Towards Tight Approximation Bounds for Graph Diameter and Eccentricities

Arturs Backurs
MIT
Cambridge, Massachusetts
backurs@mit.edu

Liam Roditty
Bar Ilan University
liam.roditty@biu.ac.il

Gilad Segal
Bar Ilan University
giladsegal123@gmail.com

Virginia Vassilevska Williams
MIT
Cambridge, Massachusetts
virgi@mit.edu

Nicole Wein
MIT
Cambridge, Massachusetts
nwein@mit.edu

ABSTRACT

Among the most important graph parameters is the Diameter, the largest distance between any two vertices. There are no known very efficient algorithms for computing the Diameter exactly. Thus, much research has been devoted to how fast this parameter can be *approximated*. Chechik et al. [SODA 2014] showed that the diameter can be approximated within a multiplicative factor of $3/2$ in $\tilde{O}(m^{3/2})$ time. Furthermore, Roditty and Vassilevska W. [STOC 13] showed that unless the Strong Exponential Time Hypothesis (SETH) fails, no $O(n^{2-\epsilon})$ time algorithm can achieve an approximation factor better than $3/2$ in sparse graphs. Thus the above algorithm is essentially optimal for sparse graphs for approximation factors less than $3/2$. It was, however, completely plausible that a $3/2$ -approximation is possible in linear time. In this work we conditionally rule out such a possibility by showing that unless SETH fails no $O(m^{3/2-\epsilon})$ time algorithm can achieve an approximation factor better than $5/3$.

Another fundamental set of graph parameters are the Eccentricities. The Eccentricity of a vertex v is the distance between v and the farthest vertex from v . Chechik et al. [SODA 2014] showed that the Eccentricities of all vertices can be approximated within a factor of $5/3$ in $\tilde{O}(m^{3/2})$ time and Abboud et al. [SODA 2016] showed that no $O(n^{2-\epsilon})$ algorithm can achieve better than $5/3$ approximation in sparse graphs. We show that the runtime of the $5/3$ approximation algorithm is also optimal by proving that under SETH, there is no $O(m^{3/2-\epsilon})$ algorithm that achieves a better than $9/5$ approximation. We also show that no near-linear time algorithm can achieve a better than 2 approximation for the Eccentricities. This is the first lower bound in fine-grained complexity that addresses near-linear time computation.

We show that our lower bound for near-linear time algorithms is essentially tight by giving an algorithm that approximates Eccentricities within a $2 + \delta$ factor in $\tilde{O}(m/\delta)$ time for any $0 < \delta < 1$. This beats all Eccentricity algorithms in Cairo et al. [SODA 2016]

and is the first constant factor approximation for Eccentricities in directed graphs.

To establish the above lower bounds we study the S - T Diameter problem: Given a graph and two subsets S and T of vertices, output the largest distance between a vertex in S and a vertex in T . We give new algorithms and show tight lower bounds that serve as a starting point for all other hardness results.

Our lower bounds apply only to sparse graphs. We show that for dense graphs, there are near-linear time algorithms for S - T Diameter, Diameter and Eccentricities, with almost the same approximation guarantees as their $\tilde{O}(m^{3/2})$ counterparts, improving upon the best known algorithms for dense graphs.

CCS CONCEPTS

• **Theory of computation** → **Problems, reductions and completeness**; Shortest paths;

KEYWORDS

Fine-grained complexity, Approximation algorithms, Diameter, Eccentricities

ACM Reference Format:

Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. 2018. Towards Tight Approximation Bounds for Graph Diameter and Eccentricities. In *Proceedings of 50th Annual ACM SIGACT Symposium on the Theory of Computing (STOC'18)*. ACM, New York, NY, USA, 14 pages. <https://doi.org/10.1145/3188745.3188950>

1 INTRODUCTION

Among the most important graph parameters are the graph's diameter and the eccentricities of its vertices. The eccentricity of a vertex v is the (shortest path) distance to the furthest vertex from v , and the diameter is the largest overall distance between any two vertices in the graph.

The eccentricities and diameter measure how fast information can spread in networks. Efficient algorithms for their computation are highly desired (see e.g. [14, 33, 37]). Unfortunately, the fastest known algorithms for these parameters are very slow. For unweighted graphs on n vertices and m edges, the fastest diameter algorithm runs in $\tilde{O}(\min\{mn, n^\omega\})$ time [23] where $\omega < 2.373$ is the exponent of square matrix multiplication [32, 46, 53]. For weighted graphs, the fastest eccentricity and diameter algorithms actually

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

STOC'18, June 25–29, 2018, Los Angeles, CA, USA

© 2018 Association for Computing Machinery.

ACM ISBN 978-1-4503-5559-9/18/06...\$15.00

<https://doi.org/10.1145/3188745.3188950>

compute all distances in the graph, i.e. they solve the All-Pairs Shortest Paths (APSP) problem. The fastest known algorithms for APSP in weighted graphs run in $\min\{\tilde{O}(mn), n^3/\exp(\sqrt{\log n})\}$ [38, 39, 52].

Whether one can solve Diameter faster than APSP is a well-known open problem (e.g. see Problem 6.1 in [21] and [6, 16]). Whether one can solve Eccentricities faster than APSP was addressed by [50] (for dense graphs) and by [34] (for sparse graphs). Vassilevska W. and Williams [50] showed that Eccentricities and APSP are equivalent under subcubic reductions, so that either both of them admit $O(n^{3-\varepsilon})$ time algorithms for $\varepsilon > 0$, or neither of them do. Lincoln et al. [34] proved that under a popular conjecture about the complexity of weighted Clique, the $O(mn)$ runtime for Eccentricities cannot be beaten by any polynomial factor for any sparsity of the form $m = \Theta(n^{1+1/k})$ for integer k .

Due to the hardness of exact computation, efficient approximation algorithms are sought. A folklore $\tilde{O}(m + n)$ time algorithm achieves a 2-approximation for the diameter in directed weighted graphs and a 3-approximation for the eccentricities in undirected weighted graphs. Aingworth et al. [6] presented an almost-3/2 approximation¹ algorithm for Diameter running in $\tilde{O}(n^2 + m\sqrt{n})$ time. Roditty and Vassilevska W. [41] improved the result of [6] with an $\tilde{O}(m\sqrt{n})$ expected time almost-3/2 approximation algorithm. Chechik et al. [20] obtained a (genuine) 3/2 approximation algorithm for Diameter (in directed graphs) and a (genuine) 5/3-approximation algorithm for Eccentricities (in undirected graphs), running in $\tilde{O}(\min\{m^{3/2}, mn^{2/3}\})$ time. These are the only known non-trivial approximation algorithms for Diameter in directed graphs. So far, there are no known faster than mn algorithms for approximating the Eccentricities in directed graphs within any constant factor.

Cairo et al. [15] generalized the above results for *undirected* graphs and obtained a time-approximation tradeoff: for every $k \geq 1$ they obtained an $\tilde{O}(mn^{1/(k+1)})$ time algorithm that achieves an almost- $2 - 1/2^k$ approximation for Diameter and an almost $3 - 4/(2^k + 1)$ -approximation for Eccentricities.

Our contributions. We address the following natural question:

Main Question: *Are the known approximation algorithms for Diameter and Eccentricities optimal?*

A partial answer is known. Under the Strong Exponential Time Hypothesis (SETH), every $3/2 - \varepsilon$ approximation algorithm (for $\varepsilon > 0$) for Diameter in undirected unweighted graphs with $O(n)$ nodes and edges must use $n^{2-o(1)}$ time [41]. Similarly, every $5/3 - \varepsilon$ approximation algorithm for the Eccentricities of undirected unweighted graphs with $O(n)$ nodes and edges must use $n^{2-o(1)}$ time [2]. This however does not answer the question of whether the runtimes of the known 3/2 and 5/3 approximation algorithms can be improved. It is completely plausible that there is a 3/2-approximation algorithm for Diameter or a 5/3-approximation for Eccentricities running in linear time.

We address our Main Question for both sparse and dense graphs. Our results are shown in Table 1.

Sparse graphs. Our first result (Theorem E.2) is that unless SETH fails, every algorithm that can distinguish between diameter 5 and 8 in undirected unweighted sparse graphs requires $n^{1.5-o(1)}$ time.

THEOREM 1.1 (3/2-DIAMETER APPROX. IS TIGHT). *Under SETH, no $O(n^{3/2-\delta})$ time algorithm for $\delta > 0$ can output a $1.6 - \varepsilon$ approximation for $\varepsilon > 0$ for the Diameter of an undirected unweighted sparse graph.*

In particular, any 3/2-approximation algorithm in sparse graphs must take $n^{1.5-o(1)}$ time. Hence the $\tilde{O}(m^{3/2})$ time 3/2-approximation algorithm of [20, 41] is optimal in two ways - improving the approximation ratio to $3/2 - \varepsilon$ causes a runtime blow-up to $n^{2-o(1)}$ ([41]) and improving the runtime to $O(m^{1.5-\delta})$ causes an approximation ratio blow-up to 8/5.

Our lower bound instance says that in $O(m^{1.5-\delta})$ time one cannot return 6 when the diameter is 8. One may be tempted to extend the above lower bound, by showing that, say, in $O(m^{4/3-\delta})$ time one cannot even return 5 when the diameter is 8. This approach, however fails: in the full version of this paper we give an $O(m^2/n)$ time algorithm that does return 5 in this case, and in general when the diameter is $2h$, it returns at least $h + 1$. Notice that when the diameter is $2h$, the folklore linear time algorithm returns an estimate of only h . Hence for sparse graphs, our algorithm runs in linear time and outperforms the folklore algorithm. Also, for constant even diameter, it gives a better than 2 approximation.

We obtain stronger Diameter hardness results for weighted graphs and for *directed* unweighted graphs. In particular, assuming SETH:

(1) For weighted sparse graphs, no $O(n^{1.5-\delta})$ time algorithm for $\delta > 0$ can output a $5/3 - \varepsilon$ Diameter approximation (for $\varepsilon > 0$). See the full version of this paper for the proof.

(2) For directed unweighted sparse graphs, using a general time-accuracy tradeoff lower bound, we show that no near-linear time algorithm can achieve an approximation factor better than 5/3. See the full version of this paper for the proof.

We summarize our Diameter lower bounds and compare them to the known upper bounds in Figure 1.

We address our Main Question for Eccentricities as well. Our main result for Eccentricities is Theorem D.1 (see the Appendix). Its first consequence is as follows:

THEOREM 1.2 (5/3-ECCENTRICITIES ALG. IS TIGHT). *Under SETH, no $O(n^{3/2-\delta})$ time algorithm for $\delta > 0$ can output a $1.8 - \varepsilon$ approximation for $\varepsilon > 0$ for the Eccentricities of an undirected unweighted sparse graph.*

In other words, the $\tilde{O}(m^{3/2})$ time 5/3-approximation algorithm of [20, 41] is tight in two ways. Improving the approximation ratio to $5/3 - \varepsilon$ causes a runtime blow-up to $n^{2-o(1)}$ ([2]) and improving the runtime to $O(m^{1.5-\delta})$ causes an approximation ratio blow-up to 1.8.

More generally, we prove (in Theorem D.1): for every $k \geq 2$, under SETH, distinguishing between Eccentricities $2k - 1$ and $4k - 3$ in unweighted undirected sparse graphs requires $n^{1+1/(k-1)-o(1)}$ time. Thus, no near-linear time algorithm can achieve a $2 - \varepsilon$ -approximation for Eccentricities for $\varepsilon > 0$.

The best (folklore) near-linear time approximation algorithm for Eccentricities currently only achieves a 3-approximation, and only

¹An almost- c approximation of X is an estimate X' so that $X \leq X' \leq cX + O(1)$.

Table 1: Our results. All of the lower bounds hold even for sparse graphs.

Runtime	Approximation	Comments
Diameter Upper Bounds		
$\tilde{O}(n^2)$ expected	nearly $3/2$	undirected unweighted
$O(n^{2.05})$	nearly $3/2$	undirected unweighted
$O(m^2/n)$	< 2 for constant even diameter	directed unweighted
Diameter Lower Bounds (under SETH)		
$\Omega(n^{3/2-o(1)})$	$8/5 - \epsilon$	undirected unweighted, implies [20, 41] alg is tight
$\Omega(n^{3/2-o(1)})$	$5/3 - \epsilon$	undirected weighted
$\Omega(n^{1+1/(k-1)-o(1)})$	$(5k-7)/(3k-4) - \epsilon$	directed unweighted, any $k \geq 3$
Eccentricities Upper Bounds		
$\tilde{O}(m\sqrt{n})$	2	directed weighted, approximation factor is tight
$\tilde{O}(m/\delta)$	$2 + \delta$	directed weighted, essentially tight
$\tilde{O}(n^2)$	nearly $5/3$	undirected unweighted
$O(n^{2.05})$	nearly $5/3$	undirected unweighted
Eccentricities Lower Bounds (under SETH)		
$\Omega(n^{1+1/(k-1)-o(1)})$	$2 - 1/(2k-1) - \epsilon$	undirected unweighted, any $k \geq 2$, tight for extremal k
$\Omega(n^{2-o(1)})$	$2 - \epsilon$	directed unweighted, essentially tight
S – T Diameter Upper Bounds		
$O(m)$	3	tight
$\tilde{O}(m\sqrt{n})$	2	tight
$\tilde{O}(n^2)$	nearly 2	
$O(n^{2.05})$	nearly 2	
S – T Diameter Lower Bounds (under SETH)		
$\Omega(n^{1+1/(k-1)-o(1)})$	$3 - 2/k - \epsilon$	any $k \geq 2$, tight for extremal k

Figure 1: Our hardness results for diameter. The x -axis is the approximation factor and the y -axis is the runtime exponent. The black line represents our lower bounds. Black dots represent existing algorithms. Blue dots represent existing algorithms whose approximation is potentially off by an additive term (the algorithms of [15]). Clear dots represent algorithms that are not known to exist but are predicted by our lower bounds.

in undirected graphs. There is no known constant factor approximation algorithm for directed graphs! Is our limitation result for linear time Eccentricity algorithms far from the truth?

We show that our lower bound result is essentially tight, for both directed and undirected graphs by producing the first non-trivial

near-linear time approximation algorithm for the Eccentricities in weighted directed graphs (Theorem 3.2).

THEOREM 1.3 (2-APPROX. FOR ECCENTRICITIES IN NEAR-LINEAR TIME.). *Under SETH, no $n^{1+o(1)}$ time algorithm can output a $2 - \epsilon$ approximation for $\epsilon > 0$ for the Eccentricities of an undirected unweighted sparse graph.*

For every $\delta > 0$, there is an $\tilde{O}(m/\delta)$ time algorithm that produces a $(2 + \delta)$ -approximation for the Eccentricities of any directed weighted graph.

The approximation hardness result is the first result within fine-grained complexity that gives tight hardness for *near linear time* algorithms.

The $2 + \delta$ approximation ratio that our algorithm produces beats *all* approximation ratios for the Eccentricities given by Cairo et al. [15]. It also constitutes the first known constant factor approximation algorithm for the Eccentricities of directed graphs.

Our approximation algorithm also implies as a corollary an approximation algorithm for the Source Radius problem studied in [2] with the same runtime and approximation factor $(2 + \delta)$. Abboud et al. [2] showed that, under the Hitting Set Conjecture, any $(2 - \epsilon)$ -approximation algorithm (for $\epsilon > 0$) for Source Radius requires $n^{2-o(1)}$ time, and hence our Source Radius algorithm is also essentially tight.

Our lower bound in Theorem 1.3 holds already for undirected unweighted graphs, and the upper bound works even for directed weighted graphs. The algorithm produces a $(2 + \delta)$ -approximation, which while close, is not quite a 2-approximation. We design a genuine 2-approximation algorithm running in $\tilde{O}(m\sqrt{n})$ time that also works for directed weighted graphs. We then complement it with a tight lower bound under SETH - in sparse directed graphs, if you go below factor 2 in the accuracy, the runtime blows up to quadratic. For both of these proofs, see the full version of the paper.

THEOREM 1.4 (TIGHT 2-APPROX. FOR ECCENTRICITIES). *Under SETH, no $n^{2-\delta}$ time algorithm for $\delta > 0$ can output a $2 - \epsilon$ approximation for the Eccentricities of a directed unweighted sparse graph.*

There is an $\tilde{O}(m\sqrt{n})$ time algorithm that produces a 2-approximation for the Eccentricities of any directed weighted graph.

We thus give an essentially complete answer to our Main Question for Eccentricities. Our results are summarized in Figures 2a and 2b.

Our conditional lower bounds are all based on a common construction: a conditional lower bound for a problem called *S-T Diameter*. In *S-T Diameter*, one is given a graph $G = (V, E)$ and two subsets $S, T \subseteq V$, not necessarily disjoint, and one seeks to compute $D_{S,T} := \max_{s \in S, t \in T} d(s, t)$.

S-T Diameter is a problem of independent interest. It is related to the bichromatic furthest pair problem studied in geometry (e.g. as in [29]), but for graphs (if we set $T = V \setminus S$).

It is easy to see that if one can compute the *S-T Diameter*, then one can also compute the Diameter in the same time: just set $S = T = V$. We show that actually, when it comes to exact computation, the *S-T Diameter* and Diameter in weighted graphs are computationally equivalent - they have the same asymptotic running time. See the full version of the paper for the proof.

S-T Diameter also has very similar approximation algorithms to Diameter. We give a 3-approximation running in linear time (based on the folklore Diameter 2-approximation algorithm), and a 2-approximation running in $\tilde{O}(m^{3/2})$ time (based on the 3/2-approximation algorithm of [20, 41]). See the full version of the paper for the proofs of these results.

We prove the following theorem for *S-T Diameter* (Theorem C.1), the proof of which is the starting point for all of our conditional lower bounds.

THEOREM 1.5. *Under SETH, for every $k \geq 2$, every algorithm that can distinguish between *S-T Diameter* k and $3k - 2$ in undirected unweighted graphs requires $n^{1+1/(k-1)-o(1)}$ time.*

The theorem immediately implies that under SETH, the aforementioned 2 and 3-approximation algorithms we designed are optimal.

Dense graphs. So far our lower bounds apply only to sparse graphs. Can we address our Main Question for dense graphs as well? In particular, can we extend our runtime lower bounds of the form $n^{1+1/\ell-o(1)}$ to $mn^{1/\ell-o(1)}$, thus matching the known algorithms for larger values of m ?

We show that the answer to our Main Question for dense graphs is “no”. For undirected unweighted graphs, we obtain $\tilde{O}(n^2)$ time algorithms for Diameter achieving an almost 3/2-approximation, and for all Eccentricities achieving an almost 5/3-approximation algorithm. These algorithms run in near-linear time in dense graphs, improving the previous best runtime of $\tilde{O}(m\sqrt{n})$ by Roditty and Vassilevska W. [41], and subsuming (for dense unweighted graphs) the results of Cairo et al. [15]. See the full version of the paper for these algorithms.

THEOREM 1.6. *There is an expected $O(n^2 \log n)$ time algorithm that for any undirected unweighted graph with diameter $D = 3h + z$ for $h \geq 0, z \in \{0, 1, 2\}$, returns an estimate D' such that $2h - 1 \leq D' \leq D$ if $z = 0, 1$ and $2h \leq D' \leq D$ if $z = 2$.*

There is an expected $O(n^2 \log n)$ time algorithm that for any undirected unweighted graph returns estimates $\epsilon'(v)$ of the eccentricities $\epsilon(v)$ of all vertices such that $3\epsilon(v)/5 - 1 \leq \epsilon'(v) \leq \epsilon(v)$ for all v .

We also show that one can improve the estimates slightly with an $O(n^{2.05})$ time algorithm, achieving the same guarantees as the $\tilde{O}(m\sqrt{n})$ time algorithms of [15, 41]. See the full version of the paper for this algorithm.

2 PRELIMINARIES

Let $G = (V, E)$ be a graph, where $|V| = n$ and $|E| = m$. For every $u, v \in V$ let $d_G(u, v)$ be the length of the shortest path from u to v . When the graph G is clear from the context we omit the subscript G .

The eccentricity $\epsilon(v)$ of a vertex v is defined as $\max_{u \in V} d(v, u)$. The diameter D of a graph is $\max_{v \in V} \epsilon(v)$. In a directed graph we have $\epsilon^{out}(v) = \max_{u \in V} d(v, u)$ (resp., $\epsilon^{in}(v) = \max_{u \in V} d(u, v)$).

Let $\deg(v)$ be the degree of v and let $N_s(u)$ be the set of the s closest vertices of v , where ties are broken by taking the vertex with the smaller id. In a directed graph let $\deg^{out}(v)$ (resp., $\deg^{in}(v)$) be the outgoing (incoming) degree of v . and let $N_s^{out}(v)$ (resp., $N_s^{in}(v)$) be the set of the s closest outgoing (incoming) vertices

Figure 2: Our algorithms and hardness results for Eccentricities. The lower bounds are for unweighted graphs and the upper bounds are for weighted graphs. The x -axis is the approximation factor and the y -axis is the runtime exponent. The black line represents our lower bounds. Black dots represent existing algorithms (including our algorithm at $(2, 3/2)$ in figure b). Blue dots represent existing algorithms whose position may not be exactly as it appears in the figure. Here, the blue dots represent our $(2 + \delta)$ -approximation algorithm running in $\tilde{O}(m/\delta)$ time. Clear dots represent algorithms that are not known to exist but are predicted by our lower bounds.

of v , where ties are broken by taking the vertex with the smaller id. For a subset $S \subseteq V$ of vertices and a vertex $v \in V$ we write $d(S, v) := \min_{s \in S} d(s, v)$ to denote the distance from the set S to the vertex v .

Let $k \geq 2$. The k -Orthogonal Vectors Problem (k -OV) is as follows: given k sets $S_1, \dots, S_k$, where each S_i contains n vectors in $\{0, 1\}^d$, determine whether there exist $v_1 \in S_1, \dots, v_k \in S_k$ so that their generalized inner product is 0, i.e. $\sum_{i=1}^d \prod_{j=1}^k v_j[i] = 0$.

Williams [51] (see also [49]) showed that if for some $\varepsilon > 0$ there is an $n^{k-\varepsilon}$ poly (d) time algorithm for k -OV, then CNF-SAT on formulas with N variables and m clauses can be solved in $2^{N(1-\varepsilon/k)}$ poly (m) time. In particular, such an algorithm would contradict the Strong Exponential Time Hypothesis (SETH) of Impagliazzo, Paturi and Zane [28] which states that for every $\varepsilon > 0$ there is a K such that K -SAT on N variables cannot be solved in $2^{(1-\varepsilon)N}$ poly N time (say, on a word-RAM with $O(\log N)$ bit words).

This also motivates the following k -OV Conjectures (implied by SETH) for all constants $k \geq 2$: k -OV requires $n^{k-o(1)}$ time on a word-RAM with $O(\log n)$ bit words. Most of our conditional lower bounds are based on the k -OV Conjecture for a particular constant k , and thus they also hold under SETH.

A main motivation behind SETH is that despite decades of research, the best upper bounds for K -SAT on N variables and M clauses remain of the form $2^{N(1-c/K)}$ poly (M) for constant c (see e.g. [27, 36, 42]). The best algorithms for the k -OV problem for any constant $k \geq 2$ on n vectors and dimension $c \log n$ run in $n^{2-1/O(\log c)}$ (Abboud, Williams and Yu [3] and Chan and Williams [17]).

3 OVERVIEW

We provide an overview of our results in this section. The full results and proofs are in the Appendix and in the full version of the paper. We first give an overview of our conditional lower bounds for sparse graphs, and then also describe the ideas behind our algorithms.

All our conditional lower bounds are based on our reduction from k -OV (for arbitrary $k \geq 2$) to S - T Diameter. We thus begin with describing the S - T Diameter construction.

3.1 Overview of Our S - T Diameter Lower Bounds.

THEOREM 3.1. *Let $k \geq 2$. Given a k -OV instance consisting of sets $W_0, W_1, \dots, W_{k-1} \subseteq \{0, 1\}^d$, each of size n , we can in $O(kn^{k-1}d^{k-1})$ time construct an unweighted, undirected graph with $O(n^{k-1} + kn^{k-2}d^{k-1})$ vertices and $O(kn^{k-1}d^{k-1})$ edges that satisfies the following properties.*

- (1) *The graph consists of $k+1$ layers of vertices $S = L_0, L_1, L_2, \dots, L_k = T$. The number of nodes in the sets is $|S| = |T| = n^{k-1}$ and $|L_1|, |L_2|, \dots, |L_{k-1}| \leq n^{k-2}d^{k-1}$.*
- (2) *S consists of all tuples $(a_0, a_1, \dots, a_{k-2})$ where for each i , $a_i \in W_i$. Similarly, T consists of all tuples $(b_1, b_2, \dots, b_{k-1})$ where for each i , $b_i \in W_i$.*
- (3) *If the k -OV instance has no solution, then $d(u, v) = k$ for all $u \in S$ and $v \in T$.*
- (4) *If the k -OV instance has a solution $a_0, a_1, \dots, a_{k-1}$ where for each i , $a_i \in W_i$ then if $\alpha = (a_0, \dots, a_{k-2}) \in S$ and $\beta = (a_1, \dots, a_{k-1}) \in T$, then $d(\alpha, \beta) \geq 3k - 2$.*
- (5) *If the k -OV instance has a solution $a_0, a_1, \dots, a_{k-1}$ where for each i , $a_i \in W_i$ then for any tuple $(b_{\lfloor \frac{k-1}{2} \rfloor}, \dots, b_{k-2})$, if $\alpha = (a_0, a_1, \dots, a_{\lfloor \frac{k-3}{2} \rfloor}, b_{\lfloor \frac{k-1}{2} \rfloor}, \dots, b_{k-2}) \in S$ and $\beta = (a_1, \dots, a_{k-1}) \in T$, then $d(\alpha, \beta) \geq 3k - 2 - 2\lceil \frac{k-1}{2} \rceil$.*
- (6) *For all i from 1 to $k-1$, for all $v \in L_i$ there exists a vertex in L_{i-1} adjacent to v and a vertex in L_{i+1} adjacent to v .*

From properties 3 and 4 in the above Theorem, we get that if there is some $k \geq 2$, $\varepsilon > 0$ and $\delta > 0$ so that there is an $O(M^{1+1/(k-1)-\varepsilon})$ time $(3 - 2/k - \delta)$ -approximation algorithm for S - T diameter in M -edge graphs, then k -OV has an $n^{k-\gamma}$ poly (d) -time algorithm for

some $\gamma > 0$ and SETH is false; this shows that our S - T Diameter algorithms are optimal. Properties 5 and 6 are useful for the rest of our constructions.

Here is how the graph G_k is built. Let $k = t + 2$ for any $t \geq 0$. Figure 3 in the Appendix shows the construction of the graph G_4 (i.e. $t = 2$).

Let $W_0, W_1, \dots, W_{t+1}$ be the sets of the $(t+2)$ -OV instance, each containing n vectors in $\{0, 1\}^d$. G_k is a layered graph on $t+3$ layers, $L_0, \dots, L_{t+2}$, where the edges go between adjacent layers L_i, L_{i+1} . We set $S = L_0$ and $T = L_{t+2}$ for the S - T diameter instance; $D_{S,T} \geq t+2$ because of the layering.

First we describe the nodes. L_0 consists of n^{t+1} nodes, each corresponding to a $t+1$ -tuple $(a_0, a_1, \dots, a_t)$ where for each i , $a_i \in W_i$. Similarly, L_{t+2} consists of n^{t+1} nodes, each corresponding to a $t+1$ -tuple $(b_1, b_2, \dots, b_{t+1})$ where for each i , $b_i \in W_i$. Layer L_1 consists of $n^t d^{t+1}$ nodes, each corresponding to a tuple $(a_0, \dots, a_{t-1}, \bar{x})$ where for each i , $a_i \in W_i$ and $\bar{x} = (x_0, \dots, x_t)$ is a $(t+1)$ -tuple of coordinates in $[d]$. Similarly, L_{t+1} consists of $n^t d^{t+1}$ nodes, each corresponding to a tuple $(b_2, \dots, b_{t+1}, \bar{x})$ where for each i , $b_i \in W_i$ and $\bar{x}$ is a $(t+1)$ -tuple of coordinates. For every $j \in \{2, \dots, t\}$, L_j consists of $n^t d^{t+1}$ nodes $(a_0, \dots, a_{t-j}, b_{t+3-j}, \dots, b_{t+1}, \bar{x})$, where for each i , $a_i \in W_i$, $b_i \in W_i$ and $\bar{x} = (x_0, \dots, x_t)$ is a $(t+1)$ -tuple of coordinates in $[d]$. In other words, there is a vector from W_i for every $i \notin \{t-j+1, t-j+2\}$.

Now we define the edges. Consider a node $(a_0, \dots, a_t) \in L_0$. For every $\bar{x} = (x_0, \dots, x_t)$, connect $(a_0, \dots, a_t)$ to $(a_0, \dots, a_{t-1}, \bar{x}) \in L_1$ if and only if for every $j \in \{0, \dots, t\}$, a_j is 1 in coordinates $x_0, \dots, x_{t-j}$. For $i \in \{1, \dots, t\}$ define the edges between L_i and L_{i+1} : For $(a_0, \dots, a_{t-i}, b_{t+3-i}, \dots, b_{t+1}, \bar{x}) \in L_i$ ² and for any $c_{t+2-i} \in W_{t+2-i}$, add an edge to $(a_0, \dots, a_{t-i-1}, c_{t+2-i}, b_{t+3-i}, \dots, b_{t+1}, \bar{x}) \in L_{i+1}$. Here we “forget” vector a_{t-i} and replace it with c_{t+2-i} , leaving everything else the same.

Finally, the edges between L_{t+1} and L_{t+2} are as follows. Consider some $(b_1, \dots, b_{t+1}) \in L_{t+2}$. For every $\bar{x} = (x_0, \dots, x_t)$, connect $(b_1, \dots, b_{t+1})$ to $(b_2, \dots, b_{t+1}, \bar{x}) \in L_{t+1}$ if and only if for every $j \in \{1, \dots, t+1\}$, b_j is 1 in coordinates $x_{t+1-j}, \dots, x_t$.

The main idea behind the proof of Theorem 3.1 is as follows.

If there is no k -OV solution, then for every k -tuple $a_0 \in W_0, \dots, a_{t+1} \in W_{t+1}$, there is some coordinate x for which $a_i[x] = 1$ for all $i \in \{0, \dots, t+1\}$. Hence, consider any $(a_0, \dots, a_t) \in S$ and $(b_1, \dots, b_{t+1}) \in T$, and for each $j \in \{1, \dots, t\}$, let x_j be the coordinate in which $a_0, \dots, a_j, b_{j+1}, \dots, b_{t+1}$ are all 1. Then, if we let $\bar{x} = (x_1, \dots, x_t)$, by the construction of G_k , there is a path $(a_0, \dots, a_t)$ to $(a_0, \dots, a_{t-1}, \bar{x})$ to $(a_0, \dots, a_{t-2}, b_{t+1}, \bar{x})$ etc, all the way to $(b_2, \dots, b_{t+1}, \bar{x})$ and then to $(b_1, \dots, b_{t+1})$, where to go from $(a_0, \dots, a_{t-2}, b_{t+1}, \bar{x})$ we forget a_{t-2} and replace it with b_{t+1} , thus taking the edge to $(a_0, \dots, a_{t-3}, b_{t+2}, \dots, b_{t+1}, \bar{x})$.

If there is a k -OV solution $a_0 \in W_0, \dots, a_{t+1} \in W_{t+1}$, then consider $\alpha = (a_0, \dots, a_t) \in S$ and $\beta = (a_1, \dots, a_{t+1}) \in T$. Let the shortest path between α and β be P and consider Figure 4 in the Appendix.

Let P' be a subpath of P that goes from α to some $\alpha^* \in S$ so that the last layer in G_k that P' touches is L_i . Then P' can have “forgotten” at most the last i vectors in $(a_0, \dots, a_t)$, and hence the first $t-i$ vectors in α^* must be $(a_0, \dots, a_{t-i})$. Similarly, if a subpath

of P goes from some $\beta^* \in T$ to β so that it does not touch any layers L_r with $r < j = t+2-j'$, then β^* can at most have forgotten $a_1, \dots, a_{j'}$ and so both β and β^* share $(a_{j'+1}, \dots, a_{t+1})$. We use this to show that either P is already strictly longer than $3k-4$, or that some α^* and β^* exist as above with $t-i \geq j'$ and so that there is a path of length exactly k between them. This path then shows that there is a coordinate x in which all $a_0, \dots, a_{t-i}$ and $a_{j'+1}, \dots, a_{t+1}$ are all 1 (see Figure 3 in the Appendix). Since $t-i \geq j'$, we must have that all $a_0, \dots, a_{t+1}$ are 1 in x , giving a contradiction since the a_i were supposed to be orthogonal. Hence we get that the distance between α and β is more than $3k-4$, and since the graph is layered $k+1$ -partite, the distance must be at least $3k-2$.

3.2 Overview of Our Diameter and Eccentricity Lower Bounds

For all of our constructions with the exception of the directed eccentricities lower bound, we begin with the S - T diameter lower bound construction from Theorem 3.1. Here, if the k -OV instance has no solution, $D_{S,T} \leq k$ and if the instance has a solution $D_{S,T} \geq 3k-2$. To adapt this construction for Eccentricities, we require that all vertices in S have low eccentricity when there is no k -OV solution and that at least one vertex in S has high eccentricity when there is a k -OV solution. The unmodified S - T Diameter construction is insufficient because pairs of vertices in S could be far from each other in the “no” case. To address this, we simply add a vertex y and connect it with a weighted edge to all vertices in S . With further simple modifications, we achieve a construction with the desired properties.

In the case of diameter, our task is more challenging because we need to ensure that if the OV instance has no solution then *all* pairs of vertices have small enough distance. We begin by augmenting the S - T Diameter construction by adding a matching between S and a new set S' as well as a matching between T and a new set T' . Without any further modifications, pairs of vertices $u, v \in S \cup S'$ (or $u, v \in T \cup T'$) could be far from one another. The challenge is to add extra gadgetry to make these pairs close for “no” instances while maintaining that in “yes” instances the distance between the diameter endpoints $s' \in S', t' \in T'$ is large. That is, for “yes” instances, we want a shortest path between the diameter endpoints s' and t' to contain the vertex $s \in S$ matched to s' and the vertex $t \in T$ matched to t' so that we can use the fact that $d(s, t) \geq 3k-2$. In other words, we do not want there to be a shortcut from s' to some vertex in S that allows us to use a path of length k from S to T . For example, we cannot simply create a vertex x and connect it to all vertices in $S \cup S'$ because this would introduce shortcuts from S' to S .

We will describe some intuition for the changes for the S side of the graph. We will pretend that the T side of the graph remains unchanged (when in fact they will be symmetrical). Recall that $s' \in S', t' \in T'$ are the endpoints of the diameter and let t be the vertex matched to t' . To solve the problem outlined in the above paragraph, we observe that in the “yes” case there are three types of vertices $s \in S$. (1) close: $d(s, t) = k$, (2) far: $d(s, t) \geq 3k-2$ (property 4 of Theorem 3.1), and (3) intermediate: $d(s, t) \geq 3k-2 - 2\lceil \frac{k-1}{2} \rceil$ (property 5 of Theorem 3.1). For close s , we need $d(s', s)$ to be large so that there is no shortcut from s' to t' through s . For far s , it is ok

²Here if $i = 1$, there are no b 's in the tuple.

if $d(s', s)$ is small because $d(s, t)$ is large enough to ensure that paths from s' to t' through s are still long enough. For intermediate s , $d(s', s)$ cannot be small, but it also need not be large. To fulfill these specifications, we add a small clique (the graph is still sparse) and connect each of its vertices to only *some* of the vertices in S and/or S' according to the implications of property 5 of Theorem 3.1. We ensure that for close s , there is a large distance from s' to s because we need to take an edge to the clique, an edge inside of the clique, and then an edge from the clique to s . For intermediate s , however, we need only take an edge to the clique and then from the clique (not an edge inside of the clique) which is sufficient to achieve the required distance. These intermediate s are important as they allow every vertex in the clique to have an edge to some vertex in S and thus be close enough to the T side of the graph in the “no” case.

3.3 Algorithms for Sparse Graphs: Overview

The main idea of all known Diameter approximation algorithms is the same: attempt to discover a node x that is very close to one of the endpoints of the diameter path. Such a vertex x then must be far from the other end point. The same idea works for S - T Diameter: if we find a node x at distance at most q from s^* (where $s^* \in S$ and $t^* \in T$ achieve the S - T diameter D), then the closest node s_x in S to x is at distance at most $2q$ from s^* and hence s_x must be at distance at least $D - 2q$ from t^* . We modify the prior Diameter approximation algorithms appropriately and obtain our S - T Diameter algorithms.

For Eccentricities, our $\tilde{O}(m\sqrt{n})$ time 2-approximation algorithm for directed graphs is very similar to the prior Eccentricity approximation algorithms. In contrast, our near-linear time $(2 + \delta)$ -approximation algorithm is very different from all previously known algorithms. Its main goal is to find for every v , some vertex v' that is *far* from v , in order to conclude that the eccentricity of v is large. Previous algorithms attempted to find a node v'' that is *close* to v .

Our algorithm proceeds in iterations and maintains a set S of nodes for which we still do not have a good eccentricity estimate. In each iteration either we get a good estimate for many new vertices and hence remove them from S , or we remove all vertices from S that have large eccentricities, and for the remaining nodes in S we have a better upper bound on their eccentricities. After a small number of iterations we have a good estimate for all vertices of the graph. This algorithm also implies an almost 2-approximation of the graph radius problem in near-linear time. We include the full proof here.

THEOREM 3.2. *Suppose that we are given a weighted, directed m edge n node graph. The weights of all edge are non-negative integers bounded by $n^{O(1)}$. For any $1 > \tau > 0$ we can in $\tilde{O}(m/\tau)$ time output quantities $\varepsilon'(v)$ such that for all $v \in V$ we have $\frac{1-\tau}{2}\varepsilon(v) \leq \varepsilon'(v) \leq \varepsilon(v)$.*

PROOF. We maintain a subset $S \subseteq V$ of vertices v for which we still do not have an estimate $\varepsilon'(v)$. Initially $S = V$ and we will end with $|S| \leq O(1)$. When $|S| \leq O(1)$ we can evaluate $\varepsilon(v)$ for all $v \in S$ in the total time of $O(m)$. Also we maintain a value D that upper bounds the largest eccentricity of a vertex in S . That is, $\varepsilon(v) \leq D$ for all $v \in S$. Initially we set $D = n^C$ for some large enough constant $C > 0$ (we assume that the input graph is strongly connected). The algorithm proceeds in phases. Each phase takes $\tilde{O}(m)$ time and

either $|S|$ decreases by a factor of at least 2 or D decreases by a factor of at least $1/(1 - \tau)$. After $O(\log(n)/\tau)$ phases either $|S| \leq O(1)$ or $D < 1$.

For a subset $S \subseteq V$ of vertices and a vertex $x \in V$ we define a set $S_x \subseteq S$ to contain those $|S_x| = |S|/2$ vertices from S that are closest to x (according to distance $d(\cdot, x)$). The ties are broken by taking the vertex with the smaller id. Given a subset $S \subseteq V$ of vertices and a threshold D , a phase proceeds as follows.

- We sample a set $A \subseteq S$ of $O(\log n)$ random vertices from the set S . With high probability for all $x \in V$ we have $A \cap S_x \neq \emptyset$.
- Let $w \in S$ be a vertex that maximizes $d(A, w)$. We can find it using Dijkstra's algorithm.
- We consider two cases.

Case $d(S \setminus S_w, w) \geq \frac{1-\tau}{2}D$. For all $x \in S \setminus S_w$ we have $\frac{1-\tau}{2}D \leq \varepsilon(x) \leq D$ and we assign the estimate $\varepsilon'(x) = \frac{1-\tau}{2}D$. This gives us that $\frac{1-\tau}{2}\varepsilon(x) \leq \frac{1-\tau}{2}D = \varepsilon'(x) \leq \varepsilon(x)$ for all $x \in S \setminus S_w$. We update S to be S_w . This decreases the size of S by a factor of 2 as required.

Case $d(S \setminus S_w, w) < \frac{1-\tau}{2}D$. Set $S' = S$. For every vertex $v \in S$ evaluate $r_v := \max_{x \in A} d(v, x)$. We can evaluate these quantities by running Dijkstra's algorithm from every vertex in A and following the incoming edges. If $r_v \geq \frac{1-\tau}{2}D$, then assign the estimate $\varepsilon'(v) = \frac{1-\tau}{2}D$ and remove v from S' . Similarly as in the previous case we have $\frac{1-\tau}{2}\varepsilon(v) \leq \varepsilon'(v) \leq \varepsilon(v)$ for all $v \in S \setminus S'$. Below we will show that for every $v \in S'$ we have $\varepsilon(v) \leq (1 - \tau)D$. Thus we can update $S = S'$ and decrease the threshold D to $(1 - \tau)D$ as required.

Correctness. We have to show that, if there exists $v \in S'$ such that $\varepsilon(v) > (1 - \tau)D$, then we will end up in the first case (this is the contrapositive of the claim in the second case). Since $v \in S'$ we must have that $d(v, x) \leq \frac{1-\tau}{2}D$ for all $x \in A$. Since $\varepsilon(v) > (1 - \tau)D$, we must have that there exists v' such that $d(v, v') > (1 - \tau)D$. By the triangle inequality we get that $d(x, v') > \frac{1-\tau}{2}D$ for every $x \in A$. Let $w \in S$ be any vertex that maximizes $d(A, w)$. We must have $d(A, w) > \frac{1-\tau}{2}D$. Since $A \cap S_w \neq \emptyset$, we have $d(S \setminus S_w, w) \geq \frac{1-\tau}{2}D$ and we will end up in the first case.

The guarantee on the approximation factor follows from the description. $\square$

3.4 Algorithms for Dense Graphs: Overview

The almost-3/2 Diameter approximation algorithm of Aingworth et al. [6] runs in $\tilde{O}(n^2 + m\sqrt{n})$ time. Roditty and Vassilevska W. [41] removed the $\tilde{O}(n^2)$ term to obtain an $\tilde{O}(m\sqrt{n})$ expected time almost-3/2 approximation algorithm. For every graph with $\Omega(n^{1.5})$ edges the running time of the latter algorithm is not better than the running time of the former algorithm. Therefore, it is interesting to consider the opposite question to the one considered by [41]. Can the $\tilde{O}(m\sqrt{n})$ term be removed?

We show that this can be done for undirected unweighted graphs and present an $O(n^2 \log n)$ expected time algorithm. For a graph of diameter $D = 3h + z$, where $z \in [0, 1, 2]$ our algorithm returns an estimation $\hat{D}$ such that $2h - 1 \leq \hat{D} \leq D$, when $z \in [0, 1]$ and $2h \leq \hat{D} \leq D$, when $z = 2$.

Interestingly, our algorithm is obtained by using ideas developed originally for distance oracles and compact routing schemes. Let $a, b \in V$ and let $d(a, b) = D$, both [6] and [41] used the following idea. Sample a set $A \subseteq V$ and compute full shortest paths trees for all vertices of A . If a vertex that is close to a or b is in A we have a good approximation, if not then all sampled vertices are far from both a and b so pick that farthest one and compute for it and for its $\sqrt{n}$ closest vertices full shortest paths trees.

Our algorithm uses a different approach. As we are allowed to use quadratic time, we try to estimate the distance between every pair of vertices. To enable this approach we can no longer sample A naively. Instead, we adapt a recursive sampling algorithm to compute A , that was introduced by Thorup and Zwick [47] in the context of compact routing schemes. The expected running time of their algorithm is $\tilde{O}(mn/|A|)$. We provide a new implementation of their algorithm that runs in expected $\tilde{O}(n(n/|A|)^2)$ time.

The set A has the following important property, for every vertex $w \in V$, its *cluster* (see [48]) $\{u \mid d(u, w) < d(u, A)\}$ is of size $O(n/|A|)$. Consider now a pair of vertices u and v that are in the cluster of w . For any such pair we can efficiently compute their exact distance. Moreover, we show that for all pairs u, v that are not in the same cluster of any vertex, we can bound $d(u, v)$ from below with $d(u, A) + d(v, A) - 1$. This, combined with some other ideas, gives our approximation guarantees. We extend our approach to also provide an almost 5/3-approximation for all Eccentricities. The idea of using the bounded clusters of Thorup and Zwick [47] has been used in prior work to obtain improved distance oracles [5, 40], approximate shortest paths [10] and compact routing schemes [4].

3.4.1 Overview of the $O(n^{2.05})$ time algorithms. The main overhead of the $\tilde{O}(m\sqrt{n})$ time algorithms [15, 41] is in computing the distances from a set $S = W \cup \{w\} \cup T$ of $O(\sqrt{n} \log n)$ nodes: the set S itself can be computed in linear time using random sampling to form a set W , BFS from a dummy node to find the node w farthest from W and then BFS from w to find the set T of closest $\sqrt{n}$ nodes to w . After one knows all distances from every $s \in S$ to every $v \in V$, it takes linear time to output the Diameter and Eccentricity estimates.

The main idea of our algorithms is as follows. If the Diameter is of size $\leq O(\log n)$, then one does not need all distances between S and V , but only those that are $O(\log n)$. Small distances are easy to compute with matrix multiplication. Let A be the adjacency matrix and A_S be its submatrix formed by just the rows in S . Then we can find the distances for all pairs in $S \times V$ at distance $\leq t$ by computing $A_S \times A^{t-1}$, which can be computed by performing $t - 1$ matrix products of dimension $|S| \times n$ by $n \times n$, and this can be accomplished in $O(tn^{2.05})$ time [26, 31]. If on the other hand the Diameter is $D \geq 100 \log n$, then one can use an $\tilde{O}(n^2)$ time algorithm by Dor et al. [24] to compute estimates of all pairwise distances with an additive error at most $4 \log n$. The maximum distance estimate computed, minus $4 \log n$, will be between $0.96D$ and D , giving a really good approximation already. A similar argument works for Eccentricities, and also for S - T Diameter.

A ORGANIZATION

In Section B we describe related work. In Section C we prove our lower bounds for S - T Diameter which serve as a basis for the rest of our lower bounds. In Section D we prove our lower bound for

Eccentricities in undirected graphs. In Section E we prove our lower bound for Diameter in undirected unweighted graphs. For the remainder of our results, we defer to the full version.

B RELATED WORK

The fastest known algorithm for APSP in dense weighted graphs runs in $n^3/2^{\Theta(\sqrt{\log n})}$ by R. Williams [52]. For sparse undirected graphs, the fastest known APSP algorithm is by Pettie [38] running in $O(mn + n^2 \log \log n)$ time. The fastest APSP algorithm for sparse undirected weighted graphs is by Pettie and Ramachandran [39] and runs in $O(mn \log \alpha(m, n))$ time. Chan [16] presented $o(mn)$ time algorithms for APSP in undirected unweighted graphs with $m > n \log \log n$; these run at worst in time $O(mn \log \log n / \log n)$. In graphs with small integer edge weights bounded in absolute value by M ($M = 1$ for unweighted graphs), APSP can be computed in $\tilde{O}(Mn^\omega)$ time (Shoshan and Zwick [44] building upon Seidel [43] and Alon, Galil and Margalit [7]) in undirected graphs and in $\tilde{O}(M^{0.681} n^{2.5302})$ (by Zwick [55]) in directed graphs. Zwick [55] also showed that APSP in directed weighted graphs admits an $(1 + \epsilon)$ -approximation algorithm for any $\epsilon > 0$, running in time $\tilde{O}(n^\omega / \epsilon \log(M/\epsilon))$. For the Diameter in graphs with integer edge weights bounded by M , Cygan et al. [23] obtained an algorithm running in time $\tilde{O}(Mn^\omega)$.

The pioneering work of Aingworth et al. [6] on diameter and shortest paths approximation was the root to many subsequent works. Building upon Aingworth et al. [6], Dor, Halperin and Zwick [24] presented additive approximation algorithms for APSP in undirected unweighted graphs, achieving among other things, a $+2$ -approximation in $\tilde{O}(n^{7/3})$ time (notably, the best known bound on ω is $> 7/3$). They also presented an $\tilde{O}(n^2)$ time $+O(\log n)$ -approximation algorithm. These algorithms were generalized by Cohen and Zwick [22] who showed that in undirected weighted graphs APSP has a (multiplicative) 3-approximation in $\tilde{O}(n^2)$ time, a 7/3-approximation in $\tilde{O}(n^{7/3})$ time, and a 2-approximation in $\tilde{O}(n\sqrt{mn})$ time. Baswana and Kavitha [10] presented an $\tilde{O}(m\sqrt{n} + n^2)$ time multiplicative 2-approximation algorithm and an $\tilde{O}(m^{2/3}n + n^2)$ time 7/3-approximation algorithm for APSP in weighted undirected graphs.

Spanners are closely related to shortest paths approximation. A subgraph H is an (α, β) -spanner of $G = (V, E)$ if for every $u, v \in V$, $d_H(u, v) \leq \alpha \cdot d_G(u, v) + \beta$, where $d_G(u, v)$ is the distance between u and v in G . Any weighted undirected graph has $(2k - 1, 0)$ -spanner with $O(n^{1+1/k})$ edges [8]. Baswana and Sen [13] presented a randomized linear time algorithm for constructing a $(2k - 1, 0)$ -spanner with $O(kn^{1+1/k})$ edges. Dor, Halperin and Zwick [24] showed that a $(1, 2)$ -spanner with $O(n^{1.5})$ edges can be constructed in $\tilde{O}(n^2)$ time. Elkin and Peleg [25] showed that for every integer $k \geq 1$ and $\epsilon > 0$ there is a $(1 + \epsilon, \beta)$ -spanner with $O(\beta n^{1+1/k})$ edges, where β depends on k and ϵ but independent on n . Baswana et al. [11] presented a $(1, 6)$ -spanner $O(n^{4/3})$ edges. Woodruff [54] presented an $\tilde{O}(n^2)$ time algorithm that computes a $(1, 6)$ -spanner with $O(n^{4/3})$ edges. Chechik [18] presented a $(1, 4)$ -spanner $O(n^{7/5})$ edges. Recently, Abboud and Bodwin [1] showed that there is no additive spanner with constant error and $O(n^{4/3-\epsilon})$ edges.

Thorup and Zwick [48] introduced the notion of distance oracles, a data structure that stores for a weighted undirected graph. Its

size is $O(kn^{1+1/k})$ and given a pair of vertices $u, v \in V$ at distance $d(u, v)$ it returns in $O(k)$ time a $(2k - 1)$ -multiplicative approximation for $d(u, v)$. The original construction time was $O(mn^{1/k})$. Baswana and Sen [12] improved this to $O(n^2)$ for unweighted graphs. Baswana and Kavitha [10] extended the $O(n^2)$ running time to weighted graphs. Subsequently, Baswana, Gaur, Sen, and Upadhyay [9] obtained subquadratic construction time in unweighted graphs, at the price of increasing the approximation from $k \cdot d(u, v)$ to $k \cdot d(u, v) + O(1)$.

The original query time of the distance oracle was $O(k)$, which is not constant for any non-constant approximation. Chechik [19] gave an oracle with space $O(n^{1+1/k})$ and $O(1)$ query time, and the same approximation. Pătraşcu and Roditty [40] obtained a $(2, 1)$ -distance oracle that uses $\tilde{O}(n^{5/3})$ space and has $O(1)$ query time. Sommer [45] presented an $\tilde{O}(n^2)$ time algorithm that constructs such a distance oracle. The construction time was recently improved to $O(n^2)$ by Knudsen [30]. Pătraşcu et. al [35] presented infinity many distance oracles with fractional stretch factors that for graphs with $m = \tilde{O}(n)$ converge exactly to the integral stretch factors and the corresponding space bound of Thorup and Zwick. Thorup and Zwick [47] extended also their techniques from [48] to compact routing schemes.

The lower bounds presented in this paper was inspired by a lower bound by Pătraşcu and Roditty [40] who showed conditional hardness based on a conjecture on the hardness of a set intersection problem for the space usage of any distance oracle that can distinguish between distances 3 and 7.

C S-T DIAMETER HARDNESS

In the following we will prove that under SETH, our S-T Diameter algorithms are essentially optimal. We prove the following theorem:

THEOREM C.1. *Let $k \geq 2$ be an integer. There is an $O(kn^{k-1}d^{k-1})$ time reduction that transforms any instance of k -OV on sets of n d -dimensional vectors into a graph on $O(n^{k-1} + kn^{k-2}d^{k-1})$ nodes and $O(kn^{k-1}d^{k-1})$ edges and two disjoint sets S and T on n^{k-1} nodes each, so that if the k -OV instance has a solution, then $D_{S,T} \geq 3k - 2$, and if it does not, $D_{S,T} \leq k$.*

From the above Theorem we get that if there is some $k \geq 2$, $\varepsilon > 0$ and $\delta > 0$ so that there is an $O(M^{1+1/(k-1)-\varepsilon})$ time $(3 - 2/k - \delta)$ -approximation algorithm for S-T diameter in M -edge graphs, then k -OV has an $n^{k-\gamma}$ poly (d) -time algorithm for some $\gamma > 0$ and SETH is false.

We obtain an immediate corollary.

COROLLARY C.2. *For S-T diameter, under SETH, there is*

- *no $(2 - \varepsilon)$ -approximation algorithm running in $O(m^{2-\delta})$ time for any $\varepsilon > 0$, $\delta > 0$,*
- *no $O(m^{3/2-\delta})$ time $7/3 - \varepsilon$ -approximation algorithm for any $\varepsilon > 0$, $\delta > 0$,*
- *no $(m + n)^{1+o(1)}$ time, $3 - \varepsilon$ -approximation algorithm for any $\varepsilon > 0$.*

Proof of Theorem C.1. We will prove the theorem for $k = t + 2$ for any $t \geq 0$. Let $W_0, W_1, \dots, W_{t+1}$ be the sets of the $(t + 2)$ -OV instance, each containing n vectors in $\{0, 1\}^d$.

We will create a layered graph G on $t + 3$ layers, $L_0, \dots, L_{t+2}$, where the edges go only between adjacent layers L_i, L_{i+1} . We will set $S = L_0$ and $T = L_{t+2}$ for the S-T diameter instance. In particular, $D_{S,T} \geq t + 2$ because of the layering.

Let us describe the vertices of G .

L_0 consists of n^{t+1} vertices, each corresponding to a $t + 1$ -tuple $(a_0, a_1, \dots, a_t)$ where for each i , $a_i \in W_i$.

Similarly, L_{t+2} consists of n^{t+1} vertices, each corresponding to a $t + 1$ -tuple $(b_1, b_2, \dots, b_{t+1})$ where for each i , $b_i \in W_i$.

Layer L_1 consists of $n^t d^{t+1}$ vertices, each corresponding to a tuple $(a_0, \dots, a_{t-1}, \bar{x})$ where for each i , $a_i \in W_i$ and $\bar{x} = (x_0, \dots, x_t)$ is a $(t + 1)$ -tuple of coordinates in $[d]$. Similarly, L_{t+1} consists of $n^t d^{t+1}$ vertices, each corresponding to a tuple $(b_2, \dots, b_{t+1}, \bar{x})$ where for each i , $b_i \in W_i$ and $\bar{x}$ is a $(t + 1)$ -tuple of coordinates.

For every $j \in \{2, \dots, t\}$, L_j consists of $n^t d^{t+1}$ vertices $(a_0, \dots, a_{t-j}, b_{t+3-j}, \dots, b_{t+1}, \bar{x})$, where for each i , $a_i \in W_i$, $b_i \in W_i$ and $\bar{x} = (x_0, \dots, x_t)$ is a $(t + 1)$ -tuple of coordinates in $[d]$. In other words, there is a vector from W_i for every $i \notin \{t - j + 1, t - j + 2\}$.

Now let us define the edges.

Consider a node $(a_0, \dots, a_t) \in L_0$. For every $\bar{x} = (x_0, \dots, x_t)$, connect $(a_0, \dots, a_t)$ to $(a_0, \dots, a_{t-1}, \bar{x}) \in L_1$ if and only if for every $j \in \{0, \dots, t\}$, a_j is 1 in coordinates $x_0, \dots, x_{t-j}$.

For any $i \in \{1, \dots, t\}$ let's define the edges between L_i and L_{i+1} . For $(a_0, \dots, a_{t-i}, b_{t+3-i}, \dots, b_{t+1}, \bar{x}) \in L_i$ ³ and for any $c_{t+2-i} \in W_{t+2-i}$, add an edge to $(a_0, \dots, a_{t-i-1}, c_{t+2-i}, b_{t+3-i}, \dots, b_{t+1}, \bar{x}) \in L_{i+1}$. Here we “forget” vector a_{t-i} and replace it with c_{t+2-i} , leaving everything else the same.

Finally, the edges between L_{t+1} and L_{t+2} are as follows. Consider some $(b_1, \dots, b_{t+1}) \in L_{t+2}$. For every $\bar{x} = (x_0, \dots, x_t)$, connect $(b_1, \dots, b_{t+1})$ to $(b_2, \dots, b_{t+1}, \bar{x}) \in L_{t+1}$ if and only if for every $j \in \{1, \dots, t + 1\}$, b_j is 1 in coordinates $x_{t+1-j}, \dots, x_t$.

Figure 3 shows the construction of the graph for $t = 2$.

An important claim is as follows:

CLAIM 1. *For every $\bar{x}$, each $(a_0, \dots, a_{t-1}, \bar{x}) \in L_1$ is at distance t to every $(b_2, \dots, b_{t+1}, \bar{x}) \in L_{t+1}$.*

PROOF. Consider the path starting from $(a_0, \dots, a_{t-1}, \bar{x})$, and then for each $i \geq 1$ following the edges $(a_0, \dots, a_{t-i}, b_{t+3-i}, \dots, b_{t+1}, \bar{x}) \in L_i$ to $(a_0, \dots, a_{t-1-i}, b_{t+2-i}, \dots, b_{t+1}, \bar{x}) \in L_{i+1}$, until we reach $(b_2, \dots, b_{t+1}, \bar{x}) \in L_{t+1}$. This path exists by construction and has length t . $\square$

Now we proceed to prove the bounds on the S-T diameter.

LEMMA C.3. *If the $(t + 2)$ -OV instance has no solution, then $D_{S,T} = t + 2$.*

PROOF. If the $(t + 2)$ -OV instance has no solution, then for every $c_0 \in W_0, c_1 \in W_1, \dots, c_{t+1} \in W_{t+1}$, there is some coordinate x such that $c_0[x] = c_1[x] = \dots = c_{t+1}[x] = 1$.

Now consider the graph and any $(a_0, \dots, a_t) \in L_0, (b_1, \dots, b_{t+1}) \in L_{t+2}$. For every $j \in \{0, \dots, t\}$, let x_j be a coordinate so that $a_0, \dots, a_{t-j}, b_{t-j+1}, \dots, b_{t+1}$ are all 1 in x_j . Let $\bar{x} = (x_0, \dots, x_t)$.

By construction, $(a_0, \dots, a_t)$ has an edge to $(a_0, \dots, a_{t-1}, \bar{x})$ and $(b_2, \dots, b_{t+1}, \bar{x})$ has an edge to $(b_1, \dots, b_{t+1})$. Also, by Claim 1, $(a_0, \dots, a_{t-1}, \bar{x})$ has a path of length t to $(b_2, \dots, b_{t+1}, \bar{x})$.

³Here if $i = 1$, there are no b 's in the tuple.

Figure 3: The reduction graph from $(t+2)$ -OV for $t=2$. The figure depicts when a path of length $t+2$ exists between arbitrary $a_0a_1a_2 \in L_0$ and $b_1b_2b_3 \in L_{t+2}$. It also shows that when there is a path of length $t+2$ between $a_0a_1a_2 \in L_0$ and $a_1a_2a_3 \in L_{t+2}$, a_0, a_1, a_2, a_3 cannot be an orthogonal 4-tuple.

This shows that $D_{S,T} \leq t+2$; equality follows because the graph is layered. $\square$

Now we prove the guarantee for the case when an orthogonal tuple exists.

LEMMA C.4. *If there exist $a_0 \in W_0, \dots, a_{t+1} \in W_{t+1}$ that are orthogonal, then $D_{S,T} \geq 3t+4$.*

To prove the lemma, we will actually prove the following more general claim.

CLAIM 2. *Suppose that $a_0 \in W_0, \dots, a_{t+1} \in W_{t+1}$ are orthogonal. Let s be such that $0 \leq s \leq t$.*

Let $b_{t-s+j} \in W_{t-s+j}$ for all $j \in [1, \dots, s]$ be some other vectors, potentially different from a_{t-s+j} . Consider $\alpha = (a_0, a_1, \dots, a_{t-s}, b_{t-s+1}, \dots, b_t)$ and $\beta = (a_1, \dots, a_{t+1}) \in L_{t+2}$. Then the distance between α and β is at least $3t-2s+4$.

Symmetrically, let $c_j \in W_j$ for all $j \in [1, \dots, s]$ be some other vectors, potentially different from a_j . Consider $\alpha = (a_0, a_1, \dots, a_t)$ and $\beta = (c_1, \dots, c_s, a_{s+1}, \dots, a_{t+1}) \in L_{t+2}$. Then the distance between α and β is at least $3t-2s+4$.

If the claim is true, then using $s=0$ we get that the diameter is at least $3t+4$ so the lemma above is true. The claim for $s>0$ is useful for the rest of our constructions.

PROOF. We will show that the distance between $\alpha = (a_0, a_1, \dots, a_{t-s}, b_{t-s+1}, \dots, b_t) \in L_0$ and $\beta = (a_1, \dots, a_{t+1}) \in L_{t+2}$ is strictly more than $3t+2-2s$. Because the graph is layered and hence bipartite and $t+2 \equiv 3t+2 \pmod{2}$, the distance must be at least $3t-2s+4$.

Let's assume for contradiction that the shortest path P between α and β is of length $\leq 3t+2-2s$. First let's look at any subpath P' of P strictly within $M = L_1 \cup \dots \cup L_{t+1}$. All nodes on P' must share the same $\bar{x}$.

Furthermore, if P' starts with a node of L_1 and ends with a node of L_{t+1} , as P' needs to be a shortest path and by Claim 1, P' must be of length exactly t .

Next, notice that P cannot go from L_0 to L_{t+2} and then back to L_0 . This is because it needs to end up in L_{t+2} and any time it crosses over M , it would need to pay a distance of $t+2$, so P would have to have length at least $3t+6 > 3t+2-2s$.

Hence, P must be of the following form: a path from α through $L_0 \cup M$ back to L_0 (possibly containing only α), followed by a path crossing M to reach L_{t+2} , followed by a path through $L_{t+2} \cup M$ to L_{t+2} (possibly empty).

We will show that if P has length $\leq 3t+2-2s$ then P must contain a length $t+2$ subpath Q between a node $(a_0, \dots, a_q, w_{q+1}, \dots, w_t) \in L_0$, for some choices of the w 's and some $q \leq t-s$, and a node $(v_1, \dots, v_q, a_{q+1}, \dots, a_{t+1}) \in L_{t+2}$, for some choices of v 's.

That is, this path traverses M without weaving, by following $(a_0, \dots, a_q, w_{q+1}, \dots, w_{t-1}, \bar{x}) \in L_1$, $(a_0, \dots, a_q, w_{q+1}, \dots, w_{t-2}, a_{t+1}, \bar{x}) \in L_2, \dots, (v_2, \dots, v_q, a_{q+1}, \dots, a_{t-s}, \dots, a_{t+1}, \bar{x}) \in L_{t+1}$.

Suppose we show that such a subpath exists. Then by the construction of our graph we have that for every $i \in \{0, \dots, q\}$, $a_i[x_j] = 1$ for all $j \in \{0, \dots, t-i\}$, and that for all $i \in \{q+1, \dots, t+1\}$, $a_i[x_j] = 1$ for all $j \in \{t+1-i, \dots, t\}$. That is, for all i , $a_i[x_{t-q}] = 1$, and we get a contradiction since the a_i were supposed to be orthogonal.

Now let α^* be the last node from L_0 on P and let β^* be the first node of L_{t+1} of P . Let $a^* \in L_1$ be the node right after α^* and let $b^* \in L_{t+1}$ be the node right before β^* . Since the subpath of P between a^* and b^* is within M , it must share the same $\bar{x}$, and it must have length exactly t by Claim 1.

We will show that the subpath Q that we are looking for is the subpath of P between α^* and β^* . Its length is exactly what we want: $t+2$. It remains to show that for some $q \leq t-s$ and

some choices of w 's and v 's, $\alpha^* = (a_0, \dots, a_q, w_{q+1}, \dots, w_t)$ and $\beta^* = (v_1, \dots, v_q, a_{q+1}, \dots, a_{t+1})$.

Consider the path P_1 between $\alpha = (a_0, a_1, \dots, a_{t-s}, b_{t-s+1}, \dots, b_t)$ and α^* and the path P_2 between $\beta = (a_1, \dots, a_{t+1})$ and β^* . Let L_i be the layer in M with largest i that P_1 touches and let L_j be the layer in M with smallest j that P_2 touches.

For convenience, let us define $j' = t + 2 - j$. The length of P_1 is then at least $2i$ and the length of P_2 is at least $2j'$. The length $|P|$ of P equals $t + 2 + |P_1| + |P_2| \geq t + 2 + 2i + 2j' = t + 2 + 2(i + j')$. Since we have assumed that $|P| \leq 3t + 2 - 2s$, we must have that $t + 2 + 2(i + j') \leq 3t + 2 - 2s$ and hence $i + j' \leq t - s$.

Now, since P_1 goes at most to L_i , then from getting from α to α^* , at most the last i elements of $(a_0, a_1, \dots, a_{t-s}, b_{t-s+1}, \dots, b_t)$ can have been “forgotten”.

Hence, $\alpha^* = (a_0, \dots, a_{t-\max\{s, i\}}, b_{t-\max\{s, i\}+1}, \dots, b_{t-i}, w_{t-i+1}, \dots, w_t)$ for some w 's. (If $i \geq s$, the b 's do not appear.)

Similarly, between β and β^* , at most the first j' elements of β can have been forgotten. Thus, we have that $\beta^* = (v_1, \dots, v_{j'}, a_{j'+1}, \dots, a_{t+1})$ for some v 's.

Now, since $i + j' \leq t - s$, we must have that $j' \leq t - s - i \leq t - \max\{s, i\}$, and hence the path between α^* and β^* is the path Q we are searching for.

See Figure 4 for an illustration of L_i, L_j etc. in the case when $s = 0$. □

D LOWER BOUNDS FOR ECCENTRICITIES

D.1 Undirected Graphs

THEOREM D.1. *Let $k \geq 2$. Under the k -OV conjecture, every algorithm that can distinguish between eccentricity at most $2k - 1$ and eccentricity at least $4k - 3$ for every vertex in an $O(n)$ edge and node undirected graph, requires at least $n^{1+1/(k-1)-o(1)}$ time on a $O(\log n)$ -bit word-RAM.*

PROOF. Let's start with the S - T -diameter construction for k obtained from a given k -OV instance. We remove any internal nodes if they don't have edges to one of their adjacent layers - they don't hurt the instance. We have a graph on $O(n^{k-1}d^{k-2})$ vertices and edges with the following properties:

(1) Suppose that the k -OV instance has no k -OV solution. Then for every $s \in S, t \in T, d(s, t) = k$. Also, for every $s \in S$ and $u \notin S \cup T, d(s, u) \leq (k - 1) + k = 2k - 1$ since we can take a $\leq (k - 1)$ length path from u to some node $t \in T$ and since $d(s, t) = k$.

(2) If there is a k -OV solution, there are two nodes $s \in S, t \in T$ with $d(s, t) \geq 3k - 2$.

We modify the construction as follows. For every $s \in S$, we create an undirected path on $k - 2$ new vertices $s_1 \rightarrow s_2 \rightarrow \dots \rightarrow s_{k-2}$ and add an edge (s, s_1) ; let's call s by s_0 . Now, the distance between s_0 and s_i is i . Add a new node y and create edges (s_{k-2}, y) for every $s \in S$. Now, $d(y, s_0) = k - 1$ for every $s \in S$, and also for every $s, s' \in S$ and all $i, j \in \{0, \dots, k - 2\}$, we have that $d(s_i, s'_j) \leq 2k - 2$.

For every $s \in S, t \in T$, there is now potentially a new path between them, from s to y in $k - 1$ steps, then to some other s' in $k - 1$ steps and then to t using $\geq k$ steps. The length is $\geq 2(k - 1) + k = 3k - 2$, so when there is a k -OV solution, there is still a pair s, t at distance at least $3k - 2$.

Now, we also attach paths to the nodes in T . In particular, for each t , add an undirected path $t \rightarrow t_1 \rightarrow \dots \rightarrow t_{k-1}$. The distance between any $s \in S$ and any t_i is $i + d(s, t)$. Hence when there is no k -OV solution, the eccentricities of all s_0 for $s \in S$ are $\leq k + (k - 1) = 2k - 1$, and when there is a k -OV solution, there is $s \in S, t \in T$ so that $d(s_0, t_{k-1}) \geq (3k - 2) + (k - 1) = 4k - 3$. □

E DIAMETER LOWER BOUNDS

E.1 5 vs 8 Unweighted Undirected Construction

In this section we show that under the 3-OV Hypothesis, any algorithm that can distinguish between diameter 5 and 8 in sparse undirected unweighted graphs, requires $\Omega(n^{3/2-o(1)})$ time.

Theorem C.1 gives us the following theorem.

THEOREM E.1. *Given a 3-OV instance consisting of three sets $A, B, C \subseteq \{0, 1\}^d$, $|A| = |B| = |C| = n$, we can in $O(n^2 d^2)$ time construct an unweighted, undirected graph with $O(n^2 + nd^2)$ vertices and $O(n^2 d^2)$ edges that satisfies the following properties.*

- (1) *The graph consists of 4 layers of vertices S, L_1, L_2, T . The number of nodes in the sets is $|S| = |T| = n^2$ and $|L_1|, |L_2| \leq nd^2$.*
- (2) *S consists of all tuples (a, b) of vertices $a \in A$ and $b \in B$. Similarly, T consists of all tuples (b, c) of vertices $b \in B$ and $c \in C$.*
- (3) *If the 3-OV instance has no solution, then $d(u, v) = 3$ for all $u \in S$ and $v \in T$.*
- (4) *If the 3-OV instance has a solution $a \in A, b \in B, c \in C$ with a, b, c orthogonal, then $d((a, b) \in S, (b, c) \in T) \geq 7$. Furthermore, by Claim 2 (with $t = 1, s = 1$) for any $b' \in B$ we have $d((a, b) \in S, (b', c) \in T) \geq 5$ and $d((a, b') \in S, (b, c) \in T) \geq 5$.*
- (5) *For any vertex $u \in L_1$ there exists a vertex $s \in S$ that is adjacent to u . Similarly, for any vertex $v \in L_2$ there exists a vertex $t \in T$ that is adjacent to v . We can assume that this property holds because we can remove all vertices that do not satisfy this property from the graph and the resulting graph will still satisfy properties 3 or 4.*

In the rest of the section we use Theorem E.1 to prove the following result.

THEOREM E.2. *Given a 3-OV instance, we can in $O(n^2 d^2)$ time construct an unweighted, undirected graph with $O(n^2 + nd^2)$ vertices and $O(n^2 d^2)$ edges that satisfies the following two properties.*

- (1) *If the 3-OV instance has no solution, then for all pairs of vertices u and v we have $d(u, v) \leq 5$.*
- (2) *If the 3-OV instance has a solution, then there exists a pair of vertices u and v such that $d(u, v) \geq 8$.*

Construction of the graph. We construct a graph with the required properties by starting with the graph from Theorem E.1 and adding more vertices and edges. Figure 6 illustrates the construction of the graph. We start by adding a set S' of n^2 vertices. S' consists of all tuples (a, b) of vertices $a \in A$ and $b \in B$. We connect every $(a, b) \in S'$ to its counterpart $(a, b) \in S$. Thus, there is a matching between the sets of vertices S and S' . We also add another set S'' of n vertices. S'' contains one vertex a for every $a \in A$. For every pair of vertices from S'' we add an edge between the vertices. Thus, the n vertices form a clique. Furthermore, for every vertex $a \in S''$ we add

Figure 4: Here P contains at least 2 nodes in L_0 and at least 2 in L_{t+2} , and $s = 0$.

Undirected eccentricities from nodes in S : either $\leq 2k - 1$ or $\geq 4k - 3$.

Figure 5: The undirected eccentricities lower bound for $k = 5$.

an edge to $(a, b) \in S$ for all $b \in B$. In total we added $n^2 + n = O(n^2)$ vertices and $\binom{n}{2} + 2n^2 = O(n^2)$ edges. We do a similar construction for the set T of vertices. We add a set T' of n^2 vertices - one vertex for every tuple (b, c) of vertices $b \in B$ and $c \in C$. We connect every $(b, c) \in T'$ to $(b, c) \in T$. Finally, we add a set T'' of n vertices. T'' contains one vertex for every vector $c \in C$. For every pair of vertices from T'' we add an edge between the vertices. We connect every $c \in T''$ to $(b, c) \in T$ for all $b \in B$. This finishes the construction of the graph. In the rest of the section we show that the construction satisfies the promised two properties.

Correctness of the construction. We need to consider two cases.

Case 1: the 3-OV instance has no solution. In this case we want to show that for all pairs of vertices u and v we have $d(u, v) \leq 5$. We consider three subcases.

Case 1.1: $u \in S \cup S' \cup S'' \cup L_1$ and $v \in T \cup T' \cup T'' \cup L_2$. We observe that there exists $s \in S$ that has $d(u, s) \leq 1$. Indeed, if $u \in S$, then $s = u$ works. If $u \in S' \cup S''$, then we are done by the construction. On the other hand, if $u \in L_1$, then there exists such an $s \in S$ by property 5 from Theorem E.1. Similarly we can show that there

exists $t \in T$ such that $d(v, t) \leq 1$. Finally, by property 3 we have that $d(s, t) = 3$. Thus, we can upper bound the distance between u and v by $d(u, v) \leq d(u, s) + d(s, t) + d(t, v) \leq 1 + 3 + 1 = 5$ as required.

Case 1.2: $u, v \in S \cup S' \cup S'' \cup L_1$. From the previous case we know that there are two vertices $s_1, s_2 \in S$ such that $d(u, s_1) \leq 1$ and $d(s_2, v) \leq 1$. To show that $d(u, v) \leq 5$ it is sufficient to show that $d(s_1, s_2) \leq 3$. This is indeed true since both vertices s_1 and s_2 are connected to some two vertices in S'' and every two vertices in S'' are at distance at most 1 from each other.

Case 1.3: $u, v \in T \cup T' \cup T'' \cup L_2$. The case is analogous to the previous case.

Case 2: the 3-OV instance has a solution. In this case we want to show that there is a pair of vertices u, v with $d(u, v) \geq 8$. Let $a \in A, b \in B, c \in C$ be a solution to the 3-OV instance. We claim that $d((a, b) \in S', (b, c) \in T') \geq 8$. Let P be an optimal path between $u = ((a, b) \in S')$ and $v = ((b, c) \in T')$ that achieves the smallest distance. We want to show that P uses at least 8 edges. Let $t \in T$ be the first vertex from the set T that is on path P . Let $s \in S$ be the last vertex on path P that belongs to S and precedes t in P . We can

Figure 6: The illustration for the 5 vs 8 construction. The edges between sets S, L_1, L_2 and T are not depicted. The edges between vertices in S' and S (T and T') form a matching. Vertices in S'' (T'') form a clique.

easily check that, if $s \neq ((a, b) \in S)$, then $d(u, s) \geq 3$ and, similarly, if $t \neq ((b, c) \in T)$, then $d(t, v) \geq 3$. We consider three subcases.

Case 2.1: $s \neq ((a, b) \in S)$ and $t \neq ((b, c) \in T)$. Since s and t are separated by two layers of vertices, we must have $d(s, t) \geq 3$. Thus we get lower bound $d(u, v) \geq d(u, s) + d(s, t) + d(t, v) \geq 3 + 3 + 3 = 9 > 8$ as required.

Case 2.2: $s = ((a, b) \in S)$ and $t = ((b, c) \in T)$. In this case we use property 4 and conclude $d(u, v) \geq d(u, s) + d(s, t) + d(t, v) = 1 + d((a, b) \in S, (b, c) \in T) + 1 \geq 1 + 7 + 1 = 9 > 8$ as required.

Case 2.3: either $s = ((a, b) \in S)$ or $t = ((b, c) \in T)$ holds but not both. W.l.o.g. $s \neq ((a, b) \in S)$ and $t = ((b, c) \in T)$. If the path uses an edge in the clique on S'' before arriving at s , then $d(u, s) \geq 4$ and we get that $d(u, v) \geq d(u, s) + d(s, t) + d(t, v) \geq 4 + 3 + 1 = 8$. On the other hand, if the path does not use any edge of the clique, then $s = ((a, b') \in S)$ for some $b' \in B$. By property 4 we have $d(s, t) = d((a, b') \in S, (b, c) \in T) \geq 5$. We conclude that $d(u, v) \geq d(u, s) + d(s, t) + d(t, v) \geq 3 + 5 + 1 = 9 > 8$ as required.

ACKNOWLEDGMENTS

Arturs Backurs was supported by an IBM PhD Fellowship, the NSF and the Simons Foundation. Virginia Vassilevska Williams was supported by an NSF CAREER Award, NSF Grants CCF-1417238, CCF-1528078 and CCF-1514339, a BSF Grant BSF:2012338 and a Sloan Research Fellowship. Nicole Wein was supported by an NSF Graduate Fellowship and NSF Grant CCF-1514339.

REFERENCES

- [1] A. Abboud and G. Bodwin. 2016. The $4/3$ additive spanner exponent is tight. In *STOC*. 351–361.
- [2] Amir Abboud, Virginia Vassilevska Williams, and Joshua R. Wang. 2016. Approximation and Fixed Parameter Subquadratic Algorithms for Radius and Diameter in Sparse Graphs. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*. 377–391.
- [3] Amir Abboud, Richard Ryan Williams, and Huacheng Yu. 2015. More Applications of the Polynomial Method to Algorithm Design. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*. 218–230.
- [4] I. Abraham and C. Gavoille. 2011. On Approximate Distance Labels and Routing Schemes with Affine Stretch. In *DISC*. 404–415.
- [5] Rachit Agarwal and Philip Brighten Godfrey. 2013. Brief announcement: a simple stretch 2 distance oracle. In *ACM Symposium on Principles of Distributed Computing, PODC '13, Montreal, QC, Canada, July 22-24, 2013*. 110–112.
- [6] D. Aingworth, C. Chekuri, P. Indyk, and R. Motwani. 1999. Fast Estimation of Diameter and Shortest Paths (Without Matrix Multiplication). *SIAM J. Comput.* 28, 4 (1999), 1167–1181.
- [7] Noga Alon, Zvi Galil, and Oded Margalit. 1997. On the Exponent of the All Pairs Shortest Path Problem. *J. Comput. Syst. Sci.* 54, 2 (1997), 255–262.
- [8] I. Althöfer, G. Das, D. Dobkin, D. Joseph, and J. Soares. 1993. On sparse spanners of weighted graphs. *Discrete & Computational Geometry* 9, 1 (1993), 81–100.
- [9] Surender Baswana, Akshay Gaur, Sandeep Sen, and Jayant Upadhyay. 2008. Distance Oracles for Unweighted Graphs: Breaking the Quadratic Barrier with Constant Additive Error. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part I: Tack A: Algorithms, Automata, Complexity, and Games*. 609–621.
- [10] S. Baswana and T. Kavitha. 2010. Faster Algorithms for All-pairs Approximate Shortest Paths in Undirected Graphs. *SIAM J. Comput.* 39, 7 (2010), 2865–2896.
- [11] Surender Baswana, Telikepalli Kavitha, Kurt Mehlhorn, and Seth Pettie. 2010. Additive spanners and (α, β) -spanners. *ACM Trans. Algorithms* 7, 1 (2010), 5:1–5:26.
- [12] Surender Baswana and Sandeep Sen. 2006. Approximate distance oracles for unweighted graphs in expected $O(n^2)$ time. *ACM Trans. Algorithms* 2, 4 (2006), 557–577.

- [13] Surender Baswana and Sandeep Sen. 2007. A simple and linear time randomized algorithm for computing sparse spanners in weighted graphs. *Random Struct. Algorithms* 30, 4 (2007), 532–563.
- [14] Michele Borassi, Pierluigi Crescenzi, Michel Habib, Walter A Kusters, Andrea Marino, and Frank W Takes. 2015. Fast diameter and radius BFS-based computation in (weakly connected) real-world graphs: With an application to the six degrees of separation games. *Theoretical Computer Science* 586 (2015), 59–80.
- [15] Massimo Cairo, Roberto Grossi, and Romeo Rizzi. 2016. New Bounds for Approximating Extremal Distances in Undirected Graphs. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10–12, 2016*. 363–376.
- [16] Timothy M. Chan. 2012. All-pairs shortest paths for unweighted undirected graphs in $\tilde{O}(mn)$ time. *ACM Trans. Algorithms* 8, 4 (2012), 34:1–34:17.
- [17] Timothy M. Chan and Ryan Williams. 2016. Deterministic APSP, Orthogonal Vectors, and More: Quickly Derandomizing Razborov-Smolensky. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10–12, 2016*. 1246–1255.
- [18] S. Chechik. 2013. New Additive Spanners. In *SODA*. 498–512.
- [19] Shiri Chechik. 2015. Approximate Distance Oracles with Improved Bounds. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14–17, 2015*. 1–10.
- [20] Shiri Chechik, Daniel H. Larkin, Liam Roditty, Grant Schoenebeck, Robert Endre Tarjan, and Virginia Vassilevska Williams. 2014. Better Approximation Algorithms for the Graph Diameter. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5–7, 2014*. 1041–1052.
- [21] F.R.K Chung. 1987. Diameters of graphs: Old problems and new results. *Congressus Numerantium* 60 (1987), 295–317.
- [22] E. Cohen and U. Zwick. 2001. All-Pairs Small-Stretch Paths. *J. Algorithms* 38, 2 (2001), 335–353.
- [23] Marek Cygan, Harold N. Gabow, and Piotr Sankowski. 2015. Algorithmic Applications of Baur-Strassen’s Theorem: Shortest Cycles, Diameter, and Matchings. *J. ACM* 62, 4 (Sept. 2015), 28:1–28:30.
- [24] D. Dor, S. Halperin, and U. Zwick. 2000. All-Pairs Almost Shortest Paths. *SIAM J. Comput.* 29, 5 (2000), 1740–1759.
- [25] Michael Elkin and David Peleg. 2004. $(1+\epsilon, \beta)$ -Spanner Constructions for General Graphs. *SIAM J. Comput.* 33, 3 (2004), 608–631.
- [26] François Le Gall and Florent Urrutia. 2017. Improved Rectangular Matrix Multiplication using Powers of the Coppersmith-Winograd Tensor. *CoRR* abs/1708.05622 (2017).
- [27] E. A. Hirsch. 1998. Two new upper bounds for SAT. In *Proc. SODA*. 521–530.
- [28] R. Impagliazzo, R. Paturi, and F. Zane. 2001. Which Problems Have Strongly Exponential Complexity? *J. Comput. Syst. Sci.* 63, 4 (2001), 512–530.
- [29] Naoki Katoh and Kazuo Iwano. 1992. Finding K Farthest Pairs and K Closest/Farthest Bichromatic Pairs for Points in the Plane. In *Proceedings of the Eighth Annual Symposium on Computational Geometry (SCG '92)*. 320–329.
- [30] Mathias Bæk Tejs Knudsen. 2017. Additive Spanners and Distance Oracles in Quadratic Time. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10–14, 2017, Warsaw, Poland*. 64:1–64:12.
- [31] François Le Gall. 2012. Faster Algorithms for Rectangular Matrix Multiplication. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20–23, 2012*. 514–523.
- [32] François Le Gall. 2014. Powers of tensors and fast matrix multiplication. In *International Symposium on Symbolic and Algebraic Computation, ISSAC '14, Kobe, Japan, July 23–25, 2014*. 296–303.
- [33] T. C. Lin, M. J. Wu, W. J. Chen, and B. Y. Wu. 2016. Computing the Diameters of Huge Social Networks. In *2016 International Computer Symposium (ICS)*. 6–11.
- [34] Andrea Lincoln, Virginia Vassilevska Williams, and Ryan Williams. 2018. Tight Hardness for Shortest Cycles and Paths in Sparse Graphs. *SODA* (2018). to appear.
- [35] Mihai Patrascu, Liam Roditty, and Mikkel Thorup. 2012. A New Infinity of Distance Oracles for Sparse Graphs. In *FOCS*.
- [36] R. Paturi, P. Pudlák, M. E. Saks, and F. Zane. 2005. An improved exponential-time algorithm for k -SAT. *J. ACM* 52, 3 (2005), 337–364.
- [37] David Peleg, Liam Roditty, and Elad Tal. 2012. Distributed Algorithms for Network Diameter and Girth. In *Automata, Languages, and Programming: 39th International Colloquium, ICALP 2012, Warwick, UK, July 9–13, 2012, Proceedings, Part II*. 660–672.
- [38] S. Pettie. 2004. A new approach to all-pairs shortest paths on real-weighted graphs. *Theor. Comput. Sci.* 312, 1 (2004), 47–74.
- [39] Seth Pettie and Vijaya Ramachandran. 2005. A Shortest Path Algorithm for Real-Weighted Undirected Graphs. *SIAM J. Comput.* 34, 6 (2005), 1398–1431.
- [40] M. Patrascu and L. Roditty. 2010. Distance Oracles beyond the Thorup–Zwick Bound. In *Proc. FOCS*. 815–823.
- [41] Liam Roditty and Virginia Vassilevska Williams. 2013. Fast approximation algorithms for the diameter and radius of sparse graphs. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing (STOC '13)*. ACM, New York, NY, USA, 515–524. <https://doi.org/10.1145/2488608.2488673>
- [42] U. Schöning. 1999. A probabilistic algorithm for k -SAT and constraint satisfaction problems. In *Proc. FOCS*. 410–414.
- [43] R. Seidel. 1995. On the all-pairs-shortest-path problem in unweighted undirected graphs. *JCSS* 51 (1995), 400–403.
- [44] A. Shoshan and U. Zwick. 1999. All Pairs Shortest Paths in Undirected Graphs with Integer Weights. In *Proc. FOCS*. 605–614.
- [45] Christian Sommer. 2016. All-Pairs Approximate Shortest Paths and Distance Oracle Preprocessing. In *43rd International Colloquium on Automata, Languages, and Programming (ICALP 2016) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 55. 55:1–55:13.
- [46] A. Stothers. 2010. On the Complexity of Matrix Multiplication. *Ph.D. Thesis, U. Edinburgh* (2010).
- [47] Mikkel Thorup and Uri Zwick. 2001. Compact routing schemes. In *SPAA*. 1–10.
- [48] Mikkel Thorup and Uri Zwick. 2005. Approximate distance oracles. *J. ACM* (2005).
- [49] Virginia Vassilevska Williams. 2015. Hardness of Easy Problems: Basing Hardness on Popular Conjectures such as the Strong Exponential Time Hypothesis (Invited Talk). In *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16–18, 2015, Patras, Greece*. 17–29.
- [50] V. Vassilevska Williams and R. Williams. 2010. Subcubic equivalences between path, matrix and triangle problems. In *Proc. FOCS*. 645–654.
- [51] R. Williams. 2005. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theor. Comput. Sci.* 348, 2–3 (2005), 357–365.
- [52] Ryan Williams. 2014. Faster all-pairs shortest paths via circuit complexity. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 – June 03, 2014*. 664–673.
- [53] Virginia Vassilevska Williams. 2012. Multiplying matrices faster than Coppersmith-Winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. ACM, 887–898.
- [54] D. P. Woodruff. 2006. Lower Bounds for Additive Spanners, Emulators, and More. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS '06)*. 389–398.
- [55] U. Zwick. 2002. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *J. ACM* 49, 3 (2002), 289–317.