP and might eventually be refuted. Still, the revealed reductions between problems are of great interest and may still yield further insights: one may be able to weaken the used assumptions or to construct generators from other fine-grained reductions. Moreover, as with the case of NSETH assumption, one may be able to derive interesting consequences from both an assumption and its negation leading to a win-win situation. Below, we state a few open problems in this direction.

If one were to partition 3-SAT in Theorem 8 into t parts, where $t = \omega(1)$, then creating an explicit function equivalent to t-OV would imply lower bounds for that function under NETH. This approach would yield a more advantageous win-win situation, as the assumption that NETH is false gives stronger lower bounds [CRTY23].

Open Problem 1. *Prove that if* NETH *is true, then there exists an explicit function of monotone complexity* $2^{\omega(\sqrt{n})}$.

Moreover, the existence of small monotone circuits not only refutes NSETH but also establishes that UNSAT has input-oblivious proof size $2^{o(n)}$ that can be verified in time $2^{\frac{n}{2}+o(n)}$. Therefore, we believe that it may be possible to derive even stronger implications beyond merely refuting NSETH.

Another question arises regarding tensor rank. Is it possible to construct a small family of tensors with superlinear rank, assuming that MAX-3-SAT cannot be solved efficiently in co-nondeterministic time (this way, eliminating the dependence on rigidity from our results)?

Open Problem 2. *Prove that, if, for any $\varepsilon > 0$,* MAX-*3*-SAT *cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, then, for all $\delta > 0$ there exists a small family of tensors of size $k \times k \times k$ such that, for infinitely many k, at least one of them has rank at least $k^{1+\delta}$.*

However, we do not know of any consequences of solving MAX-3-SAT faster in the co-nondeterministic setting. This makes our assumption weak and raises the question of whether it can be refuted.

Open Problem 3. *Prove that, for some $\varepsilon > 0$,* MAX-*3*-SAT *can be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$.*

On the other hand, to get a win-win situation, it would be interesting to find nontrivial consequences of the existence of such an algorithm.

Open Problem 4. *Derive new circuit lower bounds from the existence of an algorithm solving* MAX-*3*-SAT *in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, for some $\varepsilon > 0$.*

Structure of the Paper

The paper is organized as follows. In Section 2, we introduce the notation used throughout the paper and provide the necessary background. In Section 3, we give an overview of the main proof ideas. In Section 4, we establish the win-win circuit lower bound. In Section 5, we construct rigid matrices under an assumption that MAX-3-SAT cannot be solved fast co-nondeterministically. In Section 6, we construct either three-dimensional tensors with high rank or matrices with high rigidity under the same assumption.

2 Preliminaries

For a positive integer k, $[k] = \{1, 2, \dots, k\}$, whereas for a predicate P, $[P] = 1$ if P is true and $[P] = 0$ otherwise (the Iverson bracket). For a set S and an integer k, by $\binom{S}{k}$ we denote the set of all subsets of S of size k.

2.1 Boolean Circuits

For a binary vector $v \in \{0,1\}^n$, by $\overline{v} \in \{0,1\}^n$, we denote the vector resulting from v by flipping all its coordinates (thus, $v \oplus \overline{v} = 1^n$). This extends naturally to sets of vectors: for $V \subseteq \{0,1\}^n$, $\overline{V} = \{\overline{v} \colon v \in V\}$. By $w(v) = \sum_{i \in [n]} v_i$, we denote the *weight* of v. For two vectors $u, v \in \{0,1\}^n$, we say that u *dominates* v and write $u \geqslant v$, if $u_i \geqslant v_i$ for all $i \in [t]$. A Boolean function $f \colon \{0,1\}^n \to \{0,1\}$ is called *monotone* if $f(u) \geqslant f(v)$, for all $u \geqslant v$. For disjoint sets $A, B \subseteq \{0,1\}^n$, we say that a function $f \colon \{0,1\}^n \to \{0,1\}$ separates (A, B) if $A \subseteq f^{-1}(1)$ and $B \subseteq f^{-1}(0)$.

A *Boolean circuit* C over variables $x_1, \dots, x_n$ is a directed acyclic graph with nodes of in-degree zero and two. The in-degree zero nodes are labeled by variables x_i and constants 0 or 1, whereas the in-degree two nodes are labeled by binary Boolean functions. The only gate of out-degree zero is the output of the circuit. A circuit is called *series-parallel* if there exists a labeling ℓ of its nodes such that for every wire (u, v), $\ell(u) < \ell(v)$, and no pair of wires $(u, v), (u', v')$ satisfies $\ell(u) < \ell(u') < \ell(v) < \ell(v')$.

A Boolean circuit C with variables $x_1, \ldots, x_n$ computes a Boolean function $f\colon \{0, 1\}^n \to \{0, 1\}$ in a natural way. We define the *size* of C, $\text{size}(C)$, as the number of gates in it, and the Boolean circuit complexity of a function f, $\text{size}(f)$, as the minimum size of a circuit computing it. A circuit is called *monotone* if all its gates compute disjunctions and conjunctions. It is not difficult to see that the class of Boolean functions computed by monotone circuits coincides with the class of monotone Boolean functions. For a monotone function f, $\text{size}_m(f)$ is the minimum size of a monotone circuit computing f.

A sequence $(f_n)_{n=1}^{\infty}$, where $f_n\colon \{0, 1\}^n \to \{0, 1\}$, is called a *function family*. Such a family defines a language $\cup_{n=1}^{\infty} f_i^{-1}(1)$ and we say that the family is *explicit* if this language is in NP. When saying that a language $L \subseteq \{0, 1\}^*$ can be solved by circuits of size $T(n)$, we mean that it can be represented by a function family $(f_n)_{n=1}^{\infty}$ where $\text{size}(f_n) \leqslant T(n)$, for all n.

2.2 Arithmetic Circuits

An *arithmetic circuit* C over a ring R and variables $x_1, \ldots, x_n$ is a directed acyclic graph with nodes of in-degree zero or two. The in-degree zero nodes are labeled either by variables x_i or elements of R, whereas the in-degree two nodes are labeled by either $+$ or $\times$. Every gate of out-degree zero is called an output gate. We will typically take R to be $\mathbb{Z}$ or $\mathbb{Z}_p$ for a prime number p. A single-output arithmetic circuit C over R computes a polynomial over R in a natural way. We say that C computes a polynomial $P(x_1, \ldots, x_n)$ if the two polynomials are *identical* (as opposed to saying that C computes P if the two polynomials agree on every assignments of $(x_1, \ldots, x_n) \in R^n$). We define the size of C as the number of edges in it, and the arithmetic circuit complexity of a polynomial as the minimum size of a circuit computing it.

2.3 SAT, MAX-SAT, OV, and Clique

For a CNF formula F, by $n(F)$ and $m(F)$ we denote the number of variables and clauses of F, respectively. We write just n and m, if the corresponding CNF formula is clear from the context. In SAT (UNSAT), one is given a CNF formula and the goal is to check whether it is satisfiable (unsatisfiable, respectively). In k-SAT, the given formula is in k-CNF (that is, all clauses have at most k literals). In MAX-k-SAT, one is given a k-CNF and an integer t and is asked to check whether it is possible to satisfy exactly t clauses.

When designing an algorithm for SAT, one can assume that the input formula has a linear (in the number of variables) number of clauses. This is ensured by the following *Sparsification Lemma*. By (β, k)-SAT, we denote a special case of k-SAT where the input k-CNF formula has at most βn clauses.

Theorem 6 (Sparsification Lemma, [IPZ01]). *For any* $k \in \mathbb{Z}_{\geqslant 3}$ *and* $\varepsilon > 0$*, there exists* $\alpha = \alpha(k, \varepsilon)$ *and an algorithm that, given a* k*-CNF formula* F *over* n *variables, outputs* $t \leqslant 2^{\varepsilon n}$ *formulas* $F_1, \ldots, F_t$ *in* k*-CNF such that* $n(F_i) \leqslant n$ *and* $m(F_i) \leqslant \alpha n$*, for all* $i \in [t]$*, and* $F \in \text{SAT}$ *if and only if* $\vee_{i \in [t]} F_i \in \text{SAT}$*. The running time of the algorithm is* $O(n^{O(1)} 2^{\varepsilon n})$.

Corollary 5. *There exists a function* $\beta\colon \mathbb{Z}_{\geqslant 3} \times \mathbb{R}_{>0} \to \mathbb{Z}_{>0}$ *such that, if there exists* $\varepsilon > 0$ *for which* $(\beta(k, \varepsilon), k)$-SAT *can be solved in time* $O(2^{n/2+\varepsilon n})$ *for any* $k \in \mathbb{Z}_{\geqslant 3}$*, then* k-SAT *can be solved in time* $O(2^{n/2+2\varepsilon n})$.

Proof. Let $\beta(k, \varepsilon) = \alpha(k, \varepsilon)$ (see Theorem 6 for a definition of α). Given an instance of k-SAT, we apply the Sparsification Lemma with parameter ε to get a sequence of $t \leqslant 2^{\varepsilon n}$ formulas $F_1, \dots, F_t$, each of which is a $(\beta(k, \varepsilon), k)$-SAT instance. By the assumption, the satisfiability of each F_i can be checked in time $O(2^{n/2+\varepsilon n})$. Thus, one can check the satisfiability of $\bigvee_{i\in[t]} F_i$ in time $O(t \cdot 2^{n/2+\varepsilon n}) = O(2^{n/2+2\varepsilon n})$. □

The *Strong Exponential Time Hypothesis* (SETH), introduced in [IPZ01, IP01], asserts that, for any $\varepsilon > 0$, there is k such that k-SAT cannot be solved in time $O(2^{(1-\varepsilon)n})$. The *Nondeterministic* SETH (NSETH), introduced in [CGI+16], extends SETH by asserting that SAT is difficult even for co-nondeterministic algorithms: for any $\varepsilon > 0$, there is k such that k-SAT cannot be solved in *co-nondeterministic* time $O(2^{(1-\varepsilon)n})$. Though both these statements are stronger than $\mathsf{P} \neq \mathsf{NP}$, they are known to be hard to refute: as proved by Jahanjou, Miles and Viola [JMV18], if SETH is false, then there exists a Boolean function family in E^{NP} of series-parallel circuit size $\omega(n)$. [CGI+16] noted that it suffices to refute NSETH to get the same circuit lower bound.

Theorem 7 ([JMV18, CGI+16]). *If* NSETH *is false, then there exists a Boolean function family in* E^{NP} *of series-parallel circuit size* $w(n)$.

SETH-based conditional lower bounds are known for a wide range of problems and input parameters. One of such problems is *Orthogonal Vectors* (OV): given two sets $A, B \subseteq \{0, 1\}^d$ of size n, check whether there exists $a \in A$ and $b \in B$ such that $a \cdot b = \sum_{i\in[d]} a_i b_i = 0$. It is straightforward to see that OV can be solved in time $O(n^2 d)$. Williams [Wil05] proved that under SETH, there is no algorithm solving OV in time $O(n^{2-\varepsilon} d^{O(1)})$, for any $\varepsilon > 0$. This follows from the following reduction.

Theorem 8 ([Wil05]). *There exists an algorithm that, given a CNF formula* F *with* n *variables and* m *clauses, outputs two sets* $A_F, B_F \subseteq \{0, 1\}^m$ *such that* $|A_F| = |B_F| = 2^{n/2}$ *and* $F \in \mathsf{SAT}$ *if and only if* $(A_F, B_F) \in \mathsf{OV}$*. The running time of the algorithm is* $O(mn \cdot 2^{n/2})$.

In a similar manner, one can reduce a CNF formula to the t-OV problem involving a greater number of sets. Consider the t-OV problem, where one is given t sets $A^1, \dots, A^t \subseteq \{0, 1\}^d$, each of size n, and the objective is to determine whether there exist elements $a^1 \in A^1, \dots, a^t \in A^t$ such that

$$\sum_{i\in[d]} a_i^1 \cdot \ldots \cdot a_i^t = 0.$$

Lemma 2. *There exists an algorithm which, given a CNF formula* F *with* n *variables and* m *clauses, along with an integer* t*, constructs* t *sets* $A_F^1, \dots, A_F^t \subseteq \{0, 1\}^m$ *such that* $|A^1| = \ldots = |A^t| = 2^{n/t}$ *and* $F \in \mathsf{SAT}$ *if and only if* $(A_F^1, \dots, A_F^t) \in t$-OV*. The running time of the algorithm is* $O(mn \cdot 2^{n/t})$.

The proof follows the approach presented in [Wil05]. An instance $(A^1, \dots, A^t) \notin t$-OV if and only if for all vectors $a^1 \in A^1, \dots, a^t \in A^t$, the vectors share a common coordinate with value one. This can equivalently be formulated by stating that the union $Q = A^1 \cup \ldots \cup A^t$ possesses the property that for every $a^1, \dots, a^t \in Q$, the vectors share a common one. Consequently, Lemma 1 states that the instance $(A^1, \dots, A^t) \notin t\text{-OV}$ if and only if there exists a circuit C, composed of $\mathsf{THR}^{lt+1}_{l+1}$ gates and variables, such that $C(\overline{x}) = 0$ for every $x \in A^1 \cup \ldots \cup A^t$.

A similar reduction allows to solve MAX-3-SAT by finding a 4-clique in a 3-uniform hypergraph. A subset of l nodes in a k-hypergraph is called an l-*clique*, if any k of them form an edge in the graph.

Theorem 9 ([Wil07, LVW18]). *There exists an algorithm that, given a* 3*-CNF formula* F *with* n *variables and an integer* t*, outputs a* 4*-partite* 3*-uniform hypergraph* G *with parts of size* $k = n^{O(1)}2^{n/4}$ *such that* G *has a* 4*-clique if and only if it is possible to satisfy exactly* t *clauses of* F*.*

Proof. To construct the graph G, partition the variables of F into four groups A_0, A_1, A_2, A_3 of size $n/4$. Then, label each clause of F by some $i \in \{0, 1, 2, 3\}$ such that the clause does not contain variables from A_i. Then, for any $i \in \{0, 1, 2, 3\}$, assigning variables from all parts except from A_i, determines how many clauses labeled i are satisfied. Define tensors $\mathcal{T}_0, \mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3 \in \mathbb{Z}_{\geqslant 0}^{2^{\frac{n}{4}} \times 2^{\frac{n}{4}} \times 2^{\frac{n}{4}}}$ as follows: for $U_1, U_2, U_3 \in \{0, 1\}^{\frac{n}{4}}$, let $\mathcal{T}_i[U_1, U_2, U_3]$ be equal to the number of clauses with label i when the variables from groups $A_{(i+1) \bmod 4}$, $A_{(i+2) \bmod 4}$, and $A_{(i+3) \bmod 4}$ are assigned as in U_1, U_2, and U_3. Then, it is possible to satisfy t clauses in F if and only if there exist $U_0, U_1, U_2, U_2 \in \{0, 1\}^{\frac{n}{4}}$ such that

$$\mathcal{T}_0[U_1, U_2, U_3] + \mathcal{T}_1[U_2, U_3, U_0] + \mathcal{T}_2[U_3, U_0, U_1] + \mathcal{T}_3[U_0, U_1, U_2] = t.$$

As F contains at most $O(n^3)$ clauses, $t = O(n^3)$. This makes it possible to enumerate in polynomial time all values of the four terms above. For an integer q and $i \in \{0, 1, 2, 3\}$, define a tensor $\mathcal{A}_{i,q} \in \{0, 1\}^{2^{\frac{n}{4}} \times 2^{\frac{n}{4}} \times 2^{\frac{n}{4}}}$ as

$$\mathcal{A}_{i,q}[U_1, U_2, U_3] = [T_i[U_1, U_2, U_3] = q].$$

Then, one can satisfy t clauses in F if and only if

$$\sum_{\substack{U_0, U_1, U_2, U_3 \in \{0,1\}^{\frac{n}{4}} \\ t_0 + t_1 + t_2 + t_3 = t}} \mathcal{A}_{0,t_0}[U_1, U_2, U_3] \cdot \mathcal{A}_{1,t_1}[U_2, U_3, U_0] \cdot \mathcal{A}_{2,t_2}[U_3, U_0, U_1] \cdot \mathcal{A}_{3,t_3}[U_0, U_1, U_2] > 0.$$

For fixed t_0, t_1, t_2, t_3, such that $t_0 + t_1 + t_2 + t_3 = t$, the tensors $\mathcal{A}_{0,t_0}, \mathcal{A}_{1,t_1}, \mathcal{A}_{2,t_2}, \mathcal{A}_{3,t_3}$ may be viewed as a description of edges of a 4-partite 3-uniform hypergraph G_{t_0,t_1,t_2,t_3}. There is a 4-clique in G_{t_0,t_1,t_2,t_3} if and only if one can satisfy t_0 clauses with label 0, t_1 clauses with label 1, and so on. Let G be a superimposition of all such graphs. Then, G contains a 4-clique if and only if one can satisfy exactly t clauses of F. □

2.4 Rigidity and Tensor Rank

For a field $\mathbb{F}$, by $\mathbb{F}^{a \times b}$ we denote the set of all matrices of size $a \times b$ over $\mathbb{F}$. Similarly, by $\mathbb{F}^{a \times b \times c}$ we denote the set of all three-dimensional tensors of shape $a \times b \times c$ over $\mathbb{F}$. For two tensors A and B (of arbitrary shape), by $A \otimes B$ we denote a tensor product of A and B. For a matrix $M \in \mathbb{F}^{a \times b}$, by $|M|$ we denote the number of nonzero entries of M.

For a matrix $M \in \mathbb{F}^{a \times b}$, we say that it has r-rigidity s if it is necessary to change at least s entries of M to reduce its rank to r. That is, for each decomposition $M = R + S$ such that $\operatorname{rank}(R) \leqslant r$, it holds that $|S| \geqslant s$.

The rank of a three-dimensional tensor is a natural extension of the matrix rank. For a tensor $\mathcal{A} \in \mathbb{F}^{n \times n \times n}$, we define its rank, $\mathrm{rank}(\mathcal{A})$, as the smallest integer r such that there exist r tuples of vectors $a_l, b_l, c_l \in \mathbb{F}^n$ for which

$$\mathcal{A} = \sum_{l \in [r]} a_l \otimes b_l \otimes c_l,$$

or equivalently,

$$\mathcal{A}[i, j, k] = \sum_{l \in [r]} a_l[i] b_l[j] c_l[k],$$

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

We denote by ω the smallest real number such that any two $n \times n$ matrices can be multiplied in time $O(n^{\omega+\varepsilon})$ for any $\varepsilon > 0$ using only field operations[2].

Consider the three-dimensional tensor $\mathcal{A}_n \in \mathbb{F}^{n^2 \times n^2 \times n^2}$:

$$\mathcal{A}_n[(i, j), (i, k), (k, j)] = 1,$$

for all $i, j, k \in [n]$, with all other entries being zero. Using an approach based on the work of Strassen [Str69], for any positive integer k, if $\mathrm{rank}(\mathcal{A}_k) = q$, then one can construct an arithmetic circuit of size $O(n^{\log_k(q)})$ to perform multiplication of two $n \times n$ matrices. Thus, ω satisfies the following equation:

$$\omega = \inf_{k \in \mathbb{Z}_{>0}} \log_k \mathrm{rank}(\mathcal{A}_k).$$

In other words, for sufficiently large k, we have that $\mathrm{rank}(\mathcal{A}_k) \geqslant k^{\omega}$. Specifically, if $n = k^2$, then $\mathcal{A}_k \in M_{n \times n \times n}$, and $\mathrm{rank}(\mathcal{A}_k) \geqslant n^{\omega/2}$. Therefore, if $\omega > 2$, this yields superlinear lower bounds on arithmetic circuits and on the rank of the multiplication tensor.

The best known upper bound on ω is 2.371552 [VXXZ24]. Further details on the matrix multiplication tensor can be found in [AV24, Blä13].

3 Proof Ideas

In this section, we give high level ideas of the main results.

3.1 Monotone Circuit Lower Bound

To prove a lower bound $2^{\Omega(n/\log n)}$ for monotone circuit size of coNP under NSETH, we assume that all monotone functions from coNP have monotone circuit size $2^{o(n/\log n)}$ and show how this can be used to solve UNSAT nondeterministically in less than 2^n steps.

Given a k-CNF F, we construct an instance (A_F, B_F) of OV of size $2^{n/2}$ using Theorem 8. We show (see Lemma 3) that (A_F, B_F) is a yes-instance if and only if there exists a monotone Boolean function separating $(A_F, \overline{B_F})$. Hence, one can guess a small monotone circuit separating $(A_F, \overline{B_F})$ and verify that it is correct. Overall, the resulting nondeterministic algorithm proceeds as follows.

1. In time $O(n^2 2^{n/2})$, generate the sets A_F and B_F.
2. Guess a monotone circuit C of size $O(2^{(1-\varepsilon)n/2})$.

[2] The value of ω may depend on the field over which the calculations are performed [Sch81].

3. Verify that C separates $(A_F, \overline{B_F})$. To do this, check that $C_F(a) = 1$, for every $a \in A_F$. Then, check that $C_F(b) = 0$, for every $b \in \overline{B_F}$. The running time of this step is

$$O(2^{(1-\varepsilon)n/2} \cdot |A_F| + 2^{(1-\varepsilon)n/2} \cdot |\overline{B_F}|) = O(2^{(1-\varepsilon)n/2} \cdot 2^{n/2}) = O(2^{(1-\varepsilon/2)n}).$$

4. If C separates $(A_F, \overline{B_F})$, then $F \in \mathsf{UNSAT}$.

This already shows how small monotone circuits could help to break NSETH, though it does not provide a single explicit function with this property. In the full proof in Section 4, we introduce such a function. We also use a weaker assumption (than NSETH).

3.2 Rigid Matrices and High Rank Tensors

To construct matrices of high rigidity under the assumption that MAX-3-SAT cannot be solved fast co-nondeterministically, we proceed as follows. Take a 3-CNF formula over n variables and an integer t and transform it, using Theorem 9, into a 4-partite 3-uniform hypergraph G with each part of size $k = n^{O(1)}2^{\frac{n}{4}}$. The graph G contains a 4-clique if and only if one can satisfy t clauses of F. We show that checking whether G has a 4-clique is equivalent to evaluating a certain expression over three-dimensional tensors. We then show that if all slices of these tensors have low rigidity, then one can solve 4-*Clique* on G in co-nondeterministic time $O(k^{4-\varepsilon})$: to achieve this, one guesses a decomposition of a matrix into a sum of a low rank matrix and a matrix with few nonzero entries. The idea is that a low rank matrix can be guessed quickly as a decomposition into a rank-one matrices (which are just products of two vectors), whereas the second matrix can be guesses quickly as one needs to guess the nonzero entries only. In turn, this allows one to solve MAX-3-SAT faster than 2^n co-nondeterministically. Thus, this reduction is a generator of rigid matrices: it takes a 3-CNF formula and outputs a matrix. The same idea can be used to generate either tensors with high rank or matrices with high rigidity.

4 Circuit Lower Bounds

4.1 Boolean Functions of High Monotone Circuit Size

In this section, we prove Theorem 1.

Theorem 1. *If, for some $\varepsilon > 0$ and $k \in \mathbb{Z}_{\geqslant 3}$, k-SAT cannot be solved in input-oblivious co-nondeterministic time $O(2^{(1/2+\varepsilon)n})$, then there exists a monotone Boolean function family in coNP of monotone circuit size $2^{\Omega(n/\log n)}$.*

Combining this with Corollary 1 and Theorem 7, we get win-win circuit lower bounds. Proving any of these two circuit lower bounds is a challenging open problem.

Corollary 2. *At least one of the following two circuit lower bounds holds:*

1. E^{NP} *requires series-parallel circuits of size $\omega(n)$;*
2. *There exists a monotone Boolean function family in coNP of monotone circuit size $2^{\Omega(n/\log n)}$ and a monotone Boolean function family in P/poly that requires monotone circuits of size $2^{\Omega(n)}$ and can be computed in linear time.*

For the proof of Theorem 1, we need two technical lemmas. Recall that the reduction from SAT to OV (see Theorem 8), given a formula F, produces two sets $A_F, B_F \subseteq \{0,1\}^{m(F)}$ such that $|A_F| = |B_F| = 2^{n(F)/2}$ and $F \in \mathsf{SAT}$ if and only if $(A_F, B_F) \in \mathsf{OV}$.

Lemma 3. *Let F be a CNF formula. Then, $F \notin \mathsf{SAT}$ if and only if $(A_F, \overline{B_F})$ can be separated by a monotone function.*

For a CNF formula F, let $f_F \colon \{0,1\}^{m(F)} \to \{0,1\}$ be defined as follows:

$$f_F(x) = [\forall b \in \overline{B_F} \colon b \not\geqslant x]. \tag{1}$$

It is immediate that f is monotone and that $\overline{B_F} \subseteq f_F^{-1}(0)$.

Proof of Lemma 3. Assume that $F \notin \mathsf{SAT}$. We show that the function f_F separates $(A_F, \overline{B_F})$. To do this, it suffices to show that $A_F \subseteq f_F^{-1}(1)$. If this is not the case, then there is $a \in A_F$ such that $f_F(a) = 0$, that is, there exists $b \in \overline{B_F}$ such that $b \geqslant a$. Hence, there is no $i \in [m]$ such that $b_i = 0$ and $a_i = 1$. In turn, this means that, for $\overline{b} \in B_F$, there is no $i \in [m]$ such that $\overline{b}_i = 1$ and $a_i = 1$, meaning that $(A_F, B_F) \in \mathsf{OV}$, contradicting $F \notin \mathsf{SAT}$.

For the reverse direction, assume that for some monotone function $h \colon \{0,1\}^m \to \{0,1\}$, it holds that $A_F \subseteq h^{-1}(1)$ and $\overline{B_F} \subseteq h^{-1}(0)$. By the monotonicity of h, for every $a \in A_F$ and every $b \in \overline{B_F}$, $b \not\geqslant a$. Hence, for every $a \in A_F$ and every $b \in \overline{B_F}$, there exists $i \in [m]$ such that $b_i = 0$ and $a_i = 1$. Switching from $\overline{B_F}$ to B_F, we get that, for every $a \in A_F$ and every $b \in B_F$, there exists $i \in [m]$ such that $b_i = 1$ and $a_i = 1$, meaning that $(A_F, B_F) \notin \mathsf{OV}$ and $F \notin \mathsf{SAT}$. □

By $\mathcal{F}_{n,k,\beta}$ denote the set of k-CNF formulas with n variables and at most βn clauses. Any formula $F \in \mathcal{F}_{n,k,\beta}$ can be encoded in binary using at most $\gamma n \log n$ bits (for some $\gamma = \gamma(k, \beta)$): each variable is encoded using $\log n$ bits, all other symbols (parentheses as well as negations, disjunctions, and conjunctions require a constant number of bits). Hence,

$$|\mathcal{F}_{n,k,\beta}| \leqslant 2^{\gamma n \log n}.$$

Let us call $\mathcal{W}_{n,k}$ the set of all binary strings of length n and weight k:

$$\mathcal{W}_{n,k} = \{x \in \{0,1\}^n \colon w(x) = k\}.$$

Lemma 4. *There exists an injective encoding $e \colon \{0,1\}^{\leqslant n} \to \mathcal{W}_{4n,2n}$ such that computing and inverting e takes time linear in n.*

Proof. For $x \in \{0,1\}^{\leqslant n}$, let $e(x)$ as follows be any balanced string of length $4n$ that starts with $1^{|x|}0$ followed by x. This encoding is injective, as the initial segment $1^{|x|}0$ allows us to uniquely identify x within $e(x)$. Both computing and inverting e can be accomplished in linear time with respect to n. Indeed, computing $e(x)$ is straightforward. Conversely, given $e(x)$, one can first extract the size of x and then decode x. □

As a simple corollary, applying Lemma 4 to Boolean formulas, one can obtain the following result.

Lemma 5. *There exists a parameter* $l = O(n \log n)$ *and an injective encoding*

$$e\colon \mathcal{F}_{n,k,\beta} \to \mathcal{W}_{l,l/2}$$

such that computing and inverting e *takes time polynomial in* n.

Proof. Given a formula $F \in \mathcal{F}_{n,k,\beta}$, we can encode it in binary using at most $\gamma n \log n$ bits (as stated above). Then, applying Lemma 4, we obtain the desired result. □

Proof of Theorem 1. Assuming that coNP has small monotone circuits, we design a fast co-nondeterministic algorithm for SAT. Assume that all monotone functions over N variables in coNP can be computed by monotone circuits of size $2^{o(N/\log N)}$.

Let F be a k-CNF over n variables. Thanks to Corollary 5, we may assume that $m(F) \leqslant \beta n$, where $\beta = \beta(k, \frac{\varepsilon}{2})$. Then, solving $\mathcal{F}_{n,k,\beta}$ in co-nondeterministic time $O(2^{n/2+\varepsilon' n})$ for $\varepsilon' = \frac{\varepsilon}{2}$ will be sufficient for a contradiction.

Below, we define a universal function f containing f_F, for all $F \in \mathcal{F}_{n,k,\beta}$, as subfunctions (recall the definition (1) of f_F). Let $N = l + m$, where $l = O(n \log n)$ and e is the function from Lemma 5 and $m = \beta n$ (hence, $N = O(n \log n)$). We define $f\colon \{0,1\}^N \to \{0,1\}$ as follows. Let (c, x), where $c \in \{0,1\}^l$ and $x \in \{0,1\}^m$, be an input of f. Then,

$$f(c,x) = \begin{cases} 1, & \text{if } w(c) > l/2, \\ 0, & \text{if } w(c) < l/2, \\ 1, & \text{if } w(c) = l/2 \text{ and } e^{-1}(c) = \varnothing, \\ 0, & \text{if } w(c) = l/2 \text{ and } e^{-1}(c) \in \mathsf{SAT}, \\ f_F(x), & \text{if } w(c) = l/2 \text{ and } F = e^{-1}(c) \notin \mathsf{SAT}. \end{cases}$$

We now ensure three important properties of f.

1. *The function* f *is monotone.* For the sake of contradiction, assume that $(c, x) \geqslant (c', x')$, but $0 = f(c,x) < f(c',x') = 1$. Since $f(c',x') = 1$, $w(c') \geqslant l/2$. If $w(c') > l/2$, then $f(c,x) = 1$, since $w(c) \geqslant w(c') > l/2$. Hence, assume that $w(c') = w(c) = l/2$. Since $c \geqslant c'$, we conclude that $c = c'$ (implying that $e^{-1}(c) \neq \varnothing$). If $e^{-1}(c) \in \mathsf{SAT}$, then $f(c',x') = 0$. Hence $e^{-1}(c) \notin \mathsf{SAT}$, thus $f(c,x) = f_F(x)$ and $f(c',x') = f_F(x')$, where $F = e^{-1}(c) = e^{-1}(c')$, and f_F is clearly monotone.

2. *The function* f *is explicit:* $f \in \mathsf{coNP}$. Assume that $f(c,x) = 0$. This can happen for one of the three following reasons.

   - $w(c) < l/2$. This is easily verifiable.
   - $w(c) = l/2$ and $e^{-1}(c) \in \mathsf{SAT}$. To verify this, one computes $F = e^{-1}(c)$ (in time $O(\mathsf{poly}(n))$, due to Lemma 5), guesses its satisfying assignment and verifies it.
   - $w(c) = l/2$, $F = e^{-1}(c) \notin \mathsf{SAT}$, and $f_F(x) = 0$. Recall that if $F \in \mathsf{SAT}$, then we can guess and verify its satisfying assignment, so in this case there is no need to verify the unsatisfiability of F. Since $f_F(x) = 0$, there exists $b \in \overline{B_F}$ such that $b \geqslant x$. To verify this, one computes $F = e^{-1}(c)$, guesses the corresponding assignment to the second half of the variables of F, ensures that it produces the vector $\overline{b}$, and verifies that $b \geqslant x$.

3. Assume that f can be computed by a monotone circuit C of size $2^{o(N/\log N)}$. We show that, then, for any $k \in \mathbb{Z}_{\geqslant 3}$, k-SAT can be solved in input-oblivious co-nondeterministic time $O(2^{n/2+\varepsilon' n})$. Since $N = O(n \log n)$, then $\mathsf{size}(C) = O(2^{\varepsilon' n})$.

   Let $F \in \mathcal{F}_{n,k,\beta}$ be an unsatisfiable formula with n variables and no more than βn clauses. The following algorithm verifies its unsatisfiability in input-oblivious nondeterministic time $O(2^{n/2+\varepsilon' n})$.

   (a) In time $O(n^2 2^{n/2})$, generate the sets A_F and B_F.
   (b) Guess a monotone circuit C. This can be done in time $O(\mathsf{size}(C)) = O(2^{\varepsilon' n})$.
   (c) Compute $c = e(F)$ and substitute the first l variable of C by c. Call the resulting circuit C_F.
   (d) Verify that C_F separates $(A_F, \overline{B_F})$. To do this, check that $C_F(a) = 1$, for every $a \in A_F$. Then, check that $C_F(b) = 0$, for every $b \in \overline{B_F}$. The running time of this step is

   $$O((|A_F| + |\overline{B_F}|) \cdot \mathsf{size}(C_F)) = O(2^{n/2} \cdot 2^{\varepsilon' n} = O(2^{n/2+\varepsilon' n}).$$

   (e) If C_F is monotone and separates $(A_F, \overline{B_F})$, then we are certain that $F \notin \mathsf{SAT}$, thanks to Lemma 3.

□

Now, we show how to improve the bound for the P/poly class by making a slight adjustment to the proof of Theorem 1.

Theorem 2. *If, for some $\varepsilon > 0$ and $k \in \mathbb{Z}_{\geqslant 3}$, k-SAT cannot be solved in co-nondeterministic time $O(2^{(1/2+\varepsilon)n})$, then there exists a monotone Boolean function family in P/poly that requires monotone circuits of size $2^{\Omega(n)}$ and can be computed in linear time.*

Combining this with Theorem 7, we obtain another win-win circuit lower bound.

Corollary 6. *At least one of the following two circuit lower bounds holds:*

- E^{NP} *requires series-parallel circuits of size $\omega(n)$;*
- *There is an explicit monotone function $f \in$ P/poly that requires monotone circuits of size $2^{\Omega(n)}$.*

Proof of Theorem 2. Assuming that all monotone functions from P/poly have monotone circuits of size $2^{o(n)}$, we construct a fast co-nondeterministic algorithm similar to Theorem 1. Thanks to Corollary 5, we may assume that $m(F) \leqslant \beta n$, where $\beta = \beta(k, \varepsilon')$, setting $\varepsilon' = \frac{\varepsilon}{2}$. Thus, if for any unsatisfiable $F \in \mathcal{F}_{n,k,\beta}$, there exists a small monotone circuit separating A_F and $\overline{B_F}$, then one can solve k-SAT in co-nondeterministic time $2^{\frac{n}{2}+\varepsilon' n}$ (recall the proof of Theorem 1).

Now, we construct a family of monotone functions, each belonging to P/poly, such that if all of them have monotone circuit size $2^{o(n)}$, then this ensures that any UNSAT formula from $\mathcal{F}_{n,k,\beta}$ has a small monotone circuit separating A_F and $\overline{B_F}$.

For a set $A_F \subseteq \{0,1\}^{m(F)}$ of size $2^{n(F)/2}$, denote by $A'_F \subseteq \{0,1\}^{m(F)+l}$ the set defined as

$$A'_F = \{e(\mathsf{ind}(x)) \circ x \colon x \in A_F\},$$

where $\mathrm{ind}(x)$ is a satisfying assignment for the first half of the elements of F that produces x in the reduction (i.e., the index of the row that contains x), and e is the encoding from Lemma 4. Also, define $B'_F = \{0^l \circ x \mid x \in B_F\}$. Recall that $m(F) + l = O(n)$. It is easy to see that $(A_F, B_F) \in \mathsf{OV}$ if and only if $(A'_F, B'_F) \in \mathsf{OV}$. Thus, applying Lemma 3, we need to determine whether A'_F and $\overline{B'_F}$ can be separated by a monotone function.

For fixed sets A'_F and $\overline{B'_F}$, if they can be separated by a monotone function, then there exists a linear-time algorithm that computes some monotone function separating A'_F and $\overline{B'_F}$. The algorithm takes as input $c \circ x$, where $c \in \{0,1\}^l$ and $x \in \{0,1\}^{m(F)}$, and proceeds as follows:

- If $w(c) < \frac{l}{2}$, it outputs 0.
- If $w(c) > \frac{l}{2}$, it outputs 1.
- If $e^{-1}(c) = \varnothing$, then it outputs 0.
- Otherwise, it assigns $e^{-1}(c)$ to the first half of the variables. Let $y \in \{0,1\}^{m(F)}$ be the vector of clauses satisfied by this assignment. Then, the algorithm returns $[y \leqslant x]$.

It is clear that this algorithm separates A'_F and $\overline{B'_F}$ whenever such separation is possible and that the resulting function is indeed monotone.

Now, assuming that k-SAT cannot be solved in co-nondeterministic time $O(2^{\frac{n}{2}+\varepsilon n})$, for infinitely many instances of SAT, there exists a pair of sets (A, B) which cannot be separated by a monotone function of size $2^{o(n)}$. Hence, infinitely often, the function produced by the algorithm above has monotone circuit size $2^{\Omega(n)}$, despite being computable in linear time. Thus, there exists a sequence of formulas

$$\{F_i \colon m(F_i) = i \wedge F_i \in \mathsf{UNSAT}\}_{i=2}^{\infty}$$

such that the family of monotone functions from $\mathsf{P/poly}$

$$\{g_{F_i}\}_{i=2}^{\infty},$$

where $g_{F_i} \colon \{0,1\}^i \to \{0,1\}$ is the function described by the algorithm above, cannot be computed in co-nondeterministic time $2^{\frac{n}{2}+\varepsilon n}$. □

4.2 Functions of High Threshold Size

In this section, we aim to derive circuit lower bounds under the assumption of NETH. It is important to observe that the preceding reduction is insufficient for solving 3-SAT in time $2^{o(n)}$, as the reduction to the OV problem operates in time $O(2^{n/2})$. However, by reducing 3-SAT to t-OV for some $t = \omega(1)$, the reduction can be executed in time $2^{o(n)}$.

Theorem 3. *If* NETH *holds, then for any* $t = \omega(1)$ *and infinitely many* $n = \omega(t)$*, there exists a function* $f \in Q_t^n$ *such that any circuit* $C \leqslant f$*, composed of* $\mathsf{THR}_{l+1}^{lt+1}$ *gates, satisfies* $\mathrm{size}(C) = 2^{\Omega(n)}$*. Moreover,* $f \in \mathsf{P/poly}$ *and can be computed in linear time.*

Proof. Recall the proof of Theorem 2. Given an arbitrary $\varepsilon > 0$ and a 3-SAT instance F, apply Corollary 5 to ensure that the number of clauses $m(F)$ satisfies $m(F) = \beta n$, where $\beta = \beta(3, \varepsilon)$. Then, add artificial variables to make n a multiple of t. Next, apply Lemma 2 to construct the sets $A^1, \dots, A^t$ such that $(A^1, \dots, A^t) \notin t$-$\mathsf{OV}$ if and only if $F \in \mathsf{UNSAT}$. Subsequently, nondeterministically guess a circuit C of size $2^{o(n)}$, composed only of $\mathsf{THR}^{lt+1}_{l+1}$ gates for arbitrary $l \geqslant 1$. Then verify that $C(\overline{x}) = 0$ for every $x \in A^1 \cup \ldots \cup A^t$. The total verification time is $O(t \cdot \mathrm{size}(C) \cdot 2^{n/t}) = 2^{o(n)}$. Lemma 1 guarantees that if $F \in \mathsf{UNSAT}$, then such a circuit C must exist. Therefore, if no such circuit C of size $2^{o(n)}$ is found, it must be the case that its size is $2^{\Omega(n)}$ or $F \in \mathsf{SAT}$. □

5 Matrices of High Rigidity

In this section, we show that low rigidity matrices can be utilized to solve MAX-3-SAT more efficiently. Thus, by assuming that MAX-3-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, for any $\varepsilon > 0$, we get a generator of high rigidity matrices. Throughout this section, rigidity decompositions are considered over a finite field $\mathbb{F}$.

Theorem 4. *If, for every $\varepsilon > 0$, MAX-3-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, then, for all $\delta > 0$, there is a generator $g\colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k}$ computable in time polynomial in k such that, for infinitely many k, there exist a seed s for which $g(s)$ has $k^{\frac{1}{2}-\delta}$-rigidity $k^{2-\delta}$.*

Proof. Take a 3-CNF formula over n variables and an integer t and transform it, using Theorem 9, into a 4-partite 3-uniform hypergraph with parts H_0, H_1, H_2, H_3 of size $k = n^{O(1)} 2^{\frac{n}{4}}$. Recall that G contains a 4-clique if and only if one can satisfy t clauses of F.

Let $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3 \in \{0,1\}^{k \times k \times k}$ be three-dimensional tensors encoding the edges of G. $\mathcal{A}_i$ is responsible for storing edges spanning nodes from all parts except for H_i: for example, G has an edge (u_1, u_2, u_3), where $u_1 \in H_1, u_2 \in H_2$, and $u_3 \in H_3$, if and only if $\mathcal{A}_0[u_1, u_2, u_3] = 1$. Let

$$R = \sum_{j_0, j_1, j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] \cdot \mathcal{A}_1[j_2, j_3, j_0] \cdot \mathcal{A}_2[j_3, j_0, j_1] \cdot \mathcal{A}_3[j_0, j_1, j_2].$$

Then, G has a 4-clique if and only if $R > 0$. Also, for $j_0, j_1 \in [k]$, let

$$R_{j_0, j_1} = \sum_{j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] \cdot \mathcal{A}_1[j_2, j_3, j_0] \cdot \mathcal{A}_2[j_3, j_0, j_1] \cdot \mathcal{A}_3[j_0, j_1, j_2].$$

Thus,

$$R = \sum_{j_0, j_1 \in [k]} R_{j_0, j_1}.$$

Now, for fixed $j_0, j_1 \in [k]$, define vectors $v, u \in \{0,1\}^k$ as follows: $v[i] = \mathcal{A}_3[j_0, j_1, i]$ and $u[l] = \mathcal{A}_2[l, j_0, j_1]$. Also, define two matrices $M, L \in \{0,1\}^{k \times k}$: $M[i, l] = \mathcal{A}_0[j_1, i, l]$ and $L[i, l] = \mathcal{A}_1[i, l, j_0]$. Hence,

$$R_{j_0, j_1} = \sum_{j_2, j_3 \in [k]} M[j_2, j_3] \cdot L[j_2, j_3] \cdot u[j_3] \cdot v[j_2].$$

Now, assume that M and L have r-rigidity s, that is,

$$M = R_M + S,$$
$$L = R_L + T,$$

where $R_M, R_L, S, T \in \mathbb{F}^{k \times k}$, $\operatorname{rank}(R_M), \operatorname{rank}(R_L) \leqslant r$ and $|S|, |T| \leqslant s$.

Since $\mathrm{rank}(R_M) \leqslant r$, $\mathrm{rank}(R_L) \leqslant r$, it follows that there exist vectors: $a_{i,M}, a_{i,L} \in \mathbb{F}^k$ and $b_{i,M}, b_{i,L} \in \mathbb{F}^k$ such that

$$M = \left(\sum_{i \in [r]} a_{i,M} \cdot b_{i,M}^T \right) + S,$$
$$L = \left(\sum_{i \in [r]} a_{i,L} \cdot b_{i,L}^T \right) + T.$$

We guess these vectors for the matrices M and L. Additionally, we guess only nonzero entries of S and T, since they are sparse.

For fixed j_0, the time complexity to guess and verify the decomposition of L is $O(rk^2 + s + k^2)$, which includes the time to multiply the vectors, sum them, add S, and verify that the result equals L. The same applies when fixing j_1. Thus, the overall time complexity is $O(rk^3 + ks + k^3) = O(rk^3 + ks)$, for all j_0, j_1.

Hence,

$$\begin{aligned} R_{j_0,j_1} &= \sum_{j_2 \in [k]} \sum_{j_3 \in [k]} \left(S + \sum_{i \in [r]} a_{i,M} \cdot b_{i,M}^T \right) [j_2, j_3] \cdot \left(T + \sum_{l \in [r]} a_{l,L} \cdot b_{l,L}^T \right) [j_2, j_3] \cdot u[j_3] \cdot v[j_2] = \\ &= \sum_{j_2 \in [k]} \sum_{j_3 \in [k]} \left(\sum_{i \in [r]} a_{i,M} \cdot b_{i,M}^T \right) [j_2, j_3] \cdot \left(\sum_{l \in [r]} a_{l,L} \cdot b_{l,L}^T \right) [j_2, j_3] \cdot u[j_3] \cdot v[j_2] + \\ &+ \underbrace{\sum_{j_2 \in [k]} \sum_{j_3 \in [k]} u[j_3] \cdot v[j_2] \cdot (S[j_2, j_3] \cdot T[j_2, j_3] + S[j_2, j_3] \cdot R_L[j_2, j_3] + R_M[j_2, j_3] \cdot T[j_2, j_3])}_{R'_{j_0,j_1}}. \end{aligned}$$

Since S and T are sparse, we can compute the R'_{j_0,j_1} in time $O(s)$, for fixed j_0, j_1. Now, it remains to evaluate

$$\begin{aligned} R_{j_0,j_1} - R'_{j_0,j_1} = &\sum_{j_2 \in [k]} \sum_{j_3 \in [k]} \left(\sum_{i \in [r]} a_{i,M} \cdot b_{i,M}^T \right) [j_2, j_3] \cdot \left(\sum_{l \in [r]} a_{l,L} \cdot b_{l,L}^T \right) [j_2, j_3] \cdot u[j_3] \cdot v[j_2] = \\ &\sum_{j_2 \in [k]} \sum_{j_3 \in [k]} \left(\sum_{i \in [r]} a_{i,M}[j_2] \cdot b_{i,M}[j_3] \right) \cdot \left(\sum_{l \in [r]} a_{l,L}[j_2] \cdot b_{l,L}[j_3] \right) \cdot u[j_3] \cdot v[j_2] = \\ &\sum_{i \in [r]} \sum_{l \in [r]} \sum_{j_2 \in [k]} \sum_{j_3 \in [k]} a_{i,M}[j_2] \cdot b_{i,M}[j_3] \cdot a_{l,L}[j_2] \cdot b_{l,L}[j_3] \cdot u[j_3] \cdot v[j_2] = \\ &\sum_{i \in [r]} \sum_{l \in [r]} \left(\sum_{j_2 \in [k]} a_{i,M}[j_2] \cdot a_{l,L}[j_2] \cdot v[j_2] \right) \left(\sum_{j_3 \in [k]} b_{i,M}[j_3] \cdot b_{l,L}[j_3] \cdot u[j_3] \right). \end{aligned}$$

This sum can be computed in time $O(r^2k)$. Thus, the total running time (for all j_0, j_1) is $O(rk^3 + ks + k^2s + r^2k^3) = O(k^2s + r^2k^3)$. If $r = k^{\frac{1}{2}-\delta}$ and $s = k^{2-\delta}$ for some $\delta > 0$, the running time becomes $O(k^{4-\delta})$.

We can construct a generator g that takes the following inputs: the encoding of a 3-CNF formula, an integer t, j_0, j_1, and a 0/1 bit. If this bit is 0, we output M; otherwise, we output L. If the input to the generator is not valid, it outputs an empty family. Therefore, if MAX-3-SAT cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$ for any $\varepsilon > 0$, then for infinitely many n, the generator outputs at least one matrix with $k^{\frac{1}{2}-\delta}$-rigidity $k^{2-\delta}$, for any $\delta > 0$, and $k = n^{O(1)}2^{\frac{n}{4}}$. The generator takes $n^{O(1)} = \log^{O(1)}(k)$ input bits, works in time polynomial in k, and outputs a matrix of dimensions $k \times k$. □

As a simple corollary, one might weaken the assumption and obtain weaker matrix rigidity, while still improving the known explicit construction by [SSS97]. Recall that [SSS97] constructed matrices with r-rigidity $\frac{n^2}{r}\log(n/r)$.

Corollary 7. *If* MAX-3-SAT *cannot be solved in co-nondeterministic time* $O(2^{0.92n})$*, then there exists a generator* $g\colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k}$ *computable in time polynomial in* k *such that, for infinitely many* k*, there exists a seed* s *for which* $g(s)$ *has* $k^{0.34}$*-rigidity* $k^{1.68}$.

Proof. The generator is the one constructed in the proof of Theorem 4. Substituting $r = k^{0.34}$ and $s = k^{1.68}$ into the running time $O(k^2 s + r^2 k^3)$ yields the running time $O(k^{3.68}) = O(2^{0.92n})$, thereby completing the proof. Note that $r \cdot s = k^{2.02}$, so further weakening the assumption will result in worse rigidity than that achieved in [SSS97]. □

6 Tensors of High Rank

In this section, we show how to generate high rank tensors under an assumption that MAX-3-SAT is hard. Throughout this section, we assume that $\mathbb{F}$ is a finite field over which rigidity and tensor decompositions are considered. We start by proving two auxiliary lemmas. The first one shows how rectangular matrix multiplication can be reduced to square matrix multiplication.

Lemma 6. *For* $a, b \geqslant n$*, the product of two matrices* $A \in \mathbb{F}^{a\times n}$ *and* $B \in \mathbb{F}^{n\times b}$ *can be computed in time* $O(abn^{\omega-2})$.

Proof. Partition A and B into $n \times n$-matrices $A_1, \dots, A_{a/n}$ and $B_1, \dots, B_{b/n}$. Then, to compute $A \cdot B$, it suffices to compute $A_i \cdot B_j$, for all $i \in [a/n]$ and $j \in [b/n]$. The resulting running time is

$$O\left(\frac{a}{n} \cdot \frac{b}{n} \cdot n^{\omega}\right) = O(abn^{\omega-2}).$$

□

The next lemma shows how one can compute the value of a specific function for all inputs using fast matrix multiplication algorithms.

Lemma 7. *Let* $q \geqslant k$ *and* $A, B, C \in \mathbb{F}^{q\times k}$. *Let also* $f\colon [k]^3 \to \mathbb{F}$ *be defined as follows:*

$$f(i,j,m) = \sum_{l\in[q]} A[l,i]B[l,j]C[l,m].$$

One can compute $f(i,j,m)$*, for all* $(i,j,m) \in [k]^3$ *simultaneously, in time* $O(q^2k^{\omega-1})$.

Proof. Fix $i \in [k]$ and let $D \in \mathbb{F}^{q \times k}$ be defined by $D[l,j] = B[l,j] \cdot A[l,i]$. Then,

$$f(i,j,m) = \sum_{l \in [q]} D[l,j]C[l,m] = (D^T \cdot C)[j,m].$$

The product $D^T \cdot C$ can be computed in time $O(q^2 k^{\omega-2})$ using Lemma 6. Summing over all $i \in [k]$, gives the desired upper bound. □

Now, we are ready to prove the main result of this section.

Theorem 10. *Let* $r \geqslant \sqrt{n}$, $q \geqslant n$, *and* $k = n^{O(1)} 2^{n/4}$. *There exist functions* $g_1 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k}$ *and* $g_2 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k \times k}$ *computable in time* $k^{O(1)}$ *such that, if, for any* e, $g_1(e)$ *has* r*-rigidity* s *and* $g_2(e)$ *has rank at most* q, *then,* MAX-3-SAT *for formulas with* n *variables can be solved in co-nondeterministic time*

$$O(k^3 r + q^2 k^{\omega-1} + k^2 s + q r^2 k^{\omega-1}).$$

Proof. Take a 3-CNF formula over n variables and an integer t and transform it, using Theorem 9, into a 4-partite 3-uniform hypergraph with parts H_0, H_1, H_2, H_3 of size $k = n^{O(1)} 2^{\frac{n}{4}}$. Recall that G contains a 4-clique if and only if one can satisfy t clauses of F. Let $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \mathcal{A}_3 \in \{0,1\}^{k \times k \times k}$ be three-dimensional tensors encoding the edges of G. $\mathcal{A}_i$ is responsible for storing edges spanning nodes from all parts except for H_i: for example, G has an edge (u_1, u_2, u_3), where $u_1 \in H_1$, $u_2 \in H_2$, and $u_3 \in H_3$, if and only if $\mathcal{A}_0[u_1, u_2, u_3] = 1$. Let

$$R = \sum_{j_0, j_1, j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] \cdot \mathcal{A}_1[j_2, j_3, j_0] \cdot \mathcal{A}_2[j_3, j_0, j_1] \cdot \mathcal{A}_3[j_0, j_1, j_2].$$

Then, G has a 4-clique if and only if $R > 0$. Further, let

$$R_{j_0} = \sum_{j_1, j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] \cdot \mathcal{A}_1[j_2, j_3, j_0] \cdot \mathcal{A}_2[j_3, j_0, j_1] \cdot \mathcal{A}_3[j_0, j_1, j_2].$$

Then, $R = R_1 + \cdots + R_k$. For fixed j_0, let $M, L, T \in \{0,1\}^{k \times k}$ be defined by

$$\begin{aligned} M[j_2, j_3] &= \mathcal{A}_1[j_2, j_3, j_0], \\ L[j_3, j_1] &= \mathcal{A}_2[j_3, j_0, j_1], \\ T[j_1, j_2] &= \mathcal{A}_3[j_0, j_1, j_2]. \end{aligned}$$

Thus,

$$R_{j_0} = \sum_{j_1, j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] M[j_2, j_3] L[j_3, j_1] T[j_1, j_2].$$

Note that $\mathcal{A}_0$ is a three-dimensional tensor, whereas M, L, and T are matrices. Assume that

$\operatorname{rank}(\mathcal{A}_0) \leqslant q$ and that each of M, L, and T has r-rigidity s, that is,

$$\mathcal{A}_0[j_1, j_2, j_3] = \sum_{i \in [q]} a_{i,0}[j_1] b_{i,0}[j_2] c_{i,0}[j_3],$$

$$M = R_M + S_M = \left(\sum_{m \in [r]} a_{m,M} \cdot b_{m,M}^T \right) + S_M,$$

$$L = R_L + S_L = \left(\sum_{l \in [r]} a_{l,L} \cdot b_{l,L}^T \right) + S_L,$$

$$T = R_T + S_T = \left(\sum_{t \in [r]} a_{t,T} \cdot b_{t,T}^T \right) + S_T,$$

where $R_M, S_M, R_L, S_L, R_T, S_T \in \mathbb{F}^{k \times k}$, $|S_M| \leqslant s$, $|S_L| \leqslant s$, $|S_T| \leqslant s$, and $\operatorname{rank}(R_M) \leqslant r$, $\operatorname{rank}(R_L) \leqslant r$, $\operatorname{rank}(R_T) \leqslant r$.

We guess such a representation of $\mathcal{A}_0$, M, L, and T (guessing only nonzero entries in S_M, S_L, and S_T). For fixed j_0, one can verify the representations of M, L, and T in time $O(k^2 r)$. For all j_0, this gives $O(k^3 r)$. Since $\mathcal{A}_0$ does not depend on j_0, it suffices to guess and verify its decomposition just once. Verifying the decomposition of $\mathcal{A}_0$ can be dome in time $O(q^2 k^{\omega-1})$ using Lemma 7.

Therefore,

$$R_{j_0} = \sum_{j_1, j_2, j_3 \in [k]} \mathcal{A}_0[j_1, j_2, j_3] \cdot R_M[j_2, j_3] \cdot R_L[j_3, j_1] \cdot R_T[j_1, j_2] + \\ + \underbrace{\sum_{j_1, j_2, j_3 \in [k]} S_M[j_2, j_3] \cdot f_M(j_1, j_2, j_3) + S_L[j_3, j_1] \cdot f_L(j_1, j_2, j_3) + S_T[j_1, j_2] \cdot f_T(j_1, j_2, j_3),}_{R'_{j_0}}$$

for some functions f_M, f_L, and f_T, which are linear combinations of elements of $\mathcal{A}_0[j_1, j_2, j_3]$, $R_M[j_2, j_3]$, $S_M[j_2, j_3]$, $R_L[j_3, j_1]$, $S_L[j_3, j_1]$, $R_T[j_1, j_2]$, and $S_T[j_1, j_2]$, and each of which can be computed in $O(1)$ time at any specific point. Since $|S_M|, |S_L|, |S_T| \leqslant s$, we can compute R'_{j_0} in time $O(ks)$, for each j_0, and in time $O(k^2 s)$ overall, as each of S_M, S_L, and S_T has at most s nonzero elements.

To compute $R_{j_0} - R'_{j_0}$, it suffices to compute

$$\sum_{j_1,j_2,j_3\in[k]} \mathcal{A}_0[j_1,j_2,j_3] \cdot R_M[j_2,j_3] \cdot R_L[j_3,j_1] \cdot R_T[j_1,j_2] =$$

$$\sum_{j_1,j_2,j_3\in[k]} \left(\sum_{i\in[q]} a_{i,0}[j_1] \cdot b_{i,0}[j_2] \cdot c_{i,0}[j_3]\right) \cdot \left(\sum_{m\in[r]} a_{m,M}[j_2] \cdot b_{m,M}[j_3]\right) \cdot$$

$$\cdot \left(\sum_{l\in[r]} a_{l,L}[j_3] \cdot b_{l,L}[j_1]\right) \cdot \left(\sum_{t\in[r]} a_{t,T}[j_1] \cdot b_{t,T}[j_2]\right) =$$

$$\sum_{m,l,t\in[r]} \sum_{i\in[q]} \underbrace{\left(\sum_{j_1\in[k]} a_{i,0}[j_1] \cdot b_{l,L}[j_1] \cdot a_{t,T}[j_1]\right)}_{h_1(i,l,t)} \cdot \underbrace{\left(\sum_{j_2\in[k]} b_{i,0}[j_2] \cdot a_{m,M}[j_2] \cdot b_{t,T}[j_2]\right)}_{h_2(i,t,m)} \cdot$$

$$\cdot \underbrace{\left(\sum_{j_3\in[k]} c_{i,0}[j_3] \cdot b_{m,M}[j_3] \cdot a_{l,L}[j_3]\right)}_{h_3(i,m,l)} =$$

$$\sum_{m,l,t\in[r]} \sum_{i\in[q]} h_1(i,l,t) \cdot h_2(i,t,m) \cdot h_3(i,m,l),$$

where $h_1, h_2, h_3 \colon [q] \times [r] \times [r] \to \mathbb{F}$ are defined as follows:

$$h_1(i,l,t) = \sum_{j_1\in[k]} a_{i,0}[j_1] \cdot b_{l,L}[j_1] \cdot a_{t,T}[j_1],$$
$$h_2(i,t,m) = \sum_{j_2\in[k]} b_{i,0}[j_2] \cdot a_{m,M}[j_2] \cdot b_{t,T}[j_2],$$
$$h_3(i,m,l) = \sum_{j_3\in[k]} c_{i,0}[j_3] \cdot b_{m,M}[j_3] \cdot a_{l,L}[j_3].$$

We compute h_1, h_2, and h_3 for all inputs simultaneously, then evaluate the sum using these precomputed values. Assuming the values of h_1, h_2, and h_3 are computed already, the sum can be computed as follows.

$$\sum_{i\in[r]} \sum_{m,l,t\in[r]} h_1(i,l,t) \cdot h_2(i,t,m) \cdot h_3(i,m,l) =$$

$$\sum_{i\in[r]} \sum_{l,m\in[r]} h_3(i,m,l) \cdot \left(\sum_{t\in[r]} h_1(i,l,t) \cdot h_2(i,t,m)\right).$$

Let $V_i, Z_i \in \mathbb{F}^{r\times r}$ be two matrices, such that $V_i[l,t] = h_1(l,t)$ and $Z_i[t,m] = h_2(i,t,m)$, hence we need to evaluate:

$$\sum_{i\in[r]}\sum_{l,m\in[r]} h_3(i,m,l)\cdot\left(\sum_{t\in[r]} V_i[l,t]\cdot Z_i[t,m]\right) = \sum_{i\in[r]}\sum_{l,m\in[r]} h_3(i,m,l)\cdot(V_i\cdot Z_i)[l,m].$$

Hence, we can calculate $V_i \cdot Z_i$ in time $O(r^\omega q)$ for all $i \in [q]$ and then evaluate the sum in time $O(r^2 q)$ for fixed j_0 and $O(kr^\omega q)$ time overall.

Below, we show how to compute h_1 for all inputs. For h_2 and h_3, this can be done in the same fashion. Let $P \in \mathbb{F}^{q\times k}$ and $W \in \mathbb{F}^{(r\times r)\times k}$ be defined by $P[i,j_1] = a_{i,0}[j_1]$ and $W[(l,t),j_1] = b_{l,L}[j_1]\cdot a_{t,T}[j_1]$. Then,

$$h_1(i,l,t) = \sum_{j_1\in[k]} P[i,j_1]\cdot W[(l,t),j_1] = (P\cdot W^T)[i,(l,t)].$$

Lemma 6 guarantees that this can be computed in time $O(qr^2k^{\omega-2})$, for each j_0, and in overall time $O(qr^2k^{\omega-1})$. Thus, the overall running time of the described algorithm is

$$O(k^3r + q^2k^{\omega-1} + k^2s + kr^\omega q + qr^2k^{\omega-1}) = O(k^3r + q^2k^{\omega-1} + k^2s + qr^2k^{\omega-1}).$$

□

The following theorem provides a trade-off between matrix rigidity and tensor rank.

Theorem 11. *Let $\alpha \in [0.5, 1]$ and $\beta \in [1, 1.5]$ be constants satisfying $\beta \leqslant \frac{5-\omega}{2}$ and $\alpha \leqslant \frac{5-\beta-\omega}{2}$. If* MAX-3-SAT *cannot be solved in co-nondeterministic time $O(2^{(1-\varepsilon)n})$, for any $\varepsilon > 0$, then, for all $\delta > 0$, there exist functions $g_1\colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k}$ and $g_2\colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k\times k\times k}$ computable in time $k^{O(1)}$, such that, for infinitely many k, at least one of the following is satisfied, for at least one e:*

- $g_1(e)$ *has $k^{\alpha-\delta}$-rigidity $k^{2-\delta}$;*
- $\mathrm{rank}(g_2(e))$ *is at least $k^{\beta-\delta}$.*

Proof. Using Theorem 4, one can construct a generator g_1 that outputs $k \times k$ matrices with $k^{\frac{1}{2}-\delta}$-rigidity $k^{2-\delta}$ for $k = n^{O(1)}2^{n/4}$. Hence, from now on, we are interested in rigidity at least $k^{0.5}$.

Let g_1 be the function that outputs matrices M, L, and T from Theorem 10. Thus, g_1 takes a 3-CNF formula, t, j_0, and a number from $\{0,1,2\}$ as input. The function g_2 outputs a three-dimensional tensor $\mathcal{A}_0$, where the input to g_2 is a 3-CNF formula and t. If the input to the generators is not valid, they output empty families. Both these generators have a seed of size $n^{O(1)}$, which corresponds to $\log^{O(1)} k$ and work in time polynomial in k. Assuming that all matrices $g_1(s)$ have r-rigidity s and all tensors $g_2(s)$ have rank at most q, implies, using Theorem 10, that MAX-3-SAT can be solved in co-nondeterministic time

$$O(k^3r + q^2k^{\omega-1} + k^2s + qr^2k^{\omega-1}).$$

Let $s = k^{2-\delta}$, $r = k^{\alpha-\delta}$, and $q = k^{\beta-\delta}$. Then, the inequalities $\beta \leqslant \frac{5-\omega}{2}$ and $\alpha \leqslant 1 - \frac{\beta}{3}$ imply that the resulting algorithm solves MAX-3-SAT in co-nondeterministic time $O(k^{4-\varepsilon}) = O(2^{(1-\varepsilon)n})$:

$$O(k^3 k^{\alpha-\delta} + k^{(5-\omega)+\omega-1-2\delta}) + k^{4-\delta} + k^{\beta+(5-\beta-\omega)+\omega-3\delta}) = O(k^{4-\delta}).$$

□

One way to balance matrix rigidity and tensor rank is to ensure that a tensor has superlinear rank while maximizing matrix rigidity, as demonstrated in the next corollary.

Corollary 8. *If* MAX-3-SAT *cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$ *for any* $\varepsilon > 0$*, then for all* $\delta > 0$ *there are two polynomial time generators* $g_1 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k}$ *and* $g_2 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k \times k}$ *such that for infinitely many* k *at least one of the following is satisfied:*

- $g_1(s)$ *has* $k^{2-\frac{\omega}{2}-\delta}$*-rigidity* $k^{2-\delta}$ *, for some* s.
- $\operatorname{rank}(g_2(s))$ *is at least* $k^{1+\delta}$*, for some* s.

One can also improve matrix rigidity in our trade-off by conditioning on the value of ω.

Theorem 5. *If, for any* $\varepsilon > 0$ MAX-3-SAT*, cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$*, then, for all* $\delta > 0$ *and some* $\Delta > 0$*, there are two generators* $g_1 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k}$ *and* $g_2 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k \times k}$ *computable in time polynomial in* k *such that, for infinitely many* k*, at least one of the following is satisfied:*

- $g_1(s)$ *has* $k^{1-\delta}$*-rigidity* $k^{2-\delta}$*, for some* s*;*
- $\operatorname{rank}(g_2(s))$ *is at least* $k^{1+\Delta}$*, for some* s.

Proof. We modify the second generator in Corollary 8 by adding a new input on which the generator will output a tensor of matrix multiplication $\mathcal{A}_{\sqrt{k}}$ of size $k \times k \times k$. If $\omega = 2$, then the statement follows from Corollary 8. If $\omega \geqslant 2 + 2\Delta$ for some $\Delta > 0$, then for infinitely many k, we have $\operatorname{rank}(\mathcal{A}_{\sqrt{k}}) \geqslant k^{1+\delta}$. □

If one wants to obtain improved matrix rigidity under weaker assumptions, then the following corollary holds, similar to Corollary 7.

Corollary 9. *If* MAX-3-SAT *cannot be solved in co-nondeterministic time* $O(2^{0.85n})$*, then, for some* $\Delta > 0$*, there exist two generators* $g_1 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k}$ *and* $g_2 \colon \{0,1\}^{\log^{O(1)} k} \to \mathbb{F}^{k \times k \times k}$*, computable in time polynomial in* k*, such that, for infinitely many* k*, at least one of the following conditions holds:*

- $g_1(s)$ *has* $k^{0.7}$*-rigidity* $k^{1.4}$*, for some* s*;*
- $\operatorname{rank}(g_2(s))$ *is at least* $k^{1+\Delta}$*, for some* s.

We are now ready to prove our conditional answer to the open problem from [GT18].

Corollary 4. *If, for every* $\varepsilon > 0$, MAX-3-SAT *cannot be solved in co-nondeterministic time* $O(2^{(1-\varepsilon)n})$*, then, for any* $\delta > 0$*, one can construct an explicit family of* $2^{\log^{O(1)} n}$ *functions such that, for infinitely many* n*, at least one of them is either bilinear and requires canonical circuits of size* $2^{\Omega(n^{2/3-\delta})}$ *or trilinear and requires arithmetic circuits of size* $\Omega(n^{1.25})$.

Proof. It suffices to verify that $\alpha = \frac{2}{3}$ and $\beta = 1.25$ satisfy the inequalities in the statement of Theorem 11 for any possible value of $\omega < 2.38$. □

Acknowledgments

We are grateful to the anonymous reviewers for many helpful comments and additional references that allowed us not only to improve the writing, but also to prove stronger versions of some of our results.

References

[AB87] Noga Alon and Ravi B. Boppana. The monotone circuit complexity of Boolean functions. *Comb.*, 7(1):1–22, 1987.

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

[AFT11] Boris Alexeev, Michael A. Forbes, and Jacob Tsimerman. Tensor rank: Some lower and upper bounds. In *CCC*, pages 283–291. IEEE Computer Society, 2011.

[AL25] Josh Alman and Jingxun Liang. Low rank matrix rigidity: Tight lower bounds and hardness amplification. *CoRR*, abs/2502.19580, 2025.

[And85] Alexander E. Andreev. A method for obtaining lower bounds on the complexity of individual monotone functions. *Doklady Akademii Nauk*, 282(5):1033–1037, 1985.

[And87] Alexander E. Andreev. A method for obtaining efficient lower bounds for monotone complexity. *Algebra and Logic*, 26(1):1–18, 1987.

[AV24] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. *TheoretiCS*, 3, 2024.

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

[BCH$^+$24] Andreas Björklund, Radu Curticapean, Thore Husfeldt, Petteri Kaski, and Kevin Pratt. Chromatic number in 1.9999^n time? Fast deterministic set partitioning under the asymptotic rank conjecture. *CoRR*, abs/2404.04987, 2024.

[BGKM23] Vishwas Bhargava, Sumanta Ghosh, Mrinal Kumar, and Chandra Kanta Mohapatra. Fast, algebraic multivariate multipoint evaluation in small characteristic and applications. *J. ACM*, 70(6):42:1–42:46, 2023.

[BHPT24] Amey Bhangale, Prahladh Harsha, Orr Paradise, and Avishay Tal. Rigid matrices from rectangular PCPs. *SIAM J. Comput.*, 53(2):480–523, 2024.

[BK24] Andreas Björklund and Petteri Kaski. The asymptotic rank conjecture and the set cover conjecture are not both true. In *STOC*, pages 859–870. ACM, 2024.

[BKM$^+$24] Tatiana Belova, Alexander S. Kulikov, Ivan Mihajlin, Olga Ratseeva, Grigory Reznikov, and Denil Sharipov. Computations with polynomial evaluation oracle: ruling out superlinear SETH-based lower bounds. In *SODA*, pages 1834–1853. SIAM, 2024.

[Blä99] Markus Bläser. A $\frac{5}{2}n^2$-lower bound for the rank of $n \times n$ matrix multiplication over arbitrary fields. In *FOCS*, pages 45–50. IEEE Computer Society, 1999.

[Blä13] Markus Bläser. Fast matrix multiplication. *Theory Comput.*, 5:1–60, 2013.

[BS83] Walter Baur and Volker Strassen. The complexity of partial derivatives. *Theor. Comput. Sci.*, 22:317–330, 1983.

[Bsh89] Nader H. Bshouty. A lower bound for matrix multiplication. *SIAM J. Comput.*, 18(4):759–765, 1989.

[CDI$^+$13] Gil Cohen, Ivan Bjerre Damgård, Yuval Ishai, Jonas Kölker, Peter Bro Miltersen, Ran Raz, and Ron D. Rothblum. Efficient multiparty protocols via log-depth threshold formulae - (extended abstract). In *CRYPTO (2)*, volume 8043 of *Lecture Notes in Computer Science*, pages 185–202. Springer, 2013.

[CGI$^+$16] Marco L. Carmosino, Jiawei Gao, Russell Impagliazzo, Ivan Mihajlin, Ramamohan Paturi, and Stefan Schneider. Nondeterministic extensions of the strong exponential time hypothesis and consequences for non-reducibility. In *ITCS*, pages 261–270. ACM, 2016.

[CHR24] Lijie Chen, Shuichi Hirahara, and Hanlin Ren. Symmetric exponential time requires near-maximum circuit size. In *STOC*, pages 1990–1999. ACM, 2024.

[CKR22] Bruno Pasqualotto Cavalar, Mrinal Kumar, and Benjamin Rossman. Monotone circuit lower bounds from robust sunflowers. *Algorithmica*, 84(12):3655–3685, 2022.

[CL21] Lijie Chen and Xin Lyu. Inverse-exponential correlation bounds and extremely rigid matrices from a new derandomized XOR lemma. In *STOC*, pages 761–771. ACM, 2021.

[CRTY23] Lijie Chen, Ron D. Rothblum, Roei Tell, and Eylon Yogev. On exponential-time hypotheses, derandomization, and circuit lower bounds. *J. ACM*, 70(4):25:1–25:62, 2023.

[DE19] Zeev Dvir and Benjamin L. Edelman. Matrix rigidity and the Croot-Lev-Pach Lemma. *Theory Comput.*, 15:1–7, 2019.

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

[DL20] Zeev Dvir and Allen Liu. Fourier and circulant matrices are not rigid. *Theory Comput.*, 16:1–48, 2020.

[Dvi11] Zeev Dvir. On matrix rigidity and locally self-correctable codes. *Comput. Complex.*, 20(2):367–388, 2011.

[FGHK16] Magnus Gausdal Find, Alexander Golovnev, Edward A. Hirsch, and Alexander S. Kulikov. A better-than-$3n$ lower bound for the circuit complexity of an explicit function. In *FOCS*, pages 89–98. IEEE Computer Society, 2016.

[FM98] Matthias Fitzi and Ueli M. Maurer. Efficient byzantine agreement secure against general adversaries. In *DISC*, volume 1499 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 1998.

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

[GGKS20] Ankit Garg, Mika Göös, Pritish Kamath, and Dmitry Sokolov. Monotone circuit lower bounds from resolution. *Theory Comput.*, 16:1–30, 2020.

[GGNS23] Karthik Gajulapalli, Alexander Golovnev, Satyajeet Nagargoje, and Sidhant Saraogi. Range avoidance for constant depth circuits: Hardness and algorithms. In *APPROX/RANDOM*, volume 275 of *LIPIcs*, pages 65:1–65:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[GK98] Dima Grigoriev and Marek Karpinski. An exponential lower bound for depth 3 arithmetic circuits. In *STOC*, pages 577–582. ACM, 1998.

[GKRS19] Mika Göös, Pritish Kamath, Robert Robere, and Dmitry Sokolov. Adventures in monotone complexity and TFNP. In *ITCS*, volume 124 of *LIPIcs*, pages 38:1–38:19. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.

[GKW21] Alexander Golovnev, Alexander S. Kulikov, and R. Ryan Williams. Circuit depth reductions. In *ITCS*, volume 185 of *LIPIcs*, pages 24:1–24:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021.

[GLW22] Venkatesan Guruswami, Xin Lyu, and Xiuhan Wang. Range avoidance for low-depth circuits and connections to pseudorandomness. In *APPROX/RANDOM*, volume 245 of *LIPIcs*, pages 20:1–20:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[Gol22] Oded Goldreich. Improved bounds on the AN-complexity of $O(1)$-linear functions. *Comput. Complex.*, 31(2):7, 2022.

[GR98] Dima Grigoriev and Alexander Razborov. Exponential complexity lower bounds for depth 3 arithmetic circuits in algebras of functions over finite fields. In *FOCS*, pages 269–278. IEEE Computer Society, 1998.

[Gri80] Dima Grigoriev. Application of separability and independence notions for proving lower bounds of circuit complexity. *Journal of Soviet Mathematics*, 14:1450–1457, 1980.

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

[GW20] Oded Goldreich and Avi Wigderson. On the size of depth-three Boolean circuits for computing multilinear functions. In *Computational Complexity and Property Testing*, volume 12050 of *Lecture Notes in Computer Science*, pages 41–86. Springer, 2020.

[Hås89] Johan Håstad. Almost optimal lower bounds for small depth circuits. *Adv. Comput. Res.*, 5:143–170, 1989.

[Hås90] Johan Håstad. Tensor rank is NP-complete. *J. Algorithms*, 11(4):644–654, 1990.

[HM97] Martin Hirt and Ueli M. Maurer. Complete characterization of adversaries tolerable in secure multi-party computation (extended abstract). In *PODC*, pages 25–34. ACM, 1997.

[HM00] Martin Hirt and Ueli M. Maurer. Player simulation and general adversary structures in perfect multiparty computation. *J. Cryptol.*, 13(1):31–60, 2000.

[HR00] Danny Harnik and Ran Raz. Higher lower bounds on monotone size. In *STOC*, pages 378–387. ACM, 2000.

[HS65] Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[HS66] F. C. Hennie and Richard E Stearns. Two-tape simulation of multitape turing machines. *J. ACM*, 13(4):533–546, 1966.

[IP01] Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001.

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[JMV18] Hamidreza Jahanjou, Eric Miles, and Emanuele Viola. Local reduction. *Inf. Comput.*, 261:281–295, 2018.

[Juk99] Stasys Jukna. Combinatorics of monotone computations. *Comb.*, 19(1):65–85, 1999.

[Juk12] Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012.

[KL80] Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *STOC*, pages 302–309. ACM, 1980.

[Kor21] Oliver Korten. The hardest explicit construction. In *FOCS*, pages 433–444. IEEE, 2021.

[KP22] Alexander Kozachinskiy and Vladimir V. Podolskii. Multiparty Karchmer-Wigderson games and threshold circuits. *Theory Comput.*, 18:1–33, 2022.

[Lan14] J. M. Landsberg. New lower bounds for the rank of matrix multiplication. *SIAM J. Comput.*, 43(1):144–149, 2014.

[Li24] Zeyong Li. Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform. In *STOC*, pages 2000–2007. ACM, 2024.

[LMM$^+$22] Shachar Lovett, Raghu Meka, Ian Mertz, Toniann Pitassi, and Jiapeng Zhang. Lifting with sunflowers. In *ITCS*, volume 215 of *LIPIcs*, pages 104:1–104:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022.

[Lok09] Satyanarayana V. Lokam. Complexity lower bounds using linear algebra. *Found. Trends Theor. Comput. Sci.*, 4(1-2):1–155, 2009.

[LVW18] Andrea Lincoln, Virginia Vassilevska Williams, and R. Ryan Williams. Tight hardness for shortest cycles and paths in sparse graphs. In *SODA*, pages 1236–1252. SIAM, 2018.

[LY22] Jiatu Li and Tianqi Yang. $3.1n - o(n)$ circuit lower bounds for explicit functions. In *STOC*, pages 1180–1193. ACM, 2022.

[MR13] Alex Massarenti and Emanuele Raviolo. The rank of $n \times n$ matrix multiplication is at least $3n^2 - 2\sqrt{2}n^{\frac{3}{2}} - 3n$. *Linear Algebra and its Applications*, 438(11):4500–4509, 2013.

[MR14] Alex Massarenti and Emanuele Raviolo. Corrigendum to "The rank of $n \times n$ matrix multiplication is at least $3n^2 - 2\sqrt{2}n^{\frac{3}{2}} - 3n$". *Linear Algebra and its Applications*, 445:369–371, 2014.

[Ned20] Jesper Nederlof. Bipartite TSP in $O(1.9999^n)$ time, assuming quadratic time matrix multiplication. In *STOC*, pages 40–53. ACM, 2020.

[PF79] Nicholas Pippenger and Michael J. Fischer. Relations among complexity measures. *J. ACM*, 26(2):361–381, 1979.

[PPSZ05] Ramamohan Paturi, Pavel Pudlák, Michael E. Saks, and Francis Zane. An improved exponential-time algorithm for k-SAT. *J. ACM*, 52(3):337–364, 2005.

[PR17] Toniann Pitassi and Robert Robere. Strongly exponential lower bounds for monotone computation. In *STOC*, pages 1246–1255. ACM, 2017.

[Pra24] Kevin Pratt. A stronger connection between the asymptotic rank conjecture and the set cover conjecture. In *STOC*, pages 871–874. ACM, 2024.

[PV91] Pavel Pudlák and Zdeněk Vavřín. Computation of rigidity of order $\frac{n^2}{r}$ for one simple matrix. *Commentationes Mathematicae Universitatis Carolinae*, 32(2):213–218, 1991.

[Ram20] C. Ramya. Recent progress on matrix rigidity - A survey. *CoRR*, abs/2009.09460, 2020.

[Raz85] Alexander Razborov. Lower bounds on the monotone complexity of some Boolean function. In *Soviet Math. Dokl.*, volume 31, pages 354–357, 1985.

[Raz87] Alexander Razborov. Lower bounds for the size of circuits of bounded depth with basis (AND, XOR). *Math. notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

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

[Raz03] Ran Raz. On the complexity of matrix product. *SIAM J. Comput.*, 32(5):1356–1369, 2003.

[RR20] Sivaramakrishnan Natarajan Ramamoorthy and Cyrus Rashtchian. Equivalence of systematic linear data structures and matrix rigidity. In *ITCS*, volume 151 of *LIPIcs*, pages 35:1–35:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

[RS03] Ran Raz and Amir Shpilka. Lower bounds for matrix product in bounded depth circuits with arbitrary gates. *SIAM J. Comput.*, 32(2):488–513, 2003.

[Sch81] Arnold Schönhage. Partial and total matrix multiplication. *SIAM J. Comput.*, 10(3):434–455, 1981.

[Sha49] Claude E. Shannon. The synthesis of two-terminal switching circuits. *Bell Syst. Tech. J.*, 28(1):59–98, 1949.

[Shp03] Amir Shpilka. Lower bounds for matrix product. *SIAM J. Comput.*, 32(5):1185–1200, 2003.

[Smo87] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *STOC*, pages 77–82. ACM, 1987.

[SSS97] Mohammad Amin Shokrollahi, Daniel A. Spielman, and Volker Stemann. A remark on matrix rigidity. *Inf. Process. Lett.*, 64(6):283–285, 1997.

[Str69] Volker Strassen. Gaussian elimination is not optimal. *Numerische mathematik*, 13(4):354–356, 1969.

[Str73] Volker Strassen. Die berechnungskomplexität von elementarsymmetrischen funktionen und von interpolationskoeffizienten. *Numerische Mathematik*, 20:238–251, 1973.

[Str75] Volker Strassen. Die berechnungskomplexität der symbolischen differentiation von interpolationspolynomen. *Theor. Comput. Sci.*, 1(1):21–25, 1975.

[Str86] Volker Strassen. The asymptotic spectrum of tensors and the exponent of matrix multiplication. In *FOCS*, pages 49–54. IEEE Computer Society, 1986.

[SW99] Amir Shpilka and Avi Wigderson. Depth-3 arithmetic formulae over fields of characteristic zero. In *CCC*, page 87. IEEE Computer Society, 1999.

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

[Vas18] Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the international congress of mathematicians: Rio de janeiro 2018*, pages 3447–3487. World Scientific, 2018.

[VK22] Ben Lee Volk and Mrinal Kumar. A polynomial degree bound on equations for non-rigid matrices and small linear circuits. *ACM Trans. Comput. Theory*, 14(2):6:1–6:14, 2022.

[VXXZ24] Virginia Vassilevska Williams, Yinzhan Xu, Zixuan Xu, and Renfei Zhou. New bounds for matrix multiplication: from alpha to omega. In *SODA*, pages 3792–3835. SIAM, 2024.

[Wil05] R. Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005.

[Wil07] R. Ryan Williams. *Algorithms and resource requirements for fundamental problems*. PhD thesis, Carnegie Mellon University, 2007.

[Wil13] R. Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013.

[Wil24] Ryan Williams. The orthogonal vectors conjecture and non-uniform circuit lower bounds. In *FOCS*, pages 1372–1387. IEEE, 2024.

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