Fine-Grained Hardness for Edit Distance to a Fixed Sequence

Amir Abboud
Weizmann Institute of Science, Rehovot, Israel

Virginia Vassilevska Williams
MIT, Cambridge, MA, US

Abstract

Nearly all quadratic lower bounds conditioned on the Strong Exponential Time Hypothesis (SETH) start by reducing k-SAT to the Orthogonal Vectors (OV) problem: Given two sets A,B of n binary vectors, decide if there is an orthogonal pair $a \in A, b \in B$. In this paper, we give an alternative reduction in which the set A does not depend on the input to k-SAT; thus, the quadratic lower bound for OV holds even if one of the sets is fixed in advance.

Using the reductions in the literature from OV to other problems such as computing similarity measures on strings, we get hardness results of a stronger kind: there is a family of sequences $\{S_n\}_{n=1}^{\infty}$, $|S_n| = n$ such that computing the Edit Distance between an input sequence X of length n and the (fixed) sequence S_n requires $n^{2-o(1)}$ time under SETH.

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

Keywords and phrases SAT, edit distance, fine-grained complexity, conditional lower bound, sequence alignment

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.7

Category Track A: Algorithms, Complexity and Games

Funding *Amir Abboud*: Work partly done at the IBM Almaden Research Center.
Virginia Vassilevska Williams: Supported by an NSF CAREER Award, NSF Grants CCF-1528078, CCF-1514339 and CCF-1909429, a BSF Grant BSF:2012338, a Google Research Fellowship and a Sloan Research Fellowship.

1 Introduction

The first step in nearly all hardness proofs for polynomial time problems, that are conditioned on the Strong Exponential Time Hypothesis (SETH), is a seminal reduction of Williams [42] from k-SAT to the Orthogonal Vectors problem.

▶ **Definition 1** (The OV Problem). *Given two sets $A, B \subseteq \{0,1\}^{d(n)}$ of n binary vectors of dimension $d(n)$, decide if there is a pair $a \in A, b \in B$ that are orthogonal, i.e. $\forall i \in [d(n)] : a[i] = 0 \vee b[i] = 0$.*

The SETH states that k-SAT cannot be solved in $(2-\varepsilon)^n$ time, where $\varepsilon > 0$ is independent of k. The aforementioned reduction takes such a k-CNF formula on n variables and produces two sets of $N = 2^{n/2}$ binary vectors in $d(N) = O(\log N)$ dimensions. Thus, a subquadratic $O(N^{2-\varepsilon})$ algorithm for OV gives a $(2-\varepsilon')^n$ algorithm for k-SAT and refutes SETH. The current best algorithms for OV are mildly subquadratic $O(N^2/f(N))$ where $f(N) = N^{o(1)}$ [11, 25].

It turns out that OV is at the core of so many other problems, making their complexity quadratic. Dozens of fine-grained reductions from OV to various important problems from diverse fields have been designed in the last decade, resulting in a long list of quadratic SETH-based lower bounds [40]. For example, to prove their $n^{2-o(1)}$ lower bound for the Edit

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).
Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 7; pp. 7:1–7:14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Distance problem, Backurs and Indyk [14] encode each set of vectors A, B with a sequence S_A, S_B of length $O(nd(n))$, such that the edit distance between them reveals the existence of an orthogonal pair.

1.1 Results

In this paper we revisit the simple reduction of Williams from k-SAT to OV, that has been presented in countless lectures on fine-grained complexity, and expose a surprising room for improvement: there is an alternative reduction (with vectors of a mildly larger dimension) that encodes the k-CNF formula with *only one* of the sets. That is, the set of vectors A is *fixed* in all the instances produced by the new reduction. The implications for the SETH-hard problems may sound bizarre: it takes n^2 time to compute the Edit Distance even if one of the two sequences is always the same (for all inputs of length n). Before discussing the implications further, let us try to clarify the result with a high level technical overview.

1.1.1 Main idea

The difference between the two reductions is simple to explain. Assume we are given a k-CNF formula ϕ on n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$ such that $\phi = (C_1 \wedge \cdots \wedge C_m)$. In both reductions, we enumerate all partial assignments $\alpha \in \{\texttt{false}, \texttt{true}\}^{n/2}$ to the first half of the variables $x_1, \ldots, x_{n/2}$, and also all partial assignments $\beta \in \{\texttt{false}, \texttt{true}\}^{n/2}$ to the other half of the variables $x_{n/2+1}, \ldots, x_n$, and the goal is to find a satisfying pair, i.e. a pair α, β that when put together make a full satisfying assignment $(\alpha\beta)$.

To do this, Williams encodes each partial assignment, α or β, with a binary vector in m dimensions that has a 0 at the j^{th} coordinate if the partial assignment satisfies the clause C_j and 1 otherwise; it follows that a pair of vectors is orthogonal iff $(\alpha\beta)$ is a satisfying assignment. The vectors corresponding to the α's (the partial assignment to the first half of variables) go to the set A and the vectors corresponding to the β's go to the set B; notice that the vectors in both sets depend on the clauses of ϕ.

In the new reduction, the vectors corresponding to the α's are defined *as if ϕ has all possible clauses of size k*, i.e. as if ϕ is the complete k-CNF formula. The j^{th} coordinate depends on whether α satisfies the j^{th} clause in a certain canonical ordering of all k-CNF clauses over n variables. Thus, the set A does not depend on the (real) input formula ϕ. Then, the definition of the vectors for the β's is similar but with an extra condition: the j^{th} coordinate is automatically set to 0 if the j^{th} clause in the canonical ordering does not exist in ϕ, regardless of whether β satisfies it or not. As a result, all clauses that do not appear in ϕ cannot affect the orthogonality of any pair in $A \times B$, and the correctness of the reduction is maintained.

The only downside of the new reduction is that the size of the vectors grows from $m = O(n)$ (because of the sparsification lemma) to $O(n^{2k})$. That is, from $O(\log N)$ to $\log^{O(1)} N$. However, the dimension is still $N^{o(1)}$ and this is sufficient to deduce many $n^{2-o(1)}$ SETH-based lower bounds. Indeed, as often observed, these lower bounds can be based on the (safer than SETH) hypothesis that OV is hard when the dimension is $N^{o(1)}$.[1] Some exceptions, where the reductions crucially rely on the dimension being logarithmic, are [7, 27, 31].

To formalize the new result as a theorem, in Section 2 we introduce the $\mathrm{OV}_{\mathcal{A}}$ problem in which we are given as input only one set B of n vectors and are asked if there is an orthogonal pair in $A_n \times B$ where A_n is the n^{th} set in an (efficiently producible) family of sets $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$.

[1] Recent works [33, 5] have shown that refuting this "moderate dimension OV" hypothesis has consequences that are potentially more remarkable than refuting SETH.

1.1.2 Implications for other problems

Combining the new reduction with the existing reductions from OV in a black-box way leads to some interesting consequences. For example, the reductions from OV to the computation of certain distance measures on pairs of sequences, curves, or time-series, such as Edit Distance [14], Longest Common Subsequence [3, 22, 19], Local Alignment [9], Fréchet Distance [20], and Dynamic Time Warping Distance [22, 3], all proceed by encoding each of the sets A, B *independently* with a sequence S_A, S_B. Therefore, the quadratic $n^{2-o(1)}$ lower bounds implied by SETH hold even if only one sequence S_B is given as input, while S_A only depends on $n = |S_B|$. Another example is the regular expression matching problem [15, 21] of deciding whether a string x can be generated from a regular expression y. Again, the new reduction shows a quadratic lower bound even if the string (or the expression) are fixed. We find it surprising that such severe-looking restrictions of the problems do not reduce the complexity.

The corollaries go beyond sequence problems. The lower bounds for Bichromatic Closest Pair [12, 28] now also hold when one of the two sets is fixed; note that this is incomparable to the recent lower bound for Monochromatic Closest Pair [30]. In the Subtree Isomorphism problem we are given two rooted, unordered, unlabelled trees and are asked if one is contained in the other (a pattern and a host). The quadratic lower bound [2] can now be shown even for a fixed pattern or a fixed host. The implications are less clear-cut for many other graph problems; for example, the basic reduction from OV to diameter in sparse graphs [39] may now produce slightly simpler graphs but it is hard to characterize in what way. It is likely that a white-box usage of the new reduction will lead to interesting results; we leave this for future work.

1.1.3 Generalizing to k-OV

The SETH lower bound for OV generalizes to an $n^{k-o(1)}$ lower bound, for all $k \geq 2$, for the k-OV problem: given k sets of binary vectors $A_1, \ldots, A_k$ in d dimensions, decide if there are k vectors $a_i \in A_i$ that are orthogonal in the sense that $\forall j \in [d] : (a_1[j] \wedge \cdots \wedge a_k[j]) = 0$. The hardness of k-OV has been used to prove the hardness of several other problems (where a reduction from 2-OV is not known) [8, 10, 33, 38, 1, 16, 32, 29, 23]; e.g. an $n^{k-o(1)}$ lower bound for k-LCS, the problem of computing the longest common subsequence of k strings.

The reduction to OV extends to k-OV in the same way: instead of partitioning the variables into two sets of size $n/2$, we split them into k sets of n/k, resulting in k sets of $N = 2^{n/k}$ vectors. Applying the new idea, in Section 3, we get a reduction to instances where the first $k-1$ sets are fixed, and only one set is given as input. Consequently, the k-LCS problem has an $n^{k-o(1)}$ lower bound even if $k-1$ of the sequences are fixed.

1.1.4 Hardness for compressible instances

A surprising feature of SETH-based hardness results for problems in P is that they are proved for highly compressible instances. The reductions produce OV instances of size $N = 2^{n/2}$ that are fully determined by a k-CNF formula of size $O(n^k)$ and can therefore be losslessly encoded with $O(\log N)$ bits. This is surprising because a priori one might expect the worst case instances to require $\Omega(N)$ bits to specify. The reductions can even be adapted (with major modifications) [1] to prove the hardness for instances where the data is compressible with standard grammar-compression schemes such as the Lempel-Ziv family. The new results go a step further: the hardness holds even if one of the inputs is fully determined.

The preprocessing model

A few recent papers study the complexity of the central problems of fine-grained complexity in models with preprocessing [17, 24, 35, 36, 34]. For instance, it was shown that if we can preprocess one of the strings of an Edit Distance instance in near-linear time, then we can obtain a better approximation in sublinear time [34].

It is not difficult to get strong lower bounds for OV (and Edit Distance) in these models by combining the old reduction with a partitioning trick (similar to [41]): an algorithm that solves OV in subquadratic time $n^{2-\varepsilon}$ after preprocessing each set (separately) in arbitrary polynomial time, say n^{100}, also refutes SETH, because we can split A and B into $n^{2-1/200}$ sets of size $n^{1/200}$, preprocess each in a total of $n^{1-1/200} \cdot n^{100/200} < n^{1.5}$ and then solve OV for each pair of parts in a total of $n^{2(1-1/200)} \cdot n^{1/200(2-\varepsilon)} = n^{2-\varepsilon'}$.

The new reduction gives even more power since it implies the hardness with a fixed set, that intuitively, can be preprocessed indefinitely; formally, we get conditional lower bounds even against algorithms with preprocessing using arbitrary polynomial *space*, rather than time (see Section 5).[2] This also has implications for dynamic algorithms where a preprocessing phase is often allowed [8, 37], strengthening the lower bounds from polynomial time to space. Observe that if we relax the requirements further to allow exponential space, a linear time algorithm becomes possible: using $2^{O(n)}$ space we can construct a look-up table storing the answers to all possible inputs.

1.1.5 Generalization to Formula-SAT

Hansen, Williams, and the authors [6] have observed that Edit Distance and other problems in P are even harder than OV; if we solve them in subqadratic time, not only do we solve SAT faster on CNF formulas (the simplest kind of formulas), but we also solve it on arbitrary formulas, small depth circuits, and branching programs. Thus, the quadratic lower bounds for Edit Distance and LCS (but not OV) can be based on the safer Formula-SETH (or BP-SETH or NC-SETH): the hypothesis that SAT on arbitrary formulas of size $2^{o(n)}$ cannot be solved in $(2-\varepsilon)^n$ time [6, 4, 26]. These reductions start by reducing Formula-SAT to a problem similar in spirit to OV, called Formula-Satisfying-Pair, that can then be reduced to Edit Distance. It turns out that the new idea of encoding the formula only in one set can be applied in this case as well (see Section 6), and so all the Formula-SETH lower bounds still hold if one of the sequences is fixed.

1.1.6 Roadmap and preliminaries

We start with the new reduction from k-SAT to OV in Section 2. Then, in Section 3, we generalize it to OV on $k \geq 2$ sets. In Section 4 we explain the implication for Edit Distance. Then, we discuss conditional lower bounds for the preprocessing model in Section 5. And finally, in Section 6 we extend the new reduction to formulas beyond CNF's.

We use the notation $[n] = \{1, \ldots, n\}$ and `false`, `true` for boolean truth values.

---

[2] These results were mentioned in [34] and credited to this work as a personal communication.

2 The new reduction: k-SAT to OV with a fixed set

In this section we give the main result of the paper: a reduction from k-SAT to the Orthogonal Vectors problem (Definition 1) where one of the two sets is fixed for all inputs of size n.

To formalize this, we define the $\mathrm{OV}_{\mathcal{A}}$ problem, which is the same as OV but where the input set A is not given as input. Instead, if the input set B has size n, then we will always choose A to be the set $A_n \in \mathcal{A}$ where $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$ is a family of vector sets, containing one set of each size n. The formal definition below also incorporates the dimension of the vectors $d(n)$ which is taken to be a function of the number of vectors n, as is standard when studying OV.

▶ **Definition 2** (The $\mathrm{OV}_{\mathcal{A}}$ Problem)**.** *For a family $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$ of vector-sets, such that $A_n \subseteq \{0,1\}^{d(n)}$ is a set of n binary vectors of dimension $d(n)$, we define the $OV_{\mathcal{A}}$ problem as: Given a set B of n binary vectors of dimension $d(n)$ decide if there is a pair $a \in A_n, b \in B$ that are orthogonal, i.e. $\forall i \in [d(n)] : a[i] = 0 \vee b[i] = 0$.*

We can now give the main theorem of this paper, giving a quadratic lower bound for $\mathrm{OV}_{\mathcal{A}}$ under SETH. The family $\mathcal{A}$ for which the result holds will be clarified in the proof; we remark that it is quite natural and that is easy to produce each set A_n algorithmically in linear time.

▶ **Theorem 3** (Main)**.** *For any function $d(n) = \log^{\omega(1)} n$, there is a family of vector-sets $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$ of dimension $d(n)$ such that the $OV_{\mathcal{A}}$ problem requires $n^{2-o(1)}$ under SETH. Moreover, each set A_n can be produced in $O(n \cdot d(n))$ time and log space.*

Proof. Fix integers $n, k \geq 1$, and let $\mathcal{C}_{n,k}$ be the set of width k clauses over n variables,

$$\mathcal{C}_{n,k} = \{(\ell_1 \vee \cdots \vee \ell_k) \mid \forall i \in [k] : \ell_i \in \{x_1, \ldots, x_n\} \cup \{x_1, \ldots, x_n\}\},$$

and pick an arbitrary ordering over the clauses such that $\mathcal{C}_{n,k} = \{C_1, \ldots, C_M\}$ where $M = \binom{2n}{k} = O(n^{2k})$.

We now define the set $A_N \in \mathcal{A}$ for $N = 2^{n/2}$ as follows.[3] For each truth assignment α to the variables $x_1, \ldots, x_{n/2}$, i.e. a partial assignment, we create a vector v_α. That is, $\forall \alpha \in \{\texttt{false}, \texttt{true}\}^{n/2}$ representing the assignment $x_1 = \alpha(1), \ldots, x_{n/2} = \alpha(n/2)$, we define the vector v_α as follows:

$$\forall j \in [M] : v_\alpha[j] = \begin{cases} 0, & \text{if } \alpha \text{ satisfies the clause } C_j \\ 1, & \text{otherwise.} \end{cases}$$

That is, to set the j^{th} coordinate of v_α we check if the partial assignment α already satisfies the clause C_j (the j^{th} clause in $\mathcal{C}$) and choose 0 if so, and 1 otherwise.

Notice that the dimension of the vectors is $O(n^{2k}) = \log^{O(1)} N$ and it is asymptotically dominated by $d(N) = \log^{\omega(1)} N$. Moreover, A_N can be produced in $2^{n/2} \cdot n^{2k} = O(Nd(N))$ time, and $O(n + k \log n) = O(\log N)$ space.

We are now ready to reduce k-SAT to the $\mathrm{OV}_{\mathcal{A}}$ problem. Given a k-CNF formula ϕ on n variables $x_1, \ldots, x_n$ as input, we define a set B of $N = 2^{n/2}$ vectors as follows. For each truth assignment $\beta \in \{\texttt{false}, \texttt{true}\}^{n/2}$ to the variables $x_{n/2+1}, \ldots, x_n$, i.e. a partial assignment to the second half of the variables, we create a vector v_β such that:

$$\forall j \in [M] : v_\beta[j] = \begin{cases} 0, & \text{if either } \beta \text{ satisfies the clause } C_j\text{, or } C_j \notin \phi \\ 1, & \text{otherwise, i.e. } \mathcal{C}_j \in \phi \text{ but } \beta \text{ does not satisfy it.} \end{cases}$$

[3] While this only defines A_N for values of N that are equal to $2^{n/2}$ with integer n, it is easy to extend these A_N's into a family $\mathcal{A}$, e.g. by padding with dummy vectors that are all 1.

Thus, while v_α considered all clauses $C_j \in \mathcal{C}$ similarly, the vectors $v_\beta \in B$ depend on the input formula ϕ and automatically set to 0 all coordinates corresponding to clauses that do not appear in ϕ.

Finally, we claim that B is a "yes" $\mathrm{OV}_{\mathcal{A}}$ instance iff ϕ is satisfiable, and therefore if $\mathrm{OV}_{\mathcal{A}}$ can be solved in truly subquadratic time then for all k, k-SAT can be solved in $(2^{n/2})^{2-\varepsilon} = 2^{(1-\varepsilon/2)n}$ for some $\varepsilon > 0$, refuting SETH. The correctness follows from the following claim proved below.

▷ Claim 4. There is an orthogonal pair $a \in A_N, b \in B$ iff ϕ is satisfiable.

There exist a pair $a \in A_N, b \in B$ that are orthogonal iff there are partial assignments α, β to the first and second half of the variables, respectively, such that $v_\alpha \in A_N$ and $v_\beta \in B$ are orthogonal, meaning that for all $j \in [M]$ either $v_\alpha[j] = 0$ or $v_\beta = 0$, i.e. either α satisfies C_j or β satisfies C_j or C_j is not a clause in ϕ at all. The latter is equivalent to saying that the truth assignment $(\alpha\beta)$ composed of α and β satisfies all clauses that are in ϕ, and we conclude that there is an orthogonal pair iff ϕ has a satisfying assignment. ◀

3 Extension to k-OV

In this section we extend the result of Section 2 to the k-OV problem.

▶ **Definition 5** (The k-OV Problem). *Given k sets $A_1, \dots, A_k \subseteq \{0,1\}^{d(n)}$ of n binary vectors of dimension $d(n)$, decide if there are k vectors $a_i \in A_i$ that are orthogonal, i.e. $\forall j \in [d(n)] : a_1[j] = 0 \vee \cdots \vee a_k[j] = 0$.*

As before, we define a version of the problem in which only one of the sets is given as input and the other $k-1$ are fixed.

▶ **Definition 6** (The k-$\mathrm{OV}_{\mathcal{A}_1,\dots,\mathcal{A}_k}$ Problem). *For any $k-1$ families $\mathcal{A}_i = \{A_{i,n}\}_{n=1}^{\infty}, i \in [k-1]$ of vector-sets, such that $\forall i \in [k-1] : A_{i,n} \subseteq \{0,1\}^{d(n)}$ is a set of n binary vectors of dimension $d(n)$, we define the k-$OV_{\mathcal{A}_1,\dots,\mathcal{A}_k}$ problem as: Given a set A_k of n binary vectors of dimension $d(n)$ decide if there are k vectors $\forall i \in [k-1] : a_i \in A_{i,n}$ and $a_k \in A_k$ that are orthogonal, i.e. $\forall j \in [d(n)] : a_1[j] = 0 \vee \cdots \vee a_k[j] = 0$.*

We are now ready to extend Theorem 3.

▶ **Theorem 7.** *For all $k \geq 2$ and any function $d(n) = \log^{\omega(1)} n$, there exist $k-1$ families of vector-sets $\mathcal{A}_i = \{A_{i,n}\}_{n=1}^{\infty}, i \in [k-1]$ of dimension $d(n)$ such that the k-$OV_{\mathcal{A}_1,\dots,\mathcal{A}_k}$ problem requires $n^{k-o(1)}$ under SETH. Moreover, each set $A_{i,n}$ can be produced in $O(n \cdot d(n))$ time and log space.*

Proof. Recall from the proof of Theorem 3, the definition of $\mathcal{C}_{n,w}$ the set of all width w clauses over n variables.[4]

For all $i \in [k-1]$, we define the set $A_{i,N} \in \mathcal{A}_i$, where $N = 2^{n/k}$ as follows. For each partial assignment $\alpha^{(i)}$ to the variables $x_{(i-1)\cdot n/k+1}, \dots, x_{i\cdot(n/k)}$, we create a vector $v_{\alpha^{(i)}}$ and add it to $A_{i,N}$. Let C_j denote the j^{th} clause in a canonical ordering of the clauses in $\mathcal{C}_{n,w}$ and define $v_{\alpha^{(i)}}$ as:

[4] Note that we changed the notation of the clause size from k to w to avoid conflict with k the number of sets. That is, we are now reducing w-SAT to k-OV.

$$\forall j \in [|\mathcal{C}_{n,w}|] : v_{\alpha^{(i)}}[j] = \begin{cases} 0, & \text{if } \alpha^{(i)} \text{ satisfies the clause } C_j \\ 1, & \text{otherwise.} \end{cases}$$

Notice that the dimension of the vectors is $O(n^{2w}) = \log^{O(1)} N$ and it is asymptotically dominated by $d(N) = \log^{\omega(1)} N$. Moreover, $A_{i,N}$ can be produced in $2^{n/k} \cdot n^{2k} = O(Nd(N))$ time, and $O(n + w \log n) = O(\log N)$ space.

We are now ready to reduce w-SAT to the k-$\mathrm{OV}_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$ problem. Given a w-CNF formula ϕ on n variables $x_1, \ldots, x_n$ as input, we define a set A_k of $N = 2^{n/k}$ vectors as follows. For each partial assignment $\alpha^{(k)}$ to the variables $x_{(k-1)\cdot n/k+1}, \ldots, x_n$, we create a vector $v_{\alpha^{(k)}}$ such that:

$$\forall j \in [|\mathcal{C}_{n,w}|] : v_{\alpha^{(k)}}[j] = \begin{cases} 0, & \text{if either } \alpha^{(k)} \text{ satisfies the clause } C_j, \text{ or } C_j \notin \phi \\ 1, & \text{otherwise, i.e. } \mathcal{C}_j \in \phi \text{ but } \alpha^{(k)} \text{ does not satisfy it.} \end{cases}$$

Thus, while $v_{\alpha^{(i)}} \in A_{i,N}$ for $i \in [k-1]$ considered all clauses $C_j \in \mathcal{C}_{n,w}$ similarly, the vectors $v_{\alpha^{(k)}} \in A_k$ depend on the input formula ϕ and automatically set to 0 all coordinates corresponding to clauses that do not appear in ϕ.

Finally, we claim that A_k is a "yes" k-$\mathrm{OV}_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$ instance iff ϕ is satisfiable, and therefore if k-$\mathrm{OV}_{\mathcal{A}_1,\ldots,\mathcal{A}_k}$ can be solved faster than $n^{k-o(1)}$ time then for all w, w-SAT can be solved in $(2^{n/k})^{k-\varepsilon} = 2^{(1-\varepsilon/k)n}$ for some $\varepsilon > 0$, refuting SETH. The correctness follows from the following claim, which can be proved similarly to Claim 4.

▷ Claim 8. There exist k-orthogonal vectors $a_i \in A_{i,N}$ for $i \in [k-1]$ and $a_k \in A_k$ iff ϕ is satisfiable. ◀

4 Corollaries for Edit Distance and other problems

The hardness of $\mathrm{OV}_{\mathcal{A}}$ has immediate consequences to all the many OV-hard problems, establishing their hardness even in restricted settings. Let us formalize the implication for Edit Distance, and briefly remark on how it applies to the other problems.

▶ **Definition 9** (The Edit Distance Problem). *Given two sequences X, Y of length n, return the minimum number operations that can transform X into Y. The allowed edit-operations are insertions, deletions, and substitutions of single characters.*

In a similar way to our definition of $\mathrm{OV}_{\mathcal{A}}$ in Section 2, we formalize a restricted problem where only one of the two sequences is given as input.

▶ **Definition 10** (The $\mathrm{ED}_{\mathcal{X}}$ Problem). *For a family $\mathcal{X} = \{X_n\}_{n=1}^{\infty}$ of sequences, such that X_n is a sequence of length n, we define the $ED_{\mathcal{X}}$ problem as: Given a sequence Y of length n, return the minimum number edit-operations that can transform X_n into Y.*

We can now prove the statement in the title of the paper, showing a quadratic SETH lower bound for $\mathrm{ED}_{\mathcal{X}}$.

▶ **Theorem 11.** *There is a family of sequences $\mathcal{X} = \{X_n\}_{n=1}^{\infty}, |X_n| = n$ such that the $ED_{\mathcal{X}}$ problem requires $n^{2-o(1)}$ under SETH. Moreover, each sequence X_n can be produced in $O(n)$ time and log space.*

Proof. By Theorem 3 it is enough to reduce the $\mathrm{OV}_{\mathcal{A}}$ problem to $\mathrm{ED}_{\mathcal{X}}$. To do that, we start with an $\mathrm{OV}_{\mathcal{A}}$ instance B of size n and dimension $d(n) = n^{o(1)}$, for the family $\mathcal{A}$ constructed by Theorem 3, and we apply the reduction of Backurs and Indyk [14] (or the later one which uses a smaller alphabet [22]) to B and A_n. We get two sequences $X = S_X(A_n)$ and $Y = S_Y(B)$ of length $N = f(n) = O(n \cdot d(n))$, for some specific function $f : \mathbb{N} \to \mathbb{N}$, such that the Edit Distance between X and Y is less than a certain value τ_n iff A, B contains an orthogonal pair. Moreover, the encodings S_X and S_Y have the property that they take linear time and log space to compute, and most importantly, that S_X only depends on A_n and n but not on B. Therefore, we can construct the family of sequences $\mathcal{X}$ by setting $X_N = S_X(A_n)$ where $N = f(n)$. We get that solving $\mathrm{ED}_{\mathcal{X}}$ in $O(N^{2-\varepsilon})$ time, for some $\varepsilon > 0$, leads to an $O(n^{2-\varepsilon'})$ time algorithm for solving $\mathrm{OV}_{\mathcal{A}}$, for some $\varepsilon' > 0$, refuting SETH. ◀

To get the corollaries for LCS, k-LCS, Subtree Isomorphism, and the other problems mentioned in Section 1.1, the same arguments work: we simply have to check that the reductions from OV operate on each set separately.

5 Lower bounds for the Preprocessing model

In this section we present the implications of the new reductions for the limitations of preprocessing. As discussed in Section 1.1, any quadratic lower bound for OV already implies a quadratic lower bound even if the algorithm is allowed to preprocess one of the sets in arbitrary polynomial time. However, having a reduction from k-SAT to OV with a fixed set leads to stronger conditional lower bounds that address algorithms with arbitrary polynomial *space* preprocessing. These lower bounds are no longer based on SETH but on a plausible strengthening of it to *nonuniform* algorithms.

Recall [13] that $TIME[T(n)]/A(n)$ is the class of problems that can be solved by an $O(T(n))$ time algorithm that is given an advice string X_n of length $O(A(n))$ for all inputs of size n.

▶ **Hypothesis 12** (Nonuniform-SETH). *There is no $\varepsilon > 0$ such that for all $k \geq 3$ the k-SAT problem is in $TIME[(2-\varepsilon)^n]/2^{(1+o(1))\cdot n/2}$.*

This is a strong hypothesis, but it is not at all clear how the nonuniformity can help in solving SAT faster, and therefore it might be as plausible as SETH. Moreover, even the extreme version of this hypothesis, where the size of the advice is increased from $2^{n/2\cdot(1+o(1))}$ all the way to $2^{n\cdot(1-\varepsilon)}$ remains plausible. In fact, for our conditional lower bound below, we can weaken the hardness hypothesis to allow significantly smaller advice length: $O(2^{\varepsilon n})$ for any constant $\varepsilon > 0$, and the same conclusion for OV would follow. To obtain this stronger result it suffices to give a modification to Theorem 3 so that it reduces to *asymmetric* $\mathrm{OV}_{\mathcal{A}}$ where the sets A and B have different sizes; the details of this standard modification are omitted from this paper. In any case, the main message of the following theorem is to highlight a connection between algorithms in the preprocessing model and a breakthrough in the nonuniform complexity of SAT.

▶ **Theorem 13.** *Suppose there is an algorithm that, given two sets A, B of n binary vectors in $d(n) = \log^{\omega(1)} n$ dimensions, can preprocess the set A using $S(n) = O(n^c)$ bits of space, and subsequently solve OV on A, B in $O(n^{2-\varepsilon})$ time, for some $\varepsilon > 0, c \geq 1$. Then, Nonuniform-SETH is false.*

Proof. Fix $\varepsilon > 0, c \geq 1$ and suppose that an algorithm ALG can solve OV in subquadratic $O(n^{2-\varepsilon})$ time after preprocessing the set A using $S(n) = n^c$ space. We start by reducing k-SAT on N variables to $\mathrm{OV}_{\mathcal{A}}$ on sets of size $n = 2^{N/2}$ using Theorem 3. Then, to solve

$\mathrm{OV}_{\mathcal{A}}$ instances of size n using ALG we can take the set A_n and split it into $n^{1-1/tc}$ instances of size $n^{1/tc}$ each, for some $t \geq 2$ to be specified later. We use ALG to preprocess each of these parts in an unknown amount of time but only $O((n^{1/tc})^c)$ space; the total space that it uses for all the parts is $s_n = n^{1-1/tc} \cdot O(n^{c/tc}) = O(n^{1+1/t-1/tc})$. Let X_n be the bit-string of length s_n that encodes the state of the memory after the algorithm is done preprocessing these sets, i.e. the concatenation of the $n^{1-1/tc}$ memory tapes. Observe, crucially, that the string X_n only depends on n, even though generating it might have take an unknown amount of time. We will choose it to be the advice string for our algorithm for all inputs of size n. Then, using the string X_n our algorithm can solve any $\mathrm{OV}_{\mathcal{A}}$ instance B of size n in truly subquadratic time: we split B into $n^{1-1/tc}$ parts of size $n^{1/tc}$ each, and then solve each pair of parts using ALG and the string X_n in subquadratic $(n^{1/tc})^{2-\varepsilon}$ time, giving a total of $(n^{1-1/tc})^2 \cdot (n^{1/tc})^{2-\varepsilon} = n^{2-\varepsilon'}$ time, for $\varepsilon' = \varepsilon/tc > 0$. Going back to k-SAT, our algorithm runs in time $2^{N/2\cdot(2-\varepsilon')}$ time and has used $O(n^{1+1/t}) = O(2^{N/2\cdot(1+1/t)}) = 2^{N/2\cdot(1+o(1))}$ bits of advice, where the latter equality holds because we can choose t to be arbitrarily large, refuting Nonuniform-SETH. ◀

6 Extension to Formula-SAT and Formula-Satisfying-Pair

In this section, we extend the results of Section 2 so that the starting point is Formula-SAT rather than CNF-SAT, and the end problem is Formula-Satisfying-Pair (with a fixed formula and set A) rather than OV (with a fixed set A). Throughout, we consider deMorgan formulas that have AND/OR gates of fan-in two, and we assume that all the NOT gates are at the bottom, meaning that the leaves of each formula are either variables or their negation. The size of a formula is the total number of gates and the depth is the maximum number of levels from root to leaf.

▶ **Definition 14** (Formula-SAT). *Given a formula F over n variables, decide if it is satisfiable.*

Notice that CNF formulas are a special kind of depth 2 formulas. The following hypothesis is a more believable version of SETH.

▶ **Hypothesis 15** (Formula-SETH [6]). *There is no $\varepsilon > 0$ such that Formula-SAT on formulas of size $2^{o(n)}$ can be solved in $O((2-\varepsilon)^n)$ time.*

This hypothesis is sometimes referred to as Branching-Program-SETH (BP-SETH) since it is equivalent to a hypothesis about SAT on branching programs, or as NC-SETH which is a similar assumption about SAT on polylog depth circuits, which are equivalent to formulas of $2^{\mathrm{poly}\log n}$ size.

Just like the SETH-based lower bounds go via the OV problem, the Formula-SETH lower bounds often go via a problem such as the Formula-Satisfying-Pair problem.

▶ **Definition 16** (Formula-Satisfying-Pair [4]). *Given a formula $F = F(x_1, \ldots, x_m, y_1, \ldots, y_m)$ of size $2m$ where each variable is used exactly once, and two sets $A, B \subseteq \{0,1\}^m$ of size n, decide whether there is a pair $a \in A, b \in B$ such that $F(a_1, \ldots, a_m, b_1, \ldots, b_m) = \mathtt{true}$.*

A simple reduction, similar to the one by Williams [42], shows an $n^{2-o(1)}$ lower bound for Formula-Satisfying-Pair with $m = n^{o(1)}$ under the Formula-SETH. And with intricate gadgetry, Formula-Satisfying-Pair can be reduced to Edit Distance, LCS, Fréchet, and other problems establishing Formula-SETH lower bounds for them as well [6, 4]. The main result of this section is to prove a Formula-SETH lower bound for Formula-Satisfying-Pair with a fixed formula F and a fixed set A. As a result, the Formula-SETH lower bounds for Edit Distance and the other problems also hold when one sequence is fixed.

▶ **Definition 17** (FSP$_{\mathcal{F},\mathcal{A}}$). *For a family $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$ of vector-sets, such that $A_n \subseteq \{0,1\}^{d(n)}$ is a set of n binary vectors of dimension $d(n)$, and a family $\mathcal{F} = \{F_n\}_{n=1}^{\infty}$ of formulas, such that F_n is over $d(n)$ variables and has size $2d(n)$ we define the FSP$_{\mathcal{F},\mathcal{A}}$ problem as: Given a set B of n binary vectors of dimension $d(n)$ decide if there is a pair $a \in A_n, b \in B$ such that $F_n(a,b) =$ `true`.*

After formalizing the problem with fixed formula and set A we are ready to state the theorem. The rest of this section is dedicated to the proof.

▶ **Theorem 18.** *There is a family of vector-sets $\mathcal{A} = \{A_n\}_{n=1}^{\infty}$ of dimension $d(n) = n^{o(1)}$ and a family of formulas $\mathcal{F} = \{F_n\}_{n=1}^{\infty}$ over $d(n)$ variables and of size $2d(n)$ such that the FSP$_{\mathcal{F},\mathcal{A}}$ problem requires $n^{2-o(1)}$ under Formula-SETH. Moreover, each set A_n and formula F_n can be produced in $O(n \cdot d(n))$ time.*

Our approach is to imitate the main idea in the proof of Theorem 2. There, we looked at the set of all clauses, which is similar to thinking about the super-set of all k-CNF formulas. Now, we are faced with arbitrary formulas, and it is not so clear what the corresponding super-set would be: the set of all gates does not make much sense. To make this work, we go through the following intermediate problem, which is similar to Formula-SAT but has a structure that is easier to work with when constructing a fixed formula for FSP$_{\mathcal{F},\mathcal{A}}$.

For a depth bound $depth(n)$ we define the *Canonical-Depth-$d(n)$* formula to be the formula over n variables defined by a full binary tree in which all the gates are (?) indicating a gate that could either be AND or OR. Moreover, each leaf pointing to a variable x_i is also labelled with a (?) indicating that it could either be x_i or $\bar{x_i}$. We also fix a canonical numbering of the $s(n) = 2^{d(n)}$ gates of this formula. To get a "real" formula, we must specify $s(n)$ bits indicating for each (?) gate whether it is AND, OR, or if it is a leaf whether it is a negation or not. A natural way to make the specification is to make the j^{th} gate AND if the j^{th} bit is 1 and OR otherwise, and to make a leaf-gate a negation iff the corresponding bit is 1.

▶ **Definition 19** (Canonical-Formula-SAT). *Given $s(n)$ bits for specifying the gates of a canonical-depth-$d(n)$ formula over n variables, decide if the resulting formula is satisfiable.*

The reduction has two steps, described in Sections 6.1 and 6.2.

6.1 From Formula-SAT to Canonical-Formula-SAT

This step is not immediate only because an arbitrary $s(n)$-sized formula could have a structure that is very far from a full-binary tree. Nonetheless, we can transform it into such using standard techniques without blowing up the size by more than polynomial factors. And since our interest is in $s(n) = 2^{o(n)}$, polynomial blowups do not matter.

Given a formula F of size $s(n)$ we begin by applying the depth-reduction of Bonet-Buss [18] to get an equivalent formula F' of depth $depth(n) = O(\log s(n))$. Then, we enforce that all paths from root to leaves have length exactly $d(n)$: if a leaf is higher, we add an equivalent subtree, e.g. by replacing x_i with $(x_i \wedge \texttt{true}) \wedge (\texttt{true} \wedge \texttt{true})$ and so on. The total size of the final formula F'' is $2^{depth(n)} = s^{O(1)}$.

Then, to complete the reduction, we simply go over the gates of F'' and generate a string g of $s(n)$ bits that encodes it with the above natural representation. Thus, g is a "yes" instance for Canonical-Formula-SAT iff F is satisfiable.

6.2 From Canonical-Formula-SAT to $\mathsf{FSP}_{\mathcal{F},\mathcal{A}}$

The next idea is to transform the canonical formula $F^{(?)}$ that is full of (?) gates along with a bit-string g specifying the gates, into a formula F with real gates that takes the bits of g as inputs. In more details, we take the j^{th} gate in $F^{(?)}$, that takes input from the two gates G_1 and G_2 and we replace it with the following subformula (a similar but different formula is used for leaf-gates):

$$G_j = (g_j \wedge (G_1 \wedge G_2)) \vee (\bar{g_j} \wedge (G_1 \vee G_2))$$

This subformula encodes our natural representation where $g_j = 1$ iff the gate is AND. After these transformations, the size of the formula blows up by $2^{depth(n)}$ since we have to make two copies of each subformula at every level, but this is still $s^{O(1)}$. Notice that the formula F is now fixed, and the formula we started from is only encoded with the g_j's which can be thought of as inputs to F (note that, as opposed to the x_i's these inputs are not free, and so they do not increase the complexity of SAT).

We are now ready to define the $\mathsf{FSP}_{\mathcal{F},\mathcal{A}}$ instances that encode the satisfiability of F. Let $N = 2^{n/2}$. We define the formula F_N, i.e. the N^{th} member of the family of formulas $\mathcal{F}$, to be equal to F after we duplicate each variable so that it is used once in the formula. The vectors $a \in A_N$ have dimension $O(s(n)) = N^{o(1)}$ and are defined as follows. For each partial assignment $\alpha \in \{\texttt{true}, \texttt{false}\}^{n/2}$ to the variables $x_1, \ldots, x_{n/2}$ we define the vector a such that for all $j \in [s(n)]$ a_j is set to 1 iff the j^{th} gate in F is a leaf gate marked with a literal x_h or $\bar{x_h}$ and α makes this literal true. Note that the vectors in A do not depend on the g_j's at all.

Finally, the vectors $b \in B$ do depend on the g_j's. For each partial assignment β to the variables $x_{n/2+1}, \ldots, x_n$, we construct a vector b. For all $j \in [s(n)]$ we set b_j as follows. If the j^{th} gate in F is a leaf gate marked with a literal x_h or $\bar{x_h}$ and β makes this literal true, then we set $b_j = 1$. If the j^{th} gate is a leaf gate marked with a variable g_h (or its negation) then we set $b_j = g_h$ (or its negation). Notice that the g's affect all vectors in B in the same way (as is the case in the proof of Theorem 3).

To conclude the correctness of this reduction, observe that an evaluation of F_N on a pair of vectors a, b is equivalent to the evaluation of F on the corresponding partial assignments α, β and the given g.

References

1 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-grained complexity of analyzing compressed data: Quantifying improvements over decompress-and-solve. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 192–203, 2017.

2 Amir Abboud, Arturs Backurs, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Or Zamir. Subtree isomorphism revisited. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1256–1271, 2016.

3 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight hardness results for lcs and other sequence similarity measures. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 59–78. IEEE, 2015.

4 Amir Abboud and Karl Bringmann. Tighter connections between formula-sat and shaving logs. *arXiv preprint*, 2018. `arXiv:1804.08978`.

5 Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More consequences of falsifying SETH and the orthogonal vectors conjecture. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 253–266, 2018.

6 Amir Abboud, Thomas Dueholm Hansen, Virginia Vassilevska Williams, and Ryan Williams. Simulating branching programs with edit distance and friends: or: a polylog shaved is a lower bound made. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 375–388, 2016.

7 Amir Abboud, Aviad Rubinstein, and Ryan Williams. Distributed pcp theorems for hardness of approximation in p. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 25–36. IEEE, 2017.

8 Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 434–443, 2014.

9 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, pages 39–51, 2014.

10 Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching triangles and basing hardness on an extremely popular conjecture. *SIAM Journal on Computing*, 47(3):1098–1122, 2018.

11 Amir Abboud, Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 218–230. SIAM, 2014.

12 Josh Alman and Ryan Williams. Probabilistic polynomials and hamming nearest neighbors. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 136–150. IEEE, 2015.

13 Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.

14 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 51–58, 2015.

15 Arturs Backurs and Piotr Indyk. Which regular expression patterns are hard to match? In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 457–466. IEEE, 2016.

16 Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. Towards tight approximation bounds for graph diameter and eccentricities. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 267–280, 2018.

17 Nikhil Bansal and Ryan Williams. Regularity lemmas and combinatorial algorithms. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 745–754. IEEE, 2009.

18 Maria Luisa Bonet and Samuel R Buss. Size-depth tradeoffs for boolean formulae. *Information Processing Letters*, 49(3):151–155, 1994.

19 Karl Bringman and Marvin Künnemann. Multivariate fine-grained complexity of longest common subsequence. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1216–1235. SIAM, 2018.

20 Karl Bringmann. Why walking the dog takes time: Frechet distance has no strongly subquadratic algorithms unless SETH fails. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 661–670, 2014.

21 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A dichotomy for regular expression membership testing. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 307–318. IEEE, 2017.

22 Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 79–97. IEEE Computer Society, 2015. `doi:10.1109/FOCS.2015.15`.

23 Karl Bringmann, Marvin Künnemann, and André Nusser. Fréchet distance under translation: conditional hardness and an algorithm via offline dynamic grid reachability. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2902–2921. SIAM, 2019.

24 Timothy M Chan and Moshe Lewenstein. Clustered integer 3sum via additive combinatorics. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 31–40, 2015.

25 Timothy M Chan and Ryan Williams. Deterministic apsp, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 1246–1255. SIAM, 2016.

26 Lijie Chen, Shafi Goldwasser, Kaifeng Lyu, Guy N Rothblum, and Aviad Rubinstein. Fine-grained complexity meets ip= pspace. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1–20. SIAM, 2019.

27 Lijie Chen and Ryan Williams. An equivalence class for orthogonal vectors. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 21–40. SIAM, 2019. `doi:10.1137/1.9781611975482.2`.

28 Karthik CS, Roee David, and Bundit Laekhanukit. On the complexity of closest pair via polar-pair of point-sets. *SIAM Journal on Discrete Mathematics*, 33(1):509–527, 2019.

29 Karthik CS, Bundit Laekhanukit, and P Manurangsi. On the parameterized complexity of approximating dominating set. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC*, pages 1283–1296, 2018.

30 Karthik CS and Pasin Manurangsi. On closest pair in euclidean metric: Monochromatic is as hard as bichromatic. *ITCS*, 2019.

31 Mina Dalirrooyfard, Andrea Lincoln, and Virginia Vassilevska Williams. New techniques for proving fine-grained average-case hardness. In *61st IEEE Annual Symposium on Foundations of Computer Science, FOCS 2020, Durham, NC, USA, November 16-19, 2020*, pages 774–785. IEEE, 2020. `doi:10.1109/FOCS46700.2020.00077`.

32 Lech Duraj, Marvin Künnemann, and Adam Polak. Tight conditional lower bounds for longest common increasing subsequence. *Algorithmica*, 81(10):3968–3992, 2019.

33 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. *ACM Trans. Algorithms*, 15(2):23:1–23:35, 2019. `doi:10.1145/3196275`.

34 Elazar Goldenberg, Aviad Rubinstein, and Barna Saha. Does preprocessing help in fast sequence comparisons? In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 657–670, 2020.

35 Isaac Goldstein, Tsvi Kopelowitz, Moshe Lewenstein, and Ely Porat. Conditional lower bounds for space/time tradeoffs. In *Workshop on Algorithms and Data Structures*, pages 421–436. Springer, 2017.

36 Alexander Golovnev, Siyao Guo, Thibaut Horel, Sunoo Park, and Vinod Vaikuntanathan. Data structures meet cryptography: 3sum with preprocessing. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 294–307, 2020.

37 Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and strengthening hardness for dynamic problems via the online matrix-vector multiplication conjecture. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 21–30, 2015.

38 Robert Krauthgamer and Ohad Trabelsi. Conditional lower bounds for all-pairs max-flow. *ACM Transactions on Algorithms (TALG)*, 14(4):1–15, 2018.

39 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 515–524, 2013.

40 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the ICM*, 2018.

41 Virginia Vassilevska Williams and R. Ryan Williams. Subcubic equivalences between path, matrix, and triangle problems. *J. ACM*, 65(5):27:1–27:38, 2018. doi:10.1145/3186893.

42 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.*, 348(2-3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.