Compact Distance Oracles with Large Sensitivity and Low Stretch

Davide Bilò[1], Keerti Choudhary[2], Sarel Cohen[3], Tobias Friedrich[4], Simon Krogmann[4], and Martin Schirneck[5]

[1] Department of Information Engineering, Computer Science and Mathematics, University of L'Aquila, Italy davide.bilo@univaq.it
[2] Department of Computer Science and Engineering, Indian Institute of Technology Delhi, India keerti@iitd.ac.in
[3] School of Computer Science, The Academic College of Tel Aviv-Yaffo, Israel sarelco@mta.ac.il
[4] Hasso Plattner Institute, University of Potsdam, Germany firstname.lastname@hpi.de
[5] Faculty of Computer Science, University of Vienna, Austria martin.schirneck@univie.ac.at

Abstract. An *f-edge fault-tolerant distance sensitive oracle* (f-DSO) with stretch $\sigma \geqslant 1$ is a data structure that preprocesses an input graph G. When queried with the triple (s, t, F), where $s, t \in V$ and $F \subseteq E$ contains at most f edges of G, the oracle returns an estimate $\widehat{d}_{G-F}(s,t)$ of the distance $d_{G-F}(s,t)$ between s and t in the graph $G-F$ such that $d_{G-F}(s,t) \leqslant \widehat{d}_{G-F}(s,t) \leqslant \sigma \cdot d_{G-F}(s,t)$.

For any positive integer $k \geqslant 2$ and any $0 < \alpha < 1$, we present an f-DSO with sensitivity $f = o(\log n / \log\log n)$, stretch $2k-1$, space $O(n^{1+\frac{1}{k}+\alpha+o(1)})$, and an $\widetilde{O}(n^{1+\frac{1}{k}-\frac{\alpha}{k(f+1)}})$ query time.

Prior to our work, there were only three known f-DSOs with subquadratic space. The first one by Chechik et al. [Algorithmica 2012] has a stretch of $(8k-2)(f+1)$, depending on f. Another approach is storing an *f-edge fault-tolerant $(2k-1)$-spanner* of G. The bottleneck is the large query time due to the size of any such spanner, which is $\Omega(n^{1+1/k})$ under the Erdős girth conjecture. Bilò et al. [STOC 2023] gave a solution with stretch $3+\varepsilon$, query time $O(n^{\alpha})$ but space $O(n^{2-\frac{\alpha}{f+1}})$, approaching the quadratic barrier for large sensitivity.

In the realm of subquadratic space, our f-DSOs are the first ones that guarantee, at the same time, large sensitivity, low stretch, and non-trivial query time. To obtain our results, we use the approximate distance oracles of Thorup and Zwick [JACM 2005], and the derandomization of the f-DSO of Weimann and Yuster [TALG 2013] that was recently given by Karthik and Parter [SODA 2021].

Keywords: Approximate shortest paths · Distance sensitivity oracle · Fault-tolerant data structure · Spanner · Subquadratic space.

1 Introduction

There are applications, like routing on edge devices, where we want to quickly find out the distances between pairs of vertices, but we cannot store the entire graph topology due to memory restrictions. This problem is solved by a class of data structures called *distance oracles* (DO). Typically, not a single structure serves every use case and constructions need to provide reasonable trade-offs between the space requirement, query time, and stretch, that is, the quality of the estimated distance. We are interested in the fault-tolerant setting. Here, the data structure must additionally be able to tolerate multiple edge failures in the underlying graph. An *f-edge fault-tolerant distance sensitivity oracles* (f-DSO) for a graph $G = (V, E)$ is able to report, for any two $s, t \in V$ and set $F \subseteq E$ of at most f failing edges, an estimate $\widehat{d}_{G-F}(s,t)$ of the *replacement distance* $d_{G-F}(s,t)$ in the graph $G - F$. The parameter f is the *sensitivity* of the oracle. We say the *stretch* of the data structure is σ if $d_{G-F}(s,t) \leqslant \widehat{d}_{G-F}(s,t) \leqslant \sigma \cdot d_{G-F}(s,t)$, for any admissible query (s,t,F).

Several f-DSOs with different space-stretch-time trade-offs have been designed in the last decades, most of which can only handle a very small number $f \leqslant 2$ of failures [3,4,5,8,14,18,19,20,23,24,31]. We highlight those with sensitivity $f \geqslant 3$. The f-DSO of Duan and Ren [21] requires $O(fn^4)$ space,[1] returns exact distances in $f^{O(f)}$ query time, but the preprocessing algorithm that builds it requires exponential-in-f time $n^{\Omega(f)}$. The data structure by Chechik, Cohen, Fiat, and Kaplan [15] is more compact, requiring $O(n^{2+o(1)} \log W)$ space, and can be preprocessed in time[2] $\widetilde{O}(n^{5+o(1)} \log W)$, where W is the weight of the heaviest edge of G. The oracle has stretch $1+\varepsilon$, for any constant $\varepsilon > 0$, with an $O(f^5 \log n \log\log W)$ query time, and handles up to $f = o(\log n / \log\log n)$ failures. Finally, the f-DSO of Chechik, Langberg, Peleg, and Roditty [17] requires a subquadratic space of $O(fkn^{1+1/k} \log(nW))$, where $k \geqslant 1$ is an integer parameter, and has a fast query time of $\widetilde{O}(f \log\log d_{G-F}(s,t))$, but guarantees only a stretch of $(8k-2)(f+1)$ that depends on the number f of failures.

Another common way to provide approximate replacement distances in the presence of transient edge failures are fault-tolerant spanners [29]. An *f-edge fault-tolerant spanner with stretch σ (fault-tolerant σ-spanner)* is a subgraph H of G such that $d_{H-F}(s,t) \leqslant \sigma \cdot d_{G-F}(s,t)$, for every suitable triple (s,t,F), with $|F| \leqslant f$. For any positive integer k, Chechik, Langberg, Peleg, and Roditty [16] gave an algorithm computing a fault-tolerant $(2k-1)$-spanner with $O(fn^{1+1/k})$ edges. This was recently improved by Bodwin, Dinitz, and Robelle by reducing the size to $O(f^{1-1/k} n^{1+1/k})$ [9] and eventually to $f^{1/2} n^{1+1/k} \cdot \mathsf{poly}(k)$ for any even k and $f^{1/2-1/(2k)} n^{1+1/k} \cdot \mathsf{poly}(k)$ for odd k [10]. The authors of the last work also show almost matching lower bounds of $\Omega(f^{1/2-1/(2k)} n^{1+1/k} + fn)$ for general $k > 2$, and $\Omega(f^{1/2} n^{3/2})$ for $k = 2$ assuming the Erdős girth conjecture [22].

The main problem with the spanner approach is the high query time. In fact, to retrieve the approximate distance between a given pair of vertices, one

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

[2] For a non-negative function $g(n)$, we use $\widetilde{O}(g)$ to denote $O(g \cdot \mathsf{polylog}(n))$.

has to compute the single-source distance from one of them, say with Dijkstra's algorithm, in time that is at least linear in the size of the spanner. Therefore, an important problem in the field of fault-tolerant data structures to design f-DSOs with subquadratic space, that simultaneously guarantee a non-trivial $o(n^{1+1/k})$ query time, a low stretch of $2k-1$, and a large sensitivity f.[3]

Very recently, Bilò, Chechik, Choudhary, Cohen, Friedrich, Krogmann, and Schirneck [6] addressed the same problem. They presented, for all $\varepsilon > 0$, and constants $1/2 > \alpha > 0$, and f, a $(3+\varepsilon)$-approximate f-DSO for unweighted graphs taking space $\widetilde{O}_{\varepsilon}(n^{2-\frac{\alpha}{f+1}}(\log n)^{f+1})$ and has a query time of $\widetilde{O}_{\varepsilon}(n^{\alpha})$. While their query time is sub-linear, their space converges to quadratic for large sensitivity.

In contrast, we design a deterministic oracle for weighted graphs that can handle up to $f = o(\log n/\log\log n)$ edge failures and provides a trade-off between stretch, space, and query time. Namely, for any positive integer $k \geqslant 2$ and constant $1 - \frac{1}{k} > \alpha > 0$, our data structure has stretch $2k-1$, requires $kn^{1+\alpha+\frac{1}{k}+o(1)}$ space, and can be queried in time $\widetilde{O}(n^{1+\frac{1}{k}-\frac{\alpha}{k(f+1)}})$. The query time improves to $\mathsf{poly}(D, f, \log n)$ for graphs in which the pair-wise hop distances are bounded by D. If, for example, D is polylogarithmic, the query time is as well. We note that the query time of our f-DSO for general graphs is $\Omega(n)$ for all choices of α.

Both [6] and this work approach the problem by handling hop-short and hop-long paths separately, as is common in the area [23,35], and use the distance oracle of Thorup and Zwick [34] on the lowest level. Apart from that, the techniques are different. We highlight ours below.

Our techniques. Our f-DSO for bounded hop diameter is the result of combining the approximate distance oracles of Thorup and Zwick [34] with randomized replacement path covering (RPC), a collection of certain subgraphs of G, introduced by Weimann and Yuster [35]. Such coverings are very large, even larger than the underlying graph itself. They are thus unusable when emphasizing subquadratic space, barring additional processing. The main issue when compressing an RPC is retaining the information which subgraph is relevant for which query. We provide two different ways to solve this. One is based on the idea of using sparse spanners as proxies for the subgraphs in the covering, and the other one uses the recent derandomization technique of Karthik and Parter [26]. To lift this to an arbitrary hop diameter, we borrow from fault-tolerant spanners. There, a single graph is constructed up front to handle all queries. To achieve a compact oracle with a query time better than any spanner, we instead turn this process around and use the hop-short f-DSO to combine only the subgraphs we need.

Other related work. Demetrescu and Thorup [18] designed the first exact 1-DSO for directed edge-weighted graphs with $O(n^2 \log n)$ space and $O(\log n)$ query time. Demetrescu, Thorup, Chowdhury, and Ramachandran [19] improved the query time to $O(1)$ and generalized the oracle to handle also a single vertex failure. Later, in two consecutive papers, Bernstein and Karger improved the

[3] Subquadratic space f-DSOs with stretch $2k-1$ can only exist for $k \geqslant 2$. There is an $\Omega(n^2)$-bit lower bound for exact f-DSOs, regardless of the query time [34].

preprocessing time from $\widetilde{O}(mn^2)$ to $\widetilde{O}(mn)$ [4,5]. Khanna and Baswana [27] designed 1-DSO for unweighted graphs having size $O(k^5 n^{1+1/k} \frac{\log^3 n}{\varepsilon^4})$, a stretch of $(2k-1)(1+\varepsilon)$, and $O(k)$ query time. The problem of 1-DSO was also studied with a special focus on the preprocessing time [14,23,24,31,35].

For the case of multiple failures, other than the results we explicitly mentioned in the introduction [15,17,21], it is worth mentioning the 2-DSO of Duan and Pettie [20] with $O(n^2 \log^3 n)$ size and $O(\log n)$ query time and the work by van den Brand and Saranurak [11].

Outline. Section 2 provides an overview of our approach and presents our results. The preliminaries and notation needed to follow the technical part are given in Section 3. In Section 4, we first describe the randomized subquadratic-space f-DSO for short hop distances and then derandomize it, not only to obtain a deterministic construction but also to accelerate the query time to $\mathsf{poly}(D, f, \log n)$. Section 5 then describes how to use this to develop a deterministic subquadratic-space f-DSO also for hop-long replacement paths.

Some of the proofs are deferred to **??** due to space reasons.

2 Overview

Our first goal is to develop an f-DSO whose space is subquadratic in n, provided that the hop diameter[4] D and the sensitivity f are not too large. One of the first DSOs was given by Weimann and Yuster [35]. It reports exact distances but, on graphs with a large hop diameter, it is too large and too slow. We first give an overview of their techniques and then describe the steps we take to reduce the space as well as the query time using approximation.

Given the graph $G = (V, E)$ as well as positive integers L and f, the DSO in [35] samples a family $\mathcal{G}$ of $\widetilde{O}(fL^f)$ random spanning subgraphs of G, that is, all the subgraphs have the same vertex set V. Each graph $G_i \in \mathcal{G}$ is generated by removing each edge of G with probability $1/L$. With high probability,[5] for all vertices $s, t \in V$ and sets $F \subseteq E$ of at most f edge failures, if there is a replacement path from s to t that has at most L edges and none of them is in F, then there exists a subgraph $G_i \in \mathcal{G}$ that does not contain any edge of F but such an replacement path. Let $\mathcal{G}_F \subseteq \mathcal{G}$ be the subfamily of all the G_i in which at least all of F was removed. In other words, if s and t have a *hop-short* shortest path in $G - F$, at least one of their replacement paths survives in a graph in $\mathcal{G}_F$.

To handle hop-short replacement paths, it is enough to go over the subgraphs and report the minimum distance $d_{G_i}(s, t)$ over all $G_i \in \mathcal{G}_F$. For the *hop-long* replacement paths on more than L edges, a random subset $B \subseteq V$

[4] The *hop diameter* of a weighted graph is the minimum integer D such that all shortest paths between pairs of vertices have at most D edges.

[5] An event occurs *with high probability* (w.h.p.) if it has success probability at least $1 - n^{-c}$ for some constant $c > 0$. In fact, c can always be made arbitrarily large without affecting the asymptotics.

of $\widetilde{O}(fn/L)$ of *pivots* is sampled. This way any hop-long replacement path decomposes into short subpaths such that both endpoints are in B. To answer a hop-long query (s, t, F), a dense weighted graph H^F is created on the vertex set $V(H^F) = B \cup \{s, t\}$ such that for any two $u, v \in V(H^F)$ the edge $\{u, v\}$ has weight $\min_{G_i \in \mathcal{G}_F} d_{G_i}(u, v)$. Those edges represent the subpaths. The oracle's eventual answer to the query is the distance $d_{H^F}(s, t)$ in H^F.

The replacement distances reported by the DSO are exact w.h.p. However, this approach has several drawbacks. The most important one for us is that each of the graphs G_i has $\Omega(m)$ edges, raising the space to store them all to $\Omega(fL^f m)$, which is super-quadratic in n for dense graphs G. Also, the query time is rather high, the bottleneck is computing the weight of the $O(|B|^2) = \widetilde{O}(f^2n^2/L^2)$ edges of H^F for the hop-long paths.

The key observation for improving this result in graphs with a small hop diameter is that there *all* replacement paths are hop-short. Afek, Bremler-Barr, Kaplan, Cohen, and Merritt [1] showed that for undirected, weights graphs G and failure sets $F \subseteq E$ with $|F| \leqslant f$, every shortest path in $G - F$ is a concatenation of at most $f + 1$ shortest paths in G interleaved with at most f edges. So if D is a bound on the hop diameter of G, the hop diameter of $G - F$ is at most $L = (f{+}1)D + f$. With this definition of L, we can safely ignore hop-long replacement paths. Note that the assumption of G being undirected is essential here: The Afek et al. [1] result fails in directed graphs. Moreover, there is no hope for a subquadratic DSO in that case. Thorup and Zwick [34] showed that every data structure reporting pairwise distances in a directed graph must take $\Omega(n^2)$ space. This holds even if the data structure does not support a single edge failure and only provides an arbitrary finite approximation of the distance.

Nevertheless, we can use approximation in order to reduce the space of the DSO for undirected graphs. Instead of storing the subgraphs G_i, we replace them by the *distance oracle* (DO) of Thorup and Zwick [34]. For any positive integer k and G_i, we get a DO of size $O(kn^{1+1/k})$ that, when queried with two vertices s, t, reports the distance $d_{G_i}(s, t)$ but with a *stretch* of $2k - 1$. That means, the returned value $\widehat{d}(s, t)$ satisfies $d_{G_i}(s, t) \leqslant \widehat{d}(s, t) \leqslant (2k{-}1) \cdot d_{G_i}(s, t)$. While the use of more efficient data structures reduces the space of our DSO to $\widetilde{O}(fL^f n^{1+1/k})$, discarding the actual subgraphs G_i makes it impossible to recover the information which edges have been removed in which graph, that is, to compute the subfamily $\mathcal{G}_F$. We provide two different ways to solve this. The first one is to use spanners. The DO in [34] is accompanied by a spanner of the same size and we show that if the spanner associated with G_i does not contain an edge of F then it is safe to rely on G_i for the replacement distances, even if the graph itself has some failing edges from F.

Interestingly, the other solution comes from derandomization. Karthik and Parter [26] showed how to make the subgraph creation deterministic, albeit now with $O((cfL \log m)^{f+1})$ such graphs for some constant $c > 0$. This makes the resulting DSO less compact and also increases the preprocessing time. However, they presented a way to compute the now deterministic family $\mathcal{G}_F$ using error-correcting codes. This allows us to significantly improve the query time if the diameter is small. For this, we show how to implement the encoding procedure without using additional storage space.

We present our results in the following setting. We consider graphs with *polynomial edge weights*, meaning that they are edge-weighted by positive reals from a range of size $\mathsf{poly}(n)$, where n is the number of vertices. While the weights themselves may have arbitrary precision, the number of values that can be written as sums of at most n weights is again polynomial. Therefore, we can encode any graph distance in a constant number of $O(\log n)$-bit machine words. The restriction on the range is justified as follows. Let $W = \max_{e \in E} w(e)/\min_{e \in E} w(e)$ be the ratio between the maximum and minimum weight. Chechik et al. [15, Lemma 4.1] gave a reduction from approximate DSOs for general weighted graphs to approximate DSOs for graphs with polynomial weights that increases the space and preprocessing time only by a factor $O(\frac{\log W}{\log n})$, the query time by a factor $O(\log \log W)$, and the stretch by a factor $1 + \frac{1}{n}$.

In the statements below, k controls the stretch vs. space trade-off is an arbitrary positive integer, possibly even depending on the number of vertices n. However, there are only space improvements to be had for values $k = O(\log n)$.

Theorem 1. *Let $G = (V, E)$ be an undirected graph with polynomial edge weights, and hop diameter D. For all positive integers k and $f = o(\log n/\log\log n)$, there is an f-DSO for G that has stretch $2k-1$ and satisfies the following properties.*

1. *(Randomized.) The DSO takes space $D^f kn^{1+\frac{1}{k}+o(1)}$, has a preprocessing time of $D^f kmn^{\frac{1}{k}+o(1)}$, and answers queries correctly w.h.p. in time $D^f n^{o(1)}$.*
2. *(Deterministic.) The DSO takes space $D^{f+1}kn^{1+\frac{1}{k}+o(1)}$, has preprocessing time of $D^{f+1}kmn^{\frac{1}{k}+o(1)}$, and query time $O(f^3 D \frac{\log n \log\log n}{\log D})$.*

Corollary 1. *If G has a polylogarithmic hop diameter, then there is an f-DSO for G with stretch $2k-1$ that takes $kn^{1+\frac{1}{k}+o(1)}$ space, has a preprocessing time of $kmn^{\frac{1}{k}+o(1)}$, and $\widetilde{O}(1)$ query time.*

We also devise a solution for graphs with an arbitrarily large hop diameter. To do so, we have to compute the correct distances for hop-long replacement paths. In [35], this was the role of the dense subgraph H^F on the pivots in B. Imagine we would sparsify it using the spanner construction above. This would significantly reduce the number of edges we need and stretch the distance $d_{H^F}(s,t)$ to at most $2k-1$ times the correct replacement distance. But computing first the graph and then the distance would still take a lot of time. Instead, the idea of our solution is to prepare a spanner on vertex set B for each subgraph and to combine only those we need for the result. This way, we achieve both low memory and $o(n^{1+1/k})$ query time, as stated in the following theorem.

We remark again that Bilò et al. [6, Theorem 1.1] gave an oracle for unweighted graphs whose query time is sublinear, at the expense of the space being only marginally subquadratic.

Theorem 2. *Let $G = (V, E)$ be an undirected graph polynomial edge weights. For all positive integers k and $f = o(\log n/\log\log n)$, and every $0 < \alpha < 1$, there* *is an f-DSO for G with stretch $2k-1$, space $kn^{1+\alpha+\frac{1}{k}+o(1)}$, preprocessing time $kmn^{1+\alpha+\frac{1}{k}+o(1)}$, and query time $\widetilde{O}(n^{1+\frac{1}{k}-\frac{\alpha}{k(f+1)}})$.*

3 Preliminaries

Shortest paths and hop diameter. We let $G = (V, E)$ denote the undirected base graph with n vertices and m edges, edge-weighted by a function $w\colon E \to \mathcal{W}$, where the set of admissible weights $\mathcal{W} \subseteq \mathbb{R}^+$ is of size $|\mathcal{W}| = \mathsf{poly}(n)$. We tacitly assume $m = \Omega(n)$. For any undirected graph H (that may differ from the input G) we denote by $V(H)$ and $E(H)$ the set of its vertices and edges, respectively. Let P be a path in H from a vertex $s \in V(H)$ to $t \in V(H)$, we say that P is an *s-t-path* in H. We denote by $|P| = \sum_{e \in E(P)} w(e)$ the *length* of P, that is, its total weight. For vertices $u, v \in V(P)$, we let $P[u..v]$ denote the subpath of P from u to v. For two paths P, Q in H that share an endpoint, we use $P \circ Q$ for their concatenation. For $s, t \in V(H)$, the *distance* $d_H(s,t)$ is the minimum length of any s-t-path in H; if s and t are disconnected, we set $d_H(s,t) = +\infty$. When talking about the base graph G, we drop the subscripts if this does not create any ambiguities. The *hop diameter* of H is the maximum number of edges of any shortest path between pairs of vertices in $V(H)$.

Spanners and distance sensitivity oracles. A *spanner of stretch $\sigma \geqslant 1$*, or *σ-spanner*, for H is a subgraph $S \subseteq H$ such that for any two vertices $s, t \in V(S) = V(H)$, the inequality $d_H(s,t) \leqslant d_S(s,t) \leqslant \sigma \cdot d_H(s,t)$ holds. For a set $F \subseteq E$ of edges, let $G - F$ be the graph obtained from G by removing all edges in F. For any two $s, t \in V$, a *replacement path* $P(s,t,F)$ is a shortest path from s to t in $G - F$. Its length $d(s,t,F) = d_{G-F}(s,t)$ is the *replacement distance*. For a positive integer f, an *f-distance sensitivity oracle* (DSO) reports, upon query (s,t,F) with $|F| \leqslant f$, the replacement distance $d(s,t,F)$. It has *stretch $\sigma \geqslant 1$*, or is *σ-approximate*, if the reported value $\widehat{d}(s,t,F)$ satisfies $d(s,t,F) \leqslant \widehat{d}(s,t,F) \leqslant \sigma \cdot d(s,t,F)$ for any admissible query. We measure the space complexity of a data structure in the number of $O(\log n)$-bit machine words. The size of the input G does not count against the space unless it is stored explicitly.

Error-correcting codes. For a positive integer h, we set $[h] = \{0, 1, \ldots, h-1\}$. For positive integers q, p, and ℓ with $p \leqslant \ell$, a *code with alphabet size q, message length p, and block length ℓ* is a set $C \subseteq [q]^\ell$ such that $|C| \geqslant q^p$. An *encoding* for C is a computable injective mapping $[q]^p \to C$. Two codewords $x, y \in C$ have *(relative) distance* $\mathit{\Delta}(x,y) = |\{j \in [\ell] \mid x_j \neq y_j\}|/\ell$. For a positive real $\delta > 0$, code C is *error-correcting with (relative) distance δ*, if for any two $x, y \in C$, $\mathit{\Delta}(x,y) \geqslant \delta$. In this case, we say C is a $[p, \ell, \delta]_q$-code. It will be sufficient to focus on Reed-Solomon codes, which are $[p, q, 1 - \frac{p-1}{q}]_q$-codes for any $p \leqslant q$. When choosing q (and therefore $\ell = q$) as a power of 2 and $p < q$, there is an encoding algorithm for Reed-Solomon codes that takes $O(\ell \log p)$ time and $O(\ell)$ space using fast Fourier transform [30].

4 Small Hop Diameter

We first describe the simpler randomized version of our distance sensitivity oracle for graphs with small hop diameter. Afterwards, we derandomize it using more involved techniques like error-correcting codes. Throughout, we assume that the base graph G has edge weights from a polynomial-sized range.

4.1 Preprocessing

In the setting of Theorem 1, all shortest paths have at most D edges. Let $f = o(\log n/\log\log n)$ be the sensitivity of the oracle and $L \geqslant \max(f, 2)$ be an integer parameter which will be fixed later (depending on D). An (L, f)-*replacement path covering* (RPC) [26] is a family $\mathcal{G}$ of spanning subgraphs of G such that for any set $F \subseteq E$, $|F| \leqslant f$, and pair of vertices $s, t \in V$ such that s and t have a shortest path in $G - F$ on at most L edges, there exists a subgraph $G_i \in \mathcal{G}$ that does not contain any edge of F but an s-t-path of length $d(s, t, F)$. That means, some replacement path $P(s, t, F)$ from $G - F$ also exists in G_i. Let $\mathcal{G}_F \subseteq \mathcal{G}$ be the subfamily of all graphs that do not contain an edge of F. The definition of an RPC implies that if s and t have a replacement path w.r.t. F on at most L edges, then $\min_{G_i \in \mathcal{G}_F} d_{G_i}(s, t) = d(s, t, F)$.

To build the DSO, we first construct an (L, f)-RPC. This can be done by generating $|\mathcal{G}| = cfL^f \ln n$ random subgraphs for a sufficiently large constant $c > 0$. Each graph G_i is obtained from G by deleting any edge with probability $1/L$ independently of all other choices. As shown in [35], the family $\mathcal{G} = \{G_i\}_i$ is an (L, f)-RPC with high probability It is also easy to see using Chernoff bounds[6] that for any failure set F, $|\mathcal{G}_F| = O(|\mathcal{G}|/L^{|F|}) = \widetilde{O}(fL^{f-|F|})$.

We do not allow ourselves the space to store all subgraphs. We therefore replace each G_i by a distance oracle D_i, a data structure that reports, for any two $s, t \in V$, (an approximation of) the distance $d_{G_i}(s, t)$. For any positive integer k, Thorup and Zwick [34] devised a DO that is computable in time $\widetilde{O}(kmn^{1/k})$, has size $O(kn^{1+1/k})$, query time $O(k)$, and a stretch of $2k-1$. Roddity, Thorup, and Zwick [32] derandomized the oracle, and Chechik [12,13] reduced the query time to $O(1)$ and the space to $O(n^{1+1/k})$. Additionally, we store, for each G_i, a spanner S_i. The same work [34] contains a spanner construction with stretch $2k - 1$ that is compatible with the oracle, meaning that the oracle D_i reports exactly the value $d_{S_i}(s, t)$. The spanner is computable in time $\widetilde{O}(kmn^{1/k})$ and has $O(kn^{1+1/k})$ edges. We store it as a set of edges. There are static dictionary data structures known that achieve this in $O(kn^{1+1/k})$ space such that we can check in $O(1)$ worst-case time whether an edge is present or retrieve an edge. They can be constructed in time $\widetilde{O}(kn^{1+1/k})$ [25]. The total preprocessing time of the distance sensitivity oracle is $\widetilde{O}(|\mathcal{G}|m + |\mathcal{G}|kmn^{1/k}) = \widetilde{O}(fL^f kmn^{1/k})$ and it takes $\widetilde{O}(fL^f kn^{1+1/k})$ space.

[6] There is a slight omission in [35, Lemma 3.1] for $|\mathcal{G}_F|$ is only calculated for $|F| = f$.

4.2 Query Algorithm

Assume for now that the only allowed queries to the DSO are triples (s, t, F) of vertices $s, t \in V$ and a set $F \subseteq E$ of at most f edges such that any shortest path from s to t in $G - F$ has at most L edges. We will justify this assumption later with the right choice of L. The oracle has to report the replacement distance $d(s, t, F)$. Recall that $\mathcal{G}_F$ is the family of all graphs in $\mathcal{G}$ that have at least all edges of F removed. Since $\mathcal{G}$ is an (L, f)-RPC, all we have to do is compute (a superset of) $\mathcal{G}_F$ and retrieve (an approximation of) $\min_{G_i \in \mathcal{G}_F} d_{G_i}(s, t)$. The issue is that we do not have access to the graphs G_i directly.

The idea is to use the spanners S_i as proxies. This is justified by the next lemma that follows from a connection between the spanners and oracles presented in [34]. Let $D_i(s, t)$ denote the answer of the distance oracles D_i.

Lemma 1. *Let $G_i \in \mathcal{G}$ be a subgraph, S_i its associated spanner, and D_i its $(2k-1)$-approximate distance oracle. For any two vertices $s, t \in V$ and set $F \subseteq E$ with $|F| \leqslant f$, if $F \cap E(S_i) = \emptyset$, then $d(s, t, F) \leqslant D_i(s, t) \leqslant (2k-1)\, d_{G_i}(s, t)$.*

Let $\mathcal{G}^S = \{S_i\}_{i \in [r]}$ be the collection of spanners for all $G_i \in \mathcal{G}$, and $\mathcal{G}^S_F \subseteq \mathcal{G}^S$ those that do not contain an edge of F. Below, we hardly distinguish between a set of spanners (or subgraphs) and their indices, thus e.g. $S_i \in \mathcal{G}^S_F$ is abbreviated as $i \in \mathcal{G}^S_F$. Since $E(S_i) \subseteq E(G_i)$ and using the convention, we get $\mathcal{G}^S_F \supseteq \mathcal{G}_F$.[7] To compute $\mathcal{G}^S_F$, we cycle through all of $\mathcal{G}^S$ and probe each dictionary with the edges in F, this takes $O(f|\mathcal{G}|) = \widetilde{O}(f^2 L^f)$ time and dominates the query time. If $i \in \mathcal{G}^S_F$, then we query the distance oracle D_i with the pair (s, t) in constant time. As answer to the query (s, t, F), we return $\min_{i \in \mathcal{G}^S_F} D_i(s, t)$. By Lemma 1, the answer is at least as large as the sought replacement distance and, since there is a graph $G_i \in \mathcal{G}_F \subseteq \mathcal{G}^S_F$ with $d_{G_i}(s, t) = d(s, t, F)$, it is at most $(2k-1)\, d(s, t, F)$.

Let D be an upper bound on the hop diameter of G. As mentioned above, Afek et al. [1, Theorem 2] showed that the maximum hop diameter of all graphs $G - F$ for $|F| \leqslant f$ is bounded by $(f+1)D + f$. Using this value for L implies that indeed all queries admit a replacement path on at most L edges. For the DSO, it implies a preprocessing time of $\widetilde{O}(f L^f kmn^{1/k}) = \widetilde{O}(f^{f+1} D^f kmn^{1/k})$, which for $f = o(\log n / \log\log n)$ is $D^f kmn^{1/k + o(1)}$. The space requirement is $\widetilde{O}(f L^f kn^{1+1/k}) = \widetilde{O}(f^{f+1} D^f kn^{1+1/k}) = D^f kn^{1+1/k+o(1)}$, and the query time $\widetilde{O}(f^2 L^f) = \widetilde{O}(f^{f+2} D^f) = D^f n^{o(1)}$. This proves the first part of Theorem 1.

4.3 Derandomization

We now make the DSO deterministic via a technique by Karthik and Parter [26]. The derandomization will allow us to find the relevant subgraphs faster, so we do not need the spanners anymore. Recall that the distance oracles D_i were already

[7] We do mean here that $\mathcal{G}^S_F$ is a superset of $\mathcal{G}_F$. Since the spanner contain fewer edges than the graphs, F may be missing from $E(S_i)$ even though $F \cap E(G_i) \neq \emptyset$. This is fine as long as we take the *minimum* distance over all spanners from $\mathcal{G}^S_F$

derandomized in [32]. The only randomness left is the generation of the subgraphs G_i. Getting a deterministic construction offers an alternative approach to dealing with the issue that discarding the subgraphs for space reasons deprives us of the information which edges have been removed. Intuitively, we can now reiterate this process at query time to find the family $\mathcal{G}_F$. Below we implement this idea in a space-efficient manner.

We identify the edge set $E = \{e_0, e_1, \ldots, e_{m-1}\}$ with $[m]$. Let q be a positive integer. Assume that $p = \log_q m$ is integral, otherwise one can replace $\log_q m$ with $\lceil \log_q m \rceil$ without any changes. We interpret any edge $e_i \in E$ as a base-q number $(c_0, c_1, \ldots, c_{p-1}) \in [q]^p$ by requiring $i = \sum_{j=0}^{p-1} c_j q^j$. Consider an error-correcting $[p, \ell, \delta]$-code with distance $\delta > 1 - \frac{1}{fL}$ and (slightly abusing notation) let C be the $(m \times \ell)$-matrix with entries in $[q]$ whose i-th row is the codeword encoding the message $e_i = (c_0, c_1, \ldots, c_{p-1})$. The key contribution of the work by Karthik and Parter [26] is the observation that the *columns* of C form a family of hash functions $\{h_j \colon E \to [q]\}_{j \in [\ell]}$ such that for any pair of disjoint sets $P, F \subseteq E$ with $|P| \leqslant L$ and $|F| \leqslant f$ there exists an index $j \in [\ell]$ with $\forall x \in P, y \in F \colon h_j(x) \neq h_j(y)$.

An (L, f)-replacement path covering can be constructed from this as follows. Choose q as the smallest power of 2 greater[8] than $fL \log_L m$. Note that $q \leqslant \frac{2fL \log_2 m}{\log_2 L} \leqslant \frac{4fL \log_2 n}{\log_2 L}$. A Reed-Solomon code with alphabet size q, message length $p = \log_q m$, and block length $\ell = q$ has distance greater than $1 - \frac{1}{fL}$ [26, Corollary 18]. The resulting covering $\mathcal{G}$ consists of $|\mathcal{G}| = O(\ell \cdot q^f) = O(q^{f+1})$ subgraphs, each one indexed by a pair (j, S) where $j \in [\ell]$ and $S \subseteq [q]$ is a set with $|S| \leqslant f$. In the subgraph $G_{(j,S)}$, an edge e_i is removed if and only if $h_j(e_i) \in S$. It is verified in [26] that the family $\mathcal{G} = \{G_{(j,S)}\}_{j,S}$ is indeed an (L, f)-RPC. Moreover, for a fixed set $F = \{e_{i_1}, \ldots, e_{i_{|F|}}\}$ of edge failures, define $\mathcal{G}_F \subseteq \mathcal{G}$ to be the subfamily consisting of the graphs indexed by $(j, \{h_j(e_{i_1}), \ldots, h_j(e_{i_{|F|}})\})$ for each $j \in [\ell]$. Then, the construction ensures that no graph in $\mathcal{G}_F$ contains any edge of F and, for each pair of vertices $s, t \in V$ with an replacement path (w.r.t. F) on at most L edges, there is graph in $\mathcal{G}_F$ in which s and t are joined by a path of length $d(s, t, F)$. The ℓ graphs in $\mathcal{G}_F$ contain all the information we need for the short replacement distances with respect to the failure set F. The number of subgraphs in the covering is $O((4fL \log_L n)^{f+1})$, this is a factor $O((4f \log_L n)^f L)$ larger than what we had for the randomized variant.

In turn, we can make use of the extreme locality of the indexing scheme for $\mathcal{G}$. Since we chose q as a power of 2, we get the letters of the message $e_i = (c_0, c_1, \ldots, c_{p-1})$ by reading off blocks of $\log_2 q$ bits of the binary representation of i. The codeword of C corresponding to e_i is computable in time $O(p + \ell \log p) = O(fL(\log_L n) \log \log n)$ and space $O(p + \ell) = O(fL \log_L n)$ with the encoding algorithm of Lin, Al-Naffouri, Han, and Chung [30]. The whole matrix C and from it the family $\mathcal{G}$ can be generated in time $O(fLm(\log_L n) \log \log n + |\mathcal{G}| m) =$

[8] The original construction in [26] sets q as a prime number. We use a power of 2 instead to utilize the encoding algorithm in [30]. All statements hold verbatim for both cases.

$O((4fL\log_L n)^{f+1}m)$. Note that the codeword of e_i is $(h_1(e_i), h_2(e_i), \dots, h_\ell(e_i))$. So even after discarding C and the subgraphs, we can find the indices of graphs in $\mathcal{G}_F$ by encoding the edges of F in time $O(|F|\ell\log p) = O(f^2L(\log_L n)\log\log n)$ and rearranging the values into the ℓ sets $\{h_j(e_{i_1}), \dots, h_j(e_{i_{|F|}})\}$. In particular, using the algorithm in [30], we do not need to store the generator matrix of the code C.

The remaining preprocessing is similar as in Section 4.1, but we need neither the spanners nor the dictionaries anymore. We set $L = O(fD)$ again and, for each subgraph $G_{(j,S)}$, we only build the distance oracle $D_{(j,S)}$. This dominates the preprocessing time $\widetilde{O}(|\mathcal{G}|kmn^{1/k}) = \widetilde{O}(4^{f+1}f^{2f+2}D^{f+1}(\log_D n)^{f+1}kmn^{1/k})$ $= D^{f+1}kmn^{1/k+o(1)}$. The total size is now $\widetilde{O}(|\mathcal{G}|kn^{1+1/k}) = D^{f+1}kn^{1+1/k+o(1)}$. However, due to the derandomization, the query time is now much faster than before, in particular, polynomial in f, D and $\log n$. We do not have to cycle through all spanners anymore and instead compute $\mathcal{G}_F$ and query the DO only for those ℓ graphs. As a result, the time to report the replacement distance is $O(f^2L(\log_L n)\log\log n + \ell) = O(f^3D(\log_D n)\log\log n)$, completing Theorem 1.

5 Large Hop Diameter

We also devise a distance sensitivity oracle for graphs with an arbitrary hop diameter while maintaining a small memory footprint. For this, we have to handle hop-long replacement paths, that is, those that have more than L edges. We obtain a subquadratic-space distance sensitivity oracle with the same stretch of $2k-1$ but an $o(n^{1+1/k})$ query time. This is faster than computing the distance in any possible spanner.

5.1 Deterministic Pivot Selection

We say a query with vertices $s, t \in V$ and set $F \subseteq E$, $|F| \leqslant f$ has long replacement paths if every $P(s,t,F)$ has at least L edges. Those need to be handled in general DSOs. This is usually done by drawing a random subset $B \subset V$ of $\widetilde{O}(fn/L)$ pivots, as in [35], or essentially equivalent sampling every vertex independently with probability $\widetilde{O}(f/L)$ [23,33]. With high probability, B hits one path for every query with long replacement paths.

There are different approaches known to derandomize this depending on the setting [2,5,7,8,28]. In our case, we can simply resort to the replacement path covering to obtain $\mathcal{P}$ since we have to preprocess it anyway. We prove the following lemma for the more general class of arbitrary positive weights. Note that the key properties of an (L,f)-RPC remain in place as all definitions are with respect to the number of edges on the replacement paths. We make it so that B hits the slightly shorter paths with $L/2$ edges (instead of L). We are going to use this stronger requirement in Lemma 4.

Lemma 2. *Let $G = (V,E)$ be an undirected graph with positive edge weights. Let Q be the set of all queries (s,t,F), with $s,t \in V$ and $F \subseteq E$, $|F| \leqslant f$,* *for which every s-t-replacement path w.r.t. F has at least $L/2$ edges. Given an (L, f)-replacement path covering $\mathcal{G}$ for G, there is a deterministic algorithm that computes in time $\widetilde{O}(|\mathcal{G}|(mn+Ln^2/f))$ a set $B \subseteq V$ of size $\widetilde{O}(fn/L)$ such that, for all $(s,t,F) \in Q$, there is a replacement path $P = P(s,t,F)$ with $B \cap V(P) \neq \emptyset$. At the same time, one can build a data structure of size $O(|\mathcal{G}||B|^2)$ that reports, for every $G_i \in \mathcal{G}$ and $x, y \in B$, the distance $d_{G_i}(x,y)$ in constant time.*

5.2 Preprocessing

Our solution for large hop diameter builds on the deterministic DSO in Section 4.3. As for the case of a small hop diameter, we construct an (L, f)-replacement path covering $\mathcal{G}$ and, for each $G_{(i,S)} \in \mathcal{G}$, the distance oracle $D_{(i,S)}$. Recall that this part takes time $\widetilde{O}(|\mathcal{G}|kmn^{1/k})$ and $O(|\mathcal{G}|kn^{1+1/k})$ space.

We invoke Lemma 2 to obtain the set B. Additionally, for each subgraph, we build a complete weighted graph $H_{(i,S)}$ on the vertex set B where the weight of edge $\{x,y\}$ is $d_{G_{(i,S)}}(x,y)$, which we retrieve from the data structure mentioned in Lemma 2. We then compute a $(2k-1)$-spanner $T_{(i,S)}$ for $H_{(i,S)}$ with $O(k|B|^{1+1/k})$ edges via the same deterministic algorithm by Roditty, Thorup, and Zwick [32]. We store the new spanners for our DSO. The time to compute them is $\widetilde{O}(|\mathcal{G}|k|B|^{2+1/k})$ and, since $|B| = \widetilde{O}(fn/L)$, the preprocessing time is

$$\widetilde{O}(|\mathcal{G}|(kmn^{1/k} + |B|m + k|B|^{2+1/k})) = \widetilde{O}(|\mathcal{G}|(mn + k|B|^{2+1/k})) \leqslant L^{f+1}mn^{1+o(1)} + L^{f-1-1/k}kn^{2+1/k+o(1)} \leqslant L^{f+1}kmn^{1+1/k+o(1)}.$$

To obtain the bounds of Theorem 2, we set $L = n^{\frac{\alpha}{f+1}}$. Parameter $0 < \alpha < 1$ allows us to balance the space and query time. With this, we get a preprocessing time of $kmn^{1+\alpha+1/k+o(1)}$. and a space of $O(|\mathcal{G}|(kn^{1+1/k} + n + k|B|^{1+1/k}) = \widetilde{O}(|\mathcal{G}|kn^{1+1/k}) = L^{f+1}kn^{1+1/k+o(1)} = kn^{1+\alpha+1/k+o(1)}$.

5.3 Updated Query Algorithm

The algorithm to answer a query (s,t,F) starts similarly as before. We use the error-correcting codes to compute the subfamily $\mathcal{G}_F$ and the estimate $\widehat{d_1}(s,t,F) = \min_{(i,S)\in\mathcal{G}_F} D_{(i,S)}(s,t)$ is retrieved. However, this is no longer guaranteed to be an $(2k-1)$-approximation if the query is hop-long, i.e., if every shortest s-t-path in $G - F$ has at least L edges. It could be that no replacement paths survive and, in the extreme case, s and t are disconnected in each $G_{(i,S)} \in \mathcal{G}_F$, while they still have a finite distance in $G - F$. To account for long queries, we join all the spanners T_i for $i \in \mathcal{G}_F$. In more detail, we build a multigraph[9] H^F on the vertex set $V(H^F) = B \cup \{s,t\}$ whose edge set (restricted to pairs of pivots) is the disjoint union of all the sets $\{E(T_i)\}_{i\in\mathcal{G}_F}$ and, for each subgraph $(i,S) \in \mathcal{G}_F$ and pivot $x \in B$ contains the edges $\{s,x\}$ and $\{x,t\}$ with respective weights $D_{(i,S)}(s,x)$ and $D_{(i,S)}(x,t)$, where $D_{(i,S)}$ is the corresponding DO. The

[9] The multigraph is only used to ease notation.

oracle then computes the second estimate $\widehat{d}_2(s,t,F) = d_{H^F}(s,t)$ and returns $\widehat{d}(s,t,F) = \min\{\widehat{d}_1(s,t,F),\ \widehat{d}_2(s,t,F)\}$.

Lemma 3. *The distance sensitivity oracle has stretch $2k-1$ and the query takes time $\widetilde{O}(\frac{n^{1+1/k}}{L^{1/k}}) = \widetilde{O}(n^{1+\frac{1}{k}-\frac{\alpha}{k(f+1)}})$.*

Proving this lemma is enough to complete Theorem 2. In order to do so, we first establish the fact that $d_{H^F}(s,t)$ is a $(2k-1)$-approximation for $d(s,t,F)$ in case of a long query.

Lemma 4. *Let $s,t \in V$ be two vertices and $F \subseteq E$ a set of edges with $|F| \leqslant f$. It holds that $d(s,t,F) \leqslant d_{H^F}(s,t)$. If additionally every shortest s-t-path in $G-F$ has more than L edges, then we have $d_{H^F}(s,t) \leqslant (2k-1)\, d(s,t,F)$.*

Acknowledgements. The authors thank Merav Parter for raising the question of designing distance sensitivity oracles that require only subquadratic space.

This project received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Grant agreement No. 101019564 "The Design of Modern Fully Dynamic Data Structures (MoDynStruct)").

References

1. Afek, Y., Bremler-Barr, A., Kaplan, H., Cohen, E., Merritt, M.: Restoration by Path Concatenation: Fast Recovery of MPLS Paths. Distributed Computing **15**, 273–283 (2002). https://doi.org/10.1007/s00446-002-0080-6
2. Alon, N., Chechik, S., Cohen, S.: Deterministic Combinatorial Replacement Paths and Distance Sensitivity Oracles. In: Proceedings of the 46th International Colloquium on Automata, Languages, and Programming, (ICALP). pp. 12:1–12:14 (2019). https://doi.org/10.4230/LIPIcs.ICALP.2019.12
3. Baswana, S., Khanna, N.: Approximate Shortest Paths Avoiding a Failed Vertex: Near Optimal Data Structures for Undirected Unweighted Graphs. Algorithmica **66**, 18–50 (2013). https://doi.org/10.1007/s00453-012-9621-y
4. Bernstein, A., Karger, D.R.: Improved Distance Sensitivity Oracles via Random Sampling. In: Proceedings of the 19th Symposium on Discrete Algorithms (SODA). pp. 34–43 (2008), https://dl.acm.org/doi/abs/10.5555/1347082.1347087
5. Bernstein, A., Karger, D.R.: A Nearly Optimal Oracle for Avoiding Failed Vertices and Edges. In: Proceedings of the 41st Symposium on Theory of Computing (STOC). pp. 101–110 (2009). https://doi.org/10.1145/1536414.1536431
6. Bilò, D., Chechik, S., Choudhary, K., Cohen, S., Friedrich, T., Krogmann, S., Schirneck, M.: Approximate Distance Sensitivity Oracles in Subquadratic Space. In: Proceedings of the 55th Symposium on Theory of Computing (STOC) (2023). https://doi.org/10.1145/3564246.3585251, to appear.
7. Bilò, D., Choudhary, K., Cohen, S., Friedrich, T., Schirneck, M.: Deterministic Sensitivity Oracles for Diameter, Eccentricities and All Pairs Distances. In: Proceedings of the 49th International Colloquium on Automata, Languages, and Programming (ICALP). pp. 22:1–22:19 (2022). https://doi.org/10.4230/LIPIcs.ICALP.2022.22

8. Bilò, D., Cohen, S., Friedrich, T., Schirneck, M.: Near-Optimal Deterministic Single-Source Distance Sensitivity Oracles. In: Proceedings of the 29th European Symposium on Algorithms (ESA). pp. 18:1–18:17 (2021). https://doi.org/10.4230/LIPIcs.ESA.2021.18
9. Bodwin, G., Dinitz, M., Robelle, C.: Optimal Vertex Fault-Tolerant Spanners in Polynomial Time. In: Proceedings of the 32nd Symposium on Discrete Algorithms (SODA). pp. 2924–2938 (2021). https://doi.org/10.1137/1.9781611976465.174
10. Bodwin, G., Dinitz, M., Robelle, C.: Partially Optimal Edge Fault-Tolerant Spanners. In: Proceedings of the 33rd Symposium on Discrete Algorithms (SODA). pp. 3272–3286 (2022). https://doi.org/10.1137/1.9781611977073.129
11. Brand, J.v.d., Saranurak, T.: Sensitive Distance and Reachability Oracles for Large Batch Updates. In: Proceedings of the 60th Symposium on Foundations of Computer Science (FOCS). pp. 424–435 (2019). https://doi.org/10.1109/FOCS.2019.00034
12. Chechik, S.: Approximate Distance Oracles with Constant Query Time. In: Proceedings of the 46th Symposium on Theory of Computing (STOC). pp. 654–663 (2014). https://doi.org/10.1145/2591796.2591801
13. Chechik, S.: Approximate Distance Oracles with Improved Bounds. In: Proceedings of the 47th Symposium on Theory of Computing (STOC). pp. 1–10 (2015). https://doi.org/10.1145/2746539.2746562
14. Chechik, S., Cohen, S.: Distance Sensitivity Oracles with Subcubic Preprocessing Time and Fast Query Time. In: Proccedings of the 52nd Symposium on Theory of Computing (STOC). pp. 1375–1388 (2020). https://doi.org/10.1145/3357713.3384253
15. Chechik, S., Cohen, S., Fiat, A., Kaplan, H.: $(1+\varepsilon)$-Approximate f-Sensitive Distance Oracles. In: Proceedings of the 28th Symposium on Discrete Algorithms (SODA). pp. 1479–1496 (2017). https://doi.org/10.1137/1.9781611974782.96
16. Chechik, S., Langberg, M., Peleg, D., Roditty, L.: Fault Tolerant Spanners for General Graphs. SIAM Journal on Computing **39**, 3403–3423 (2010). https://doi.org/10.1137/090758039
17. Chechik, S., Langberg, M., Peleg, D., Roditty, L.: f-Sensitivity Distance Oracles and Routing Schemes. Algorithmica **63**, 861–882 (2012). https://doi.org/10.1007/s00453-011-9543-0
18. Demetrescu, C., Thorup, M.: Oracles for Distances Avoiding a Link-Failure. In: Proceedings of the 13th Symposium on Discrete Algorithms (SODA). pp. 838–843 (2002), https://dl.acm.org/doi/10.5555/545381.545490
19. Demetrescu, C., Thorup, M., Chowdhury, R.A., Ramachandran, V.: Oracles for Distances Avoiding a Failed Node or Link. SIAM Journal on Computing **37**, 1299–1318 (2008). https://doi.org/10.1137/S0097539705429847
20. Duan, R., Pettie, S.: Dual-Failure Distance and Connectivity Oracles. In: Proceedings of the 20th Symposium on Discrete Algorithms (SODA). pp. 506–515 (2009), https://dl.acm.org/doi/10.5555/545381.545490
21. Duan, R., Ren, H.: Maintaining Exact Distances under Multiple Edge Failures. In: Proceedings of the 54th Symposium on Theory of Computing (STOC). pp. 1093–1101 (2022). https://doi.org/10.1145/3519935.3520002
22. Erdős, P.: Extremal Problems in Graph Theory. Theory of Graphs and its Applications pp. 29–36 (1964)
23. Grandoni, F., Vassilevska Williams, V.: Faster Replacement Paths and Distance Sensitivity Oracles. ACM Transaction on Algorithms **16**, 15:1–15:25 (2020). https://doi.org/10.1145/3365835

24. Gu, Y., Ren, H.: 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). pp. 76:1–76:20 (2021). https://doi.org/10.4230/LIPIcs.ICALP.2021.76
25. Hagerup, T., Miltersen, P.B., Pagh, R.: Deterministic Dictionaries. Journal of Algorithms **41**, 69–85 (2001). https://doi.org/10.1006/jagm.2001.1171
26. Karthik C.S., Parter, M.: Deterministic Replacement Path Covering. In: Proceedings of the 32nd Symposium on Discrete Algorithms (SODA). pp. 704–723 (2021). https://doi.org/10.1137/1.9781611976465.44
27. Khanna, N., Baswana, S.: Approximate Shortest Paths Avoiding a Failed Vertex: Optimal Size Data Structures for Unweighted Graphs. In: Proceedings of the 27th Symposium on Theoretical Aspects of Computer Science (STACS). pp. 513–524 (2010). https://doi.org/10.4230/LIPIcs.STACS.2010.2481
28. King, V.: Fully Dynamic Algorithms for Maintaining All-Pairs Shortest Paths and Transitive Closure in Digraphs. In: Proceedings of the 40th Symposium on Foundations of Computer Science (FOCS). pp. 81–91 (1999). https://doi.org/10.1109/SFFCS.1999.814580
29. Levcopoulos, C., Narasimhan, G., Smid, M.H.M.: Efficient Algorithms for Constructing Fault-Tolerant Geometric Spanners. In: Proceedings of the 30th Symposium on Theory of Computing (STOC). pp. 186–195 (1998). https://doi.org/10.1145/276698.276734
30. Lin, S., Al-Naffouri, T.Y., Han, Y.S., Chung, W.: Novel Polynomial Basis with Fast Fourier Transform and Its Application to Reed-Solomon Erasure Codes. IEEE Transactions on Information Theory **62**, 6284–6299 (2016). https://doi.org/10.1109/TIT.2016.2608892
31. Ren, H.: Improved Distance Sensitivity Oracles with Subcubic Preprocessing Time. Journal of Computer and System Sciences **123**, 159–170 (2022). https://doi.org/10.1016/j.jcss.2021.08.005
32. Roditty, L., Thorup, M., Zwick, U.: Deterministic Constructions of Approximate Distance Oracles and Spanners. In: Proceedings of the 32nd International Colloquium on Automata, Languages and Programming (ICALP). pp. 261–272 (2005). https://doi.org/10.1007/11523468_22
33. Roditty, L., Zwick, U.: Replacement Paths and k Simple Shortest Paths in Unweighted Directed Graphs. ACM Transaction on Algorithms **8**, 33:1–33:11 (2012). https://doi.org/10.1145/2344422.2344423
34. Thorup, M., Zwick, U.: Approximate Distance Oracles. Journal of the ACM **52**, 1–24 (2005). https://doi.org/10.1145/1044731.1044732
35. Weimann, O., Yuster, R.: Replacement Paths and Distance Sensitivity Oracles via Fast Matrix Multiplication. ACM Transactions on Algorithms **9**, 14:1–14:13 (2013). https://doi.org/10.1145/2438645.2438646