

New Approximate Distance Oracles and Their Applications

Avi Katria 

Bar Ilan University, Ramat Gan, Israel

Liam Roditty 

Bar Ilan University, Ramat Gan, Israel

Abstract

Let $G = (V, E)$ be an undirected graph with n vertices and m edges, and let $\mu = m/n$. A *distance oracle* is a data structure designed to answer approximate distance queries, with the goal of achieving low stretch, efficient space usage, and fast query time. While much of the prior work focused on distance oracles with constant query time, this paper presents a comprehensive study of distance oracles with non-constant query time. We explore the tradeoffs between space, stretch, and query time of distance oracles in various regimes. Specifically, we consider both weighted and unweighted graphs in the regimes of stretch < 2 and stretch ≥ 2 . In addition, we demonstrate several applications of our new distance oracles to the n -Pairs Shortest Paths (n -PSP) problem and the All Nodes Shortest Cycles ($ANSC$) problem. Our main contributions are:

- **Weighted graphs:** We present a new three-way trade-off between stretch, space, and query time, offering a natural extension of the classical Thorup–Zwick distance oracle [STOC’01 and JACM’05] to regimes with larger query time. Specifically, for any $0 < r < 1/2$ and integer $k \geq 1$, we construct a $(2k(1-2r)-1)$ -stretch distance oracle with $\tilde{O}(m+n^{1+1/k})$ space and $\tilde{O}(\mu n^r)$ query time. This construction provides an asymptotic improvement over the classical $(2k-1)$ -stretch and $O(n^{1+1/k})$ -space tradeoff of Thorup and Zwick in sparse graphs, at the cost of increased query time. We also improve upon a result of Dalirrooyfard et al. [FOCS’22], who presented a $(2k-2)$ -stretch distance oracle with $O(m+n^{1+1/k})$ space and $O(\mu n^{1/k})$ query time. In our oracle we reduce the stretch from $(2k-2)$ to $(2k-5)$ while preserving the same space and query time.
- **Unweighted graphs:** We present a $(2k-5, 4+2_{\text{ODD}})$ -approximation¹ distance oracle with $O(n^{1+1/k})$ space and $O(n^{1/k})$ query time. This improves upon a $(2k-2, 2_{\text{ODD}})$ -approximation distance oracle of Dalirrooyfard et al. [FOCS’22] while maintaining the same space and query time. We also present a distance oracle that given $u, v \in V$ returns an estimate $\hat{d}(u, v) \leq d(u, v) + 2\lceil d(u, v)/3 \rceil + 2$, using $O(n^{4/3+2\varepsilon})$ space and $O(n^{1-3\varepsilon})$ query time. To the best of our knowledge, this is the first distance oracle that simultaneously achieves a multiplicative stretch < 2 , and a space complexity $O(n^{1.5-\alpha})$, for some $\alpha > 0$.
- **Applications for n -PSP and $ANSC$:** We present an $\tilde{O}(m^{1-1/(k+1)}n)$ -time algorithm for the n -PSP problem, that for every input pair $\langle s_i, t_i \rangle$, where $i \in [n]$, returns an estimate $\hat{d}(s_i, t_i)$ such that $\hat{d}(s_i, t_i) \leq d(s_i, t_i) + 2\lceil d(s_i, t_i)/2k \rceil$. By allowing a small additive error, this result circumvents the conditional running time lower bound of $\Omega\left(m^{2-\frac{2}{k+1}} \cdot n^{\frac{1}{k+1}-o(1)}\right)$, established by Dalirrooyfard et al. [FOCS’22] for achieving $(1+1/k)$ -stretch. Additionally, we present an $\tilde{O}(mn^{1-1/k})$ -time algorithm for the $ANSC$ problem that computes, for every $u \in V$, an estimate $\hat{c}_u$ such that $\hat{c}_u \leq SC(u) + 2\lceil SC(u)/2(k-1) \rceil$, where $SC(u)$ denotes the length of the shortest cycle containing u . This improves upon the $\tilde{O}(m^{2-2/k}n^{1/k})$ -time algorithm of Dalirrooyfard et al. [FOCS’22], while achieving the same approximation guarantee.

We obtain our results by developing several new techniques, among them are the *borderline vertices* technique and the *middle vertex* technique, which may be of independent interest.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Distance oracles, Fine-grained algorithms, Graph algorithms, Data structures

¹ $2_{\text{ODD}} = 2 \cdot (d(u, v) \bmod 2)$.



© Avi Katria and Liam Roditty;

licensed under Creative Commons License CC-BY 4.0

36th International Symposium on Algorithms and Computation (ISAAC 2025).

Editors: Ho-Lin Chen, Wing-Kai Hon, and Meng-Tsung Tsai; Article No. 43; pp. 43:1–43:17

Leibniz International Proceedings in Informatics



LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2025.43

Related Version Full Version: <https://arxiv.org/abs/2509.00890> [27]

Funding Liam Roditty: Supported in part by BSF grants 2016365 and 2020356.

1 Introduction

Let $G = (V, E)$ be an undirected graph with $n = |V|$ vertices and $m = |E|$ edges with non-negative real edge weights, and let $\mu = m/n$ be the average degree. The distance $d_G(u, v)$ between u and v is the length of the shortest path² between u and v in G . An estimation $\hat{d}(u, v)$ of $d(u, v)$ is an α -stretch if $d(u, v) \leq \hat{d}(u, v) \leq \alpha \cdot d(u, v)$.

In their seminal work, Thorup and Zwick [36] presented a data structure for distance approximations called a *distance oracle*. A distance oracle is a data structure with $o(n^2)$ space (otherwise we can trivially store the distance matrix), that supports distance queries between vertices. For any integer $k \geq 1$, they [36] showed that in $O(kmn^{1/k})$ expected time, it is possible to preprocess a weighted undirected graph and create a distance oracle of size $O(kn^{1+1/k})$. For every $u, v \in V$ the query of the distance oracle returns a $(2k - 1)$ -stretch for $d(u, v)$ in $O(k)$ time. Different aspects of distance oracles, such as construction time, query time, and stretch, have been studied since their introduction, more than two decades ago. For more details see for example [10, 29, 38, 18, 19, 23, 25, 26, 21, 4, 3, 12, 13, 15].

Thorup and Zwick [36] employed Erdős' girth conjecture³ to establish a lower bound on the space/stretch tradeoff for distance oracles. The conjecture asserts the existence of graphs with $\Omega(n^{1+1/k})$ edges and girth $\geq 2k + 2$. Under this assumption, they proved that any distance oracle with stretch $t \leq 2k - 1$ must use $\Omega(n^{1+1/k})$ bits on some input, for all $k \geq 1$. However, these lower bounds apply to graphs with $m = \Omega(n^{1+1/k})$ edges.

This motivates studying the $(2k - 1)$ -stretch/ $O(n^{1+1/k})$ -space tradeoff in sparser graphs, where $m = o(n^{1+1/k})$.⁴ This approach has been applied in two different stretch regimes. Distance oracles with stretch ≥ 2 (see, for example, [7, 6, 9, 33, 5, 25]) and distance oracles with stretch < 2 (see, for example, [6, 8, 21, 28, 31]).

1.1 Distance oracles with stretch ≥ 2

Our main result for stretch ≥ 2 establishes a three-way tradeoff among the key parameters of distance oracles: stretch, space, and query time. This new tradeoff improves upon previously known results for specific values of stretch, space, and query time. We prove the following:

► **Theorem 1.** *Let $k \geq 5$ and let $0 < c < \frac{k}{2} - 1$ be an integer. There is an $\tilde{O}(m + n^{1+1/k})$ space distance oracle that given two query vertices $u, v \in V$ computes in $O(\mu n^{c/k})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq (2k - 1 - 4c) \cdot d(u, v)$. The distance oracle is constructed in $\tilde{O}(mn^{\frac{c+2}{k}})$ expected time.*

An interesting tradeoff that follows from Theorem 1 is a better space/stretch tradeoff than the classical Thorup and Zwick [36] $(2k - 1)$ -stretch / $\tilde{O}(n^{1+1/k})$ space tradeoff, at a cost of a larger query time. Specifically, by setting $r = \frac{c}{k}$, we get a $(2k(1 - 2r) - 1)$ -stretch

² We omit G when it is clear from the context.

³ The girth of a graph is the length of its shortest cycle.

⁴ We remark that a distance oracle in such graphs must use $\Omega(m)$ space, because of similar arguments to the lower bound of [36].

distance oracle that uses $\tilde{O}(m + n^{1+1/k})$ space, at the price of $\tilde{O}(\mu n^r)$ query time. By setting $c = 1$ in Theorem 1, we improve upon a recent result of Dalirrooyfard, Jin, V. Williams, and Wein [22]. They presented a $(2k - 2)$ -stretch distance oracle that uses $\tilde{O}(m + n^{1+1/k})$ space and $\tilde{O}(\mu n^{1/k})$ query time.⁵ We improve the stretch from $(2k - 2)$ to $(2k - 5)$ while using the same space and query time.

To obtain Theorem 1, we develop a new technique called the *borderline vertices* technique. Given $u, v \in V$, the borderline vertices τ_u and τ_v are two vertices on the shortest path between u and v that satisfy special properties that allow us to better exploit the structure of the Thorup and Zwick distance oracle. Using the borderline vertices technique, we also manage to achieve the following distance oracle, which improves upon the construction time of Theorem 1 at the cost of increasing the stretch.

► **Theorem 2.** *Let $k \geq 4$ and let $0 < c < \frac{k}{2} - 1$ be an integer. There is an $\tilde{O}(m + n^{1+1/k})$ space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(\mu n^{c/k})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq (2k - 4c) \cdot d(u, v)$. The distance oracle is constructed in $\tilde{O}(mn^{\frac{c+1}{k}})$ expected time.*

Agarwal, Godfrey, and Har-Peled [9] considered also small stretches, and presented a 2-stretch distance oracle that uses $\tilde{O}(m + n^{3/2})$ space and has $\tilde{O}(\mu n^{1/2})$ query time, that is constructed in $\tilde{O}(mn^{1/2})$ time, and a 3-stretch distance oracle that uses $\tilde{O}(m + n^{4/3})$ space and has $\tilde{O}(\mu n^{1/3})$ query time, and is constructed in $\tilde{O}(mn^{2/3})$ time. These results suggest that the following general stretch/space/query time tradeoff may exist for every k .

► **Problem 1.** *For which values of k it is possible to construct a k -stretch distance oracle that uses $\tilde{O}(m + n^{1+1/k})$ space and has $\tilde{O}(\mu n^{1/k})$ query time.*

Agarwal, Godfrey, and Har-Peled [9] solved Problem 1 for $k = 2, 3$. By setting $c = 1$ in Theorem 2 and Theorem 1, respectively, we obtain a 4-stretch distance oracle that uses $\tilde{O}(m + n^{5/4})$ space and has $\tilde{O}(\mu n^{1/4})$ query time, and a 5-stretch distance oracle that uses $\tilde{O}(m + n^{6/5})$ space and has $\tilde{O}(\mu n^{1/5})$ query time. Thus, we solve Problem 1 for $k = 4, 5$.

In addition, using the borderline vertices technique we obtain $(2k - 3)$ -stretch distance oracle that uses $\tilde{O}(m + n^{1+1/k})$ space, has $\tilde{O}(\mu n^{1/k})$ query time, and is constructed in $\tilde{O}(mn^{1/k})$ time. For $k = 3$ this improves the construction time of [9] for a 3-stretch distance oracle from $\tilde{O}(mn^{2/3})$ to $\tilde{O}(mn^{1/3})$. Our results for weighted graphs are summarized in Table 2.

Next, we turn our focus to *unweighted* graphs. An estimation $\hat{d}(u, v)$ of $d(u, v)$ is an (α, β) -approximation if $d(u, v) \leq \hat{d}(u, v) \leq \alpha d(u, v) + \beta$. (α, β) -approximations were extensively studied in the context of distance oracles, graph spanners, and emulators. (For more details see for example [24, 37, 11, 17, 16, 30, 1, 2, 9, 7, 13]).

From the girth conjecture, it follows that in unweighted graphs, for every (α, β) -distance oracle that uses $\tilde{O}(n^{1+1/k})$ space, it must hold that $\alpha + \beta \geq 2k - 1$. However, this inequality must hold only for adjacent vertices. For non-adjacent vertex pairs, we prove the following simple lower bound.

► **Theorem 3.** *Assuming the Erdős girth conjecture, any distance oracle that uses $o(n^{1+1/k})$ space must have an input graph $G = (V, E)$ and two vertices $u, v \in V$ such that $d(u, v) = 2$ and $\hat{d}(u, v) \geq 2k + 2$.*

⁵ We remark that even more recently, Chechik, Hoch, and Lifshitz [20] presented a $(2k - 3)$ -stretch n -PSP algorithm, however, they did not present a new distance oracle.

A *spanner* is a subgraph $H \subseteq G$ that approximates distances *without* supporting distance queries. Baswana, Kavitha, Mehlhorn, and Pettie [11] and Parter [30] presented a k -spanner for every $u, v \in V$ such that $d(u, v) = 2$, and matched the lower bound of Theorem 3 for spanners. An interesting question is whether this lower bound can be matched by distance oracles with efficient distance queries as well. Thus, we formulate the following problem.

► **Problem 2.** *For which values of k it is possible to construct a distance oracle with $\tilde{O}(n^{1+1/k})$ space that uses $\tilde{O}(n^{1/k})$ query time, such that $d(u, v) \leq \hat{d}(u, v) \leq k \cdot d(u, v)$, for every $u, v \in V$ that satisfy $d(u, v) = 2$.*

Agarwal, Godfrey, and Har-Peled [9] presented for unweighted graphs a $(2, 1)$ -approximation distance oracle that uses $\tilde{O}(n^{3/2})$ space, has $\tilde{O}(n^{1/2})$ query time, and is constructed in $\tilde{O}(mn^{1/2})$ time. They also presented a $(3, 2)$ -approximation distance oracle that uses $\tilde{O}(n^{4/3})$ space, has $\tilde{O}(n^{1/3})$ query time, that is constructed in $\tilde{O}(mn^{2/3})$ time.

By analyzing more carefully the $(2, 1)$ -approximation distance oracle of [9] it is possible to show that it is actually a $(2, 1_{\text{ODD}})$ -approximation⁶ distance oracle, and therefore solves Problem 2 for $k = 2$.⁷ In the following two theorems, we show that it is possible to solve Problem 2 for $k = 3$ and $k = 4$ as well.

► **Theorem 4.** *There is a $\tilde{O}(n^{4/3})$ space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{1/3})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq 3d(u, v) + 2_{\text{ODD}}$. The distance oracle is constructed in $\tilde{O}(mn^{1/3})$ expected time.*

► **Theorem 5.** *There is a $\tilde{O}(n^{5/4})$ space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{1/4})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq 4d(u, v) + 3_{\text{ODD}}$. The distance oracle is constructed in $\tilde{O}(mn^{1/2})$ expected time.*

In Theorem 4 we improve the distance oracle of [9] from more than a decade ago. In particular we improve the construction time from $\tilde{O}(mn^{2/3})$ to $\tilde{O}(mn^{1/3})$, and the approximation from $(3, 2)$ to $(3, 2_{\text{ODD}})$. Dalirrooyfard, Jin, V. Williams, and Wein [22] presented a $(2k - 2, 2_{\text{ODD}})$ -stretch distance oracle that uses $\tilde{O}(n^{1+1/k})$ space and $\tilde{O}(n^{1/k})$ query time. By setting $k = 4$ in the $(2k - 2, 2_{\text{ODD}})$ -stretch distance oracle of [22] they obtain a $(6, 2_{\text{ODD}})$ -stretch distance oracle. In Theorem 5 we improve the approximation from $(6, 2_{\text{ODD}})$ to $(4, 3_{\text{ODD}})$ while maintaining the same space and query time.

To obtain our new distance oracles for unweighted graphs, presented in Theorem 4 and Theorem 5, we develop a new technique for using the middle vertex in the path, called the *middle vertex* technique. Let $u, v \in V$, the middle vertex τ is a vertex on the shortest path between u and v that satisfies $d(u, \tau) = d(u, v)/2 + 0.5_{\text{ODD}}$ and $d(v, \tau) = d(u, v)/2 - 0.5_{\text{ODD}}$. As with the borderline vertices technique for *sparse weighted* graphs, the middle vertex technique enables us to better exploit the structure of the distance oracles of Thorup and Zwick in *dense unweighted* graphs.

By combining the middle vertex technique with the borderline vertices technique, we manage to achieve two more distance oracles for dense unweighted graphs. The first distance oracle improves the approximation of the distance oracle of Theorem 5 from $(4, 3_{\text{ODD}})$ to $(3, 2 + 2_{\text{ODD}})$ at the cost of increasing the query time to $\tilde{O}(n^{1/2})$.

⁶ x_{ODD} is defined as $x \cdot 1_{\text{ODD}}$, where 1_{ODD} is $d(u, v) \bmod 2$

⁷ In their work, the additive error comes from the fact that if $B(u) \cap B(v) = \emptyset$ then $\min(h(u), h(v)) \leq d(u, v)/2 + 1$. However, this can be improved by showing that if $B(u) \cap B(v) = \emptyset$ then $\min(h(u), h(v)) \leq d(u, v)/2 + 1_{\text{ODD}}$.

► **Theorem 6.** *There is a $\tilde{O}(n^{5/4})$ space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{1/2})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq 3d(u, v) + 2 + 2_{\text{ODD}}$. The distance oracle is constructed in $\tilde{O}(mn^{1/2})$ expected time.*

The second distance oracle improves the $(2k - 2, 2_{\text{ODD}})$ -approximation of [22] to $(2k - 5, 4 + 2_{\text{ODD}})$ -approximation, while using the same space and query time.

► **Theorem 7.** *Let $k \geq 5$. There is an $\tilde{O}(n^{1+1/k})$ space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{1/k})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq (2k - 5)d(u, v) + 4 + 2_{\text{ODD}}$. The distance oracle is constructed in $\tilde{O}(mn^{\frac{3}{k}})$ expected time.*

Our results for unweighted graphs are summarized in Table 1.

1.2 Distance oracles with stretch < 2

For distance oracles with stretch < 2 in weighted graphs, we present the following new three-way tradeoff between: stretch, space, and query time. We prove:

► **Theorem 8.** *Let $0 < c < 1/3$ be a real constant. There is a $\tilde{O}(m + n^{2-2c})$ -space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(\mu^t n^{tc})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq (1 + 2/t)d(u, v)$. The distance oracle is constructed in $\tilde{O}(mn^{1-c})$ time.*

Using this tradeoff, we obtain the first distance oracle with stretch < 2 , sublinear query time, and $\tilde{O}(m + n^{1.5-\alpha})$ space, for $\alpha > 0$. In particular, by setting $t = 3$ and $c = 1/3 - \epsilon$, we obtain a $5/3$ -stretch distance oracle with $\tilde{O}(m + n^{4/3+2\epsilon})$ -space and $O(\mu^3 n^{1-3\epsilon})$ -query time. For comparison, Agarwal [6] presented a distance oracle with stretch $1 + \frac{1}{t+0.5}$, using $\tilde{O}(m + n^{2-c})$ space and $\tilde{O}(\mu^t n^{(t+1)c})$ query time. Setting $t = 1$ and $c = 1/2 - \epsilon$ in the construction of [6] yields a $5/3$ -stretch oracle with $\tilde{O}(m + n^{1.5+\epsilon})$ -space and $\tilde{O}(\mu n^{0.5-\epsilon})$ -query time. (See Table 2.)

For unweighted graphs, we present two new tradeoffs. The first improves upon the tradeoff of the distance oracle of Bilò, Chechik, Choudhary, Cohen, Friedrich, and Schirneck [13], while preserving the same space and query time. Specifically, they presented a distance oracle for unweighted graphs that uses $\tilde{O}(n^{2-c})$ space, has $\tilde{O}(n^{tc})$ query time, and returns an estimate $\hat{d}(u, v) \leq (1 + 1/t)d(u, v) + 2$. In the following theorem, we slightly improve the stretch bound to $\hat{d}(u, v) \leq d(u, v) + 2 \left\lceil \frac{d(u, v)}{2t} \right\rceil$, while using the same space and query time.

► **Theorem 9.** *Let $t \geq 1$ be an integer and let $0 < c < 1/2$ be a real constant. There is a $\tilde{O}(n^{2-c})$ -space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{c \cdot t})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq d(u, v) + 2 \left\lceil \frac{d(u, v)}{2t} \right\rceil$. The distance oracle is constructed in $\tilde{O}(mn^{1-c})$ time.*

Next, by adapting methods of Theorem 8 to unweighted graphs, we obtain another tradeoff that uses less space than the tradeoff of Theorem 9 at the cost of a larger query time. We prove:

► **Theorem 10.** *Let $t \geq 1$. Let $0 < c < 1/3$ be a real constant. There is a $\tilde{O}(n^{2-2c})$ -space distance oracle that given two query vertices $u, v \in V$ computes in $\tilde{O}(n^{tc})$ -time a distance estimation $\hat{d}(u, v)$ that satisfies $d(u, v) \leq \hat{d}(u, v) \leq d(u, v) + 2 \lceil d(u, v)/t \rceil + 2$. The distance oracle is constructed in $\tilde{O}(mn^{1-c})$ time.*

■ **Table 1** Distance oracles in unweighted graphs. x_{ODD} is $x \cdot (d(u, v) \bmod 2)$.

Space	Time	Old Approximation	New Approximation
$\tilde{O}(kn^{1+1/k})$	$\tilde{O}(n^{1/k})$	$(2k - 2, 2_{\text{ODD}})$ [22]	$(2k - 5, 4 + 2_{\text{ODD}})$ (Th7)
$\tilde{O}(n^{4/3})$	$\tilde{O}(n^{1/3})$	$(3, 2)$ [7]	$(3, 2_{\text{ODD}})$ (Th4)
$\tilde{O}(n^{5/4})$	$\tilde{O}(n^{1/4})$	$(6, 2_{\text{ODD}})$ [22]	$(4, 3_{\text{ODD}})$ (Th5)
$\tilde{O}(n^{5/4})$	$\tilde{O}(n^{1/2})$	–	$(3, 2 + 2_{\text{ODD}})$ (Th6)
$\tilde{O}(n^{6/5})$	$\tilde{O}(n^{1/5})$	$(8, 2_{\text{ODD}})$ [22]	$(5, 4 + 2_{\text{ODD}})$
$\tilde{O}(n^{2-c})$	$\tilde{O}(n^{tc})$	$\delta + \delta/t + 2$ [13]	$\delta + 2\lceil \delta/2t \rceil$ (Th9)
$\tilde{O}(n^{3/2+\epsilon})$	$\tilde{O}(n^{1-2\epsilon})$	$1.5\delta + 2$ [13]	$\delta + 2\lceil \delta/4 \rceil$ (Th9)
$\tilde{O}(n^{2-2c})$	$\tilde{O}(n^{tc})$	–	$\delta + 2\lceil \delta/t \rceil + 2$ (Th10)
$\tilde{O}(n^{4/3+2\epsilon})$	$\tilde{O}(n^{1-3\epsilon})$	–	$\delta + 2\lceil \delta/3 \rceil + 2$ (Th10)

In particular, using Theorem 10 we obtain the first distance oracle with $o(n^{1.5-\alpha})$ -space, for $\alpha > 0$, that achieves a multiplicative error strictly better than 2. Specifically, by setting $t = 3$ and $c = 1/3 - \epsilon$, we get a distance oracle with $\tilde{O}(n^{4/3+2\epsilon})$ space and $\tilde{O}(n^{1-3\epsilon})$ query time. These results are summarized in Table 1.

1.3 Applications for n -PSP and $ANSC$

Recently, Dalirrooyfard, Jin, V. Williams, and Wein [22] (followed by Chechik, Hoch and Lifshitz [20]) studied two fundamental problems in graphs, the n -pairs shortest paths (n -PSP) problem and the all-nodes shortest cycles ($ANSC$) problem. The n -PSP problem is defined as follows. Given a set of vertex pairs (s_i, t_i) , for $1 \leq i \leq O(n)$, compute the distance $d(s_i, t_i)$. The $ANSC$ problem is defined as follows. Compute the length of the shortest simple cycle that contains v , for every $v \in V$.

A straightforward approach for solving the n -PSP problem is to first construct a distance oracle and then query it for every pair in the input set. The total runtime of this approach is the sum of the construction time and n times the query time of the distance oracle. While this method naturally yields an n -PSP algorithm, a distance oracle must satisfy additional requirements that an n -PSP algorithm does not. In particular, a distance oracle must use limited space and support queries between any of the $\Omega(n^2)$ pairs of vertices.

Using our new distance oracles, we obtain improved time/stretch tradeoffs for both the n -PSP problem and the $ANSC$ -problem, improving some of the results of [22, 20].

In [22], a lower bound based on the combinatorial $4k$ clique hypothesis for the n -PSP problem is presented. They showed that $\Omega(m^{2-\frac{2}{k+1}} \cdot n^{\frac{1}{k+1}-o(1)})$ time is required to achieve $(1 + 1/k - \epsilon)$ stretch. As mentioned in [22] using the $(1 + 1/k)$ -distance oracle of [6] it is possible to solve n -PSP with $(1 + 1/k)$ -stretch in $O(m^{2-\frac{2}{k+1}} n^{\frac{1}{k+1}})$ time, which almost matches the lower bound of [22]. In this paper, we show that it is possible to significantly improve the running time of [6] and to bypass the lower bound of [22] at the cost of a small additive error (of up to 2). We obtain an $O(m^{1-\frac{1}{k+1}} n)$ time algorithm that returns $\hat{d}(s_i, t_i) \leq d(s_i, t_i) + 2 \left\lceil \frac{d(s_i, t_i)}{2k} \right\rceil \leq (1 + 1/k)d(u, v) + 2$, as presented in the following theorem.

► **Theorem 11.** *Let $k \geq 2$. Given an unweighted graph G and vertex pairs (s_i, t_i) for $1 \leq i \leq O(n)$, there is an algorithm for n -PSP that computes an estimation $\hat{d}(s_i, t_i)$ such that $d(s_i, t_i) \leq \hat{d}(s_i, t_i) \leq d(s_i, t_i) + 2 \left\lceil \frac{d(s_i, t_i)}{2k} \right\rceil$ in $\tilde{O}(m^{1-\frac{1}{k+1}} n)$ time.*

In [22] they presented an $O(m + n^{2/k})$ time n -PSP algorithm that returns $(2k-2)(2k-1)$ -stretch. In the following theorem we improve the stretch from $(2k-1)(2k-2)$ to $(2k-1)(2k-3)$ while using the same running time.

► **Theorem 12.** *Let $k \geq 3$. Given a weighted graph G and vertex pairs (s_i, t_i) for $1 \leq i \leq O(n)$, there is an algorithm for $n - \text{PSP}$ that computes an estimation $\hat{d}(s_i, t_i)$ such that $d(s_i, t_i) \leq \hat{d}(s_i, t_i) \leq (2k - 1)(2k - 3)$ in $\tilde{O}(m + n^{2/k})$ time.*

For the *ANSC* problem, [22] presented an $O(m^{2-2/k}n^{1/k})$ running time algorithm that for every $u \in V$ returns estimate $\hat{c}(u)$ such that $\hat{c}_u \leq SC(u) + 2 \left\lceil \frac{SC(u)}{2(k-1)} \right\rceil$, where $SC(u)$ is the length of the shortest simple cycle that contains u . In the following theorem, we improve the running time from $O(m^{2-2/k}n^{1/k})$ of [22] to $\tilde{O}(mn^{1-\frac{1}{k}})$.

► **Theorem 13.** *Given an undirected unweighted graph G , let k be a positive integer. There is a randomized algorithm for *ANSC* that computes for every $u \in V$ an estimation $\hat{c}_u$ such that $SC(u) \leq \hat{c}_u \leq 1 + 2 \left\lceil \frac{SC(u)}{2(k-1)} \right\rceil$ in $\tilde{O}(mn^{1-\frac{1}{k}})$ -time.*

In addition, we present the first, to the best of our knowledge, stretch < 2 algorithm for the *ANSC* problem in *weighted graphs*. Specifically, we present an $O(m^{2-1/k})$ time algorithm for weighted graphs that for every $u \in V$ returns an estimate $\hat{c}$ such that $\hat{c}_u \leq 1 + \frac{1}{k-1}SC(u)$, as presented in the following theorem.

► **Theorem 14.** *Given an undirected weighted graph G , let k be a positive integer. There is a randomized algorithm for *ANSC* that computes for every $u \in V$ an estimation $\hat{c}_u$ such that $SC(u) \leq \hat{c}_u \leq (1 + \frac{1}{k-1})SC(u)$ in $\tilde{O}(m^{2-\frac{1}{k}})$ -time.*

Our results for the *ANSC* problem are summarized in Table 4.

The rest of this paper is organized as follows. In the next section, we present some necessary preliminaries. In Section 3 we present a technical overview of our main techniques and some of our new distance oracles. In the full version [27] of the paper, we present the full proofs of our techniques, distance oracles, applications, and lower bound.

2 Preliminaries

In this section, we present several definitions and existing tools that we use to develop our new techniques and ideas which we then apply in order to obtain new distance oracles. Let $G = (V, E)$ be an undirected graph with $n = |V|$ vertices and $m = |E|$ edges. Throughout the paper, we consider both unweighted graphs and weighted graphs with non-negative real edge weights.

Let $u, v \in V$. The distance $d(u, v)$ between u and v is the length of a shortest path between u and v . Let $P(u, v)$ be the shortest path between u and v . Let $N(u)$ be the vertices that are neighbors of u and let $\deg(u) = |N(u)|$ be the degree of u . Let $X \subseteq V$. Let $N(X)$ be the vertices that are neighbors of some vertex $u \in X$, i.e., $N(X) = \{w \in V \mid \exists x \in X : (w, x) \in E\}$. The distance $d(u, X)$ between u and X is the distance between u and the closest vertex to u from X , that is, $d(u, X) = \min_{x \in X} (d(u, x))$. Let $p(u, X) = \arg \min_{x \in X} (d(u, x))$ (ties are broken in favor of the vertex with a smaller identifier).

Next, we define bunches and clusters as in [36]. Let $u \in V$ and let $X, Y \subseteq V$. Let $B(u, X, Y) = \{v \in X \mid d(u, v) < d(u, Y)\}$ be the bunch of u with respect to X and Y . Let $C(u, Y) = \{v \in V \mid d(u, v) < d(v, Y)\}$ be the cluster of u with respect to Y .

The starting point in many algorithms and data structures for distance approximation, and in particular in Thorup and Zwick[36]’s distance oracles, is a hierarchy of vertex sets $A_0, A_1, \dots, A_k$, where $A_0 = V$, $A_k = \emptyset$ and for $0 \leq i \leq k - 1$, $A_{i+1} \subseteq A_i$ and $|A_i| = n^{1-i/k}$. For every $u \in V$, let $p_i(u) = p(u, A_i)$ and let $h_i(u) = d(u, A_i)$. Using this hierarchy, Thorup and Zwick defined k bunches for every vertex $u \in V$ as follows: For every $0 \leq i \leq k - 1$, the bunch $B_i(u)$ is defined as $B_i(u) = B(u, A_i, A_{i+1})$. They also defined a cluster for every

■ **Algorithm 1** $\text{Intersection}(u, w, U, W)$.

```

1 if  $U \cap W \neq \emptyset$  then
2    $\text{return } \min_{x \in U \cap W} d'(u, x) + d'(x, w);$ 
3 return  $\infty$ ;
```

vertex $w \in A_i \setminus A_{i+1}$ as follows: $C(w) = C(w, A_{i+1})$. Throughout the paper when we save $B_i(u)$, for every $u \in V$ and $0 \leq i \leq k-1$ (or $C(u) = C(u, A_{i+1})$, for every $u \in A_i \setminus A_{i+1}$), we mean that we can check whether $x \in B_i(u)$ (or $x \in C(u)$) in constant time and if $x \in B_i(u)$ (or $x \in C(u)$) we can retrieve $d(u, x)$ in constant time as well.

The simplest instance of this hierarchy is when $k = 2$ and the hierarchy is composed of the sets A_0, A_1, A_2 , where $A_0 = V$, $A_1 \subset V$, and $A_2 = \emptyset$. Throughout this paper, in such a case, we will omit subscripts and will refer to A_1 simply as A . Moreover, since in this case $B_0(u) = B(u, V, A)$ and $B_1(u) = B(u, A, \emptyset) = A$, for every $u \in V$, we use $B(u)$ to denote $B_0(u)$ and A to denote $B_1(u)$ and since we have only $p_0(u)$ and $p_1(u)$, where $p_0(u)$ is u , we use $p(u)$ to denote $p_1(u)$ and $h(u)$ to denote $h_1(u)$.

Thorup and Zwick [36] construct the sets $A_1, \dots, A_{k-1}$, by adding to A_{i+1} , where $i \in [0, k-2]$, every vertex of A_i , independently at random with probability p . Constructing the sets in such a way allows Thorup and Zwick [36] to prove the following:

► **Lemma 15** ([36]). *Given an integer parameter $k \geq 2$, we can compute in $\tilde{O}(n)$ time sets $A_1, \dots, A_{k-1}$, such that $|A_i| = O(n^{1-i/k})$ for every $i \in [1, k-1]$ with high probability, and for every $i \in [0, k-1]$ the size of $B_i(u)$ is $O(n^{1/k})$, with high probability (w.h.p). The cost of computing $B_i(u)$, for every $u \in V$, is $\tilde{O}(mn^{1/k})$ expected time.*

► **Lemma 16** ([35]). *Given a parameter p , we can compute a set A of size $\tilde{O}(np)$ in $\tilde{O}(mp^{-1})$ expected time such that, $|C(w, A)| = O(1/p)$, for every vertex $w \in V \setminus A$, and $|B(v, V, A)| = O(1/p)$ for every $v \in V$.*

We remark that both Lemma 15 and Lemma 16 have also slower deterministic constructions [34, 35].

Next, we present procedure **Intersection**. The input to **Intersection** is two vertices $u, w \in V$ and two sets $U, W \subseteq V$, such that, $u \in U$ and $w \in W$. For every $u' \in U$ and every $w' \in W$, we assume that distance estimations $d'(u, u')$ and $d'(w, w')$ are known. The output of **Intersection** is $\min_{x \in U \cap W} d'(u, x) + d'(w, x)$, if $U \cap W \neq \emptyset$ and ∞ , otherwise. (See Algorithm 1.) In most of our uses of **Intersection** we have $d'(u, u') = d(u, u')$ and $d'(w, w') = d(w, w')$. When this is not the case, we explicitly describe d' and its relation to d . The following lemma regarding the running time of **Intersection** is straightforward.

► **Lemma 17.** *$\text{Intersection}(u, w, U, W)$ runs in $O(\min(|U|, |W|))$.*

Proof. Assume, without loss of generality (wlog), that $|U| \geq |W|$. By checking for every vertex $y \in W$ in $O(1)$ time whether $y \in U$ we compute $U \cap W$. For each vertex $x \in U \cap W$, we compute in $O(1)$ time the value $d'(u, x) + d'(x, w)$. Thus, the total running time is $\min(|U|, |W|) \cdot O(1) = O(\min(|U|, |W|))$, as required. ◀

The following property of $\text{Intersection}(u, w, U, W)$ is the main feature of **Intersection**, and will be used throughout the paper.

► **Property 18.** *Let $P = P(u, w)$. If $P \cap (U \cap W) \neq \emptyset$, and there exists $x \in P \cap (U \cap W)$ such that $d'(u, x) = d(u, x)$ and $d'(x, w) = d(x, w)$ then $\text{Intersection}(u, w, U, W)$ returns $d(u, w)$.*

Algorithm 2 $\text{TZQuery}(u, v)$.

```

1 for  $i \in [k]$  do
2   if  $p_{i-1}(u) \in B_i(v)$  then return  $h_{i-1}(u) + d(p_{i-1}(u), v)$ ;
3    $(u, v) \leftarrow (v, u)$ 

```

Let TZQuery be the query of the distance oracle presented by Thorup and Zwick [36]. We let $\text{MTZQuery}(u, v)$ be $\min(\text{TZQuery}(u, v), \text{TZQuery}(v, u))$. Pseudocode for TZQuery exists in Algorithm 2.

The following lemma appears explicitly in [36] regarding the correctness and running time of MTZQuery .

► **Lemma 19** ([36]). *The running time of MTZQuery is $O(k)$ and it holds that*

$$\text{MTZQuery}(u, v) \leq (2k - 1)d(u, v)$$

The next lemma follows from [36], and we prove it here for completeness.

► **Lemma 20.** *Let $\hat{d}(u, v)$ be $\text{MTZQuery}(u, v)$. For every integer $1 \leq i \leq k$, one of the following holds.*

- $\min(h_i(u), h_i(v)) \leq \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$
- $\hat{d}(u, v) \leq 2 \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$

Proof. Wlog, assume that $h_{i-1}(u) \leq h_{i-1}(v)$. We divide the proof into two cases. The case that $p_{i-1}(u) \in B_i(v)$ and the case that $p_{i-1}(u) \notin B_i(v)$. Consider the case that $p_{i-1}(u) \in B_i(v)$. In this case the value of $d(p_{i-1}(u), v)$ is saved in the distance oracle. From the triangle inequality it follows that $d(p_{i-1}(u), v) \leq h_{i-1}(u) + d(u, v)$. Therefore, we have that $\hat{d}(u, v) \leq h_{i-1}(u) + d(p_{i-1}(u), v) \leq 2h_{i-1}(u) + d(u, v)$. Since $h_{i-1}(u) \leq h_{i-1}(v)$ we get that $\hat{d}(u, v) \leq 2 \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$, as required.

Consider now the case that $p_{i-1}(u) \notin B_i(v)$. From the definition of $B_i(v)$ it follows that $h_i(v) \leq d(v, p_{i-1}(u)) \leq d(u, v) + h_{i-1}(u)$. Since $h_{i-1}(u) \leq h_{i-1}(v)$, we get that $h_i(v) \leq \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$, and thus $\min(h_i(u), h_i(v)) \leq \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$, as required. ◀

The following property follows by applying Lemma 20 inductively.

► **Property 21.** *Let $1 \leq i \leq k$ be an integer, and let $1 \leq j < i$. Then either:*

- $\min(h_i(u), h_i(v)) \leq \min(h_{i-1}(u), h_{i-1}(v)) + d(u, v)$
- $\hat{d}(u, v) \leq 2 \min(h_j(u), h_j(v)) + (i - j)d(u, v)$

We remark that all our distance oracles return an estimation that is the length of a path in G and therefore $d(u, v) \leq \hat{d}(u, v)$.

3 Technical overview

In the classic distance oracle of Thorup and Zwick [36], it is possible to bound $h_i(u)$ with $d(u, v) + h_{i-1}(v)$, for every $1 \leq i \leq k - 1$. The key to improving the $(2k - 1)$ -stretch of the distance oracle lies in tightening the bound on $h_i(u)$. Previous work (see, for example, [9, 32, 5, 8, 6]) improved the bound on $h_1(u)$ from $d(u, v)$ to $d(u, v)/2$ by exploiting the following structural property of $B_0(\cdot)$: if $P(u, v) \not\subseteq B_0(u) \cup B_0(v)$, then $h_1(u) + h_1(v) \leq d(u, v) + 1_{\text{ODD}}$.

In this paper, we develop two new techniques that exploit structural properties of $C(\cdot)$ to construct a distance oracle with improved stretch guarantees. These techniques enable us to bound not only $h_1(u)$ by $d(u, v)/2$, but also $h_2(u)$ by $d(u, v)$, thereby improving overall stretch. We begin by presenting the borderline vertices technique, which applies to both weighted and unweighted graphs.

3.1 The borderline vertices technique

Let $C^*(u) = C(u) \cup N(C(u))$ be an augmented cluster. Roughly speaking, using the borderline vertices technique we show that if $P(u, v) \not\subseteq C^*(u, A_c) \cap C^*(v, A_c)$ then $\max(h_{c+1}(u), h_{c+1}(v)) \leq d(u, v)$. In particular, by setting $c = 1$ we get an improvement over the result of [9], showing not only that $h_1(u) \leq d(u, v)/2$ but also that $h_2(u) \leq d(u, v)$.

Let $P = P(u, v)$ be a shortest path between u and v . The borderline vertex $\tau_u(P)$ of $C^*(u)$ in P is the farthest vertex from u in $C^*(u) \cap P$. Similarly, the borderline vertex $\tau_v(P)$ of $C^*(v)$ in P is the farthest vertex from v in $C^*(v) \cap P$. Obviously, a distance oracle cannot store $\tau_u(P)$ for every shortest path P , as this would require $\Theta(n^2)$ space. We overcome this limitation by using $\tilde{O}(\mu \cdot |C(u)|)$ query time to iterate over all vertices in the augmented cluster $C^*(u)$, and in particular $\tau_u = \tau_u(P)$.

To obtain our improved bound on $h_2(v)$, we analyze two different cases regarding $p_1(\tau_u)$. The case that $p_1(\tau_u) \in B_1(v)$ and the case that $p_1(\tau_u) \notin B_1(v)$. If $p_1(\tau_u) \in B_1(v)$, then the query algorithm can return as an estimation $\hat{d}(u, v) = u \rightarrow \tau_u \rightarrow p_1(\tau_u) \rightarrow v$, where $x \rightarrow y \rightarrow z \rightarrow w$ is $d(x, y) + d(y, z) + d(z, w)$. In this case, since $\tau_u \notin C(u, A_1)$ (as it is the farthest vertex from u in $C^*(u)$) we get that $h_1(\tau_u) \leq d(u, \tau_u) \leq d(u, v)$, and therefore $\hat{d}(u, v) \leq 3d(u, v)$.

Otherwise, if $p_1(\tau_u) \notin B_1(v)$ then $h_2(v) \leq d(v, p_1(\tau_u)) \leq d(v, \tau_u) + h_1(\tau_u)$. Since $h_1(\tau_u) \leq d(u, \tau_u)$ we get that $h_2(v) \leq d(v, \tau_u) + d(u, \tau_u) = d(u, v)$, as wanted. (See Figure 1). Using symmetric arguments for τ_v , we can also get that $h_2(u) \leq d(u, v)$.

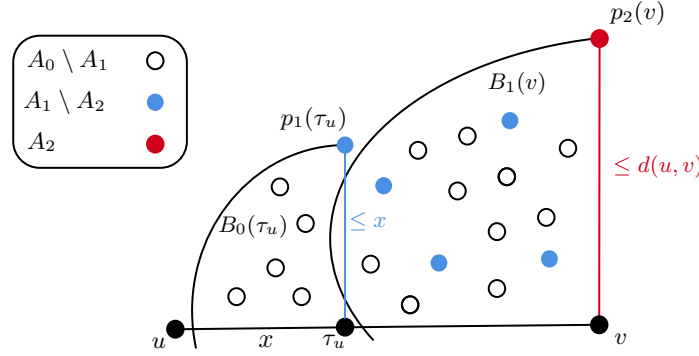
In unweighted graphs, we can avoid the μ factor from the query time at the cost of bounding $h_2(u)$ and $h_2(v)$ by $d(u, v) + 2$ rather than $d(u, v)$. The difference is that we let $\tau'_u(P)$ be defined as the farthest vertex from u in $C(u) \cap P$ (unlike $\tau_u(P)$ which considers $C^*(u) \cap P$). Next, we provide an overview of our main three-way tradeoff that uses the borderline vertices technique.

3.2 $(2k - 1 - 4c)$ -stretch distance oracle

Let $0 < c < k/2 - 1$ be an integer, let $V = A_0, A_c, A_{c+1}, \dots, A_k = \emptyset$, such that $|A_i| = n^{1-i/k}$. The storage of the distance oracle saves: the graph G , $d(s_1, s_2)$, for every $s_1 \in A_{c+1}, s_2 \in A_{k-c-2}$ and $B(u) = \cup_{i \in \{0 \cup [c, k]\}} B_i(u)$ for every $u \in V$. Where $B_0(u) = B(u, V, A_c)$.

The query works as follows. First, we set $\hat{d}(u, v)$ to $\text{Intersection}(u, v, C^*(u, A_c), C^*(v, A_c))$. Then, we iterate over every $u' \in C^*(u, A_c)$ (to iterate over τ_u), and update $\hat{d}(u, v)$ to $\min(\hat{d}(u, v), d(u, u') + \text{MTZQuery}(u', v))$. Similarly, for every $v' \in C^*(v, A_c)$ we update $\hat{d}(u, v)$ to $\min(\hat{d}(u, v), d(v, v') + \text{MTZQuery}(v', u))$. Finally, the algorithm returns $\min(\hat{d}(u, v), u \rightarrow p_{c+1}(u) \rightarrow p_{k-c-2}(v) \rightarrow v, u \rightarrow p_{k-c-2}(u) \rightarrow p_{c+1}(v) \rightarrow v)$.

It is straightforward to see that the space is $O(m + n^{1+1/k})$ and that the query takes $O(\mu|C(u, A_c)|) = O(\mu n^{c/k})$ time. The main technical contribution is in proving that $\hat{d}(u, v) \leq (2k - 4c - 1) \cdot d(u, v)$. To prove it, we use the borderline vertices technique. Since that $\tau_u \in C^*(u, A_c)$, when the algorithm iterates over $C^*(u, A_c)$, there is an iteration in which $u' = \tau_u$. In this iteration we guarantee that $\hat{d}(u, v) \leq d(u, \tau_u) + \text{MTZQuery}(\tau_u, v)$. Since τ_u is a borderline vertex, we either have that $\hat{d}(u, v) \leq u \rightarrow \tau_u \rightarrow p_c(\tau_u) \rightarrow v \leq 3d(u, v)$, and the



■ **Figure 1** $p_1(\tau_u) \notin B_1(v)$.

estimation guarantee holds, or that $h_{c+1}(v) \leq d(u, v)$. Similarly, since $\tau_v \in C^*(v, A_c)$ and we iterate over $C^*(u, A_c)$, there is an iteration in which $v' = \tau_v$. In this iteration, either a short path is found ($v \rightarrow \tau_v \rightarrow p_c(\tau_u) \rightarrow u$), or we have that $h_{c+1}(u) \leq d(u, v)$.

From the properties of **MTZQuery**, for every $c + 1 < i < k$, either a short path is found or $h_i(u) \leq d(u, v) + h_{i-1}(v)$. Since $h_{c+1}(u) \leq d(u, v)$ and $h_{c+1}(v) \leq d(u, v)$, it follows that

$$h_{k-c-2}(u) \leq \max(h_{c+1}(u), h_{c+1}(v)) + (k - 2c - 3)d(u, v) \leq (k - 2c - 2)d(u, v).$$

In the query procedure, we have that $\hat{d}(u, v) \leq u \rightarrow p_{k-c-2}(u) \rightarrow p_{c+1}(v) \rightarrow v \leq 2h_{k-c-2}(u) + d(u, v) + 2h_{c+1}(v)$. Thus:

$$\hat{d}(u, v) \leq 2(k - 2c - 2)d(u, v) + d(u, v) + 2d(u, v) = (2k - 4c - 1)d(u, v),$$

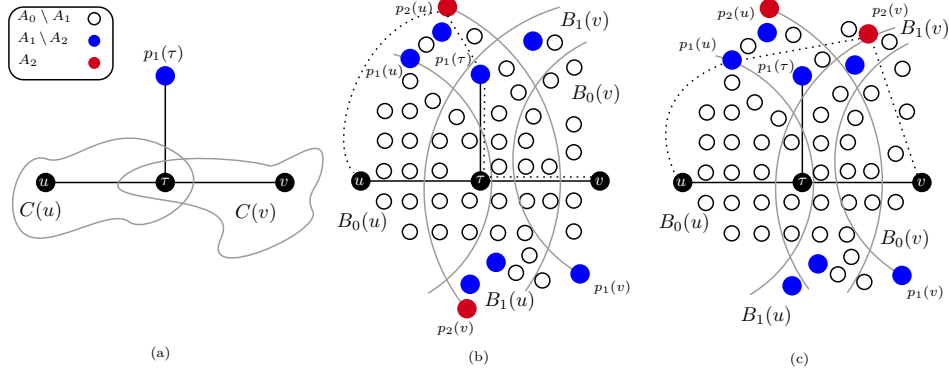
as required.

3.3 The middle vertex technique

For unweighted graphs, we develop the middle vertex technique that allows us to bound either $h_2(u)$ or $h_2(v)$ by $d(u, v)/2 + 1_{\text{ODD}}$, improving the previous $3d(u, v)/2 + 1_{\text{ODD}}$ bound from [22]. Roughly speaking, using the middle vertex technique we show that if $P(u, v) \not\subseteq C(u, A_1) \cap C(v, A_1)$ then $\min(h_2(u), h_2(v)) \leq d(u, v) + 1_{\text{ODD}}$. The middle vertex technique requires $O(n^{1/k})$ query time, matching the query time of the unweighted borderline technique. However, by using the middle vertex technique we reduce the additive error from 2 to 1_{ODD} in the bound of $\min(h_2(u), h_2(v))$.

For the sake of simplicity, let A_0, A_1, A_2, A_3 be a vertex hierarchy, where $A_0 = V$, $A_3 = \emptyset$ and let $d(u, v) = \delta$ be even. Let τ be the middle vertex on $P(u, v)$. That is, $d(u, \tau) = d(\tau, v) = \delta/2$ ($\delta/2$ is an integer since δ is even). The middle vertex τ has the following useful property. If $h_1(\tau) > \delta/2$ then $u, v \in B_0(\tau)$. The problem is that τ is unknown. However, the case that $u, v \in B_0(\tau)$ is equivalent to the case that $\tau \in C(u) \cap C(v)$. We use the $O(n^{1/k})$ query time to compute for every $w \in C(u) \cap C(v)$ the value of $d(u, w) + d(w, v)$. Therefore, even without knowing τ , in this case we have that $\hat{d}(u, v) = \delta$. (See Figure 2(a).)

If $h_1(\tau) \leq \delta/2$ then $d(u, p_1(\tau)) \leq d(u, \tau) + h_1(\tau) \leq \delta$ and $d(v, p_1(\tau)) \leq \delta$. If $p_1(\tau) \in B_1(u) \cap B_1(v)$, we exploit, again, the $O(n^{1/k})$ query time to compute for every $w \in B_1(u) \cap B_1(v)$ the value $d(u, w) + d(w, v)$. Therefore, even without knowing $p_1(\tau)$ the query algorithm returns $\hat{d}(u, v) \leq d(u, p_1(\tau)) + d(v, p_1(\tau)) \leq 2\delta$. (See Figure 2(b).) If $p_1(\tau) \notin B_1(u) \cap B_1(v)$ then without loss of generality $p_1(\tau) \notin B_1(u)$ and therefore $h_2(u) \leq d(u, p_1(\tau)) \leq \delta$. (See Figure 2(c).)



■ **Figure 2** (a) $\tau \in C(u) \cap C(v)$. (b) $p_1(\tau) \in B_1(u) \cap B_1(v)$. (c) $p_1(\tau) \notin B_1(u) \cap B_1(v)$. In the figure $p_1(\tau) \notin B_1(v)$.

Our distance oracles for unweighted graphs use the middle vertex technique. When combined with the unweighted borderline vertices technique, this enables the construction of two additional oracles.

3.4 Distance oracles with stretch < 2

Let $A \subseteq V$ be a set of size $O(n^{1-c})$, let $u, v \in V$, let $\delta = d(u, v)$ and let $P = P(u, v)$. Let $t \geq 0$ be an integer. Let $S_t(u) = \bigcup_{w \in S_{t-1}(u)} B(w)$, where $S_0(u) = \{u\}$. Let d_t be a distance function induced by S_t (for the exact definition of d_t , see Figure 3). Let $u = u_0$ (resp. $v = v_0$). For every $1 \leq i \leq t$, let $u_i \in B(u_{i-1})$ (resp. $v_i \in B(v_{i-1})$) be the farthest vertex in P from u (resp. v). (See Figure 3 for an illustration.) Let $P_t(u) = P(u, u_t)$. By the definition of $S_t(u)$ we have that $P_t(u) \subseteq S_t(u)$. We prove the following two properties:

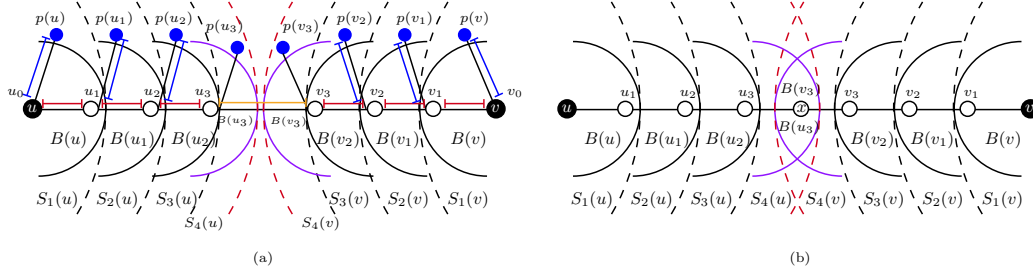
- If $P_t(u) \cap P_t(v) \neq \emptyset$ then $d_t(u, w) + d_t(v, w) = \delta$ for some $w \in P_t(u) \cap P_t(v)$. (See Figure 3(b).)
- If $P_t(u) \cap P_t(v) = \emptyset$ then $\sum_{i=0}^{t-1} h(u_i) + h(v_i) \leq \delta + 2t - 1$. (See Figure 3(a).)

Next, we overview the distance oracle that uses these properties. The distance oracle stores the graph G and $d(x, y)$, for every $\langle x, y \rangle \in A \times V$, at the cost of $O(|V| \cdot |A|) = O(m + n^{2-c})$ space. In the query of the distance oracle, we compute $S_t(u)$ and $S_t(v)$. To address the case that $P_t(u) \cap P_t(v) \neq \emptyset$ we compute for every $w \in S_t(u) \cap S_t(v)$ in $O(\min(|S_t(u)|, |S_t(v)|))$ time the value $d_t(u, w) + d_t(v, w)$. In such a case, from the first property we have $d_t(u, w) + d_t(v, w) = \delta$, for some w .

To address the case that $P_t(u) \cap P_t(v) = \emptyset$ we compute $d(u, p(w)) + d(p(w), v)$ for every $w \in S_t(u) \cup S_t(v)$. Let $w \in \bigcup_{i=0}^{t-1} \{u_i, v_i\}$ be the vertex with minimal $h(w)$ value. Using the bound $\sum_{i=0}^{t-1} h(u_i) + h(v_i) \leq d(u, v) + 2t - 1$, we can show that $h(w) \leq \lfloor (d(u, v) + 2t - 1)/2t \rfloor = \lceil d(u, v)/2t \rceil$ and get that $d(u, p(w)) + d(p(w), v) \leq \delta + 2\lceil \delta/2t \rceil$.

Next, we reduce the space from $O(n^{2-c})$ to $O(n^{2-2c})$ by saving $d(x, y)$, for every $\langle x, y \rangle \in A \times A$ instead of saving $d(x, y)$, for every $\langle x, y \rangle \in A \times V$.

This information prevents us from using paths of the form $u \rightsquigarrow p(w) \rightsquigarrow v$. We can still use paths of the form $u \rightsquigarrow u_i \rightsquigarrow p(u_i) \rightsquigarrow p(v_j) \rightsquigarrow v_j \rightsquigarrow v$. However, since we do not know the vertices u_i and v_j we need to consider all vertex pairs in $S_{t-1}(u) \times S_{t-1}(v)$. This increases the query time from $O(n^{tc})$ to $O(n^{2(t-1)c})$. To avoid $O(n^{2(t-1)c})$ query time, we show that it suffices to consider only vertex pairs in $S_i(u) \times S_{t-1-i}(v)$, where $0 \leq i \leq t-1$ in $O(tn^{(t-1)c})$ -time, and still get the same approximation without iterating over all vertex pairs in $S_{t-1}(u) \times S_{t-1}(v)$.



■ **Figure 3** (a) $P_4(u) \cap P_4(v) = \emptyset$. The blue interval between u_i (v_i) and $p(u_i)$ ($p(v_i)$), where $i \in \{0, 1, 2\}$, is bounded by the red interval between u_i (v_i) and u_{i+1} (v_{i+1}) plus 1. The orange interval plus 1 bounds $h(u_3) + h(v_3)$. (b) $P_4(u) \cap P_4(v) = \{x\}$.

We define the set $Q = \{\langle u_i, v_{t-i-1} \rangle \mid 0 \leq i \leq t-1\}$. Let $q = \langle q_u, q_v \rangle \in Q$ be the vertex pair with minimal $h(q_u) + h(q_v)$ value. We prove that $h(q_u) + h(q_v) \leq \lfloor (d(u, v) + 2t - 1)/t \rfloor = \lceil d(u, v)/t \rceil + 1$. While iterating over the pairs of $S_i(u) \times S_{t-1-i}(v)$, we encounter q , and get that $\hat{d}(u, v) \leq \delta + 2\lceil \delta/t \rceil + 2$. Thus, with $O(n^{2-2c})$ space and $O(n^{tc})$ query time we get an estimation $\hat{d}(u, v) \leq \delta + 2\lceil \delta/t \rceil + 2$. By setting $c = 1/3 - \varepsilon$ and $t = 2, 3$ we get two new distance oracles, one with $O(n^{4/3+2\varepsilon})$ space and $O(n^{2/3-2\varepsilon})$ query, and estimation $\hat{d}(u, v) \leq \delta + 2\lceil \delta/2 \rceil + 2 \leq 2\delta + 4$, and another with $O(n^{4/3+2\varepsilon})$ space and $O(n^{1-3\varepsilon})$ query, and estimation $\hat{d}(u, v) \leq \delta + 2\lceil \delta/3 \rceil + 2 \leq \frac{5}{3}\delta + 4$.

We also consider weighted graphs with non-negative real edge weights. Agarwal, Godfrey and Har-Peled [7] defined $B^*(u)$ to be $B(u) \cup N(B(u))$. We revise the definition of S_t to be $\bigcup_{w \in S_{t-1}(u)} B^*(w)$. This new definition allows us to extend the previous results to weighted graphs. The distance oracles obtained using this approach are presented in the full version[27].

We remark that the usage of the sets $S_t(u)$ and $S_t(v)$ in the query algorithm is similar to the usage of the graph H in the work of [14]. However, in our analysis of the unweighted case, we achieve a slightly better distance approximation. In addition, we introduce a new approach to obtain a distance estimation using two vertices from A that are close to $P(u, v)$ rather than a single vertex from A , as in [14].

References

- 1 Amir Abboud and Greg Bodwin. The $4/3$ additive spanner exponent is tight. *J. ACM*, 64(4):28:1–28:20, 2017. doi:10.1145/3088511.
- 2 Amir Abboud, Greg Bodwin, and Seth Pettie. A hierarchy of lower bounds for sublinear additive spanners. *SIAM J. Comput.*, 47(6):2203–2236, 2018. doi:10.1137/16M1105815.
- 3 Amir Abboud, Karl Bringmann, and Nick Fischer. Stronger 3-sum lower bounds for approximate distance oracles via additive combinatorics. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 391–404. ACM, 2023. doi:10.1145/3564246.3585240.
- 4 Amir Abboud, Karl Bringmann, Seri Khourey, and Or Zamir. Hardness of approximation in p via short cycle removal: cycle detection, distance oracles, and beyond. In Stefano Leonardi and Anupam Gupta, editors, *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pages 1487–1500. ACM, 2022. doi:10.1145/3519935.3520066.

- 5 Ittai Abraham and Cyril Gavoille. On approximate distance labels and routing schemes with affine stretch. In David Peleg, editor, *Distributed Computing - 25th International Symposium, DISC 2011, Rome, Italy, September 20-22, 2011. Proceedings*, volume 6950 of *Lecture Notes in Computer Science*, pages 404–415. Springer, 2011. doi:10.1007/978-3-642-24100-0_39.
- 6 Rachit Agarwal. The space-stretch-time tradeoff in distance oracles. In Andreas S. Schulz and Dorothea Wagner, editors, *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, volume 8737 of *Lecture Notes in Computer Science*, pages 49–60. Springer, 2014. doi:10.1007/978-3-662-44777-2_5.
- 7 Rachit Agarwal, Brighten Godfrey, and Sariel Har-Peled. Faster approximate distance queries and compact routing in sparse graphs. *CoRR*, abs/1201.2703, 2012. arXiv:1201.2703.
- 8 Rachit Agarwal and Philip Brighten Godfrey. Distance oracles for stretch less than 2. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 526–538. SIAM, 2013. doi:10.1137/1.9781611973105.38.
- 9 Rachit Agarwal, Philip Brighten Godfrey, and Sariel Har-Peled. Approximate distance queries and compact routing in sparse graphs. In *INFOCOM 2011. 30th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 10-15 April 2011, Shanghai, China*, pages 1754–1762. IEEE, 2011. doi:10.1109/INFCOM.2011.5934973.
- 10 Surender Baswana and Telikepalli Kavitha. Faster algorithms for all-pairs approximate shortest paths in undirected graphs. *SIAM J. Comput.*, 39(7):2865–2896, 2010. doi:10.1137/080737174.
- 11 Surender Baswana, Telikepalli Kavitha, Kurt Mehlhorn, and Seth Pettie. Additive spanners and (α, β) -spanners. *ACM Trans. Algorithms*, 7(1):5:1–5:26, 2010. doi:10.1145/1868237.1868242.
- 12 Davide Bilò, Shiri Chechik, Keerti Choudhary, Sarel Cohen, Tobias Friedrich, Simon Krogmann, and Martin Schirneck. Approximate distance sensitivity oracles in subquadratic space. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1396–1409. ACM, 2023. doi:10.1145/3564246.3585251.
- 13 Davide Bilò, Shiri Chechik, Keerti Choudhary, Sarel Cohen, Tobias Friedrich, and Martin Schirneck. Improved approximate distance oracles: Bypassing the thorup-zwick bound in dense graphs. *CoRR*, abs/2307.11677, 2023. doi:10.48550/arXiv.2307.11677.
- 14 Davide Bilò, Shiri Chechik, Keerti Choudhary, Sarel Cohen, Tobias Friedrich, and Martin Schirneck. Improved approximate distance oracles: Bypassing the thorup-zwick bound in dense graphs. *arXiv preprint arXiv:2307.11677*, 2023. doi:10.48550/arXiv.2307.11677.
- 15 Davide Bilò, Shiri Chechik, Keerti Choudhary, Sarel Cohen, Tobias Friedrich, and Martin Schirneck. Improved distance (sensitivity) oracles with subquadratic space. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1550–1558. IEEE, 2024. doi:10.1109/FOCS61266.2024.00097.
- 16 Greg Bodwin and Virginia Vassilevska Williams. Better distance preservers and additive spanners. *ACM Trans. Algorithms*, 17(4), October 2021. doi:10.1145/3490147.
- 17 Shiri Chechik. New additive spanners. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 498–512. SIAM, 2013. doi:10.1137/1.9781611973105.36.
- 18 Shiri Chechik. Approximate distance oracles with constant query time. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 654–663. ACM, 2014. doi:10.1145/2591796.2591801.

- 19 Shiri Chechik. Approximate distance oracles with improved bounds. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 1–10. ACM, 2015. doi:10.1145/2746539.2746562.
- 20 Shiri Chechik, Itay Hoch, and Gur Lifshitz. New approximation algorithms and reductions for n -pairs shortest paths and all-nodes shortest cycles. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 5207–5238. SIAM, 2025. doi:10.1137/1.9781611978322.177.
- 21 Shiri Chechik and Tianyi Zhang. Path-reporting distance oracles with logarithmic stretch and linear size. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 42:1–42:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ICALP.2024.42.
- 22 Mina Dalirrooyfard, Ce Jin, Virginia Vassilevska Williams, and Nicole Wein. Approximation algorithms and hardness for n -pairs shortest paths and all-nodes shortest cycles. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 290–300. IEEE, 2022. doi:10.1109/FOCS54457.2022.00034.
- 23 Michael Elkin, Ofer Neiman, and Christian Wulff-Nilsen. Space-efficient path-reporting approximate distance oracles. *Theor. Comput. Sci.*, 651:1–10, 2016. doi:10.1016/J.TCS.2016.07.038.
- 24 Michael Elkin and David Peleg. $(1+\epsilon, \beta)$ -spanner constructions for general graphs. *SIAM J. Comput.*, 33(3):608–631, 2004. doi:10.1137/S0097539701393384.
- 25 Michael Elkin and Seth Pettie. A linear-size logarithmic stretch path-reporting distance oracle for general graphs. *ACM Trans. Algorithms*, 12(4):50:1–50:31, 2016. doi:10.1145/2888397.
- 26 Michael Elkin and Idan Shabat. Path-reporting distance oracles with logarithmic stretch and size $o(n \log \log n)$. In *64th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2023, Santa Cruz, CA, USA, November 6-9, 2023*, pages 2278–2311. IEEE, 2023. doi:10.1109/FOCS57990.2023.00141.
- 27 Avi Kadia and Liam Roditty. New approximate distance oracles and their applications. *arXiv preprint arXiv:2509.00890*, 2025.
- 28 Tsvi Kopelowitz, Ariel Korin, and Liam Roditty. On the space usage of approximate distance oracles with sub-2 stretch. In Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson, editors, *51st International Colloquium on Automata, Languages, and Programming, ICALP 2024, July 8-12, 2024, Tallinn, Estonia*, volume 297 of *LIPIcs*, pages 101:1–101:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ICALP.2024.101.
- 29 Manor Mendel and Assaf Naor. Ramsey partitions and proximity data structures. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2006), 21-24 October 2006, Berkeley, California, USA, Proceedings*, pages 109–118. IEEE Computer Society, 2006. doi:10.1109/FOCS.2006.65.
- 30 Merav Parter. Bypassing erdős’ girth conjecture: Hybrid stretch and sourcewise spanners. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part II*, volume 8573 of *Lecture Notes in Computer Science*, pages 608–619. Springer, 2014. doi:10.1007/978-3-662-43951-7_49.
- 31 Ely Porat and Liam Roditty. Preprocess, set, query! *Algorithmica*, 67(4):516–528, 2013. doi:10.1007/S00453-013-9825-9.
- 32 Mihai Pătraşcu and Liam Roditty. Distance oracles beyond the thorup-zwick bound. *SIAM J. Comput.*, 43(1):300–311, 2014. doi:10.1137/11084128X.

- 33 Mihai Pătraşcu, Liam Roditty, and Mikkel Thorup. A new infinity of distance oracles for sparse graphs. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 738–747. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.44.
- 34 Liam Roditty, Mikkel Thorup, and Uri Zwick. Deterministic constructions of approximate distance oracles and spanners. In Luís Caires, Giuseppe F. Italiano, Luís Monteiro, Catuscia Palamidessi, and Moti Yung, editors, *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 261–272. Springer, 2005. doi:10.1007/11523468_22.
- 35 Mikkel Thorup and Uri Zwick. Compact routing schemes. In Arnold L. Rosenberg, editor, *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA 2001, Heraklion, Crete Island, Greece, July 4-6, 2001*, pages 1–10. ACM, 2001. doi:10.1145/378580.378581.
- 36 Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005. doi:10.1145/1044731.1044732.
- 37 Mikkel Thorup and Uri Zwick. Spanners and emulators with sublinear distance errors. In *Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2006, Miami, Florida, USA, January 22-26, 2006*, pages 802–809. ACM Press, 2006. URL: <http://dl.acm.org/citation.cfm?id=1109557.1109645>.
- 38 Christian Wulff-Nilsen. Approximate distance oracles with improved preprocessing time. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 202–208. SIAM, 2012. doi:10.1137/1.9781611973099.18.

A

 Tables of our distance oracles

A.1 Weighted graphs with stretch ≥ 2

■ **Table 2** Distance oracles in weighted graphs.

Construction	Space	Time	Stretch	Ref.	Comments
$\tilde{O}(mn^{1/k})$	$\tilde{O}(n^{1+1/k})$	$\tilde{O}(1)$	$2k - 1$	[36]	
$\tilde{O}(mn^{3/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^{c/k})$	$2k - 1 - 4c$	Th1	$1 \leq c < k/2$
$\tilde{O}(mn^{3/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^r)$	$2k(1 - 2r) - 1$	Th1	$c/k = r < 1/2$
$\tilde{O}(mn^{1/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^{1/k})$	$2k - 2$	[22]	$k \geq 3$
$\tilde{O}(mn^{1/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^{1/k})$	$2k - 3$	Th15[27]	$k \geq 3$
$\tilde{O}(mn^{2/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^{1/k})$	$2k - 4$	Th2	$k \geq 4$
$\tilde{O}(mn^{3/k})$	$\tilde{O}(m + n^{1+1/k})$	$\tilde{O}(\mu n^{1/k})$	$2k - 5$	Th1	$k \geq 5$
$\tilde{O}(mn^{2/3})$	$\tilde{O}(m + n^{4/3})$	$\tilde{O}(\mu n^{1/3})$	3	[7]	
$\tilde{O}(mn^{1/3})$	$\tilde{O}(m + n^{4/3})$	$\tilde{O}(\mu n^{1/3})$	3	Th15[27]	
$\tilde{O}(mn^{1/2})$	$\tilde{O}(m + n^{5/4})$	$\tilde{O}(\mu n^{1/4})$	4	Th2	
$\tilde{O}(mn^{3/5})$	$\tilde{O}(m + n^{6/5})$	$\tilde{O}(\mu n^{1/5})$	5	Th1	
$\tilde{O}(mn^{1-c})$	$\tilde{O}(m + n^{2-c})$	$\tilde{O}(\mu^t n^{tc})$	$1 + 1/t$	[6]	$c < 1/2$
$\tilde{O}(mn^{1-c})$	$\tilde{O}(m + n^{2-c})$	$\tilde{O}(\mu^t n^{(t+1)c})$	$1 + 1/(t + 0.5)$	[6]	$c < 1/2$
$\tilde{O}(mn^{1-c})$	$\tilde{O}(m + n^{2-2c})$	$\tilde{O}(\mu^t n^{tc})$	$1 + 2/t$	Th8	$c < 1/3$

A.2 n -PSP and $ANSC$ results

■ **Table 3** $nPSP$ in weighted graphs unless stated otherwise. $\delta = d(u, v)$.

Time	Approximation	Ref.	Comment
$\tilde{O}(m + n^{1+2/k})$	$(2k-1)(2k-2)\delta + 4k - 2$	[22]	Unweighted, $k \geq 2$
$\tilde{O}(m + n^{1+2/k})$	$(2k-1)(2k-3)\delta$	Theorem 12	$k \geq 2$
$\Omega(m^{2-\frac{2}{k+1}} n^{\frac{1}{k+1}-o(1)})$	$(1 + 1/k)\delta$	[22]	
$\tilde{O}(m^{2-\frac{2}{k+1}} n^{\frac{1}{k+1}})$	$(1 + 1/k)\delta$	[6] ⁸	
$\tilde{O}(m^{1-\frac{1}{k+1}} n)$	$\delta + 2\lceil \delta/2k \rceil$	Theorem 11	Unweighted, $k \geq 2$

■ **Table 4** $ANSC$ algorithm improvements in undirected graphs. $SC(u)$ is the shortest cycle for u .

Time	Approximation	Ref.	Comment
$\tilde{O}(m^{2-2/k} n^{1/k})$	$SC(u) + 2\lceil SC(u)/2(k-1) \rceil$	[22]	Unweighted, $k \geq 2$
$\tilde{O}(mn^{1-1/k})$	$SC(u) + 2\lceil SC(u)/2(k-1) \rceil$	Theorem 13	Unweighted, $k \geq 2$
$\tilde{O}(m^{2-\frac{1}{k}} n)$	$(1 + \frac{1}{k-1})SC(u)$	Theorem 14	Weighted, $k \geq 2$

⁸ This result is obtained by constructing the distance oracle of Agarwal [6] and querying it n times.