Distributed Constructions of Dual-Failure Fault-Tolerant Distance Preservers

Merav Parter
Weizmann Institute of Science, Rehovot, Israel
merav.parter@weizmann.ac.il

Abstract

Fault tolerant distance preservers (spanners) are sparse subgraphs that preserve (approximate) distances between given pairs of vertices under edge or vertex failures. So-far, these structures have been studied thoroughly mainly from a centralized viewpoint. Despite the fact fault tolerant preservers are mainly motivated by the error-prone nature of distributed networks, not much is known on the distributed computational aspects of these structures.

In this paper, we present distributed algorithms for constructing fault tolerant distance preservers and +2 additive spanners that are resilient to at most *two edge* faults. Prior to our work, the only non-trivial constructions known were for the *single* fault and *single source* setting by [Ghaffari and Parter SPAA'16].

Our key technical contribution is a distributed algorithm for computing distance preservers w.r.t. a subset S of source vertices, resilient to two edge faults. The output structure contains a BFS tree $BFS(s, G \setminus \{e_1, e_2\})$ for every $s \in S$ and every $e_1, e_2 \in G$. The distributed construction of this structure is based on a delicate balance between the edge congestion (formed by running multiple BFS trees simultaneously) and the sparsity of the output subgraph. No sublinear-round algorithms for constructing these structures have been known before.

2012 ACM Subject Classification Networks → Network algorithms

Keywords and phrases Fault Tolerance, Distance Preservers, CONGEST

Digital Object Identifier 10.4230/LIPIcs.DISC.2020.21

Related Version `https://arxiv.org/abs/2010.01503`

Funding Partially funded by the ISF, grant no. 713130.

1 Introduction

Fault tolerant distance preservers are sparse subgraphs that preserve distances between given pairs of nodes under edge or vertex failures. In this paper, we present the first non-trivial distributed constructions of source-wise distance preservers and additive spanners that can handle *two* edge failures. We start by providing some background on fault-tolerant preservers from a graph-theoretical perspective, and then provide the distributed algorithmic context.

Fault-Tolerant Distance Preserves. Distances preservers are sparse subgraphs that preserve the distances between a given pairs of nodes *exactly*. As distance preservers are often computed for distributed networks where parts can spontaneously fail, a desired requirement for these applications is *fault-tolerance*. For every small set of edge failures, fault tolerant preservers are required to contain *replacement paths* around the faulted set. Formally, for a pair of vertices s and t and a subset of edge failures F, a replacement path $P(s, t, F)$ is an s-t shortest path in the surviving graph $G \setminus F$. The efficient (centralized) computation of all replacement path distances for a given s-t pair and a given source vertex s has attracted a lot of attention since the 80's [16, 15, 9, 22, 12, 24, 6, 1, 7]. Most of these works focus on the single-failure case, and relatively little is known on the complexity of distance preserving computation under multiple edge faults.

Parter and Peleg [19] introduced the notion of FT-BFS structure for a given source vertex s. Roughly speaking, an FT-BFS structure is a subgraph of the original graph that preserves all $\{s\} \times V$ distances under a single failure of an edge or a vertex. An FT-MBFS structure is collection of FT-BFS structures with respect to a collection of sources $S \subseteq V$. For every n-vertex graph $G = (V, E)$ and a source set $S \subseteq V$, [19] presented an algorithm for computing an FT-MBFS subgraph $H \subseteq G$ with $O(\sqrt{|S|}n^{3/2})$ edges. This was also shown to be existentially tight. Parter [17] presented a construction of dual-failure FT-BFS structures with $O(n^{5/3})$ edges. Gupta and Khan [13] extended this construction to multiple sources S and provided a dual-failure FT-MBFS with $O(|S|^{1/3}n^{5/3})$ edges, which is also existentially tight [19]. For a general bound on the number of fault f, the state-of-the-art upper bound is $O(|S|^{1/2^f}n^{2-1/2^f})$ by Bodwin et al. [3], a lower bound of $\Omega(|S|^{1/(f+1)}n^{2-1/(f+1)})$ is known by [17]. Closing this gap is a major open problem.

Fault-tolerant (FT) additive spanners are sparse subgraphs that preserve distance under failure with some additive stretch. While various upper bound constructions are known [4, 2, 18], to this date no lower bounds are known for constant additive stretch. For example, one can compute +2 FT-additive spanners with $\widetilde{O}(n^{5/3})$ edges[1], but no lower-bound is known for this structure.

Distributed Constructions. Despite the fact that the key motivation for fault tolerant preservers comes from distributed networks, considerably less is known on their distributed constructions. In this paper, we consider the standard CONGEST model of distributed computing [20]. In this model, the network is abstracted as an n-node graph $G = (V, E)$, with one processor on each node. Initially, these processors only know their incident edges in the graph, and the algorithm proceeds in synchronous communication rounds over the graph $G = (V, E)$. In each round, nodes are allowed to exchange $O(\log n)$ bits with their neighbors and perform local computation. Throughout, the diameter of the graph $G = (V, E)$ is denoted by D.

Ghaffari and Parter [11] presented the first distributed constructions of fault tolerant distance preserving structures. For every n-vertex D-diameter graph $G = (V, E)$ and a source vertex $s \in V$, they gave an $\widetilde{O}(D)$-round algorithm for computing an FT-BFS structure with respect to s. Both the size bound of the output structure and the round complexity of their algorithm are nearly optimal. An additional useful property of that algorithm is that it also computes the length of all the $\{s\} \times V$ replacement paths. To the best of our knowledge, currently there are no non-trivial distributed constructions that support either multiple sources or more than a single fault. A natural extension of [11] to a subset of sources S (dual faults) might lead to a round complexity of $\Omega(|S|D)$ (resp., $\Omega(D^2)$ rounds[2]). These bounds are inefficient for graphs with a large diameter, or for supporting a large number of sources. Finally, while distributed constructions for additive spanners are known [21, 5, 8], to this date there are no known distributed constructions of fault-tolerant additive spanners.

1.1 Our Results

In this paper we overcome the single-source and single-fault distributed barriers. We present constructions of FT preservers and additive spanners resilient to two edge failures with *sublinear* round complexities.

---

[1] The notation $\widetilde{O}$ hides poly-logarithmic terms in the number of vertices n.
[2] These bounds indeed seem to be achievable by slightly adapting the algorithm [11].

Fault Tolerant Distance Preservers. Given an unweighted and undirected n-vertex graph $G = (V, E)$ and integer $f \geq 1$, a subgraph $H \subseteq G$ is an f-FT-MBFS structure w.r.t. S if:

$$\text{dist}(s, t, H \setminus F) = \text{dist}(s, t, G \setminus F), \text{ for every } s \in S, t \in V, F \subseteq E \text{ and } |F| \leq f \ .$$

When $f = 1$, we call H an FT-MBFS structure, and when $f = 2$ it is called a dual-failure FT-MBFS.

▶ **Theorem 1** (Distributed FT-MBFS). *There exists a randomized algorithm that given an n-vertex graph $G = (V, E)$, and a subset $S \subseteq V$ computes w.h.p. a subgraph $H \subseteq G$ such that H is an FT-MBFS w.r.t. S and $|E(H)| = O(\sqrt{|S|}n^{3/2})$ edges. The round complexity is $\widetilde{O}(D + \sqrt{n|S|})$.*

This improves upon the $O(|S|D)$ bound implied by the FT-BFS algorithm of [11], for $D \geq \sqrt{n/|S|}$. The size of the FT-MBFS subgraph H is existentially optimal (up to a logarithmic factor).

We also consider the dual-failure setting. In the centralized literature it has been widely noted that the dual-failure case is already considerably more involved compared to the single fault setting. Indeed, there has been no prior distributed constructions of distance preserving subgraphs that are resilient to two faults. We provide a simplified centralized algorithm for the dual failure setting which serves the basis for our distributed construction:

▶ **Theorem 2** (Distributed Dual-Failure FT-MBFS). *There exists a randomized algorithm that given an n-vertex graph $G = (V, E)$, and a subset $S \subseteq V$ computes w.h.p. a subgraph $H \subseteq G$ such that H is a dual-failure FT-MBFS w.r.t. S and contains $O(|S|^{1/8} \cdot n^{15/8})$ edges. The round complexity is $\widetilde{O}(D + n^{7/8}|S|^{1/8} + |S|^{5/4}n^{3/4})$.*

We note that the size of our subgraph is suboptimal, as there exist (centralized) constructions [13, 17] that compute dual-failure FT-MBFS subgraphs with $O(|S|^{1/3}n^{5/3})$ edges. These constructions, however, are inherently sequential, and it is unclear how to efficiently implement them in the distributed setting. Specifically, in the CONGEST model, a naive simultaneous computation of multiple BFS trees $\texttt{BFS}(s, G \setminus \{e_1, e_2\})$ for every $s \in S$ and $e_1, e_2 \in G$ might result in a very large *congestion* over the graph edges. To reduce this congestion, one needs to balance the edge congestion and the sparsity of the output subgraph. These two opposing forces lead to suboptimal constructions w.r.t. the size, but with the benefit of obtaining a sub-linear round-complexity. We also note that our algorithms solve the subgraph problem rather than the distance computation problem. That is, in contrast to the FT-BFS algorithm of [11], we compute only the FT preserving subgraph but not necessarily the FT distances.

Fault Tolerant Additive Spanners. We employ the distributed construction of FT-MBFS structures to provide the first non-trivial constructions of fault tolerant $+2$ additive spanners. These structures are defined as follows. Given an unweighted undirected n-vertex graph $G = (V, E)$ and integer $f \geq 1$ and a stretch parameter β, a subgraph $H \subseteq G$ is an $+\beta$ f-FT additive spanner w.r.t. S if:

$$\text{dist}(s, t, H \setminus F) \leq \text{dist}(s, t, G \setminus F) + \beta, \text{ for every } s, t \in V, F \subseteq E \text{ and } |F| \leq f \ .$$

When $f = 1$, we call H an $+2$ FT-additive spanner, and when $f = 2$ it is called a $+2$ dual-failure FT-additive spanner. By using Thm. 1 and Thm. 2 respectively, we get:

▶ **Corollary 3** ($+2$ Additive Spanner, Single Fault). *For every n-vertex graph $G = (V, E)$, there exists a randomized algorithm that w.h.p. computes $+2$ FT-additive spanner $H \subseteq G$ with $\widetilde{O}(n^{5/3})$ edges in $O(D + n^{5/6})$ rounds.*

The size of the +2 FT-additive spanner matches the state-of-the-art bound. For the dual-failure setting, our bounds are suboptimal due to the suboptimality of the dual-failure FT-MBFS structures.

▶ **Corollary 4** (+2 Additive Spanner, Two Faults). *For every n-vertex graph $G = (V, E)$, there exists a randomized algorithm that w.h.p. computes a +2 dual-failure FT-additive spanner with $\widetilde{O}(n^{17/9})$ edges within $\widetilde{O}(D + n^{8/9})$ rounds.*

No sublinear round algorithms for +2 FT-additive spanners with $o(n^2)$-edges were known before.

The High-Level Approach. We provide the high-level ideas required to compute FT-MBFS structures w.r.t. a collection of source nodes S. This construction is later on used for computing FT-additive spanners and dual-failure FT-MBFS structures. By definition, an FT-MBFS structure for S is required to contain a BFS tree w.r.t. each $s \in S$ in every graph $G \setminus \{e\}$. Upon using any consistent tie-breaking of shortest-path distances, the union of all these trees contain $O(|S|n^{3/2})$ edges [19]. Our goal is then to compute all these BFS trees efficiently in the CONGEST model. For the single source case, the key observation made in [11] is that for every vertex t, it is sufficient to send only $O(D)$ BFS tokens throughout the computation: one token for every edge e on the shortest s-t path $\pi(s,t)$. The reason is that a failing of an edge $e \notin \pi(s,t)$ does not effect the s-t distance. Since every edge (u,t) is required to pass through only $|\pi(s,t)| = O(D)$ BFS tokens, using the random-delay approach, all these trees could be computed in dilation+congestion$= \widetilde{O}(D)$ rounds, w.h.p. Extending this idea to multiple sources, ultimately leads to a round complexity of $\Omega(D|S|)$. Indeed a-priori it is unclear how to break this barrier, as for every s and $e \in \pi(s,t)$, the s-t distance in $G \setminus \{e\}$ might be different, forcing t to receive the BFS token from $\Omega(D|S|)$ BFS algorithms. Our key idea is to define for every vertex t a smaller set of *relevant pairs* (s,e) from which it is allowed to receive the BFS tokens. This set $\{(s,e)\}$ is defined by including only edges e that are sufficiently *close* to t on its $\pi(s,t)$ path for every s. The main technical issue that arises with this idea is the inconsistency in the definition of relevant pairs between nodes on a given replacement path $P(s,t,e)$. In particular, there might be cases where (s,e) is relevant for t but it is not in the relevant set of some vertex w on the $P(s,t,e)$ path. In such a case, w might block the propagation the BFS token $BFS(s, G \setminus \{e\})$ (as (s,e) is not in its relevant set) which would prevent t from receiving it. These technical issues become more severe in the dual failure setting, due to a considerably more delicate interaction between the dual-failure replacement paths. In the very high level to mitigate this problem, we add to the output structure a collection of an (FT-) BFS trees w.r.t. a *randomly* sampled set of nodes. This edge set would compensate (in a non-trivial manner) for the lost tokens of the truncated BFS constructions.

1.2 Preliminaries

Given an unweighted n-vertex graph $G = (V, E)$, for an $s, t \in V$ and $e \in G$, the replacement path $P(s,t,e)$ is an s-t shortest path in $G \setminus \{e\}$. Throughout, we assume that the shortest-path ties are broken in a consistent manner using the vertex IDs. For a given $p \in [0,1]$, let $\texttt{Sample}(V,p)$ be a subset of vertices obtained by sampling each vertex in V independently with probability p. Let $\texttt{BFS}(s,G)$ be a BFS tree rooted at s in G. Throughout, the (unique) shortest-path between any pair x, y in G is denoted by $\pi(x,y)$. Let $N(u)$ be the neighbors of u in G. Given a tree T and $u, v \in V(T)$, let $\pi(u,v,T)$ be the tree path between u and v. Throughout, all shortest-path ties are broken in a consistent manner, by preferring vertices

of lower IDs. That is, in every BFS computation (which defines the shortest-path) every vertex picks its parent to be the vertex of minimum ID among all its potential parents in the tree. For a given integer parameter σ, let $\pi_\sigma(u,v)$ denote the set of last $\min\{\sigma, |\pi(u,v)|\}$ edges (closest to v) on the path $\pi(u,v)$. For an edge $e=(x,y)$ and a subgraph $G' \subseteq G$, let $\text{dist}(e,t,G') = \min\{\text{dist}(x,t,G'), \text{dist}(y,t,G')\}$. For an s-t path P, let $\texttt{LastE}(P)$ be the last edge of the path (incident to t). For $a,b \in P$, let $P[a,b]$ be the sub-path segment between a and b in P.

▶ **Definition 5.** *For a given source vertex s, a subgraph H is an* FT-BFS structure *with respect to s if* $\text{dist}(s,t,H\setminus\{e\}) = \text{dist}(s,t,G\setminus\{e\})$ *for every $t \in V$ and $e \in E$. In the same manner, for a given subset $S \subseteq V$, a subgraph H is an* multi-source FT-BFS structure *with respect to S if* $\text{dist}(s,t,H\setminus\{e\}) = \text{dist}(s,t,G\setminus\{e\})$ *for every $s,t \in S \times V$ and $e \in E$.*

▶ **Fact 6** ([19])**.** *For every n-vertex graph $G=(V,E)$, and a subset $S \subseteq V$, let*

$$H = \bigcup_{s,t,e \in S \times V \times E} \{\texttt{LastE}(P(s,t,e))\}.$$

Then, H is an FT-MBFS *structure w.r.t. S and $|E(H)| = O(\sqrt{|S|} \cdot n^{3/2})$. This edge bound is existentially tight.*

The random delay Technique. Throughout, we make an extensive use of the random delay approach of [14, 10]. Specifically, we use the following theorem:

▶ **Theorem 7** ([10, Theorem 1.3])**.** *Let G be a graph and let $A_1,\ldots,A_m$ be m distributed algorithms in the* CONGEST *model, where each algorithm takes at most $\texttt{d}$ rounds, and where for each edge of G, at most $\texttt{c}$ messages need to go through it, in total over all these algorithms. Then, there is a randomized distributed algorithm (using only private randomness) that, with high probability, produces a schedule that runs all the algorithms in $O(\texttt{c}+\texttt{d}\cdot\log n)$ rounds, after $O(\texttt{d}\log^2 n)$ rounds of pre-computation.*

2 Simplified Meta-Algorithms

We start by presenting simplified (centralized) constructions of FT preserving subgraphs. This would serve as a more convenient starting point for the distributed constructions of these structures.

FT-MBFS Structures. Let $T_S = \bigcup_{s\in S} T_s$ where $T_s = \texttt{BFS}(s,G)$. The FT-MBFS subgraph H is given by the union of three subgraphs: T_S, and the subgraphs H_1 and H_2 defined by:

$$H_1 = \{\texttt{LastE}(P(s,t,e)) \;\mid\; s,t \in S\times V, e \in \pi_\sigma(s,t,T_s)\} \text{ where } \sigma = \sqrt{n/|S|},$$

and $H_2 = \bigcup_{r\in R} \texttt{BFS}(r,G)$, where $R = \texttt{Sample}(V, 10\log n/\sigma)$.

▶ **Lemma 8.** *$E(H) = O(\sqrt{|S|}\cdot n^{3/2}\log n)$ and H is an* FT-MBFS *with respect to S.*

Proof. The size analysis follows by noting that $|T_S| = O(|S|\cdot n)$, and in addition each vertex t adds at most $\sigma = \sqrt{n/|S|}$ edges to H_1 for every source $s \in S$. Thus, $|H_1| = O(\sigma\cdot|S|n) = O(n\sqrt{n|S|})$. Turning to H_2, by the Chernoff bound, w.h.p., $|R| = O(n\log n/\sigma)$ and thus $|H_2| = O(\sqrt{|S|}\cdot n^{3/2}\log n)$.

We next show that H is an FT-MBFS with respect to S. By Fact 6, it is sufficient to show that H contains the last edge of the replacement path $P(s,t,e)$ for every $s,t,e \in S\times V\times E$.

Fix a source $s \in S$ and a vertex $t \in V$. If $\text{dist}(s,t,G) \leq \sigma$, then $H_1 \cup T_S$ contain the last edge of $P(s,t,e)$ for every edge e. This is because $\texttt{LastE}(P(s,t,e))$ is added to H_1 for every $e \in \pi(s,t,T_s)$ and $P(s,t,e) = \pi(s,t,T_s)$ for every $e \notin \pi(s,t,T_s)$.

Thus, assume that $\text{dist}(s,t,G) \geq \sigma$ and specifically, consider an edge $e \in \pi(s,t,T_s) \setminus \pi_\sigma(s,t,T_s)$. Since $\text{dist}(s,t,G) \geq \sigma$, it also holds that $|P(s,t,e)| \geq \sigma$. Thus by the Chernoff bound, w.h.p. there is at least one vertex in R that lies in the $(\sigma/2)$-length suffix of $P(s,t,e)$. That is, w.h.p., there is a vertex $r \in V(P(s,t,e)) \cap R$ such that $\text{dist}(r,t,P(s,t,e)) \leq \sigma/2$. We next claim that there is *no* r-t shortest path in G that contains the failing edge e. This holds as $\text{dist}(r,t,G) \leq \text{dist}(r,t,G \setminus \{e\}) \leq \sigma/2$, but by the definition of the edge e, $\text{dist}(e,t,G) \geq \sigma$. By the uniqueness of the replacement paths, we have that $P(s,t,e)[r,t] = \pi(r,t,T_r)$ where $T_r = \texttt{BFS}(r,G)$, and thus $\texttt{LastE}(P(s,t,e)) \in H_2$. The claim follows. ◀

FT-MBFS Structures for 2 Faults. We next describe a simplified centralized construction of dual-failure FT-MBFS structures, this serves the basis for the distributed implementation. As we will see later on, computing these structures in the distributed setting is considerably more involved. To balance between edge congestion and sparsity of the structure, the final size of the FT-MBFS structures computed in distributed setting is larger compared to the centralized setting. For every $s,t \in V$ and $e_1,e_2 \in E$, let $P(s,t,\{e_1,e_2\})$ be the s-t shortest path in $G \setminus \{e_1,e_2\}$.

▶ **Fact 9** ([17]). *For every n-vertex graph $G=(V,E)$, and a subset $S \subseteq V$, let*

$$H = \bigcup_{s,t \in S \times V, e_1,e_2 \in E} \{\texttt{LastE}(P(s,t,\{e_1,e_2\}))\}.$$

Then H is a dual-failure FT-MBFS w.r.t. S.

Let S be the set of sources. Let R_1 be a random sample of $O(\sqrt{n|S|}\log n)$ vertices, and let R_2 be a random sample of $O(|S|^{1/4} \cdot n^{3/4} \log n)$ vertices. Let $H_1 = \bigcup_{r \in R_1} \texttt{FT-MBFS}(r,G)$ and $H_2 = \bigcup_{r \in R_2} \texttt{BFS}(r,G)$. The dual-failure FT-BFS structure w.r.t. S denoted by $\texttt{FT-BFS}_2(S)$ contains H_1, H_2 and the a subset of last edges of certain replacement paths. Let $\sigma_1 = \sqrt{n/|S|}$ and $\sigma_2 = (n/|S|)^{1/4}$. For every path P and integer σ, let P_σ be the σ-length suffix of P (when $\sigma \geq |P|$, then P_σ is simply P). Every vertex t, define the edge set E_t as

$$E_t = \bigcup_{s \in S} \bigcup_{e_1 \in \pi_{\sigma_1}(s,t)} \bigcup_{e_2 \in P_{\sigma_2}(s,t,e_1)} \{\texttt{LastE}(P(s,t,\{e_1,e_2\}))\} .$$

The final dual-failure FT-BFS structure is given by:

$$\texttt{FT-BFS}_2(S) = H_1 \cup H_2 \cup \bigcup_t E_t .$$

▶ **Lemma 10.** *$\texttt{FT-BFS}_2(S)$ is a dual-failure FT-BFS of S and contains $\widetilde{O}(|S|^{1/4} \cdot n^{7/4})$ edges.*

Proof. By the definition of the E_t sets, it remains to show that H contains the last edge of an replacement path $P(s,t,\{e_1,e_2\})$ such that either (i) $e_1 \in \pi(s,t) \setminus \pi_{\sigma_1}(s,t)$ or (ii) $e_1 \in \pi_{\sigma_1}(s,t)$ but $e_2 \in P(s,t,e_1) \setminus P_{\sigma_2}(s,t,e_1)$. We begin with (i). Since $e_1 \in \pi(s,t) \setminus \pi_{\sigma_1}(s,t)$, we have that $|P(s,t,e_1)| \geq \sqrt{n/|S|}$. Since we sample each vertex into R_1 with probability of $10 \log n \cdot \sqrt{|S|/n}$, w.h.p. the $\sigma_1/2$-length suffix of $P(s,t,\{e_1,e_2\})$ contains a vertex, say r, in R_1. Since $\text{dist}(r,t,G) \leq \sigma_1/2$, we have that $e_1 \notin \pi(r,t,G)$, and by the uniqueness of the shortest paths, we have that $P(s,t,\{e_1,e_2\}) = P(s,t,\{e_1,e_2\})[s,r] \circ P(r,t,\{e_2\})$. Since H_2 contains

the FT-BFS w.r.t. r, it contains the path $P(r,t,\{e_2\})$ and thus $\texttt{LastE}(P(s,t,\{e_1,e_2\}))$ is in $\texttt{FT-BFS}_2(S)$. We proceed with (ii). Since $e_2 \in P(s,t,e_1) \setminus P_{\sigma_2}(s,t,e_1)$, we have that $|P(s,t,\{e_1,e_2\})| \geq \sigma_2$. Since we sample each vertex into R_2 with probability of $10\log n/\sigma_2$, w.h.p. the $\sigma_2/2$-length suffix of $P(s,t,\{e_1,e_2\})$ contains a vertex, say r', in R_2. Since $\text{dist}(r,t,G) \leq \sigma_2/2$, we have that $e_1,e_2 \notin \pi(r,t,G)$. By the uniqueness of the shortest paths, we have that $P(s,t,\{e_1,e_2\}) = P(s,t,\{e_1,e_2\})[s,r] \circ \pi(r,t,G)$. The claim follows as H_1 contains the BFS tree rooted at r, and concluding that $\texttt{LastE}(P(s,t,\{e_1,e_2\}))$ is in $\texttt{FT-BFS}_2(S)$. The size bound follows by noting that $|E(H_1)| = O(|\sqrt{|R_1|}n^{3/2})$ and $|E(H_2)| = O(|R_2|n)$. In addition, since each vertex t adds the last edges of $O(|S|\cdot(n/|S|)^{3/4})$ replacement paths, we get that $|E_t| = O(|S|^{1/4}n^{3/4})$. The lemma follows. ◀

Comparison to Bodwin et al. [3]. A simplified algorithm for computing sparse FT-MBFS structures (of suboptimal size) has been also provided by [3]. Their algorithm iterates over the vertices where for every vertex t the algorithm defines a small set of edges incident to t that should be added to the output subgraph H. For every *vertex* t, the algorithm reduces the task of computing FT-MBFS structure with respect to S sources and supporting f faults[3] into the computation of an FT-MBFS structure to support S' sources and $f-1$ faults, where $|S'| = O(\sqrt{|S|n})$. The main limitation in implementing this algorithm in the distributed setting is that for each vertex t the algorithm defines a *distinct* set of sources. For $f=1$ for example, our simplified algorithm computes BFS trees w.r.t. a subset of sources S'. In contrast, in the algorithm of [3], a BFS tree is computed w.r.t. a distinct set of sources S_t for every vertex t, the union of all these S_t sets might be very large (leading to a large round complexity).

3 Distributed Construction of FT-MBFS Structures

In this section we prove Thm. 1 and present our main algorithm for computing sparse FT-MBFS structures with respect to S sources. This structure becomes useful both for the construction of FT-additive spanners, and for the computation of the dual-failure FT preservers.

3.1 The algorithm

Set $\sigma = \lceil\sqrt{n/|S|}\rceil$ and $\sigma' = 3\sigma$. The algorithm has two main steps. In the first step, a subset R of $O(n\log n/\sigma)$ vertices is uniformly sampled, and a BFS trees $T_s = \texttt{BFS}(s,G)$ is computed for every vertex $s \in S \cup R$. Let $T_S = \bigcup_{s\in S} T_s$ and $T_R = \bigcup_{r\in R} T_r$. All the edges of $T_R \cup T_S$ are added to the output subgraph H, by their corresponding endpoints.

In the second step, the algorithm computes a special subset of replacement paths, and the last edges of these replacement paths will be added to H. To define this subset, we need the following definition. For every $s,t \in S \times V$, each vertex t defines a set of *relevant edge-list* $\pi_{\sigma'}(s,v)$ that consists of the last σ' edges of its $\pi(s,v)$ paths. It also defines a shorter prefix $\pi_{\sigma}(s,v)$ that contains the last σ edges of this path.

The algorithm first lets each vertex t learn its relevant edge-list $\pi_{\sigma'}(s,t)$ for every $s \in S$. This can be done within $O(|S|\cdot\sigma' + D) = O(\sqrt{n|S|} + D)$ rounds by applying a simple pipeline strategy. From now on, the algorithm divides the time into phases of $\ell = O(\log n)$ rounds. Every $\texttt{BFS}(s, G\setminus\{e\})$ algorithm then starts in phase $\tau_{s,e}$, where $\tau_{s,e}$ is a random

[3] The task is to pick the edges of t that should be added to such a structure

variable with a uniform distribution in $[1, \sigma' \cdot |S|]$. Specifically, using the notion of k-wise independence, similarly to [11], all vertices can learn a random seed of $\mathcal{SR}$ of $O(\log n)$ bits. Using the seed $\mathcal{SR}$ and the IDs of the edge e and the source s, all vertices can compute the starting phase $\tau_{s,e}$ of each BFS construction $\texttt{BFS}(s, G \setminus \{e\})$. In the analysis section, we show that due to these random starting points, w.h.p., each edge $e' = (u, v)$ is required to send as most ℓ edges in every phase. In a standard application of a BFS computation with delay $\tau_{s,e}$, every vertex t is supposed to receive a $\texttt{BFS}(s, G \setminus \{e\})$-token (for the first time) in phase $\text{dist}_{G\setminus\{e\}}(s, t) + \tau_{s,e}$. In our case, the algorithm cannot afford to compute the entire BFS trees, but rather only certain fragments of them. Specifically, the BFS tokens are initiated and propagated following certain rules whose goal is to keep the congestion over the edges small. In the high-level, every vertex t would send its neighbor $u \in N(t)$ (such that $(u, t) \neq e$) a BFS token $\texttt{BFS}(s, G \setminus \{e\})$ only if $e \in \pi_{\sigma'}(s, u)$. In the special case where $e \notin \pi(s, t)$, the vertex t will initiate the BFS token to u in phase $\text{dist}_{G\setminus\{e\}}(s, t) + \tau_{s,e}$. In the remaining case where $e \in \pi(s, t)$, t will send u the token $\texttt{BFS}(s, G \setminus \{e\})$ to u in phase i iff (i) $e \in \pi_{\sigma'}(s, u)$ and t received the token $\texttt{BFS}(s, G \setminus \{e\})$ for the first time in phase $i - 1$.

As we will see in the analysis section, even-though each vertex t sends the BFS tokens $\texttt{BFS}(s, G \setminus \{e\})$ to neighbors u provided that $e \in \pi_{\sigma'}(s, u)$, it might be the case that a vertex u would not get the BFS token for each of its edges in $\pi_{\sigma'}(s, u)$. This might happen when the path $P(s, u, e)$ contains intermediate vertices w for which $e \notin \pi_{\sigma'}(s, w)$, which would block the propagation of the token. Fortunately, a more careful look reveals that in all the cases where $\texttt{LastE}(P(s, u, e)) \notin T_R$, the BFS token of $\texttt{BFS}(s, G \setminus \{e\})$ would complete its propagation over the entire $P(s, u, e)$ for every $e \in \pi_\sigma(s, u) \subseteq \pi_{\sigma'}(s, u)$. As we will see, this would be sufficient for the correctness of the `FT-MBFS` structure.

Second-Order Implementation Details. For the generation of the shared random seed, we use the same construction of [11] which is based on the notion of k-wise independence hash functions.

▶ **Lemma 11** ([11]). *The string of shared randomness SR can be generated and delivered to all vertices in $O(D + \log n)$ rounds.*

We argue that each vertex v by knowing the sets $\bigcup_{u \in N(v) \cup \{v\}} \pi_{\sigma'}(s, u)$, can locally compute the edges in $\pi_{\sigma'}(s, u) \setminus \pi(s, v)$ for every $u \in N(v)$.

▶ **Lemma 12.** *For every vertex v, neighbor $u \in N(v)$ and an edge $e \in \pi_{\sigma'}(s, u)$, v can locally decide if $e \in \pi(s, v)$ or not.*

Proof. Let $e \in \pi_{\sigma'}(s, u) \cap \pi(s, v)$. We will show that e can locally recognize that $e \in \pi(s, v)$. If $e \in \pi_{\sigma'}(s, v)$, then v clearly knows that $e \in \pi(s, v)$. Otherwise, if $e = (x, y) \in \pi_{\sigma'}(s, u) \setminus \pi_{\sigma'}(s, v)$, we show that y must be the endpoint of the first edge in $\pi_{\sigma'}(s, v)$. To see this, assume towards contradiction that y has no incident edge in $\pi_{\sigma'}(s, v)$. Since $e \in \pi_{\sigma'}(s, u)$, we have that $\text{dist}(x, u, G) \leq 3\sigma$. However, by the assumption, $\text{dist}(x, v, G) \geq 3\sigma + 2$, in contradiction as (u, v) are neighbors. As $y \in V(\pi_{\sigma'}(s, v))$ and $e = (x, y) \in \pi_{\sigma'}(s, u)$, v can deduce that x is the parent of y in T_s, and consequently that $(x, y) \in \pi(s, v)$ as well. ◀

Note that vertex u receives messages from all its potential parents in $\texttt{BFS}(s, G \setminus \{e\})$ at the same time, namely, at phase $\text{dist}(s, u, G \setminus \{e\}) + \tau_{s,e}$. It selects as its parent the vertex of minimum ID, which would guarantee that the shortest path ties are broken an a consistent manner, leading to a sparse structure.

Algorithm 1 The Distributed FT-MBFS Algorithm

1. Sample a set $R \subset V$ of $O(n \log n/\sigma)$ vertices uniformly at random from V.
2. Construct a BFS tree $T_s = \texttt{BFS}(G, s)$ rooted at s for every $s \in S \cup R$, and add these trees to H.
3. Number the edges of $T_S = \bigcup_{s \in S} T_s$ by numbers 1 to $|S|(n-1)$, where each edge $e \in T_s$ has a distinct number for every s containing $e \in T_s$.
4. Make each vertex v know the numbers of the edges on the σ'-length suffix $\pi_{\sigma'}(s, v)$ for every $s \in S$.
5. Let each vertex v send to each of its neighbors $u \in N(v)$ the numbers of the edges on $\bigcup_{s \in S} \pi_{\sigma'}(s, v)$.
6. Broadcast a string $\mathcal{SR}$ of $O(\log n)$ random bits.
7. For every $s \in S$, and $e \in T_s$, let $\tau_{s,e}$ be picked uniformly at random from $\{1, 2, \ldots, \sigma' \cdot |S|\}$ by setting it equal to $\mathcal{SR}[i]$ where i is the edge-number of e in T_s. Since $\mathcal{SR}$ is publicly known, given the ID of an edge e, a vertex can compute $\tau_{s,e}$ for every s.
8. Divide time into phases of $\ell = \Theta(\log n)$ rounds each.
9. Run each $\texttt{BFS}(G \setminus \{e\}, s)$ for every e and $s \in S$ at a speed of one hop per phase, following these rules for every vertex v:
   - For every edge $e \in \pi_{\sigma'}(s, u) \setminus \pi(s, v)$, and[4] every neighbor $u \in N(v)$, v sends u the BFS token $\texttt{BFS}(s, G \setminus \{e\})$ in phase $\text{dist}(s, v, G) + \tau_{s,e}$.
   - For every BFS token $BFS(s, G \setminus \{e\})$ received for the first time at phase i at v from a non-empty subset of neighbors $N'(v) \subseteq N(v)$, v does the following:
     - If $e \in \pi_\sigma(s, v)$, then v adds the edge (w, v) to H where w is the vertex of minimum-ID in $N'(u)$.
     - v sends the BFS token $\texttt{BFS}(s, G \setminus \{e\})$ in phase $i+1$ to every neighbor $u \in N(v) \setminus N'(v)$ satisfying that $e \in \pi_{\sigma'}(s, u)$.

Correctness. Unlike the single-source case, where the correctness of the algorithm was immediate by construction, here the correctness argument is more delicate. Specifically, in the FT-BFS construction of [11], for every vertex v, the BFS token $\texttt{BFS}(s, G \setminus \{e\})$ reached every vertex t for which $e \in \pi(s, t)$. In contrast, in our setting, only a subset of the replacement paths are fully constructed which poses a challenge for showing the correctness.

To show that the output subgraph H is indeed an FT-MBFS w.r.t. S, throughout, we fix a source $s \in S$, target $t \in V$ and an edge $e = (x, y)$. We need the following definitions. Let T_s be a BFS tree rooted at s for every $s \in S$. For a vertex y and a tree T_s, let $T_s(y)$ be the subtree rooted at y in T_s. A vertex w is said to be *sensitive* to an edge $e \in T_s$, if $e \in \pi(s, w)$. Observe that for every edge $e = (x, y) \in T_s$, where x is closer to s, the set of sensitive vertices to e are those that belong to $T_s(y)$.

▶ **Definition 13** (sensitive-detour). *For a given replacement path $P(s, t, e)$ let w be the first vertex on the path (closest to s) that is sensitive to e. We denote the segment $SD(s, t, e) = P(s, t, e)[w, t]$ by the* sensitive-detour *of $P(s, t, e)$.*

▶ **Observation 14.** *For every $s, t \in S \times V$ and $e = (x, y) \in G$, it holds that: (i) $SD(s, t, e) \subseteq T_s(y)$ and (ii) $P(s, t, e) = \pi(s, w') \circ (w', w) \circ SD(s, t, e)$ for a unique pair $w, w' \in P(s, t, e)$.*

Proof. (i) Let w be the first vertex in $T_s(y) \cap P(s, t, e)$, thus $SD(s, t, e) = P(s, t, e)[w, t]$. Assume towards contradiction that there exists $w' \in SD(s, t, e)$ such that $w' \notin T_s(y)$. Since

the shortest-paths are computed in a consistent manner, and $e \notin \pi(s,w')$, we get that $P(s,t,e)[s,w'] = \pi(s,w')$. Thus, $w \in \pi(s,w')$, contradiction as $w \in T_s(y)$.

(ii) Let w' be the neighbor of w (defined as above) on $P(s,t,e)$ that is closer to s. By definition, $w' \notin T_s(y)$ and thus by the uniqueness of the shortest-paths, we have $P(s,t,e) = \pi(s,w') \circ (w',w) \circ SD(s,t,e)$. ◀

▷ Claim 15. If $e \in \pi_{\sigma'}(s,w')$ for every $w' \in SD(s,t,e)$, then $\texttt{LastE}(P(s,t,e)) \in H$.

Proof. Let w be the first vertex on $SD(s,t,e)$ and let q be the preceding neighbor of w (not in $SD(s,t,e)$). Since $e \in \pi_{\sigma'}(s,w)$, the vertex q can locally detect that $e \notin \pi(s,q)$ (using Lemma 12). Note that since $q \notin SD(s,t,e)$, it holds $\text{dist}(s,q,G \setminus \{e\}) = \text{dist}(s,q,G)$. Thus, q send to w the BFS token $\texttt{BFS}(s, G \setminus \{e\})$ in phase $\text{dist}(s,q,G \setminus \{e\}) + 1 + \tau_{s,e}$. The token propagates over the $SD(s,t,e)$ segment at a speed of one hop per phase as for each $w' \in SD(s,t,e)$, $e \in \pi_{\sigma'}(s,w')$. ◁

▶ **Lemma 16.** *For every $s,t \in S \times V$ and $e \in G$, we have that $\texttt{LastE}(P(s,t,e)) \in H$.*

Proof. Fix a replacement path $P(s,t,e)$ where $e = (x,y)$. We consider the following cases.

Case (1): $e \in \pi(s,t) \setminus \pi_\sigma(s,t)$. In this case, $|P(s,t,e)| \geq \text{dist}(s,t,G) \geq \sigma$ and thus w.h.p. the $\sigma/2$-length suffix of the path contains at least one sampled vertex in R, say r. Since $\text{dist}(r,t,G) \leq \sigma/2$ but $\text{dist}(x,t,G) \geq \sigma$, the edge e does not appear on any r-t shortest path. As the shortest-path ties are broken in a consistent manner, we have that $P(s,t,e)[r,t] = \pi(r,t)$. Since the algorithm adds the BFS trees w.r.t. all vertices in R, we have that $\texttt{LastE}(\pi(r,t)) \in H$.

Case (2): $e \in \pi_\sigma(s,t)$ but $|SD(s,t,e)| \geq \sigma$. The proof for this case follows by noting that for every two vertices $u,v \in T_s(y)$, there is no u-v shortest path that go through the edge e. Assume towards contradiction that there is a u-v shortest path P that goes through e, since $\pi(x,u) \subset \pi(s,u)$, $\pi(x,v) \subset \pi(s,v)$, it holds that $e \in \pi(x,u), \pi(x,v)$, and thus:

$$|P| = \text{dist}_G(u,x) + \text{dist}(x,v) = 1 + \text{dist}_G(y,u) + 1 + \text{dist}_G(y,v) = 2 + \text{dist}(u,v) \ ,$$

contradiction that P is a u-v shortest path. Since $|SD(s,t,e)| \geq \sigma$, w.h.p., it contains at least one sampled vertex $r \in R$. As both $r,t \in T_s(y)$, $\pi(r,t)$ is free of failed edge e. Thus $P(s,t,e)[r,t] = \pi(r,t)$, concluding that $\texttt{LastE}(P(s,t,e)) \in H$. We note that this is the only case where the proof would not work for the case of a single vertex (rather than edge) fault.

Case (3): $e \in \pi_\sigma(s,t)$ but $|SD(s,t,e)| < \sigma$. This is the most interesting case as the last edge of the path $P(s,t,e)$ is not necessarily in $\bigcup_{r \in R} T_r$. We need to show that the suffix of the path $P(s,t,e)$ is computed by the algorithm, and that its last edge is added to H. Since $|SD(s,t,e)| < \sigma$, it holds that $\text{dist}(w,t,G \setminus \{e\}) \leq \sigma$ where w is the first vertex on the $SD(s,t,e)$ segment. Since $e \in \pi_\sigma(s,t)$, it holds that $\text{dist}(e,w,G \setminus \{e\}) \leq 2\sigma$. Finally, as $w \in SD(s,t,e)$ it implies that $e \in \pi_{\sigma'}(s,w)$. The claim then follows by Claim 15. ◀

Size. The first part adds the BFS trees w.r.t. $|R| = O(\sqrt{|S|n})$ vertices. In addition, in the gradual BFS constructions, for every edge $e \in \bigcup_{s \in S} \pi_\sigma(s,v)$, the vertex v adds at most one edge to H (corresponding to the last edge of $P(s,v,e)$). Since $|\bigcup_{s \in S} \pi_\sigma(s,v)| = O(\sqrt{|S|n})$, this adds $O(\sqrt{|S|}n^{3/2})$ edges.

Round Complexity.

▷ Claim 17. Each vertex t can learn the relevant edge set $\bigcup_{s\in S}\pi_{\sigma'}(s,t)$ within $O(\sqrt{n|S|})$ rounds.

Proof. For every $s\in S$, each edge e in T_s propagates down the tree for σ' time steps (i.e., until reaching all vertices at distance σ' from e). Focusing on a single-source s, each edge $e'=(x,y)$ needs to pass at most σ' messages, corresponding to the last σ' edges on the $\pi(s,y)$ path. Since there are $|S|$ sources, the total number of messages passing through a single edge is $|S|\cdot\sigma'$. Using pipeline all these messages can arrive in $O(\sqrt{n|S|})$ rounds. ◁

▶ **Lemma 18.** *W.h.p., at most $\ell=O(\log n)$ BFS tokens need to go through each edge, per phase.*

Proof. We show that w.h.p., in each phase number τ and for each edge $e'=(v,u)$, at most $O(\log n)$ BFS tokens will need to go through e' from v to u in phase τ. Note that the only BFS tokens passing over the edge $e'=(v,u)$ correspond to the BFS algorithms of $\texttt{BFS}(s,G\setminus\{e\})$ for $e\in\pi_{\sigma'}(s,u)\cup\pi_{\sigma'}(s,v)$. Thus each edge passes $O(|S|\cdot\sigma)$ tokens.

Each of the $O(|S|\cdot\sigma)$ permitted tokens passing through e' from v to u in phase τ satisfies that $\mathrm{dist}(s,v,G\setminus\{e\})+\tau_{s,e}=\tau$. Assuming that the starting phase $\tau_{s,e}$ is chosen uniformly at random from a range of size $3\sigma|S|$, the probability of this event is at most $1/(3\sigma\cdot|S|)$. Hence, over the set $3\sigma\cdot|S|$ permitted tokens, only 1 token, in expectation, is scheduled to go through from v to u in phase τ. If the random delay values $\tau_{s,e}$ were completely independent, by an application of the Chernoff bound, we would have that this number is at most $O(\log n)$, w.h.p. This will not be exactly true in our case, as we produce $\mathcal{SR}$ using a pseudo-random generators, but using k-wise independence on the generated string, for $k=\Theta(\log n)$ and from a result of Schmidt et al.[23], it is known that for this application of the Chernoff bound, it suffices to have k-wise independence between the random values, for $k=\Theta(\log n)$. ◀

We are now ready to complete the round complexity argument. By Claim 17, each vertex t computes its relevant edge set $\pi_{\sigma'}(s,t)$ within $O(D+\sqrt{|S|n})$ rounds. Within additional $O(\sqrt{|S|n})$ rounds, each vertex t can also learn the relevant edge sets of its neighbors. By Lemma 18, the computation of all BFS trees is implemented within $\widetilde{O}(D+\sigma\cdot|S|)=\widetilde{O}(D+\sqrt{|S|n})$ rounds. This completes the proof of Theorem 1.

4 Distributed Construction of Dual Failure Distance Preservers

In this section, we extend the construction of $\texttt{FT-MBFS}$ structures to support two edge failures. Throughout, For every $s,t\in S\times V$, and $e_1,e_2\in G$, recall that $P(s,t,\{e_1,e_2\})$ is the unique s-t path in $G\setminus\{e_1,e_2\}$ chosen based on a consistent tie-breaking scheme (based on vertex IDs). For a given parameter σ, let $P_\sigma(s,t,F)$ be the σ-length suffix (ending at t) of the path. When $|P_\sigma(s,t,F)|\leq\sigma$, $P_\sigma(s,t,F)$ is simply $P(s,t,F)$. Set

$$\sigma_1=(n/|S|)^{5/8}\quad\text{and}\quad\sigma_2=(n/|S|)^{1/4}\ .$$

We start by describing the algorithm based on the assumption that every vertex t has the following information:

- (I1) The distance $\mathrm{dist}(s,t,e)$ for every $s\in S$ and every $e\in\pi_{2\sigma_2}(s,t)$.
- (I2) The path segment $P_{\sigma_2}(s,t,e)$ for every $s\in S$ and every $e\in\pi_{2\sigma_2}(s,t)$.

Note that in contrast to the FT-BFS construction of [11], the FT-MBFS algorithm of Theorem 1 computes the structure but not necessarily the distances. We therefore need to augment the algorithm by a procedure that computes the information (I1,I2) for all near faults (at distance σ_2 from t).

▶ **Lemma 19.** *There is a randomized algorithm that w.h.p. computes the information (I1,I2) for every vertex t within $\widetilde{O}((n/\sigma_1)\cdot\sigma_2+(n/\sigma_1)^2+D)$ rounds.*

In Subsec. 4.1 we describe the key construction. Then in Subsec. 4.2, we prove Lemma 19.

4.1 Distributed Alg. for Dual Failure FT-MBFS Structure (Under Assumption)

Before explaining the algorithm, we need the following definition, which extends Def. 13 to the dual failure setting.

▶ **Definition 20** (sensitive-detour of a Dual-Fault Replacement Path)**.** *A vertex t is* sensitive *to the triplet (s,e_1,e_2) if $P(s,t,\{e_1,e_2\})\notin\{P(s,t,e_1),P(s,t,e_2)\}$. This necessarily implies that for a sensitive vertex it holds that $e_2\in P(s,t,e_1)$ and $e_1\in P(s,t,e_2)$. For a given $P(s,t,\{e_1,e_2\})$ path, let w be the first vertex (closest to s) that is sensitive to (s,e_1,e_2). The sensitive detour $SD(s,t,\{e_1,e_2\})$ correspond to the segment $P(s,t,\{e_1,e_2\})[w,t]$.*

Set $\sigma=\sigma_2=(n/|S|)^{1/4}$. The first step of the algorithm computes an FT-MBFS subgraph FT-MBFS$(R\cup S)$ where R is a randomly sampled set of $O(n\log n/\sigma)$ vertices. By Thm. 1, this can be done in $\widetilde{O}(\sqrt{|R|n}+D)$ rounds. The second step computes a subset of dual-failure replacement paths $\{P(s,t,\{e_1,e_2\}), s\in S,t\in V,e_1,e_2\in E\}$ that satisfy certain properties. As in the single failure case, the computation of many of the replacement paths might be incomplete. The guarantee, however, would be that any $P(s,t,\{e_1,e_2\})$ replacement path whose last edge is not in FT-MBFS$(R\cup S)$ is fully computed by the algorithm. The set of replacement paths which the algorithm attempts to compute is defined by:

$$Q_t=\{(s,e_1,e_2)\ \mid\ e_1\in P_\sigma(s,t,e_2)\ \text{ and }\ e_2\in P_\sigma(s,t,e_1),\ s\in S\}.$$

By the assumption (I1,I2), each vertex t knows the last 2σ edges of the path $P(s,t,e)$ for every $s\in S$ and every $e\in\pi_{2\sigma}(s,t)$, it can compute Q_t (see Claim 22). Observe that $|Q_t|=|S|\cdot\sigma^2$. The algorithm starts by letting each vertex exchange its Q_t set with its neighbors. The BFS tokens BFS$(s,G\setminus\{e_1,e_2\})$ are permitted to pass from a vertex u to a vertex v only if $(s,e_1,e_2)\in Q_v$. To control the congestion due to the simultaneous constructions of multiple BFS trees, the vertices share a random string $\mathcal{SR}$. Each BFS algorithm BFS$(s,G\setminus\{e_1,e_2\})$ for every $e_1,e_2\in G$ and $s\in S$ starts in phase τ_{s,e_1,e_2} chosen uniformly at random in the range $\{1,\ldots,\Theta(|S|\cdot\sigma^2)\}$. Using the seed $\mathcal{SR}$ and the IDs of s,e_1,e_2, each vertex can compute τ_{s,e_1,e_2}. These τ_{s,e_1,e_2} values are $O(\log n)$-wise independent.

Each BFS algorithm BFS$(s,G\setminus\{e_1,e_2\})$ then starts in phase τ_{s,e_1,e_2}, and proceeds in a speed of one hop per phase. Each phase consists of $\ell=\Theta(\log n)$ rounds. The rules for passing the BFS tokens of BFS$(s,G\setminus\{e_1,e_2\})$ are as follows:

- Each vertex v that is not sensitive[5] to e_1,e_2 sends the token BFS$(s,G\setminus\{e_1,e_2\})$ in round $\tau_{s,e_1,e_2}+\text{dist}(s,v,G\setminus\{e_1,e_2\})$ to every neighbor $u\in N(v)$ satisfying that $(s,e_1,e_2)\in Q_u$.

[5] In the analysis, we show that in the case where there is $u\in N(v)$ for which $(s,e_1,e_2)\in Q_u$, v can indeed detect that it is not sensitive.

- Every vertex v that is sensitive to (s, e_1, e_2) upon receiving the first BFS token $\texttt{BFS}(s, G \setminus \{e_1, e_2\})$ in phase i does as follows:
  - Let w be the minimum-ID vertex in $N(v)$ from which v has received the BFS token in that phase. Then, v adds the edge (w, v) to the output structure H.
  - v sends the token $\texttt{BFS}(s, G \setminus \{e_1, e_2\})$ in phase $i+1$ to every neighbor $u \in N(v)$ satisfying that $(s, e_1, e_2) \in Q_u$.

This completes the description of the algorithm.

Analysis. Let $Q'_t \subset Q_t$ be defined by $Q'_t = \{(s, e_1, e_2) \mid e_1 \in P_{\sigma/2}(s, t, e_2) \text{ and } e_2 \in P_{\sigma/2}(s, t, e_1),\ s \in S\}$. Let w be the first sensitive vertex (see Def. 20) w.r.t. (s, e_1, e_2) on the replacement path $P(s, t, \{e_1, e_2\})$. Recall that $SD(s, t, \{e_1, e_2\}) = P(s, t, \{e_1, e_2\})[w, t]$ is the sensitive detour of $P(s, t, \{e_1, e_2\})$.

▶ **Observation 21.** *Any vertex $w' \in SD(s, t, \{e_1, e_2\})$ is sensitive to the two edges e_1, e_2.*

Proof. Recall that w is the first sensitive vertex on $P(s, t, \{e_1, e_2\})$, and thus the first vertex of the sensitive detour. Assume towards contradiction, that there exists a vertex $w' \in SD(s, t, \{e_1, e_2\})$ that is not sensitive to (s, e_1, e_2). Let $P \in \{P(s, w', e_1), P(s, w', e_2)\}$ be such that $P = P(s, w', \{e_1, e_2\})$. By the uniqueness of the shortest paths, we have that $P(s, t, \{e_1, e_2\})[s, w'] = P \circ P(s, t, \{e_1, e_2\})[w', t]$. We then have that $w \in P$ and thus $P[s, w] = P(s, w, \{e_1, e_2\})$ contradiction that w is the first sensitive vertex on $P(s, t, \{e_1, e_2\})$. ◀

▷ Claim 22. By knowing (I1) and (I2), each vertex t can compute the set Q_t.

To prove the correctness of the output structure, by Fact 9 we need to show that $\texttt{LastE}(P(s, t, \{e_1, e_2\}))$ is in H for every $s \in S$ and every $e_1, e_2 \in E$. Throughout, we consider a fixed replacement path $P(s, t, \{e_1, e_2\})$ and assume w.l.o.g. that $e_1 \in \pi(s, t)$ and $e_2 \in P(s, t, e_1)$.

▶ **Lemma 23.** *For every $(s, e_1, e_2) \notin Q'_t$ it holds that $\texttt{LastE}(P(s, t, \{e_1, e_2\})) \in \texttt{FT-MBFS}(R)$.*

To complete the correctness argument, it remains to show that $\texttt{LastE}(P(s, t, \{e_1, e_2\})) \in H$ for every $(s, e_1, e_2) \in Q'_t$. We do it in two steps, depending on the length of the sensitive-detour.

▷ Claim 24. Let $(s, e_1, e_2) \in Q'_t$. If $|SD(s, t, \{e_1, e_2\})| \geq \sigma/3$, then $\texttt{LastE}(P(s, t, \{e_1, e_2\})) \in H$.

For now on, we consider $P(s, t, \{e_1, e_2\})$ paths such that $(s, e_1, e_2) \in Q'_t$ and with a short sensitive detour, i.e., $|SD(s, t, \{e_1, e_2\})| \leq \sigma/3$. We first show the following.

▷ Claim 25. Let $(s, e_1, e_2) \in Q'_t$ and $|SD(s, t, \{e_1, e_2\})| \leq \sigma/3$. Then, $(s, e_1, e_2) \in Q_{w'}$ for every $w' \in SD(s, t, \{e_1, e_2\})$.

Proof. Fix $w' \in SD(s, t, \{e_1, e_2\})$. Since the detour is short it holds that $\text{dist}(w', t, G \setminus \{e_1, e_2\}) \leq \sigma/3$ for every $w' \in SD(s, t, \{e_1, e_2\})$. In addition, since $e_2 \in P_{\sigma/2}(s, t, e_1)$, we have that

$$\text{dist}(e_2, w', G \setminus \{e_1\}) \leq \text{dist}(e_2, t, G \setminus \{e_1\}) + \text{dist}(t, w', G \setminus \{e_1, e_2\}) \leq \sigma\ . \tag{1}$$

As w' is sensitive, it holds that $e_2 \in P(s, w', e_1)$, and combining with Eq. (1) we have that $e_2 \in P_\sigma(s, w', e_1)$. In the same manner, since $e_1 \in P_{\sigma/2}(s, t, e_2)$ and $e_1 \in P(s, w', e_2)$, by the same reasoning we have that $e_1 \in P_\sigma(s, w', e_2)$. We conclude that $(s, e_1, e_2) \in Q_{w'}$. ◁

We next show that the BFS token $\texttt{BFS}(s, G\setminus\{e_1,e_2\})$ arrives each vertex $w' \in SD(s,t,\{e_1,e_2\})$ in phase $\text{dist}(s,w',G\setminus\{e_1,e_2\}) + \tau_{s,e_1,e_2}$. Since for every vertex $w' \in SD(s,t,\{e_1,e_2\})$ it holds that $(s,e_1,e_2) \in Q_{w'}$, it is guaranteed that the BFS token $\texttt{BFS}(s, G\setminus\{e_1,e_2\})$ arriving w' in $SD(s,t,\{e_1,e_2\})$ in phase i is sent to the next hop $w'' \in SD(s,t,\{e_1,e_2\})$ in phase $i+1$. Therefore it is sufficient to show that the first vertex, say w, on the sensitive-detour $SD(s,t,\{e_1,e_2\})$ receives the token $\texttt{BFS}(s, G\setminus\{e_1,e_2\})$ in phase $\text{dist}(s,w',G\setminus\{e_1,e_2\}) + \tau_{s,\{e_1,e_2\}}$. Let q be the neighbor of w on $P(s,t,\{e_1,e_2\})$ not in $SD(s,t,\{e_1,e_2\})$.

▷ Claim 26. q sends to w (first vertex on the sensitive-detour) the BFS token $\texttt{BFS}(s, G\setminus\{e_1,e_2\})$ in phase $\text{dist}(s,w,G\setminus\{e_1,e_2\}) + \tau_{s,e_1,e_2}$.

▶ **Corollary 27.** *For every path $P(s,t,\{e_1,e_2\})$ satisfying that (i) $(s,e_1,e_2) \in Q'_t$ and (ii) $|SD(s,t,\{e_1,e_2\})| \leq \sigma/3$, it holds that the detour $SD(s,t,\{e_1,e_2\})$ is fully computed by the algorithm (i.e., the BFS token propagates through all the vertices on the sensitive detour). Consequently, $\texttt{LastE}(P(s,t,\{e_1,e_2\})) \in H$.*

Round Complexity. We next analyze the round complexity. The computation of the structure $\texttt{FT-MBFS}(R\cup S)$ takes $O(\sqrt{(|R|+|S|)n}+D) = \widetilde{O}(n^{7/8}\cdot|S|^{1/8})$ rounds. Running the partially computed BFS trees $\texttt{BFS}(s, G\setminus\{e_1,e_2\})$ takes in total $\widetilde{O}(D+(\sigma_2)^2\cdot|S|)$. Combining with the round complexity of Lemma 19 yields the desired bound of $\widetilde{O}(D + |S|^{5/4}n^{3/4} + |S|^{1/8}n^{7/8})$.

Size. The total number of edges in $\texttt{FT-MBFS}(R\cup S)$ is bounded by $O(\sqrt{|R|+|S|}\cdot n^{3/2})$. In addition, each vertex t adds at most $|Q_t| = O(|S|\cdot\sigma^2)$ edges to H. Plugging $\sigma = (n/|S|)^{1/4}$ and $|R| = O(n\log n/\sigma)$ yields the desired edge bound of $\widetilde{O}(|S|^{1/8}n^{15/8})$.

4.2 Learning Distances and Short RP Segments of Near Faults

In this subsection we fill in the missing piece of the algorithm by proving Lemma 19, and thus establishing Theorem 2. The computation of the information (I1,I2) for every vertex t is done in two key steps depending on the structure of the $P(s,t,e)$ path.

A replacement-path $P(s,t,e)$ for $e \in \pi_{\sigma_2}(s,t)$ is said to be *easy* if $|SD(s,t,e)| \leq \sigma_1$. Otherwise, the path $P(s,t,e)$ for $e \in \pi_{\sigma_2}(s,t)$ is *hard*.

Computing the information for easy replacement paths. We will present a somewhat stronger algorithm that computes (I1,I2) for every $P(s,t,e)$ paths satisfying that $e \in \pi_{\sigma_1}(s,t)$ (rather than just $e \in \pi_{\sigma_2}(s,t)$). The algorithm simply applied the second step of the single-failure $\texttt{FT-MBFS}$ algorithm with parameter $\sigma = 8\sigma_1$. Recall that in this phase, a partial collection of replacement paths is computed which is characterized by the given parameter σ. By the proof of Lemma 16 (Case (3)), we have that each t knows $\text{dist}(s,t,e)$ for every *easy* replacement path. It therefore remains for it to learn also the σ_2-length suffix of these paths. We next show that this can be done by a simple extension of the algorithm.

▷ Claim 28. Within extra $\widetilde{O}(D+\sigma_1\cdot\sigma_2\cdot|S|)$ rounds, every vertex t can learn the σ_2-length suffix of every easy replacement path $P(s,t,e)$, $e \in \pi_{\sigma_1}(s,t)$ for every $s \in S$.

Computing the information for hard replacement paths. It remains to consider the hard replacement paths $P(s,t,e)$. I.e., paths for which $e \in \pi_{\sigma_2}(s,t)$ and their sensitive-detour is of length at least σ_1. (Unlike the previous algorithm, here we might not learn the distances $\text{dist}(s,t,G\setminus\{e\})$ for edges $e \in \pi_{\sigma_1}(s,t)\setminus\pi_{\sigma_2}(s,t)$.) We assume here that this step is applied already computing the information for the easy replacement paths.

Let R be a random sample of $O(n \log n/\sigma_1)$ vertices. The algorithm computes BFS trees $T_r = \texttt{BFS}(r, G)$ for every $r \in R$. In addition, each vertex also learns its last σ_2 edges on each $\pi(r, t, T_r)$ paths. Using the random delay approach, this can be done in $\widetilde{O}(D+(n \log n/\sigma_1)\cdot\sigma_2)$ rounds.

▶ **Lemma 29.** *One can compute LCA (Least Common Ancestor) labels in each BFS tree T_s, $s \in S$ in total time $\widetilde{O}(D + S)$. The size of each LCA label is $O(\log^2 n)$ bits (per tree T_s).*

Consider an hard replacement-path $P(s, t, e)$ and let q be the neighbor before w on the path, where w is the first sensitive vertex on $P(s, t, e)$. Let $e = (x, y)$. We claim the following, see Fig. 1 for an illustration.

▷ Claim 30. For every hard replacement path $P(s, t, e)$, there must be two vertices r_1, r_2 such that (i) $\text{dist}(r_1, r_2, G) \leq \sigma_1/16$, (ii) r_1 is not sensitive to e and r_2 is sensitive to e and (iii) $e \notin \pi(r_1, r_2)$.

Proof. We claim that for every vertex w' appearing on the $(\sigma_1/8)$-length prefix of $SD(s, t, e)$ it holds that $e \notin \pi_{\sigma_1/8}(s, w')$. Assume towards contradiction otherwise, since $e \in \pi_{\sigma_1/8}(s, w')$ and $e \in \pi_{\sigma_2}(s, t)$ (and $\sigma_2 << \sigma_1$), the tree path between w' and t in T_s is free from e and has length at most $\sigma_1/4$. As $\text{dist}(w', t, G \setminus \{e\}) = |SD(s, t, e)[w', t'] \geq \sigma_1/2$, we end with a contradiction.

Next, let w be the first vertex on $SD(s, t, e)$ and let q be the vertex that appears just before w on $P(s, t, e)$. By the uniqueness of the shortest-path, $P(s, t, e) = \pi(s, q) \circ P(s, t, e)[q, t]$. We now claim that $|\text{dist}(s, q, G)| \geq \sigma_1/8-1$. Since $e \notin \pi_{\sigma_1/8}(s, w)$, it implies that $|\text{dist}(s, w, G)| \geq \sigma_1/8$ concluding that $|\text{dist}(s, q, G)| \geq \sigma_1/8 - 1$.

Therefore the $\sigma_1/32$ suffix of $\pi(s, q)$ contains a vertex $r_1 \in R$ that is not sensitive to e. The $\sigma_1/32$ prefix of the sensitive detour $SD(s, t, e)$ contains a vertex $r_2 \in R$ that is sensitive to e. Since the distance between r_1, r_2 on $P(s, t, e)$ is at most $\sigma_1/16$ and since $\text{dist}(e, r_2, G) \geq \sigma_1/8$, we conclude that $\text{dist}(r_1, r_2, G) = \text{dist}(r_1, r_2, G \setminus \{e\})$. ◁

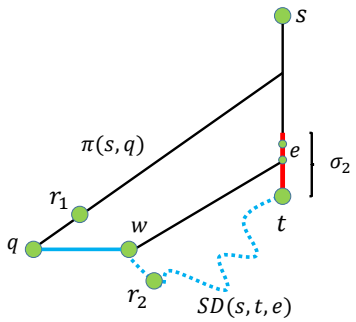

Figure 1 An illustration for the proof of Claim 30. Shown in an hard $P(s, t, e)$ path where $e = (x, y) \in \pi_{\sigma_2}(s, t)$. The vertex w is the first vertex on the sensitive detour, thus the entire $P(s, t, e)[w, t]$ is contained in the vertex set of $T_s(y)$, where is the subtree of T_s rooted at y. Dashed edges correspond to the path segment $SD(s, t, e)$. Since e is very close to t, but e is somewhat far from the vertices on the prefix of the sensitive detour, there are two vertices r_1, r_2 that satisfy the properties of the claim.

The algorithm then lets each vertex r in R send to all vertices in the graph the following:

- The list of the distances $\text{dist}(r, r', G)$ for every r' in R.
- The $\widetilde{O}(1)$-length bit LCA label of r in each tree T_s.

Overall, the total information sent is $\widetilde{O}(|R|^2+|S|\cdot|R|)$. This can be done in $\widetilde{O}(|R|^2+|S|\cdot|R|+D)$ rounds by a simple pipeline.

Now every vertex t is doing the following calculations for every edge $e \in \pi_{\sigma_1}(s,t)$ for which it did not receive a BFS token $\texttt{BFS}(s,t,G\setminus\{e\})$ in the first phase of the algorithm (of handling the easy replacement paths). Using the LCAs of all vertices in R with respect to T_s, it computes the set R_e^+ and R_e^- where $R_e^- = \{r \in R \mid e \notin \pi(s,r)\}$ and $R_e^+ = R \setminus R_e^-$. Note that $e \in \pi(s,r)$ only if the LCA of r and t is *below* the failing edge e. Since t has the $2\sigma_1$-length suffix of its $\pi(s,t)$ path, it can detect if the LCA is below the edge e. Let

$$\mathrm{dist}(s,t,G\setminus\{e\}) = \min_{r_1\in R_e^-} \min_{r_2\in R_e^+, \mathrm{dist}(r_1,r_2,G)\le\sigma_1/16} \mathrm{dist}(s,r_1,G)+\mathrm{dist}(r_1,r_2,G)+\mathrm{dist}(r_2,t,G)\ .$$

Let $r_1^* \in R_e^-$ and $r_2^* \in R_e^+$ be the vertices that minimize the $\mathrm{dist}(s,t,G\setminus\{e\})$. Then, the t lets $P_{\sigma_2}(s,t,e) = \pi_{\sigma_2}(r_2^*,t)$. This completes the description of the algorithm, the complete proof of Lemma 19, Cor. 3 and 4 are deferred to the full version.

References

1 Noga Alon, Shiri Chechik, and Sarel Cohen. Deterministic combinatorial replacement paths and distance sensitivity oracles. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece*, pages 12:1–12:14, 2019.

2 Davide Bilò, Fabrizio Grandoni, Luciano Gualà, Stefano Leucci, and Guido Proietti. Improved purely additive fault-tolerant spanners. In *Algorithms-ESA 2015*, pages 167–178. Springer, 2015.

3 Greg Bodwin, Fabrizio Grandoni, Merav Parter, and Virginia Vassilevska Williams. Preserving distances in very faulty graphs. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

4 Gilad Braunschvig, Shiri Chechik, David Peleg, and Adam Sealfon. Fault tolerant additive and (μ, α)-spanners. *Theor. Comput. Sci.*, 580:94–100, 2015.

5 Keren Censor-Hillel, Telikepalli Kavitha, Ami Paz, and Amir Yehudayoff. Distributed construction of purely additive spanners. *Distributed Computing*, 31(3):223–240, 2018.

6 Shiri Chechik and Sarel Cohen. Near optimal algorithms for the single source replacement paths problem. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2090–2109, 2019.

7 Shiri Chechik and Ofer Magen. Near optimal algorithm for the directed single source replacement paths problem. *CoRR*, abs/2004.13673, 2020.

8 Michael Elkin and Shaked Matar. Near-additive spanners in low polynomial deterministic CONGEST time. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 531–540, 2019.

9 Yuval Emek, David Peleg, and Liam Roditty. A near-linear-time algorithm for computing replacement paths in planar directed graphs. *ACM Transactions on Algorithms (TALG)*, 6(4):1–13, 2010.

10 Mohsen Ghaffari. Near-optimal scheduling of distributed algorithms. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC*, pages 3–12, 2015.

11 Mohsen Ghaffari and Merav Parter. Near-optimal distributed algorithms for fault-tolerant tree structures. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016*, pages 387–396, 2016.

12 Fabrizio Grandoni and Virginia Vassilevska Williams. Improved distance sensitivity oracles via fast single-source replacement paths. In *2012 IEEE 53rd Annual Symposium on Foundations of Computer Science*, pages 748–757. IEEE, 2012.

13 Manoj Gupta and Shahbaz Khan. Multiple source dual fault tolerant bfs trees. In *44th International Colloquium on Automata, Languages, and Programming (ICALP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

14 Frank Thomson Leighton, Bruce M Maggs, and Satish B Rao. Packet routing and job-shop scheduling ino (congestion+ dilation) steps. *Combinatorica*, 14(2):167–186, 1994.

15 Enrico Nardelli, Guido Proietti, and Peter Widmayer. Finding the most vital node of a shortest path. *Theoretical computer science*, 296(1):167–177, 2003.

16 Enrico Nardelli, Ulrike Stege, and Peter Widmayer. Low-cost fault-tolerant spanning graphs for point aets in the euclidean plane, 1997.

17 Merav Parter. Dual failure resilient bfs structure. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing*, pages 481–490, 2015.

18 Merav Parter. Vertex fault tolerant additive spanners. *Distributed Computing*, 30(5):357–372, 2017.

19 Merav Parter and David Peleg. Sparse fault-tolerant BFS structures. *ACM Trans. Algorithms*, 13(1):11:1–11:24, 2016.

20 David Peleg. *Distributed Computing: A Locality-sensitive Approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

21 Seth Pettie. Distributed algorithms for ultrasparse spanners and linear size skeletons. In *the Proc. of the Int'l Symp. on Princ. of Dist. Comp. (PODC)*, pages 253–262, 2008.

22 Liam Roditty and Uri Zwick. Replacement paths and k simple shortest paths in unweighted directed graphs. *ACM Transactions on Algorithms (TALG)*, 8(4):1–11, 2012.

23 Jeanette P Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-hoeffding bounds for applications with limited independence. *SIAM Journal on Discrete Mathematics*, 8(2):223–250, 1995.

24 Oren Weimann and Raphael Yuster. Replacement paths and distance sensitivity oracles via fast matrix multiplication. *ACM Transactions on Algorithms (TALG)*, 9(2):1–13, 2013.