Distributed PCP Theorems for Hardness of Approximation in P

(Extended Abstract)

Amir Abboud
Stanford University
Computer Science Department
Palo Alto, CA, USA
abboud@cs.stanford.edu

Aviad Rubinstein
UC Berkeley
EECS
Berkeley, CA, USA
aviad@eecs.berkeley.edu

Ryan Williams
MIT
CSAIL and EECS
Cambridge, MA, USA
rrw@mit.edu

Abstract—We present a new distributed model of probabilistically checkable proofs (PCP). A satisfying assignment $x \in \{0,1\}^n$ to a CNF formula $\phi$ is shared between two parties, where Alice knows $x_1, \ldots, x_{n/2}$, Bob knows $x_{n/2+1}, \ldots, x_n$, and both parties know $\phi$. The goal is to have Alice and Bob jointly write a PCP that $x$ satisfies $\phi$, while exchanging little or no information. Unfortunately, this model as-is does not allow for nontrivial query complexity. Instead, we focus on a non-deterministic variant, where the players are helped by Merlin, a third party who knows all of $x$.

Using our framework, we obtain, for the first time, PCP-like reductions from the Strong Exponential Time Hypothesis (SETH) to approximation problems in $P$. In particular, under SETH we show that there are no truly-subquadratic approximation algorithms for Maximum Inner Product over $\{0,1\}$-vectors, LCS Closest Pair over permutations, Approximate Regular Expression Matching, and Diameter in Product Metric. All our inapproximability factors are nearly-tight. In particular, for the first three problems we obtain nearly-polynomial factors of $2^{(\log n)^{1-o(1)}}$; only $(1+o(1))$-factor lower bounds (under SETH) were known before.

As an additional feature of our reduction, we obtain new SETH lower bounds for the exact “monochromatic” Closest Pair problem in the Euclidean, Manhattan, and Hamming metrics.

Index Terms—fine-grained complexity; similarity search; strong exponential-time hypothesis; closest pair; longest common subsequence; inapproximability

I. INTRODUCTION

Fine-Grained Complexity classifies the time complexity of fundamental problems under popular conjectures, the most productive of which has been the Strong Exponential Time Hypothesis\(^1\) (SETH). The list of “SETH-Hard” problems is long, including central problems in pattern matching and bioinformatics\(^1\), \(^2\), \(^3\), graph algorithms\(^4\), \(^5\), dynamic data structures\(^6\), parameterized complexity and exact algorithms\(^7\), \(^8\), \(^9\), computational geometry\(^10\), time-series analysis\(^11\), \(^12\), and even economics\(^13\) (a longer list can be found in\(^14\)).

For most problems in the above references, there are natural and meaningful approximate versions, and for most of them the time complexity is wide open (a notable exception is\(^4\)). Perhaps the most important and challenging open question in the field of Fine-Grained Complexity is whether a framework for hardness of approximation in $P$ is possible. To appreciate the gaps in our knowledge regarding inapproximability, consider the following fundamental problem from the realms of similarity search and statistics, of finding the most correlated pair in a dataset.

Definition 1.1 (The MAX INNER PRODUCT Problem (MAX-IP)). Given a set of $N$ binary vectors in $\{0,1\}^d$, return a pair that maximizes the inner product.

Thinking of the vectors as subsets of $[d]$, this MAX-IP problem asks to find the pair with largest overlap, a natural similarity measure. A naïve algorithm solves the problem in $O(N^2d)$ time, and one of the most-cited fine-grained results is a SETH lower bound for this problem.\(^2\) Assuming SETH, we cannot solve MAX-IP (exactly) in $N^{2-\varepsilon} \cdot 2^{o(d)}$ time, for any $\varepsilon > 0$ \(^15\).

This lower bound is hardly pleasing when one

\(^1\) SETH is a pessimistic version of $P \neq NP$, stating that for every $\varepsilon > 0$ there is a $k$ such that $k$-SAT cannot be solved in $O((2-\varepsilon)^n)$ time.

\(^2\) As a matter of fact, the lower bound is only for the bichromatic version of the problem, where we are given two sets of vector and want to find the best pair, one from each list. This distinction between monochromatic and bichromatic not so important for now, and we will only address it in Section I-B1.
of the most vibrant areas of Algorithms\footnote{In SODA’17, two entire sessions were dedicated to algorithms for similarity search.} is concerned with designing approximate but near-linear time solutions for such similarity search problems. For example, the original motivation of the celebrated MinHash algorithm was to solve the indexing version of this problem \cite{16,17}, and one of the first implementations was at the core of the AltaVista search engine. The problem has important applications all across Computer Science, most notably in Machine Learning, databases, and information retrieval, e.g. \cite{18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44}.\textbf{MAX-IP} seems to be more challenging than closely related problems where similarity is defined as small Euclidean distance rather than large inner product. For the latter, we can get near-linear time solutions for such similarity search problems. For example, the original motivation of the celebrated Hamming distance algorithm was to solve the indexing version of this problem \cite{16,17}, and one of the first implementations was at the core of the AltaVista search engine.\textbf{Open Question 1. Is there an \(O(N^{1+\varepsilon})\)-time algorithm for computing an \(f(\varepsilon)\)-approximation to MAX INNER PRODUCT over binary vectors?}\textbf{Open Question 2. Is there a PCP-like theorem for fine-grained complexity?} 

A. PCP-like Theorems for Fine-Grained Complexity

The following meta-structure is common to most SETH-based reductions: given a CNF \(\varphi\), construct \(N = O\left(2^{n/2}\right)\) gadgets, one for each assignment to the first/last \(n/2\) variables, and embed those gadgets into some problem \(A\). The embedding is designed so that if \(A\) can be solved in \(O\left(N^{2-\varepsilon}\right) = O\left(2^{(1-\frac{\varepsilon}{2})n}\right)\) time, a satisfying assignment for \(\varphi\) can be efficiently recovered from the solution, contradicting SETH.

The most obvious barrier to proving fine-grained hardness of approximation is the lack of an appropriate PCP theorem. Given a 3-SAT formula \(\varphi\), testing that an assignment \(x \in \{0,1\}^n\) satisfies \(\varphi\) requires reading all \(n\) bits of \(x\). The PCP Theorem \cite{35,36}, shows how to transform \(x \in \{0,1\}^n\) into a PCP (probabilistically checkable proof) \(\pi = \pi(\varphi,x)\), which can be tested by a probabilistic verifier who only reads a few bits from \(\pi\). This is the starting point for almost all proofs of NP-hardness of approximation. The main obstacle in using PCPs for fine-grained hardness of approximation is that all known PCPs incur a blowup in the size proof: \(\pi(\varphi,x)\) requires \(n' \gg n\) bits. The most efficient known PCP, due to Dinur \cite{37}, incurs a polylogarithmic blowup \((n' = n \cdot \text{polylog}(n))\), and obtaining a PCP with a constant blowup is a major open problem (e.g. \cite{38,39}). However, note that even if we had a fantastic PCP with only \(n' = 10n\), a reduction of size \(N' = 2^{2n} = 2^{5n}\) does not imply any hardness at all. Our goal is to overcome this barrier: the excitement from the successes of Fine-Grained Complexity?

\textit{Distributed PCPs}\n
Our starting point is that of error-correcting codes, a fundamental building block of PCPs. Suppose that Alice and Bob want to encode a message \(m = (\alpha;\beta) \in \{0,1\}^n\) in a distributed fashion. Neither Alice nor Bob knows the entire message: Alice knows the first half \((\alpha \in \{0,1\}^{rac{n}{2}})\), and Bob knows the second half \((\beta \in \{0,1\}^{rac{n}{2}})\). Alice can locally compute an encoding \(E'(\alpha)\) of her half, and Bob locally computes an encoding \(E'(\beta)\) of his. Then the concatenation of the Alice’s and Bob’s strings, \(E(m) = (E'(\alpha);E'(\beta))\), is an error-correcting encoding of \(m\).
Now let us return to distributed PCPs. Alice and Bob share a $k$-SAT\(^4\) formula $\varphi$. Alice has an assignment $\alpha \in \{0,1\}^N$ to the first half of the variables, and Bob has an assignment $\beta \in \{0,1\}^N$ to the second half. We want a protocol where Alice locally computes a string $\pi'(\alpha) \in \{0,1\}^{n'}$, Bob locally computes $\pi'(\beta) \in \{0,1\}^{n'}$, and together $\pi(\alpha,\beta) = (\pi'(\alpha), \pi'(\beta))$ is a valid probabilistically checkable proof that $x = (\alpha,\beta)$ satisfies $\varphi$. That is, a probabilistic verifier can read a constant number of bits from $(\pi'(\alpha), \pi'(\beta))$ and decide (with success probability at least $2/3$) whether $(\alpha,\beta)$ satisfies $\varphi$.

It is significant to note that if distributed PCPs can be constructed, then very strong reductions for fine-grained hardness of approximation follow, completely overcoming the barrier for fine-grained PCPs outlined above. The reason is that we can still construct $N = O(2^{\frac{n}{2}})$ gadgets, one for each half assignment $\alpha, \beta \in \{0,1\}^\frac{n}{2}$, where the gadget for $\alpha$ also encodes $\pi'(\alpha)$. The blowup of the PCP only affects the size of each gadget, which is negligible compared to the number of gadgets. In fact, this technique would be so powerful, that we could reduce SETH to problems like approximate $\ell_2$-Nearest Neighbor, where the existing sub-quadratic approximation algorithms (e.g. [25]) would falsify SETH!

Alas, distributed PCPs are unconditionally impossible (even for 2-SAT) by a simple reduction from Set Disjointness:

**Theorem 1.2** (Reingold [40]; informal). Distributed PCPs are impossible.

**Proof (sketch).** Consider the 2-SAT formula $\varphi \triangleq \bigwedge_{i=1}^N (\neg \alpha_i \lor \neg \beta_i)$. This $\varphi$ is satisfied by assignment $(\alpha,\beta) \iff$ the vectors $\alpha,\beta \in \{0,1\}^\frac{n}{2}$ are disjoint. If a PCP verifier can decide whether $(\alpha,\beta)$ satisfies $\varphi$ by a constant number of queries to $(\pi'(\alpha), \pi'(\beta))$, then Alice and Bob can simulate the PCP verifier to decide whether their vectors are disjoint, while communicating only a constant number of bits (the values read by the PCP verifier). This contradicts the randomized communication complexity lower bounds of $\Omega(n)$ for set disjointness [41], [42], [43]. □

Note that the proof shows that even distributed PCPs with $o(n)$ queries are impossible.

**Distributed and non-deterministic PCPs:** As noted above, set disjointness is very hard for randomized communication, and hard even for non-deterministic communication [44]. But Aaronson and Wigderson [45] showed that set disjointness does have $O(\sqrt{n})$ Merlin-Arthur (MA) communication complexity. In particular, they construct a simple protocol where the standard Bob and an untrusted Merlin (who can see both sets of Alice and Bob) each send Alice a message of length $O(\sqrt{n})$. If the sets are disjoint, Merlin can convince Alice to accept; if they are not, Alice will reject with high probability regardless of Merlin’s message.

Our second main insight in this paper is this: for problems where the reduction from SETH allows for an efficient OR gadget, we can enumerate over all possible messages from Merlin and Bob\(^5\). Thus we incur only a subexponential blowup\(^6\) in the reduction size, while overcoming the communication barrier. Indeed, the construction in our PCP-like theorem can be interpreted as implementing a variant of Aaronson and Wigderson’s MA communication protocol. The resulting PCP construction is distributed (in the sense described above) and non-deterministic (in the sense that Alice receives sublinear advice from Merlin).

It can be instructive to view our distributed PCP model as a 4-party (computationally-efficient) communication problem. Merlin wants to convince Alice, Bob, and Veronica (the verifier) that Alice and Bob jointly hold a satisfying assignment to a publicly-known formula. Merlin sees everything except the outcome of random coin tosses, but he can only send $o(n)$ bits to only Alice. Alice and Bob each know half of the (allegedly) satisfying assignment, and each of them must (deterministically) send a (possibly longer) message to Veronica. Finally, Veronica toses coins and is restricted to reading only $o(n)$ bits from Alice’s and Bob’s messages, after which she must output Accept/Reject.

Patrascu and Williams [7] asked whether it is possible to use Aaronson and Wigderson’s MA protocol for Set Disjointness to obtain better algorithms for satisfiability. Taking an optimistic twist, our results in this paper may suggest this is indeed possible: if any of several important and simple problems admit efficient approximation algorithms, then faster algorithms for (exact) satisfiability may be obtained via Aaronson and Wigderson’s MA protocol.

**B. Our results**

Our distributed and non-deterministic PCP theorem is formalized and proved in the full version. Since our

\(^4\)In the formulation of SETH, $k$ is a “sufficiently large constant”. However, for the purposes of our discussion here it suffices to think of $k = 3$.

\(^5\)In fact, enumerating over Merlin’s possible messages turns out to be easy to implement in the reductions; the main bottleneck is the communication with Bob.

\(^6\)Subexponential in $n$ (the number of $k$-SAT variables), which implies subpolynomial in $N \approx 2^{n/2}$. 
main interest is proving hardness-of-approximation results, we abstract the prover-verifier formulation by reducing our PCP to an Orthogonal-Vectors-like problem which we call PCP-VECTORS (see below). PCP-VECTORS turns out to be an excellent starting point for many results, yielding easy reductions for fundamental problems and giving essentially tight inapproximability bounds. We begin with the description of PCP-VECTORS, and then exhibit what we think are the most interesting applications.

a) PCP-Vectors: We introduce an intermediate problem which we call PCP-VECTORS. The purpose of introducing this problem is to abstract out the prover-verifier formulation before proving hardness of approximation in P, very much like NP-hardness of approximation reductions start from gap-3-SAT or LABEL COVER.

Definition I.3 (PCP-Vectors). The input to this problem consists of two sets of vectors \( A \subset \Sigma^L \times K \) and \( B \subset \Sigma^L \). The goal is to find vectors \( a \in A \) and \( b \in B \) that maximize

\[
s(a, b) \triangleq \Pr_{\ell \in L} \left( \bigvee_{k \in K} (a_{\ell,k} = b_k) \right).
\] (1)

Theorem I.4. Let \( \varepsilon > 0 \) be any constant, and let \( (A, B) \) be an instance of PCP-Vectors with \( N \) vectors and parameters \( |L|, |K|, |\Sigma| = N^{o(1)} \). Then, assuming SETH, \( O(N^{2-\varepsilon}) \)-time algorithms cannot distinguish between:

- (Completeness) there exist \( a^*, b^* \) such that \( s(a^*, b^*) = 1 \); and
- (Soundness) for every \( a \in A, b \in B \), we have \( s(a, b) \leq 1/2^{(\log N)^{1-o(1)}} \).

We also have a symmetric variant of PCP-Vectors (which we call Symmetric PCP-Vectors), where the vectors come from one set. There is some tradeoff between the properties of the two variants: In PCP-Vectors, we can afford to assume additional structure on the hard instances, which supports reductions to structured problems like SUBSET QUERY and REGULAR EXPRESSION. In contrast, having one set of vectors in Symmetric PCP-Vectors simplifies reductions to Closest Pair problems with one set, like MAX IP and LCS CLOSEST PAIR.

Definition I.5 (Symmetric PCP-Vectors). The input to this problem consists of a single set of vectors \( U \subset \Sigma^L \times K \). The goal is to find a pair of vectors \( u^*, v^* \in U \) (where \( u^* \neq v^* \)) that maximize

\[
s(u, v) \triangleq \Pr_{\ell \in L} \left( \bigvee_{k \in K} (u_{\ell,k} = v_{\ell,k}) \right).
\] (2)

Theorem I.6. Let \( \varepsilon > 0 \) be any constant, and let \( U \) be an instance of Symmetric PCP-Vectors with \( N \) vectors and parameters \( |L|, |K|, |\Sigma| = N^{o(1)} \). Then, assuming SETH, \( O(N^{2-\varepsilon}) \)-time algorithms cannot distinguish between:

- (Completeness) there exist \( u^* \neq v^* \in U \) such that \( s(u^*, v^*) = 1 \); and
- (Soundness) for every \( u, v \in U \) we have \( s(u, v) \leq 1/2^{(\log N)^{1-o(1)}} \).

Furthermore, we have the guarantee that for every \( \ell \in L \), there is at most one \( k \in K \) such that \( u_{\ell,k} = v_{\ell,k} \).

b) Max Inner Product: Our first application is a strong resolution of Open Question 1, under SETH. Not only is an \( O(1) \)-factor approximation impossible in \( O(N^{1+\varepsilon}) \) time, but we must pay a near-polynomial \( 2^{(\log N)^{1-o(1)}} \) approximation factor if we do not spend nearly-quadratic \( N^{2-o(1)} \) time!

Theorem I.7. Assuming SETH, for all \( \varepsilon > 0 \), every \( O(N^{2-\varepsilon}) \)-time algorithm for MAX INNER PRODUCT on \( N \) vectors from \( \{0,1\}^d \) with dimension \( d = N^{o(1)} \) must have approximation factor at least \( 2^{(\log N)^{1-o(1)}} \).

Improving our lower bound even to some \( N^{\varepsilon} \) factor would refute SETH via the known MAX-IP algorithms (see e.g. [31]). Using a standard trick, Theorem I.7 also applies to the harder (but more useful) search version widely known as MIPS.

Corollary I.8. Assuming SETH, for all \( \varepsilon > 0 \), no algorithm can preprocess a set of \( N \) vectors \( p_1, \ldots, p_N \in D \subset \{0,1\}^d \) in polynomial time, and subsequently given a query vector \( q \in \{0,1\}^d \) can distinguish in \( O(N^{1-\varepsilon}) \) time between the cases:

- (Completeness) there is a \( p_i \in D \) such that \( \langle p_i, q \rangle \geq s \); and
- (Soundness) for all \( p_i \in D \), \( \langle p_i, q \rangle \leq s/2^{(\log N)^{1-o(1)}} \),

even when \( d = N^{o(1)} \) and the similarity threshold \( s \in [d] \) is fixed for all queries \( q \).

Except for the \( (1+o(1)) \)-factor lower bound [31] which transfers to MIPS as well, the only lower bounds known were either for specific techniques [46], [47], [48], [24], or were in the cell-probe model but only ruled out extremely efficient queries [49], [50], [51], [52], [53], [54].

An important version of MAX-IP is when the vectors are in \( \{-1,1\}^d \) rather than \( \{0,1\}^d \). This

7Notice that for the asymmetric variant PCP-VECTORS, this is true without loss of generality since all \( a_{\ell,k} \) are compared to the same \( b_k \).
version is closely related to other famous problems such as the light bulb problem and the problem of learning parity with noise (see the reductions in [28]). Negative coordinates often imply trivial results for multiplicative hardness of approximation: it is possible to shift a tiny gap of $k$ vs. $k+1$ to a large multiplicative gap of 0 vs 1 by adding $k$ coordinates with $-1$ contribution. In the natural version where we search for a pair with maximum inner product in absolute value, this trick does not quite work. Still, Ahle et al. [31] exploit such cancellations to get a strong hardness of approximation result using an interesting application of Chebychev embeddings. The authors had expected that a different approach must be taken to prove constant factor hardness for the $\{0,1\}$ case. Interestingly, since it is easy to reduce $\{0,1\}$ to $\{-1,1\}$\footnote{E.g. map each 0 to a random string in $\{±1\}^d$, and map each 1 to the string $1^d$.}, our reduction also improves their lower bound for the $\{-1,1\}$ case from $2^{\tilde{O}(\sqrt{\log N})}$ to the almost-tight $2^{(\log N)^{1+o(1)}}$. This also implies an $N^{1-o(1)}$-time lower bound for queries in the indexing version of the problem.

d) Longest Common Subsequence Closest Pair:
Efficient approximation algorithms have the potential for major impact in sequence alignment problems, the standard similarity measure between genomes and biological data. One of the most cited scientific papers of all time studies BLAST, a heuristic algorithm for sequence alignment that often returns grossly sub-optimal solutions\footnote{Note that many of its sixty-thousand citations are by other algorithms achieving better results (on certain datasets).} but always runs in near-linear time, in contrast to the best-known worst-case quadratic-time algorithms. For theoreticians, to get the most insight into these similarity measures, it is common to think of them as Longest Common Subsequence (LCS) or Edit Distance. The LCS CLOSEST PAIR problem is:

Given a (data) set of $N$ strings and a (query) set of $N$ strings, all of which have length $m \ll N$, find a pair, one from each set, that have the maximum length common subsequence (noncontiguous).

The search version and the Edit Distance version are defined analogously. Good algorithms for these problems would be highly relevant for bioinformatics. The known gaps between upper and lower bounds are huge. A series of breakthroughs [61], [62], [63], [64], [65], [66] led to “good” approximation algorithms for Edit Distance: the closest pair version can be solved in near-linear time with a $2^{O(\sqrt{\log m \log \log m})}$ approximation. Meanwhile, LCS resisted all these attacks, and to our knowledge, no non-trivial algorithms are known. On the complexity side, only a $(1+o(1))$-approximation factor lower bound is known for LCS [11], [12], [67], and getting a 1.001 approximation in near-linear time is not known to have any consequences. For certain algorithmic techniques like metric embeddings there are nearly logarithmic lower bounds for Edit-Distance, but even under such restrictions the gaps are large [68], [69], [70], [71], [72].

Perhaps our most surprising result is a separation between these two classical similarity measures. Although there is no formal equivalence between the two, they have appeared to have the same complexity no matter what the model and setting are. We prove that LCS Closest Pair is much harder to approximate than Edit Distance.

**Theorem I.10.** Assuming $SETH$, there is no $(\tilde{O}(N^{1-o(1)}))$-approximation algorithm for LCS.
Our lower bounds also rule out interesting algorithms for the harder setting of Nearest-Neighbor queries: Preprocess a regular expression so that given a string, we can find a string in the language of the expression that is approximately-the-closest one to our query string. The formal statement and definitions of regular expressions are given in the full version.

Theorem I.11 (informal). Assuming SETH, no $O(N^{2-\varepsilon})$-time algorithm can, given a regular expression $R$ of size $N$ and a set $S$ of $N$ strings of length $m = N^{o(1)}$, distinguish between the two cases:

- (Completeness) some string in $S$ is in $L(R)$
- (Soundness) all strings in $S$ have Hamming distance $(1 - o(1)) \cdot m$ (or, $(1 - o(1)) \cdot m$ if the alphabet is binary) from all strings in $L(R)$.

f) Diameter in Product Metrics: The diameter (or furthest pair) problem has been well-studied in a variety of metrics (e.g. graph metrics [85], [4], [86]). There is a trivial $2$-approximation in near-linear time (return the largest distance from an arbitrary point), and for arbitrary metrics (to which we get query access) there is a lower bound stating that a quadratic number of queries is required to get a $(2 - \delta)$-approximation [87]. For $\ell_2$-metric, there is a sequence of improved subquadratic-time approximation algorithms [88], [89], [90], [91], [92], [93]. The natural generalization to the $\ell_p$-metric for arbitrary $p$ is, to the best of our knowledge, wide open.

While we come short of resolving the complexity of approximating the diameter for $\ell_p$-metrics, we prove a tight inapproximability result for the slightly more general problem for the product of $\ell_p$ metrics. Given a collection of metric spaces $M_i = \langle X_i, \Delta_i \rangle$, their $f$-product metric is defined as

$$\Delta\left((x_1, \ldots, x_k), (y_1, \ldots, y_k)\right) \triangleq f\left(\Delta_1(x_1, y_1), \ldots, \Delta_k(x_k, y_k)\right).$$

In particular, we are concerned with the $\ell_2$-product of $\ell_\infty$-spaces, whose metric is defined as:

$$\Delta_{2,\infty}(x, y) \triangleq \left[\sum_{i=1}^{d_2} \left(\max_{j=2}^{d_\infty} \left\|x_{i,j} - y_{i,j}\right\|\right)^2\right]. \quad (3)$$

(This is a special case of the more general $\Delta_{2,\infty,1}(\cdot, \cdot)$ product metric, studied by [74].)

Product metrics (or cascaded norms) are useful for aggregating different types of data [94], [95], [96], [97]. They also received significant attention from the algorithms community because they allow rich embeddings, yet are amenable to algorithmic techniques (e.g. [95], [93], [98], [74], [72], [66]).
1.12 (Diameter). Assuming SETH, there are no \( (2 − \delta) \)-approximation algorithms for PRODUCT-METRIC DIAMETER in time \( O(N^{2−\varepsilon}) \), for any constants \( \varepsilon, \delta > 0 \).

1.1. Closest Pair vs. “Bichromatic” Closest Pair:

The main results in this paper extend known hardness-in-P results to also rule out efficient approximation algorithms. An additional feature of our reduction is that it does not suffer from the following caveat, common to almost all previous work. Going back to the MAX-IP problem, for example, the known hardness results of [15], [31] hold only for the “bichromatic” variant of the problem: given sets of vectors \( A, B \), the algorithm must find a pair \( a \in A \) and \( b \in B \) that maximizes \( a \cdot b \). In contrast, our results hold both for the bichromatic variant and the “monochromatic” variant, where given a single set \( U \), one must find a pair \( x, y \in U \) (s.t. \( x \neq y \)) that maximizes \( x \cdot y \). For the latter variant, even in the exact setting (no approximation allowed), it was open whether there is a SETH-based lower bound.

As a corollary, we obtain via known reductions \(^1\) exact hardness for monochromatic variants of other problems. For example,

Corollary 1.13 (Monochromatic Euclidean and Hamming Closest Pair). Assuming SETH, there are no \( O(N^{2−\varepsilon}) \)-time exact algorithms for the (monochromatic) CLOSEST PAIR problem, with Euclidean, Manhattan, or Hamming metrics, for any constant \( \varepsilon > 0 \).

Note that in low dimensions, the monochromatic version of Euclidean Closest Pair is known to admit polynomially faster algorithms than the bichromatic version [99]. Furthermore, [100] recently showed that even in higher dimensions, hardness for the monochromatic Euclidean CLOSEST PAIR cannot be proven by reducing the bichromatic to the monochromatic.

C. Related work

For all the problems we consider, SETH lower bounds for the exact (bichromatic) version are known. See [15], [29] for the MAX-IP and SUBSET QUERIES problems, [1], [11], [12], [101] for LCS CLOSEST PAIR, [3], [84] for REGULAR EXPRESSION MATCHING, and [15] for METRIC DIAMETER.

\(^1\)In fact, our results hold for an even stronger variant: given two sets \( A, B \), we show that it is hard to distinguish between the case where there is a pair \( a \in A \) and \( b \in B \) with a large inner product, and the case where every pair \( x \neq y \in (A \cup B) \) has a small inner product.

Prior to our work, some hardness of approximation results were known using more problem-specific techniques. For example, distinguishing whether the diameter of a graph on \( O(n) \) edges is 2 or at least 3 in truly-subquadratic time refutes SETH [4], which implies hardness for \( (3/2 − \varepsilon) \) approximations. (This is somewhat analogous to the NP-hardness of distinguishing 3-colorable graphs from graphs requiring at least 4 colors, immediately giving hardness of approximation for the chromatic number.) In most cases, however, this fortunate situation does not occur. The only prior SETH-based hardness of approximation results proved with more approximation-oriented techniques are by Ahle et al. [31] for MAX-IP via clever embeddings of the vectors. As discussed above, for the case of \( \{0, 1\} \)-valued vectors, their inapproximability factor is still only \( 1 + o(1) \).

[67] show that, under certain complexity assumptions, deterministic algorithms cannot approximate the Longest Common Subsequence (LCS) of two strings to within \( 1 + o(1) \) in truly-subquadratic time. They tackle a completely orthogonal obstacle to proving SETH-based hardness of approximation: for problems like LCS with two long strings, the quality of approximation depends on the fraction of assignments that satisfy a SAT instance. There is a trivial algorithm for approximating this fraction: sample assignments uniformly at random. See further discussion on Open Question 4.

Recent works by Williams [102] (refuting the MA variant of SETH) and Ball et al. [103] also utilize low-degree polynomials in the context of SETH and related conjectures. Their polynomials are quite different from ours: they sum over many possible assignments, and are hard to evaluate (in contrast, the polynomials used in our proof correspond to a single assignment, and they are trivial to evaluate). The main technical barrier to hardness of approximation in \( \mathsf{P} \) is the blowup incurred by standard PCP constructions; in particular, we overcome it with distributed constructions. There is also a known construction of PCP with linear blowup for large (but sublinear) query complexity [38] with non-uniform verifiers; note however that merely obtaining linear blowup is not small enough for our purposes. Different models of “non-traditional” PCPs, such as interactive PCPs [104] and interactive oracle proofs (IOP) [105], [106] have been considered and found “positive” applications in cryptography (e.g. [107], [108], [105]). In particular, [109] obtain a linear-size IOP. It is an open question whether these interactive variants can imply interesting hardness of approximation results [109]. (And it would be very interesting if our distributed PCPs have any cryptographic appli-
After the first version of this paper became public, it was brought to our attention that the term "distributed PCP" has been used before in a different context by Drucker [110]. In the simplest variant of Drucker’s model, Alice and Bob want to compute \( f(\alpha, \beta) \) with minimal communication. They receive a PCP that allegedly proves that \( f(\alpha, \beta) = 1 \); Alice and Bob each query the PCP at two random locations and independently decide whether to accept or reject the PCP. As with the interactive variants of PCP, we don’t know of any implications of Drucker’s work for hardness of approximation, but we think that this is a fascinating research direction.

D. Discussion

In addition to resolving the fine-grained approximation complexity of several fundamental problems, our work opens a hope to understanding more basic questions in this area. We list a few that seem to represent some of the most fundamental challenges, as well as exciting applications.

a) LCS Closest Pair Problem over \( \{0, 1\} \): The LCS Closest Pair Problem is most interesting in two regimes: permutations (which, by definition, require a large alphabet); and small alphabet, most notably \( \{0, 1\} \). For the regime of permutations, we obtain nearly-polynomial hardness of approximation. For small alphabet \( \Sigma \), per contra, there is a trivial \( 1/|\Sigma| \)-approximation algorithm in near-linear time: pick a random \( \sigma \in \Sigma \), and restrict all strings to their \( \sigma \)-subset. Are there better approximation algorithms?

Our current hardness techniques are limited because this problem does not admit an approximation preserving OR-gadget for a large OR. In particular the \( 1/|\Sigma| \)-approximation algorithm outlined above implies that we cannot combine much more than \( |\Sigma| \) substrings in a clever way and expect the LCS to correspond to just one substring.

Open Question 3. Is there a 1.1-approximation for the LCS Closest Pair Problem on binary inputs running in \( O(n^{2-\varepsilon}) \) time, for some \( \varepsilon > 0 \)?

b) LCS Problem (with two strings): Gadgets constructed in a fashion similar to our proof of Theorem I.10 can be combined together (along with some additional gadgets) into two long strings \( A, B \) of length \( m \), in a way that yields a reduction from SETH to computing the longest common subsequence (LCS) of \( (A, B) \), ruling out exact algorithms in \( O(m^{2-\varepsilon}) \) [11], [12]. However, in the instances output by this reduction, approximating the value of the LCS reduces to approximating the fraction of assignments that satisfy the original formula; it is easy to obtain a good additive approximation by sampling random assignments. The recent work of [67] mentioned above, uses complexity assumptions on deterministic algorithms to tackle this issue, but their ideas do not seem to generalize to randomized algorithms.

Open Question 4. Is there a 1.1-approximation for LCS running in \( O(n^{2-\varepsilon}) \) time, for some \( \varepsilon > 0 \)? (Open for all alphabet sizes.)

c) Dynamic Maximum Matching: A holy grail in dynamic graph algorithms is to maintain a \((1+\varepsilon)\)-approximation for the Maximum Matching in a dynamically changing graph, while only spending amortized \( n^{o(1)} \) time on each update. Despite a lot of attention in the past few years [111], [112], [113], [114], [115], [116], [117], [118], [119], [120], current algorithms are far from achieving this goal: one can obtain a \((1+\varepsilon)\)-approximation by spending \( \Omega(\sqrt{m}) \) time per update, or one can get an \( 2\)-approximation with \( O(1) \) time updates.

For exact algorithms, we know that \( n^{o(1)} \) update times are impossible under popular conjectures [121], [6], [122], [123], [124], such as 3-SUM\(^\text{13}\), Triangle Detection\(^\text{14}\) and the related Online Matrix Vector Multiplication\(^\text{15}\). From the viewpoint of PCP’s, this question is particularly intriguing since it seems to require hardness amplification for one of these other conjectures. Unlike all the previously mentioned problems, even the exact case of dynamic matching is not known to be SETH-hard.

Open Question 5. Can one maintain an \((1+\varepsilon)\)-approximate maximum matching dynamically, with \( n^{o(1)} \) amortized update time?

New frameworks for hardness of approximation: More fundamental than resolving any particular problem, our main contribution is a conceptually new framework for proving hardness of approximation for problems in \( \mathcal{P} \) via distributed PCPs. In particular, we were able to resolve several open problems while relying on simple algebraization techniques from early days of PCPs (e.g. [127] and reference therein). It is plausible that our results can be improved by importing into our framework more advanced techniques.

\(^{13}\)The 3-SUM Conjecture, from the pioneering work of [125], states that we cannot find three numbers that sum to zero in a list of \( n \) integers in \( O(n^{2-\varepsilon}) \) time, for some \( \varepsilon > 0 \).

\(^{14}\)The conjecture that no algorithm can find a triangle in a graph on \( n \) edges in \( O(n^{4/3-\varepsilon}) \) time, for some \( \varepsilon > 0 \), or even just that \( O(m^{1+o(1)}) \) algorithms are impossible [6].

\(^{15}\)The conjecture that given a Boolean \( n \times n \) matrix \( M \) and a sequence of \( n \) vectors \( v_1, \ldots, v_n \in \{0, 1\}^n \) we cannot compute the \( n \) products \( M \cdot x_i \) in an online fashion (output \( Mx_i \) before seeing \( x_{i+1} \)) in total of \( O(n^{3-\varepsilon}) \) time [123]. See [126] for a recent upper bound.
from decades of work on PCPs — starting with verifier composition [35], parallel repetition [128], Fourier analysis [129], etc.

**d) Hardness from other sublinear communication protocols for Set Disjointness:** A key to our results is an MA protocol for Set Disjointness with sublinear communication, which trades off between the size of Merlin’s message and the size of Alice and Bob’s messages. There are other non-standard communication models where Set Disjointness enjoys a sublinear communication protocol, for example quantum communication\(^\text{16}\) [130].

**Open Question 6.** Can other communication models inspire new reductions (or algorithms) for standard computational complexity?

**e) Hardness of approximation from new models of PCPs:** This is the most open-ended question. Formulating a clean conjecture about distributed PCPs was extremely useful for understanding the limitations and possibilities of our framework — even though our original conjecture turned out to be false.

**Open Question 7.** Formulate a simple and plausible PCP-like conjecture that resolves any of the open questions mentioned in this section.

**ACKNOWLEDGEMENTS**

We thank Karthik C.S., Alessandro Chiesa, Søren Dahlgaard, Piotr Indyk, Rasmus Pagh, Ilya Razenshteyn, Omer Reingold, Nick Spooner, Virginia Vassilevska Williams, Amey Santhosh, Omer Reingold, Nick Spooner, Virginia Vassilevska Williams, Ameya Velingker, and anonymous reviewers for helpful discussions and suggestions.

This work was done in part at the Simons Institute for the Theory of Computing. We are also grateful to the organizers of Dagstuhl Seminar 16451 for a special collaboration opportunity.

A.A. was supported by the grants of Virginia Vassilevska Williams: NSF Grants CCF-1417238, CCF-1528078 and CCF-1514339, and BSF Grant BSF:2012338. A.R. was supported by Microsoft Research PhD Fellowship, NSF grant CCF-1408635 and by Templeton Foundation grant 3966. R.W. was supported by an NSF CAREER grant (CCF-1741615).

**REFERENCES**


[14] V. V. Williams, “Hardness of easy problems: basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk),” in LIPIcs-Leibniz International Proceedings in Informatics, vol. 43, 2015.


