Improved Distance (Sensitivity) Oracles with Subquadratic Space

Davide Bilò
Dep. of Inf. Eng., CS and Mathematics
University of L'Aquila
davide.bilo@univaq.it

Shiri Chechik
Department of Computer Science
Tel Aviv University
shiri.chechik@gmail.com

Keerti Choudhary
Dep. of CS and Engineering
Indian Institute of Technology Delhi
keerti@iitd.ac.in

Sarel Cohen
School of Computer Science
The Academic College of Tel Aviv-Yaffo
sarelco@mta.ac.il

Tobias Friedrich
Hasso Plattner Institute
University of Potsdam
tobias.friedrich@hpi.de

Martin Schirneck
Faculty of Computer Science
University of Vienna
martin.schirneck@univie.ac.at

***Abstract*—A distance oracle (DO) for a graph G is a data structure that, when queried with vertices s, t, returns an estimate $\widehat{d}(s,t)$ of their distance in G. The oracle has stretch (α,β) if the estimate satisfies $d(s,t) \leqslant \widehat{d}(s,t) \leqslant \alpha \cdot d(s,t) + \beta$. An f-edge fault-tolerant distance sensitivity oracle (f-DSO) additionally receives a set F of up to f edges and estimates the distance in $G-F$.**

Our first contribution is the design of new distance oracles with subquadratic space for undirected graphs. We show that introducing a small additive stretch $\beta > 0$ allows one to make the multiplicative stretch α arbitrarily small. This sidesteps a known lower bound of $\alpha \geqslant 3$ (for $\beta = 0$ and subquadratic space) [Thorup & Zwick, JACM 2005]. We present a DO for graphs with edge weights in $[0, W]$ that, for any positive integer ℓ and any $c \in (0, \ell/2]$, has stretch $(1+\frac{1}{\ell}, 2W)$, space $\widetilde{O}(n^{2-\frac{c}{\ell}})$, and query time $O(n^c)$, generalizing results by Agarwal and Godfrey [SODA 2013] to arbitrarily dense graphs.

Our second contribution is a framework that turns an (α,β)-stretch DO for unweighted graphs into an $(\alpha(1+\varepsilon),\beta)$-stretch f-DSO with sensitivity $f = o(\log(n)/\log\log n)$ retaining subquadratic space. This generalizes a result by Bilò, Chechik, Choudhary, Cohen, Friedrich, Krogmann, and Schirneck [TheoretiCS 2024]. Combining the framework with our new DO gives an f-DSO that, for any $\gamma \in (0, (\ell+1)/2]$, has stretch $((1+\frac{1}{\ell})(1+\varepsilon), 2)$, space $n^{2-\frac{\gamma}{(t+1)(f+1)}+o(1)}/\varepsilon^{f+2}$, and query time $\widetilde{O}(n^\gamma/\varepsilon^2)$. This is the first f-DSO with subquadratic space, near-additive stretch, and sublinear query time.

The full version of this work can be found at https://arxiv.org/abs/2408.10014.

***Index Terms*—distance oracle, distance sensitivity oracle, fault tolerance, shortest paths, subquadratic space**

I. Introduction

A *distance oracle* (DO) is a data structure to retrieve exact or approximate distances between any pair of vertices s, t in an undirected graph $G = (V, E)$ upon query. The problem of designing distance oracles has attracted a lot of attention in recent years due to the wide applicability in domains like network routing, traffic engineering, and distributed computing. These oracles are used in settings where one cannot afford to store the entire graph, but still wants to be able to quickly query graph distances. A DO has *stretch* (α,β) if, for any pair s and t, the value $\widehat{d}(s,t)$ returned by the DO satisfies $d(s,t) \leqslant \widehat{d}(s,t) \leqslant \alpha \cdot d(s,t) + \beta$, where $d(s,t)$ denotes the exact distance between s and t in G. As networks in most real-life applications are prone to transient failures, researchers have also studied the problem of designing oracles that additionally tolerate multiple edge failures in G. An *f-edge fault-tolerant distance sensitivity oracle* (f-DSO) with stretch (α,β) is a data structure that, when queried on a triple (s,t,F), where $F \subseteq E$ has size at most f, outputs an estimate $\widehat{d}(s,t,F)$ of the distance $d(s,t,F)$ from s to t in $G-F$ such that $d(s,t,F) \leqslant \widehat{d}(s,t,F) \leqslant \alpha \cdot d(s,t,F) + \beta$.

Several DOs and f-DSOs with different size-stretch-time trade-offs have been developed in the last decades. See, for example, [1]–[13]. Our focus is on providing new distance oracles with a subquadratic space usage for both static and error-prone graphs. A special focus of this work is how static distance oracles can be converted into fault-tolerant distance *sensitivity* oracles.

A. Approximate Distance Oracles for Static Graphs

We first discuss distance oracles. Extensive research has been dedicated in that direction in the past two decades. In their seminal paper [12], Thorup and Zwick showed that, for any positive integer k (possibly depending on the input size), an undirected graph with n vertices and m edges can be preprocessed in time $O(mn^{1/k})$ to obtain an oracle with multiplicative stretch $2k-1$, space[1] $O(kn^{1+1/k})$, and query time $O(k)$. Subsequent works [14]–[16] further improved the space to $O(n^{1+1/k})$ and the query time to $O(1)$.

In the case of $k = 2$, this results in a 3-approximate oracle taking subquadratic space $O(n^{3/2})$. A simple information

This project received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program, grant agreement No. 803118 "The Power of Randomization in Uncertain Environments (UncertainENV)". The third author is partially supported by Google India Research Awards.

[1]The space of the data structures is measured in the number of machine words on $O(\log n)$ bits.

theoretic lower bound using bipartite graphs [12] shows that distance oracles with a purely multiplicative stretch *below* 3 require space that is at least quadratic in n. Pătraşcu and Roditty [10] were arguably the first to introduce an additive stretch to simultaneously reduce the space and the multiplicative stretch in general dense graphs. They proposed a distance oracle for unweighted graphs with stretch $(2,1)$ that takes $O(n^{5/3})$ space, has a a constant query time and can be constructed in time $O(mn^{2/3})$. They also showed that DOs with multiplicative stretch $\alpha \leqslant 2$ and constant query time require $\Omega(n^2)$ space, assuming a conjecture on the hardness of set intersection queries.

Agarwal and Godfrey [17] studied $(1+\varepsilon, O(1))$-stretch DOs for sparse graphs. However, when transferred to dense graphs the space of their construction becomes $\Omega(n^2)$ and the query time is $\Omega(n)$. To the best of our knowledge, no distance oracles have been constructed that simultaneously have subquadratic space and a multiplicative stretch better than 2 for general undirected graphs. If we want to reduce α while retaining low space, we necessarily have to introduce some additive stretch β and the query time must rise beyond constant. This raises the following natural question.

Question. *Is there a distance oracle with a $1+\varepsilon$ multiplicative and constant additive stretch that takes subquadratic space and has a query time that is sublinear in n?*

We provide an affirmative answer by presenting the first oracle with stretch $(1+\varepsilon, O(1))$ for general graphs. The construction is surprisingly simple, has subquadratic space, and sublinear query time, improving over Agarwal and Godfrey's work for dense graphs.

In the following theorem and throughout, the $\widetilde{O}$-notation omits polylogarithmic factors in the number of vertices n.

Theorem 1. *Let $W \geqslant 0$ be a real number, and G an undirected graph with n vertices and edge weights in a $\mathsf{poly}(n)$-sized subset of $[0, W]$. For every positive integer $K \leqslant \sqrt{n}$ and any $\varepsilon > 0$, there exists a path-reporting distance oracle for G that has stretch $(1+\varepsilon, 2W)$, space $\widetilde{O}(n^2/K)$, and query time $O(K^{\lceil 1/\varepsilon \rceil})$ for the distance and an additional $O(1)$ time per reported edge. The data structure can be constructed in APSP time.*

The restriction of the edge weights to a polynomial-sized subset of $[0, W]$ is to ensure that any graph distance can be encoded in a constant number of $O(\log n)$-bit words. We remark that our construction in fact guarantees a stretch of $(1+\varepsilon, 2w_{s,t})$ where $w_{s,t}$ is the maximum edge weight along a shortest path from s to t in G. The stretch thus depends locally on the queried vertices rather than the global edge weights.

One cannot reduce the additive stretch in Theorem 1 to 1 (if $\varepsilon < 2$) even in unweighted graphs as the unconditional lower bound for bipartite graphs [12] stated earlier rules out any data structure that can distinguish between distances 1 and 3 in subquadratic space.

By setting $\varepsilon = 1/\ell$ and $K^\ell = n^c$, we get the following trade-off between stretch space and time.

Corollary 2. *For a positive integer t and any $0 < c \leqslant \ell/2$, there exists a distance oracle for undirected graphs with stretch $(1+\frac{1}{\ell}, 2W)$, space $\widetilde{O}(n^{2-\frac{c}{\ell}})$, and query time $O(n^c)$.*

An extension of our construction allows to trade a higher stretch for a lower space. Namely, we present a family of DOs with multiplicative stretch of $2k-1+\varepsilon$ and $o(n^{1+1/k})$ space.

Theorem 3. *Let W be a non-negative real number, and G an undirected graph with n vertices and edge weights in a $\mathsf{poly}(n)$-sized subset of $[0, W]$. For all positive integers k and K with $K = O(n^{1/(2k+1)})$, and every $\varepsilon > 0$, there exits a distance oracle for G that has stretch $(2k-1+\varepsilon, 4kW)$, space $O((\frac{n}{K})^{1+1/k} \log^{1+1/k} n)$, and query time $O(K^{2\lceil 4k/\varepsilon \rceil})$. The data structure can be constructed in APSP time.*

For $k = 1$, the oracle in Theorem 3 has the same multiplicative stretch of $1+\varepsilon$ as the one in Theorem 1 and a better space of $\widetilde{O}(n^2/K^2)$, but the additive stretch of $4W$ is larger and so is the query time. We obtain the following corollary for general k, $\varepsilon = 1/\ell$, and $K^{8k\ell} = n^c$.

Corollary 4. *For all positive integers k and ℓ, and any $0 < c \leqslant (4-\frac{4}{2k+1})\,\ell$, there exists a distance oracle with stretch $(2k-1+\frac{1}{\ell}, 4kW)$, space $\widetilde{O}(n^{1+\frac{1}{k}(1-\frac{c}{8\ell})})$, and query time $O(n^c)$.*

Probably closest to the hierarchy in Theorem 3 is the distance labeling scheme of Abraham and Gavoille [18]. Seen as a DO for unweighted graphs, for any integer $k \geqslant 2$, it has a stretch of $(2k-2, 1)$ space $\widetilde{O}(n^{1+\frac{2}{2k-1}})$, and query time $O(k)$.

B. Distance Sensitivity Oracles

Most of the proposed distance sensitivity oracles that treat the sensitivity f (the number of tolerated edge failures) as a parameter require $\Omega(n^2)$ space, have a stretch depending on f, or an $\Omega(n)$ query time. See Section I-C for a detailed discussion. Bilò, Chechik, Choudhary, Cohen, Friedrich, Krogmann, and Schirneck [19] were the first to introduce an f-DSO with subquadratic space, a constant multiplicative stretch of $3+\varepsilon$ (for any f), and a query time that can be made an arbitrarily small polynomial. More precisely, for any unweighted graph G with unique shortest paths, every integer constant $f \geqslant 2$, any $0 < \gamma < 1/2$, and $\varepsilon > 0$, they devised an f-DSO for G with stretch $3+\varepsilon$, space $\widetilde{O}(n^{2-\frac{\gamma}{f+1}}) \cdot O(\log n/\varepsilon)^{f+2}$, query time $O(n^\gamma/\varepsilon^2)$, and preprocessing time $\widetilde{O}(mn^{2-\frac{\gamma}{f+1}}) \cdot O(\log n/\varepsilon)^{f+1}$. Even more than in the case of static distance oracles, a multiplicative stretch better than 3 (let alone close to 1) remains a barrier for subquadratic-space f-DSOs. We explore whether the introduction of a small additive stretch can help here as well.

Question. *Is there a distance sensitivity oracle for general sensitivity f with a $1+\varepsilon$ multiplicative stretch, possibly at the expense of a constant additive stretch, that takes subquadratic space and has a query time that is sublinear in n?*

Indeed, we devise an f-DSO with stretch $(1+\varepsilon, 2)$ that has subquadratic space and a small polynomial query time.

Theorem 5. *Let ℓ be a positive integer constant and G be an undirected and unweighted graph with n vertices and m edges and unique shortest paths. For any $0 < \gamma \leqslant (\ell+1)/2$, sensitivity $2 \leqslant f = o(\log(n)/\log\log n)$, and approximation parameter $\varepsilon = \omega(\sqrt{\log(n)}/n^{\frac{\gamma}{2(\ell+1)(f+1)}})$, there exists an f-edge fault-tolerant distance sensitivity oracle for G that has*

- *stretch* $((1+\frac{1}{\ell})(1+\varepsilon), 2)$,
- *space* $n^{2-\frac{\gamma}{(\ell+1)(f+1)}+o(1)}/\varepsilon^{f+2}$,
- *query time* $O(n^{\gamma}/\varepsilon^2)$,
- *preproc. time* $n^{2+\gamma+o(1)} + mn^{2-\frac{\gamma}{(\ell+1)(f+1)}+o(1)}/\varepsilon^{f+1}$.

Observe that the additive stretch of $\beta = 2$ is necessary due to the unconditional lower bound of $\Omega(n^2)$ on the size of every distance oracle with a purely multiplicative stretch of $\alpha < 3$, as discussed in Section I-A. The assumption of unique shortest paths can be achieved, for example, by perturbing the edge weights with random small values. This means that an unweighted graph G becomes weighted and its edge weights are very close to 1. As an alternative, we can compute, in time $O(mn + n^2 \log^2 n)$, a set of unique shortest paths via *lexicographic pertubation* [20].

Our main technique to obtaain the new distance sensitivity oracle is to develop a reduction that, given a path-reporting DO with stretch (α, β), constructs a $((1+\varepsilon)\alpha, \beta)$-stretch f-DSO. Crucially, the reduction results in a subquadratic-space f-DSO provided that the initial DO also takes only subquadratic space. Although the problem of designing static as well as fault-tolerant distance (sensitivity) oracles has been studied extensively in the past two decades, there has been no substantial progress to obtain black-box conversions from compact DOs to compact f-DSOs. The work by Bilò et al. [19] comes close, but their construction of the f-DSO is entangled with the inner workings of the DO of Thorup and Zwick [12]. Also, their analysis relies heavily on the input distance oracle having a purely multiplicative stretch of 3. In contrast, we develop algorithms that can work with any distance oracle as long as it is able to report an approximate shortest path that adheres to the stretch bound. Our analysis is able to incorporate any multiplicative stretch $\alpha \geqslant 1$ as well as additive stretch β that satisfies very mild technical assumptions. (See Theorem 7 for the precise statement.) Plugging in the distance oracle with stretch $(1+\frac{1}{\ell}, 2)$ from Corollary 2 then allows us to achieve the $((1+\frac{1}{\ell})(1+\varepsilon), 2)$-stretch f-DSO with subquadratic space and small polynomial query time from Theorem 5.

Emulating Searches in Fault-Tolerant Trees: Our transformation that makes distance oracles fault tolerant has two major steps. In the first one, we only treat *hop-short* replacement paths, these are shortest paths in $G-F$ whose number of edges is bounded by some cut-off parameter L. We transform any (α, β)-stretch distance oracle into an f-distance sensitivity oracle for hop-short paths (f-DSO$^{\leqslant L}$) with the same stretch. The second step then combines the solutions for hop-short paths into a general distance sensitivity oracle (f-DSO).

For the the first step, we develop *fault-tolerant tree oracles*, which are a new way to retrieve hop-short replacement paths from a previously known data structure called fault-tolerant trees (FT-trees), but without actually storing those trees. FT-trees were originally introduced by Chechik, Cohen, Fiat, and Kaplan [4]. There is an FT-tree $FT(s,t)$ required for every pair of vertices and a query traverses along a root-to-leaf path in the respective tree, until a shortest s-t-path in $G-F$ is found. This solution uses super-quadratic $\Omega(n^2L^f)$ space in total and is thus too large for our purpose. We devise a technique to emulate a search in an FT-tree without access to the tree. We use those searches to generate a carefully chosen family of subgraphs of G and apply the input distance oracle to each of them. This ensures that any query (s,t,F) that satisfies $d(s,t,F) \leqslant L$, is answered with an (α,β)-approximate path.

Theorem 6. *Let G be an unweighted (possibly directed) graph, with n vertices, and let f and L be positive integer parameters possibly depending on n. Assume access to a path-reporting distance oracle that, on any spanning subgraph of G, has stretch (α,β), takes space S, query time Q, and preprocessing time T. Then, there is an f-edge fault-tolerant distance sensitivity oracle for replacement paths in G with at most L edges that has*

- *stretch* (α,β),
- *space* $O(fL\log n)^f \cdot \mathsf{S}$,
- *query time* $\widetilde{O}(f \cdot (\mathsf{Q} + \alpha L + \beta + f^2 L))$,
- *preprocessing time* $\widetilde{O}(n^2(\alpha L+\beta)^f)\cdot(O(fL\log n)^f+\mathsf{Q})+O(fL\log n)^f\cdot\mathsf{T}$.

Note that the behavior of f-DSOs (hop-short or general) are usually only discussed for valid queries, that is, triples (s,t,F) where the edges in F are actually present in G. *Checking* for validity requires $\Omega(n^2)$ space, which would make subquadratic-space f-DSO impossible. Our query algorithm never uses the assumption $F \subseteq E$ and the data structure can be queried with any triplet (s,t,F) where $F \subseteq \binom{V}{2}$ is a set of at most f pairs of vertices. In this case it returns the approximate distance for the valid query $(s,t,F \cap E)$.

General Distance Sensitivity Oracles: Bilò et al. [19] described how to apply an f-DSO$^{\leqslant L}$ (with the restriction to hop-short paths) to construct a general distance sensitivity oracle without the constraint. They used two sets of pivots, which are vertices at with the hop-short solutions are combined. However, their approach only works for f-DSO$^{\leqslant L}$ with purely multiplicative stretch of 3, resulting in an f-DSO with multiplicative stretch $3+\varepsilon$. We generalize this by introducing a new data structure called *pivot trees*, as well as pinpointing the places in their construction that need to be adapted to accompany multiplicative stretch $\alpha \neq 3$ an additive stretch $\beta > 0$. The new pivot trees allow us to quickly find relevant pivots in $G-F$ that are close enough to the endpoints s and t in the query. An additional difference to [19] is the more involved stretch analysis. The issue is that the additive part β accumulates while the answers of the f-DSO$^{\leqslant L}$ are aggregated. Fortunately, this happens only if the paths in questions are hop-long, i.e., if they have more than L edges. This allows us to introduce an inductive argument controlling the stretch

accumulation and charge the overhead to the multiplicative part instead. As a result, we are able to turn a f-DSO$^{\leqslant L}$ with stretch (α, β) into a general f-DSO that has stretch $(\alpha(1+\varepsilon), \beta)$, all this while keeping the space subquadratic.

We first provide a randomized solution and then show how to derandomize. For the randomized construction, the guarantees hold *with high probability* (w.h.p.), which we define as with probability at least $1 - n^{-c}$ for some constant $c > 0$. In fact, c can be made arbitrarily large without affecting the asymptotics. We later show how to derandomize the oracle without any loss in its features, under very mild assumption on the parameters f and L. The main source of randomization is the creation of the pivot sets. The aforementioned pivot trees not only allow us to get better bounds for the randomized data structure but even help with derandomizing it.

To the best of our knowledge, we provide the first deterministic f-DSO with subquadratic space, near-additive stretch, and sublinear query time.

Theorem 7. *Let G be an undirected, unweighted graph with n vertices, m edges, and unique shortest paths. Let f and L be positive integer parameters possibly depending on n and m such that $2 \leqslant f \leqslant L \leqslant n$ as well as $L = \omega(\log n)$. Assume access to an f-edge fault-tolerant distance sensitivity oracle for replacement paths in G with at most L edges, that has stretch (α, β), space S_L, query time Q_L, and preprocessing time T_L. Then, for every $\varepsilon = \varepsilon(n, m, f, L) > 0$, and $\beta = o\left(\frac{\varepsilon^2 L}{f^3 \log n}\right)$, there is a randomized (general) f-edge fault-tolerant distance sensitivity oracle for G that with high probability has*

- *stretch $(\alpha(1+\varepsilon), \beta)$,*
- *query time $\widetilde{O}(f^5 L^{f-1}(\mathsf{Q}_L + f)/\varepsilon^2)$,*
- *preprocessing time $\mathsf{T}_L + O(L^{3f} n) + \widetilde{O}(f^2 m n^2/L) \cdot O(\log n/\varepsilon)^{f+1}$.*

The space of the data structure is w.h.p.

$$\mathsf{S}_L + \widetilde{O}(f L^{2f-1} n) + \widetilde{O}\left(f^2 \frac{n^2}{L}\right) \cdot O\left(\frac{\log n}{\varepsilon}\right)^{f+2}.$$

If additionally $f \geqslant 4$ and $L = \widetilde{O}(\sqrt{f^3 m/\varepsilon})$, the data structure can be made deterministic with the same stretch, query time, preprocessing time, and space.

If $\widetilde{O}(f L^{2f}) = \widetilde{O}(n)$, the space in Theorem 7 simplifies to $\mathsf{S}_L + \widetilde{O}(f^2 n^2/L) \cdot O(\log n/\varepsilon)^{f+2}$. That means our construction has subquadratic space as long as the space requirement S_L of the input f-DSO for short hop-distances is subquadratic. We would like to add some context to the restriction on the additive stretch β. Assume for simplicity that f is a constant. In most cases in the literature where an f-DSO$^{\leqslant L}$ is used to build an f-DSO, L is of order $n^{\Theta(1/f)}$. We also use such a value in this work. In this case, our construction supports an additive stretch that can be as large as a small polynomial, as long as it is asymptotically smaller than $\varepsilon^2 n^{O(1/f)}$. Conversely, for a fixed β, the restriction can be interpreted as bounding how fast the approximation parameter ε can approach 0 (we do not require ε to be constant). In Theorem 5, we have $\beta = 2$, hence $\varepsilon = \omega(\sqrt{\log(n)}/n^{O(1/f)})$. Finally, the derandomization requires $f \geqslant 4$ and $L = \widetilde{O}(\sqrt{f^3 m/\varepsilon})$. Even in the unfavorable case in which both f and ε are constants, this allows for a cut-off parameter up to $\widetilde{O}(\sqrt{m})$. In very dense graphs, this is no restriction at all.

C. Related Work

Distance Oracles: Thorup and Zwick [12] showed that, for any positive integer k, any DO for undirected graphs with a multiplicative stretch strictly less than $2k+1$ must take $\Omega(n^{1+1/k})$ bits off space, assuming the Erdős girth conjecture [21]. The lower bound only applies to graphs that are sufficiently dense and to queries that involve pairs of neighboring vertices, leading to several attempts to bypass it in different settings. For example, there is a line of work on improved distance oracles for sparse graphs. Porat and Roditty [22] showed that for unweighted graphs and any $\varepsilon > 0$, one can construct a DO with multiplicative stretch $1+\varepsilon$ and query time $\widetilde{O}(m^{1-\frac{\varepsilon}{4+2\varepsilon}})$. The space of the data structure is $O(nm^{1-\frac{\varepsilon}{4+2\varepsilon}})$, which is subquadratic for $m = o(n^{1+\frac{\varepsilon}{4+\varepsilon}})$. Pătraşcu, Roditty, and Thorup [23] obtained a series of DOs with fractional multiplicative stretches for sparse graphs. For general dense graphs, Pătraşcu and Roditty [10] devised a distance oracle for unweighted graphs with stretch $(2, 1)$ that has $O(1)$ query time, $O(n^{5/3})$ space, and can be constructed in time $O(mn^{2/3})$. They also showed that (α, β)-approximate DOs with $2\alpha + \beta < 4$ require $\Omega(n^2)$ space, assuming conjecture on the hardness of set intersection queries. Compared to the Pătraşcu and Roditty upper bound [10], Baswana, Goyal, and Sen [2] marginally increased the stretch to $(2, 3)$ and space to $\widetilde{O}(n^{5/3})$ in order to reduce the preprocessing time to $\widetilde{O}(n^2)$. The stretch was later reset again to $(2, 1)$ by Sommer [24], keeping the improved time complexity. A successive work by Knudsen [25] removed all additional poly-logarithmic factors in both the construction time and space.

Agarwal and Godfrey [1], [17] investigated the possibility of constructing a distance oracle with a stretch less than 2, albeit at the expense of slower query times. They showed that, for any positive integer ℓ and any real number $c \in (0, 1]$, it is possible to design a DO of size $O(m + n^{2-c})$ and multiplicative stretch $1 + \frac{1}{\ell}$. The query time is $O((n^c \mu)^\ell)$, where $\mu = \frac{2m}{n}$ is the average degree of the graph. Furthermore, they also showed that the query time can be reduced to $O((n^c + \mu)^{2\ell-1})$ at the cost of a small additive stretch $\frac{2\ell-1}{t} W$, with W being the maximum edge weight. Though the constructions in [1], [17] have a multiplicative stretch better than 2, their DOs have two main drawbacks. The subquadratic space only holds for sparse graphs, and, while they achieve a very low stretch, the query time is super-linear for dense graphs.

Akav and Roditty [26] proposed, for any $\varepsilon \in (0, \frac{1}{2})$, an $O(m + n^{2-\Omega(\varepsilon)})$-time algorithm that computes a DO with stretch $(2+\varepsilon, 5)$ and $O(n^{11/6})$ space, thus breaking the quadratic time barrier for multiplicative stretch below 3. Chechik and Zhang [27] improved this by offering both a DO with stretch $(2, 3)$ that can be built in $\widetilde{O}(m + n^{1.987})$ time and a

DO with stretch $(2+\varepsilon, c(\varepsilon))$ that can be built in $O(m+n^{\frac{5}{3}-\varepsilon})$ time, where $c(\varepsilon)$ is exponential in $1/\varepsilon$. Both data structures have space $\widetilde{O}(n^{5/3})$ and a constant query time.

Distance Sensitivity Oracles: Most of the work on distance sensitivity oracles is about handling a very small number $f \in \{1,2\}$ of failures [3], [6], [8], [9], [11], [28]–[33]. Here, we focus on related work with sensitivity $f \geqslant 3$ as this is the setting of the second problem in this paper. In their seminal work, Weimann and Yuster [13] designed a randomized f-DSO for exact distances introducing a size-time trade-off that is controlled by a parameter $\alpha \in (0,1)$. More precisely, their oracle w.h.p. has space $\widetilde{O}(n^{3-\alpha})$, query time of $\widetilde{O}(n^{2-2(1-\alpha)/f})$, and can be built in time $\widetilde{O}(mn^{2-\alpha})$. Van den Brand and Saranurak [34] and Karczmarz and Sankowski [35] presented f-DSOs using algebraic algorithms. However, their space requirement is at least quadratic and their query time is at least linear. Duan and Ren [7] provided an alternative f-DSO for exact distances with $O(fn^4)$ space, $f^{O(f)}$ query time, that is, constant whenever f is a constant. The preprocessing time for building their oracle is exponential in f, namely, $n^{\Omega(f)}$. Recently, Dey and Gupta [5] developed an f-DSO for undirected graphs where each edge has an integral weight from $\{1 \dots W\}$ with $O((cf\log(nW))^{O(f^2)})$ query time, where $c > 1$ is some constant. It has near-quadratic space $O(f^4 n^2 \log^2(nW))$. A drawback of their oracles is again the preprocessing time of $\Omega(n^f)$, and space at least $\Omega(n^2)$.

When allowing approximation, the f-DSO of Chechik et al. [4] guarantees a multiplicative stretch of $1+\varepsilon$ with a space requirement of $O(n^{2+o(1)}\log W)$, where $\varepsilon > 0$ W is constant and W is the weight of the heaviest edge. Their oracle can handle up to $f = o(\log n/\log\log n)$ failures, has a query time of $O(f^5 \log n \log\log W)$, and can be build in $O(n^{5+o(1)}(\log W)/\varepsilon^f)$ time. In fact, the preprocessing time has recently been reduced to $O(mn^{2+o(1)}/\varepsilon^f)$ [19]

Besides the general Thorup-Zwick bound [12] that assumes the girth conjecture, they also showed *unconditionally* that for undirected graphs with a multiplicative stretch better than 3 must take $\Omega(n^2)$ bits of space. This of course also applies in the presence of failures and is the reason why all the above f-DSOs have at least quadratic space. Like for distance oracles, there has been a line of work focusing on the design of f-DSOs with subquadratic space, sidestepping the bound. These oracles must have stretch (α,β) with $\alpha+\beta \geqslant 3$ since a stretch (α,β) can also be stated as $(\alpha+\beta, 0)$. Chechik, Langberg, Peleg, and Roditty [36] designed an f-DSO that, for any integer parameter $k \geqslant 1$, has space of $O(fkn^{1+1/k}\log(nW))$, query time $\widetilde{O}(|F|\log\log d_{G-F}(s,t))$, and guarantees a multiplicative stretch of $(8k-2)(f+1)$. Note that the stretch depends on the sensitivity parameter f. Recently, two f-DSOs with subquadratic space and stretch independent from f have been developed. The first one is by Bilò, Choudhary, Cohen, Friedrich, Krogmann, and Schirneck [37] that can handle up to $f = o(\log n/\log\log n)$ edge failures and, for every integer $k \geqslant 2$, guarantees a stretch of $2k-1$. The size and the query time depend on some trade-off parameter $\alpha \in (0, 1/k)$. They are equal to $kn^{1+\alpha+\frac{1}{k}+o(1)}$ and $\widetilde{O}(n^{1+\frac{1}{k}-\frac{\alpha}{k(f+1)}})$, respectively. The second f-DSO by Bilò et al. [19] works for unweighted graphs G with unique shortest paths. For every constants $f \geqslant 2$, $0 < \gamma < 1/2$, and any $\varepsilon > 0$ (possibly depending on m, n, and f), their oracle has stretch $3+\varepsilon$, space $\widetilde{O}(n^{2-\frac{\gamma}{f+1}}) \cdot O(\log n/\varepsilon)^{f+2}$, query time $O(n^\gamma/\varepsilon^2)$, and preprocessing time $\widetilde{O}(mn^{2-\frac{\gamma}{f+1}}) \cdot O(\log n/\varepsilon)^{f+1}$.

II. Technical Overview

This extended abstract is concluded by a more detailed overview of how we achieve the results stated above. The sections corresponds to the two main parts of the paper. The first is about our new construction of distance oracles. The second part describes the framework that turns (static) DOs into distance *sensitivity* oracles. This consists of two steps: first obtain an f-DSO$^{\leqslant L}$ and then use if for the general f-DSO. In the third part of the overview, we briefly sketch our derandomization approach.

A. Improved Distance Oracles

Our distance oracles introduce a small additive stretch in order to make the multiplicative stretch arbitrarily small while, at the same time, keep the space subquadratic. We assume the input graph to be undirected, for it is known that subquadratic-space DOs are impossible for digraphs [12]. The edges may have non-negative weights from a domain of polynomial size[2] with maximum weight W. A common pattern in the design of distance oracles is to designate a subset (or hierarchy of subsets) of vertices called *centers* [12], *landmark vertices* [17] or *pivots* [14], [15]. The data structure stores, for each vertex v in the graph, the distance from v to all pivots. Also, v knows its closest pivot $p(v)$. When given two query vertices $s,t \in V$, the oracle first checks whether s and t are sufficiently "close" to work out the exact distance $d(s,t)$. The definition of "close" varies among the different constructions in the literature. If s and t are instead "far" from each other, the estimate $d(s,p(s)) + d(p(s),t)$ is returned, which is at most $d(s,t) + 2\,d(s,p(s))$. Since s and t are "far" compared to $d(s,p(s))$, the estimate has a good stretch. This observation, of course, is in no way confined to the vertex s. For any vertex v on a shortest s-t-path, $d(s,p(v)) + d(p(v),t)$ incurs an error of at most $2\,d(v,p(v))$. This gives some freedom on how much storage space and query time one is willing to spend on finding such a v with a small distance to its closest pivot.

Our twist to that method is to look at the vicinity of a vertex not in terms of a fixed radius, but by an absolute bound on the number of considered vertices. Namely, we define a cut-off value K and store, for each vertex v, the K nearest vertices, regardless of the actual distance to v. Choosing $\widetilde{O}(n/K)$ pivots ensures that every vertex has a pivot in its K-vicinity. Storing the distances from every vertex to every pivot takes $\widetilde{O}(n^2/K)$ space, so K is our saving over quadratic space.

One of two things can happen when searching along the shortest path from s to t for a suitable vertex v. If all

[2]The domain size is such that any graph distance can be encoded in a constant number of machine words.

vertices in the list $K[v]$ have a small graph distance to v, also the closest pivot $p(v)$ must be nearby. Otherwise, there are elements in $K[v]$ that have a large graph distance, which we can use to skip ahead in the path. This sets up a win-win strategy. Consider the auxiliary graph H on the same vertex set as G in which any vertex v has an edge to each member of $K[v]$. Given a query (s,t) and an approximation parameter $\varepsilon > 0$, we conduct a bidirectional breath-first search in H starting from both s and t, trimming the search at hop-distance $1/\varepsilon$. The two searches meeting in one or more vertices is our definition of s and t being "close". We can then compute $d(s,t)$ exactly by minimizing $d(s,v) + d(v,t)$ over the intersecting vertices. Otherwise, we prove that the reason why the searches remained disjoint was that we could not skip ahead fast enough. There must have been a vertex v on the shortest s-t-path for which all neighbors in $K[v]$ have a small distance to v, including $p(v)$. We take "small" to mean at most $\frac{\varepsilon}{2}\, d(s,t)+W$, where W is the maximum edge weight. The sum $d(s,p(v)) + d(p(v),t)$ thus overestimates the true distance by at most $2\,d(v,p(v))$, resulting in an $1+\varepsilon$ multiplicative stretch and $2W$ additive.

Spacewise, the bottleneck is to store all the distances between vertices and pivots. In a second construction, we devise a way to further reduce the space, at the cost of increasing both the multiplicative and additive stretch. We are now looking for two vertices u and v on the s-t-path that both have small distance to their respective pivots $p(u)$ and $p(v)$. The portion of the distance between $p(u)$ and $p(v)$ is not stored directly but instead estimated at query time by another, internal, distance oracle. Since the inner data structure only needs to answer queries between pivots, we can get a $2k-1$ multiplicative stretch (for this part) with only $O((n/K)^{1+1/k})$ space. This results in a hierarchy of new DOs with ever smaller space.

B. From Hop-Short Distance Oracles to Distance Sensitivity Oracles

An f-DSO is a data structure that receives a query (s,t,F), consisting of vertices s and t as well as a set $F \subseteq E$ of at most f edges, and answers with an approximation of the s-t-distance $d(s,t,F)$ in the graph $G-F$. Let L be an integer cut-off parameter. A query (s,t,F) is *hop-short*, if s and t are joined by a path on at most L edges in $G-F$. An f-DSO for hop-short distances (f-DSO$^{\leqslant L}$) only guarantees a good stretch for hop-short queries. Such oracles are used as stepping stones towards general f-DSOs [11], [13], [38], [39].

Recently, Bilò et al. [19] presented a new approximate f-DSO for unweighted graphs with unique shortest paths taking only subquadratic space. Implicit in their work is a pathway that takes the (static) distance oracle of Thorup and Zwick with multiplicative stretch 3 and makes it fault-tolerant increasing the stretch to $3+\varepsilon$. The parameter $\varepsilon > 0$ influences the space, query time, and preprocessing time of the data structure. The transformation has two major steps. In the first one, the DO is used to build a hop-short f-DSO$^{\leqslant L}$. The second step then combines the answer for hop-short paths into good approximations for arbitrary queries. Their ad-hoc method is highly tailored towards the DO of Thorup and Zwick [12] and does not readily generalize.

We take the same two-step approach but give an entirely new construction for the hop-short f-DSO$^{\leqslant L}$. This is necessary to make it compatible with other distance oracles. Our framework works with any path-reporting DO as a black box. The input oracle can have an arbitrary multiplicative stretch α and may even have an additive component β, requiring new techniques. The resulting distance sensitivity oracle has stretch $(\alpha(1+\varepsilon), \beta)$. The key feature of the reduction is that, if the input DO has subquadratic space, so does the f-DSO.

Fault Tolerance for Hop-Short Paths: Naively, an oracle with sensitivity f must be able to handle $O(n^2 m^f)$ queries, one for each pair of vertices and any set F of up to f edge failures. Not every failure set is relevant for every pair of vertices. If the shortest s-t-path does not contain an edge of F, it is still the shortest path in $G-F$. As a first key tool we use *fault-tolerant trees* (FT-trees) to zero in on the relevant queries. The were originally introduced by Chechik et al. [4]. We combine it with hop-short paths. There is a tree $FT(s,t)$ for every pair of vertices s and t whose shortest path in the original graph G has at most L edges. In the root that path is stored. For any edge e_1 along that path, the root has a child node which in turn stores a shortest s-t-path in the graph $G - \{e_1\}$. This corresponds to failing e_1 and looking for a *replacement path* $P(s,t,\{e_1\})$. The construction is iterated for each child node until depth f is reached. That means, for any node x at level k of $FT(s,t)$, let $F_x = \{e_1, \ldots, e_k\}$ be the edges associated with the path the root to x. The node x stores a shortest s-t-path P_x in $G-F_x$. If the shortest s-t-replacement path has more than L edges, x is made a leaf, marked by setting $P_x = \perp$.

For a query (s,t,F), in any node x starting with the root, it is checked whether $E(P_x) \cap F \neq \emptyset$. If so, the search continues with the first child node associated with an edge in $E(P_x) \cap F$. Otherwise, there are two cases. Either x stores some shortest s-t-path disjoint from F or $P_x = \perp$. In the first case, the length $|P_x|$ is the desired *replacement distance* $d(s,t,F)$. In the second, it is enough to report that $d(s,t,F) > L$. This reduces the number of relevant queries to $O(n^2 L^f)$.

The second key tool is an (L,f)-*replacement path covering* (RPC) [13], [39]. This is a family of $\mathcal{G}$ subgraphs of G such that for any set F of at most f edges and pair of vertices s and t for which there is a replacement path $P(s,t,F)$ of at most L edges, there exists a subgraph $G_i \in \mathcal{G}$ such that $E(G_i) \cap F = \emptyset$ and G_i contains $P(s,t,F)$. RPCs are common in the design of f-DSO$^{\leqslant L}$ because, if one can find G_i quickly, the replacement distance is just $d_{G_i}(s,t)$. Recently, Karthik and Parter [39] gave a construction with $O(fL\log n)^{f+1}$ graphs. We shave an $(fL\log n)$-factor from that. In the full version of [39], they give an algorithm that takes a list $(F_1,P_1),\ldots(F_\ell,P_\ell)$ of pairs of edge sets, with $|F_k| \leqslant f$ and $|P_k| \leqslant L$. It computes a family of subgraphs that only for pairs (F_k,P_k) from the list guarantees some G_i avoiding F_k but containing P_k. We use nodes of the fault-tolerant trees to compute a list that allows us to cover all relevant queries but get a smaller family

with only $O(fL\log n)^f$ graphs. This improvement may be of independent interest.

Unfortunately, in the subquadratic-space regime, both the FT-trees as well as the graphs of the replacement-path covering are too large. The former have size $O(n^2L^{f+1})$ (each node stores up to L edges), and the latter $O(m)\cdot O(fL\log n)^f$. Instead, we build a data structure that *emulates* queries in the FT-trees without actually storing them. The graphs in the RPC are replaced by a distance oracles. When using an (α,β)-approximate DO, this gives a saving over quadratic space. The main difficulty is that the emulated query procedure must follow the exact same path as in the actual (discarded) tree $FT(s,t)$ in order to find the correct distance oracle.

From Hop-Short to General Distance Sensitivity Oracles: The second step of the transformation takes the f-DSO$^{\leqslant L}$ for hop-short distances and combines it with techniques to stitch together the answer for hop-long paths from those for hop-short ones. This step follows the pathway in [19] more closely. The key differences are the introduction of what we call *pivot trees* as well as a more thorough analysis that is needed to deal with additive stretch.

We reuse the idea of FT-trees but, as before, they are too large to be stored. Above, we emulated a query without access to the actual trees. Here, we do store some of the trees but in smaller versions and not for all pairs of vertices. Since we must also handle hop-long queries, any node of an FT-tree may now store a path of up to n edges. If we were to create a child node for each edge, we would end up with a prohibitive space of $O(n^{f+1})$ for a single tree. Instead, we use larger segments and each child node now corresponds to the failure of a whole segment. Of course, this has the danger that failing a segment may inadvertently destroy replacement paths that would still exists in $G-F$, where only the failing edges are removed. We have to strike a balance between the space reduction of larger segments and the introduced inaccuracies. This leads to the concept of granularity. In an FT-tree with granularity λ, the first and last $\lambda/2$ edges form their own segments. Beyond that, we use segments whose size increase exponentially towards the middle of the stored path. The base of this exponential is $1+\Theta(\varepsilon)$, where $\varepsilon>0$ is the approximation parameter. A higher granularity means larger trees but with higher accuracy.

We offset this by building the larger trees only for very few pairs of vertices. We sample a set B_1 of $\widetilde{O}(fn/L^f)$ pivots and build an FT-tree with granularity $\lambda=\Theta(\varepsilon L)$ for each pair of vertices in $B_1\times B_1$. For some vertex v and failure set F, let $\mathrm{ball}_{G-F}(v,\lambda/2)$ be the ball of hop-distance $\lambda/2$ around v in $G-F$. We show that if both s and t have respective pivots $p_s\in\mathrm{ball}_{G-F}(s,\lambda/2)\cap B_1$ and $p_t\in\mathrm{ball}_{G-F}(t,\lambda/2)\cap B_1$, then the FT-tree $FT_\lambda(p_s,p_t)$ with granularity λ gives a very good estimate of the replacement distance $d(s,t,F)$. However, since there are so few pivots in B_1 this can only be guaranteed if the balls around s and t are very dense. We also have to give a fall-back solution in case one of the balls is sparse. We sample a set B_2 of now $\widetilde{O}(fn/L)$ pivots (much more than B_1) and build an FT-tree *without* granularity for pairs of vertices in $B_2\times V$. These trees are much smaller and we show that together with the f-DSO$^{\leqslant L}$ they are still sufficiently accurate.

The problem is to quickly find the pivots that are close to s and t in $G-F$ at query time. Simply scanning over B_1 and B_2 is way too slow. Instead, we use yet another kind of tree data structure, *pivot trees*. They are inspired by FT-trees but are not the same. We have one pivot tree per vertex in V (instead of an FT-tree for every pair). The tree belonging to s stores in each node x some shortest path P_x starting from s as before, and each child of node x represents the failure of one edge of P_x. The key difference is that the other endpoint of P_x can vary now, the path ends in the closest pivot $p_s\in B_1$ of the first type. Note that in the graph $G-F_x$ that pivot may not be the same as in $G-(F_x\cup\{e_{k+1}\})$. We only care about the vicinity $\mathrm{ball}_{G-F}(s,\lambda/2)$, so the paths P_x have at most $\lambda/2$ edges. If the closest pivot in B_1 is too far away, $\mathrm{ball}_{G-F}(s,\lambda/2)$ must be sparse. That means, also $\mathrm{ball}_{G-F}(s,\lambda/2)\cap B_2$ is small and it is feasible to store all pivots of the second type.

Unfortunately, this is still not enough. Due to our sparsification via segments with multiple edges, the answer of the FT-trees with and with out granularity are only accurate if the replacement path $P(s,t,F)$ is "far away" from all failures in F. That roughly means that any vertex of $P(s,t,F)$ is more than an $\Theta(\varepsilon)$-multiple from any endpoint of some edge in F. If this safety area is free of failures, no segment can accidentally contain an edge of F, which would disturb the return value of the FT-tree too much. If, however, there is a failure too close to the path, Chechik et al. [4] showed that there is always some surrogate target $t'\in V(F)$ such that the replacement path $P(s,t',F)$ is indeed far away from all failures and also not much of a detour compared to going directly from s to t. This causes an $1+\varepsilon_1$ factor in the multiplicative part of the stretch, where $\varepsilon_1=\Theta(\varepsilon)$. We build an auxiliary *weighted* complete graph H on the vertex set $V(F)\cup\{s,t\}$ to exploit this detour structure. We use the FT-trees and f-DSO$^{\leqslant L}$ to compute the weight of all edges in the graph. The eventual answer of our oracle is the s-t-distance in H.

The main obstacle is to prove that $d_H(s,t)$ is actually an $(\alpha(1+\varepsilon),\beta)$-approximation of $d(s,t,F)$. This is also the other decisive difference to [19] in that we need to handle both the multiplicative and the additive stretch. Consider an edge $\{u,v\}\in E(H)$. If $P(u,v,F)$ is hop-short, the f-DSO$^{\leqslant L}$ trivially gives an (α,β)-approximation of $d(u,v,F)$ which we use to compute the weight $w_H(u,v)$. If $P(u,v,F)$ has more than L edges and is "far away" from all failures, we show that $w_H(u,v)$ computed by the FT-trees is only an $(\alpha,X\beta)$-approximation. The blow-up $X=O(f\log(n)/\varepsilon)$ stems from the segments of increasing size. This is only exacerbated by the fact that the shortest s-t-path in the weighted graph H has $O(f^2)$ edges, each one contributing their own distortion to the additive stretch. However, the *additive* blow-up only happens if $P(u,v,F)$ has many edges, that is, if $d(u,v,F)$ is large. The idea is to then charge most of the additive stretch to the multiplicative part, increasing it only by another $1+\varepsilon_2=1+\Theta(\varepsilon)$ factor which is then combined with the $1+\varepsilon_1$ stemming from the auxiliary graph H. To make this work, we carefully analyze the interplay of the edges in H, using an

induction over $E(H)$ in the order of increasing replacement distance $d(u, v, F)$.

C. Derandomization

We use the *critical paths* proposed by Alon, Chechik, and Cohen [38] for the derandomization. That means identifying a small set of shortest paths in G, (greedily) computing a deterministic hitting set for them, and building the fault-tolerant data structure from there. The number of paths needs to be small in order to keep the derandomization efficient, which is the main difficulty. Our threshold of efficiency is that making the data structures deterministic should not increase their preprocessing time by more than constant factors.

In the context of derandomization, we view paths as mere sets of vertices without any further structure. This allows us to apply the approach also to other subsets of V in a unified fashion. Consider the (static) distance oracle for weighted graphs in Theorems 1 and 3. The list $K[v]$, containing the K vertices closest to v, are key components in the construction. It is indeed enough to hit all the $\{K[v]\}_{v\in V}$ to derandomize the DO. This also incurs hardly any extra work as those lists are compiled anyway in the preprocessing.

The construction of the general f-DSO from the one restricted to hop-short distances (Theorem 7) builds on the pivot sets B_1 and B_2 whose derandomization is more involved. Recall that we use λ for the granularity of the FT-trees. During the proof of correctness, it becomes apparent that the pivots of the second type in B_2 need to hit the length-$\lambda/2$ prefixes and suffixes of all concatenations of up to two replacement paths, provided that those concatenations are hop-long (have more than L edges). We use a structural result by Afek, Bremler-Barr, Kaplan, Cohen, and Merritt [40] that states that replacement paths themselves are concatenations of $O(f)$ shortest paths in the original graph G. This allows us to show that computing a deterministic hitting set of all shortest paths in G of length $\Omega(\lambda/f)$ suffices to guarantee the same covering properties as B_2. The set B_1, in turn, need to hit all the sets $\mathrm{ball}_{G-F}(u, \lambda/2)$ that are sufficiently dense (more than L^f vertices). In principle, a similar approach as for B_2 would work but that would either produce way too many pivots or take much to long. Instead, we interleave the level-wise construction of the pivot trees with derandomization phases to find a deterministic stand-in for B_1.

References

[1] R. Agarwal, "The Space-Stretch-Time Tradeoff in Distance Oracles," in *Proceedings of the 22th Annual European Symposium on Algorithms (ESA)*, 2014, pp. 49–60.

[2] S. Baswana, V. Goyal, and S. Sen, "All-pairs Nearly 2-Approximate Shortest Paths in $O(n^2\,\mathsf{polylog}\,n)$ Time," *Theoretical Computer Science*, vol. 410, pp. 84–93, 2009.

[3] S. Chechik and S. Cohen, "Distance Sensitivity Oracles with Subcubic Preprocessing Time and Fast Query Time," in *Proccedings of the 52nd Symposium on Theory of Computing (STOC)*, 2020, pp. 1375–1388.

[4] S. Chechik, S. Cohen, A. Fiat, and H. Kaplan, "$(1+\varepsilon)$-Approximate f-Sensitive Distance Oracles," in *Proceedings of the 28th Symposium on Discrete Algorithms (SODA)*, 2017, pp. 1479–1496.

[5] D. Dey and M. Gupta, "Nearly Optimal Fault Tolerant Distance Oracle," in *Proceedings of the 56th Symposium on Theory of Computing (STOC)*, 2024, pp. 944–955.

[6] R. Duan and S. Pettie, "Dual-Failure Distance and Connectivity Oracles," in *Proceedings of the 20th Symposium on Discrete Algorithms (SODA)*, 2009, pp. 506–515.

[7] R. Duan and H. Ren, "Maintaining Exact Distances under Multiple Edge Failures," in *Proceedings of the 54th Symposium on Theory of Computing (STOC)*, 2022, pp. 1093–1101.

[8] C. Demetrescu and M. Thorup, "Oracles for Distances Avoiding a Link-Failure," in *Proceedings of the 13th Symposium on Discrete Algorithms (SODA)*, 2002, pp. 838–843.

[9] F. Grandoni and V. Vassilevska Williams, "Faster Replacement Paths and Distance Sensitivity Oracles," *ACM Transaction on Algorithms*, vol. 16, pp. 15:1–15:25, 2020.

[10] M. Pătraşcu and L. Roditty, "Distance Oracles Beyond the Thorup-Zwick Bound," *SIAM Journal on Computing*, vol. 43, pp. 300–311, 2014.

[11] H. Ren, "Improved Distance Sensitivity Oracles with Subcubic Preprocessing Time," *Journal of Computer and System Sciences*, vol. 123, pp. 159–170, 2022.

[12] M. Thorup and U. Zwick, "Approximate Distance Oracles," *Journal of the ACM*, vol. 52, pp. 1–24, 2005.

[13] O. Weimann and R. Yuster, "Replacement Paths and Distance Sensitivity Oracles via Fast Matrix Multiplication," *ACM Transactions on Algorithms*, vol. 9, pp. 14:1–14:13, 2013.

[14] S. Chechik, "Approximate Distance Oracles with Constant Query Time," in *Proceedings of the 46th Symposium on Theory of Computing (STOC)*, 2014, pp. 654–663.

[15] ——, "Approximate Distance Oracles with Improved Bounds," in *Proceedings of the 47th Symposium on Theory of Computing (STOC)*, 2015, pp. 1–10.

[16] C. Wulff-Nilsen, "Approximate Distance Oracles with Improved Query Time," in *Proceedings of the 24th Symposium on Discrete Algorithms (SODA)*, 2013, pp. 539–549.

[17] R. Agarwal and P. B. Godfrey, "Distance Oracles for Stretch Less Than 2," in *Proceedings of the 24th Symposium on Discrete Algorithms (SODA)*, 2013, pp. 526–538.

[18] I. Abraham and C. Gavoille, "On Approximate Distance Labels and Routing Schemes with Affine Stretch," in *Proceedings of the 25th International Symposium on Distributed Computing (DISC)*, 2011, pp. 404–415.

[19] D. Bilò, S. Chechik, K. Choudhary, S. Cohen, T. Friedrich, S. Krogmann, and M. Schirneck, "Approximate Distance Sensitivity Oracles in Subquadratic Space," *TheoretiCS*, vol. 3, pp. 15:1–15:47, 2024.

[20] S. Cabello, E. W. Chambers, and J. Erickson, "Multiple-Source Shortest Paths in Embedded Graphs," *SIAM Journal on Computing*, vol. 42, pp. 1542–1571, 2013.

[21] P. Erdős, "Extremal Problems in Graph Theory," *Theory of Graphs and its Applications*, pp. 29–36, 1964.

[22] E. Porat and L. Roditty, "Preprocess, Set, Query!" in *Proceedings of the 19th European Symposium on Algortihms (ESA)*, 2011, pp. 603–614.

[23] M. Pătraşcu, L. Roditty, and M. Thorup, "A New Infinity of Distance Oracles for Sparse Graphs," in *Proceedings of the 53rd Symposium on Foundations of Computer Science (FOCS)*, 2012, pp. 738–747.

[24] C. Sommer, "All-pairs Approximate Shortest Paths and Distance Oracle Preprocessing," in *Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP)*, 2016.

[25] M. B. T. Knudsen, "Additive Spanners and Distance Oracles in Quadratic Time," in *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2017, pp. 64:1–64:12.

[26] M. Akav and L. Roditty, "An Almost 2-Approximation for All-Pairs of Shortest Paths in Subquadratic Time," in *Proceedings of the 14th Symposium on Discrete Algorithms (SODA)*, 2020, pp. 1–11.

[27] S. Chechik and T. Zhang, "Nearly 2-Approximate Distance Oracles in Subquadratic Time," in *Proceedings of the 33rd Symposium on Discrete Algorithms (SODA)*, 2022, pp. 551–580.

[28] C. Demetrescu, M. Thorup, R. A. Chowdhury, and V. Ramachandran, "Oracles for Distances Avoiding a Failed Node or Link," *SIAM Journal on Computing*, vol. 37, pp. 1299–1318, 2008.

[29] A. Bernstein and D. R. Karger, "Improved Distance Sensitivity Oracles via Random Sampling," in *Proceedings of the 19th Symposium on Discrete Algorithms (SODA)*, 2008, pp. 34–43.

[30] ——, "A Nearly Optimal Oracle for Avoiding Failed Vertices and Edges," in *Proceedings of the 41st Symposium on Theory of Computing (STOC)*, 2009, pp. 101–110.

[31] S. Baswana and N. Khanna, "Approximate Shortest Paths Avoiding a Failed Vertex: Near Optimal Data Structures for Undirected Unweighted Graphs," *Algorithmica*, vol. 66, pp. 18–50, 2013.

[32] D. Bilò, S. Cohen, T. Friedrich, and M. Schirneck, "Near-Optimal Deterministic Single-Source Distance Sensitivity Oracles," in *Proceedings of the 29th European Symposium on Algorithms (ESA)*, 2021, pp. 18:1–18:17.

[33] Y. Gu and H. Ren, "Constructing a Distance Sensitivity Oracle in $O(n^{2.5794}M)$ Time," in *Proceedings of the 48th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2021, pp. 76:1–76:20.

[34] J. v. d. Brand and T. Saranurak, "Sensitive Distance and Reachability Oracles for Large Batch Updates," in *Proceedings of the 60th Symposium on Foundations of Computer Science (FOCS)*, 2019, pp. 424–435.

[35] A. Karczmarz and P. Sankowski, "Sensitivity and dynamic distance oracles via generic matrices and frobenius form," in *Proceedings of the 64th Symposium on Foundations of Computer Science (FOCS)*, 2023, pp. 1745–1756.

[36] S. Chechik, M. Langberg, D. Peleg, and L. Roditty, "f-Sensitivity Distance Oracles and Routing Schemes," *Algorithmica*, vol. 63, pp. 861–882, 2012.

[37] D. Bilò, K. Choudhary, S. Cohen, T. Friedrich, S. Krogmann, and M. Schirneck, "Compact Distance Oracles with Large Sensitivity and Low Stretch," in *Proceedings of the 18th Algorithms and Data Structures Symposium (WADS)*, 2023, pp. 149–163.

[38] N. Alon, S. Chechik, and S. Cohen, "Deterministic Combinatorial Replacement Paths and Distance Sensitivity Oracles," in *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming, (ICALP)*, 2019, pp. 12:1–12:14.

[39] Karthik C.S. and M. Parter, "Deterministic Replacement Path Covering," in *Proceedings of the 32nd Symposium on Discrete Algorithms (SODA)*, 2021, pp. 704–723.

[40] Y. Afek, A. Bremler-Barr, H. Kaplan, E. Cohen, and M. Merritt, "Restoration by Path Concatenation: Fast Recovery of MPLS Paths," *Distributed Computing*, vol. 15, pp. 273–283, 2002.