Sparse Fault-Tolerant BFS Trees*

Merav Parter** and David Peleg

Department of Computer Science and Applied Mathematics,
The Weizmann Institute, Rehovot, Israel
{merav.parter,david.peleg}@weizmann.ac.il

Abstract. A *fault-tolerant* structure for a network is required to continue functioning following the failure of some of the network's edges or vertices. This paper considers *breadth-first search (BFS)* spanning trees, and addresses the problem of designing a sparse *fault-tolerant* BFS tree, or *FT-BFS tree* for short, namely, a sparse subgraph T of the given network G such that subsequent to the failure of a single edge or vertex, the surviving part T' of T still contains a BFS spanning tree for (the surviving part of) G. For a source node s, a target node t and an edge $e \in G$, the shortest $s - t$ path $P_{s,t,e}$ that does not go through e is known as a *replacement path*. Thus, our `FT-BFS` tree contains the collection of all replacement paths $P_{s,t,e}$ for every $t \in V(G)$ and every failed edge $e \in E(G)$. Our main results are as follows. We present an algorithm that for every n-vertex graph G and source node s constructs a (single edge failure) `FT-BFS` tree rooted at s with $O(n \cdot \min\{\mathtt{Depth}(s), \sqrt{n}\})$ edges, where $\mathtt{Depth}(s)$ is the depth of the BFS tree rooted at s. This result is complemented by a matching lower bound, showing that there exist n-vertex graphs with a source node s for which any edge (or vertex) `FT-BFS` tree rooted at s has $\Omega(n^{3/2})$ edges. We then consider *fault-tolerant multi-source BFS trees*, or `FT-MBFS` *trees* for short, aiming to provide (following a failure) a BFS tree rooted at each source $s \in S$ for some subset of sources $S \subseteq V$. Again, tight bounds are provided, showing that there exists a poly-time algorithm that for every n-vertex graph and source set $S \subseteq V$ of size σ constructs a (single failure) `FT-MBFS` tree $T^*(S)$ from each source $s_i \in S$, with $O(\sqrt{\sigma} \cdot n^{3/2})$ edges, and on the other hand there exist n-vertex graphs with source sets $S \subseteq V$ of cardinality σ, on which any `FT-MBFS` tree from S has $\Omega(\sqrt{\sigma} \cdot n^{3/2})$ edges. Finally, we propose an $O(\log n)$ approximation algorithm for constructing `FT-BFS` and `FT-MBFS` structures. The latter is complemented by a hardness result stating that there exists no $\Omega(\log n)$ approximation algorithm for these problems under standard complexity assumptions. In comparison with previous constructions our algorithm is deterministic and may improve the number of edges by a factor of up to $\sqrt{n}$ for some instances. All our algorithms can be extended to deal with one *vertex* failure as well, with the same performance.

* Supported in part by the Israel Science Foundation (grant 894/09), the I-CORE program of the Israel PBC and ISF (grant 4/11), the United States-Israel Binational Science Foundation (grant 2008348), the Israel Ministry of Science and Technology (infrastructures grant), and the Citi Foundation.

** Recipient of the Google European Fellowship in distributed computing; research is supported in part by this Fellowship.

H.L. Bodlaender and G.F. Italiano (Eds.): ESA 2013, LNCS 8125, pp. 779–790, 2013.
© Springer-Verlag Berlin Heidelberg 2013

1 Introduction

Background and Motivation. Modern day communication networks support a variety of logical structures and services, and depend on their undisrupted operation. As the vertices and edges of the network may occasionally fail or malfunction, it is desirable to make those structures robust against failures. Indeed, the problem of designing fault-tolerant constructions for various network structures and services has received considerable attention over the years.

Fault-resilience can be introduced into the network in several different ways. This paper focuses on a notion of fault-tolerance whereby the structure at hand is augmented or "reinforced" (by adding to it various components) so that subsequent to the failure of some of the network's vertices or edges, the surviving part of the structure is still operational. As this reinforcement carries certain costs, it is desirable to minimize the number of added components. To illustrate this type of fault tolerance, let us consider the structure of graph k-spanners (cf. [19,20,21]). A graph spanner H can be thought of as a skeleton structure that generalizes the concept of spanning trees and allows us to faithfully represent the underlying network using few edges, in the sense that for any two vertices of the network, the distance in the spanner is stretched by only a small factor. More formally, consider a weighted graph G and let $k \geq 1$ be an integer. Let $\text{dist}(u, v, G)$ denote the (weighted) distance between u and v in G. Then a k-spanner H satisfies that $\text{dist}(u, v, H) \leq k \cdot \text{dist}(u, v, G)$ for every $u, v \in V$. Introducing fault tolerance, we say that a subgraph H is an *f-edge fault-tolerant k-spanner* of G if $\text{dist}(u, v, H \setminus F) \leq k \cdot \text{dist}(u, v, G \setminus F)$ for any set $F \subseteq E$ of size at most f, and any pair of vertices $u, v \in V$. A similar definition applies to f-vertex fault-tolerant k-spanners. Sparse fault-tolerant spanner constructions were presented in [6,10]. This paper considers *breadth-first search (BFS)* spanning trees, and addresses the problem of designing *fault-tolerant* BFS trees, or `FT-BFS` trees for short. By this we mean a subgraph T of the given network G, such that subsequent to the failure of some of the vertices or edges, the surviving part T' of T still contains a BFS spanning tree for the surviving part of G. We also consider a generalized structure referred to as a *fault-tolerant multi-source BFS tree*, or `FT-MBFS` *tree* for short, aiming to provide a BFS tree rooted at each source $s \in S$ for some subset of sources $S \subseteq V$.

The notion of `FT-BFS` trees is closely related to the problem of constructing *replacement paths* and in particular to its *single source* variant, studied in [13]. That problem requires to compute the collection $\mathcal{P}_s$ of all $s - t$ replacement paths $P_{s,t,e}$ for every $t \in V$ and every failed edge e that appears on the $s - t$ shortest-path in G. The vast literature on *replacement paths* (cf. [4,13,24,26,28]) focuses on *time-efficient* computation of the these paths as well as their efficient maintenance in data structures (a.k.a *distance oracles*). In contrast, the main concern in the current paper is with optimizing the *size* of the resulting fault tolerant structure that contains the collection $\mathcal{P}_s$ of all replacement paths given a source node s. A typical motivation for such a setting is where the graph edges represent the channels of a communication network, and the system designer would like to purchase or lease a minimal collection of channels (i.e., a subgraph $G' \subseteq G$) that maintains its functionality as a "BFS tree" with respect to the source s upon any single edge or vertex failure in G. In such a context, the cost of computation at the preprocessing stage may often be negligible compared to the purchasing/leasing cost of the resulting structure. Hence, our key cost measure in this paper is the *size* of the fault tolerant structure, and our main goal is to achieve *sparse* (or *compact*) structures. Most previous work on sparse / compact fault-tolerant structures and services concerned structures that are *distance-preserving* (i.e., dealing with distances, shortest paths or shortest routes), *global* (i.e., centered on "all-pairs" variants), and *approximate* (i.e., settling for near optimal distances), such as *spanners*, *distance oracles* and *compact routing schemes*. The problem considered here, namely, the construction of FT-BFS trees, still concerns a distance preserving structure. However, it deviates from tradition with respect to the two other features, namely, it concerns a "single source" variant, and it insists on exact shortest paths. Hence our problem is on the one hand easier, yet on the other hand harder, than previously studied ones. Noting that in previous studies, the "cost" of adding fault-tolerance (in the relevant complexity measure) was often low (e.g., merely polylogarithmic in the graph size n), one might be tempted to conjecture that a similar phenomenon may reveal itself in our problem as well. Perhaps surprisingly, it turns out that our insistence on exact distances plays a dominant role and makes the problem significantly harder, outweighing our willingness to settle for a "single source" solution.

Contributions. We obtain the following results. In Sec. 2, we define the *Minimum* FT-BFS and *Minimum* FT-MBFS problems, aiming at finding the minimum such structures tolerant against a single edge or vertex fault. We show that these problems are NP-hard and moreover, cannot be approximated (under standard complexity assumptions) to within a factor of $\Omega(\log n)$, where n is the number of vertices of the input graph G. Section 3 presents lower bound constructions for these problems. For the single source case, we present a lower bound stating that for every n there exists an n-vertex graph and a source node $s \subseteq V$ for which any FT-MBFS tree from s requires $\Omega(n^{3/2})$ edges. We then show that there exist n-vertex graphs with source sets $S \subseteq V$ of size σ, on which any FT-MBFS tree from the source set S has $\Omega(\sqrt{\sigma} \cdot n^{3/2})$ edges. These results are complemented by matching upper bounds. In Sec. 4, we present a simple algorithm that for every n-vertex graph G and source node s, constructs a (single edge failure) FT-BFS tree rooted at s with $O(n \cdot \min\{\texttt{Depth}(s), \sqrt{n}\})$ edges. A similar algorithm yields an FT-BFS tree tolerant to one vertex failure, with the same size bound. In addition, for the multi source case, we show that there exists a polynomial time algorithm that for every n-vertex graph and source set $S \subseteq V$ of size $|S| = \sigma$ constructs a (single failure) FT-MBFS tree $T^*(S)$ from each source $s_i \in S$, with $O(\sqrt{\sigma} \cdot n^{3/2})$ edges.

Note that while those algorithms match the worst-case lower bounds, they might still be far from optimal for certain instances, see [18]. Consequently, in Sec. 5, we complete the upper bound analysis by presenting an $O(\log n)$ approximation algorithm for the Minimum `FT-MBFS` problem. This approximation algorithm is superior in instances where the graph enjoys a sparse `FT-MBFS` tree, hence paying $O(n^{3/2})$ edges is wasteful. In light of the hardness result for these problems (in Sec. 2), the approximability result is tight (up to constants). All our results hold for directed graphs as well.

Related Work. To the best of our knowledge, this paper is the first to study the sparsity of fault-tolerant BFS structures for graphs. The question of whether it is possible to construct a sparse fault tolerant *spanner* for an arbitrary undirected weighted graph, raised in [8], was answered in the affirmative in [6], presenting algorithms for constructing an f-vertex fault tolerant $(2k-1)$-spanner of size $O(f^2 k^{f+1} \cdot n^{1+1/k} \log^{1-1/k} n)$ and an f-edge fault tolerant $2k-1$ spanner of size $O(f \cdot n^{1+1/k})$ for a graph of size n. A randomized construction attaining an improved tradeoff for vertex fault-tolerant spanners was shortly afterwards presented in [10], yielding (with high probability) for every graph $G=(V,E)$, odd integer s and integer f, an f-vertex fault-tolerant s-spanner with $O\left(f^{2-\frac{2}{s+1}} n^{1+\frac{2}{s+1}} \log n\right)$ edges. This should be contrasted with the best stretch-size tradeoff currently known for non-fault-tolerant spanners [25], namely, $2k-1$ stretch with $\tilde{O}(n^{1+1/k})$ edges. Fault tolerant spanners for the d-dimensional Euclidean case were studied in [8,16,17].

A related network service is the *distance oracle* [3,23,26], which is a succinct data structure capable of supporting efficient responses to distance queries on a weighted graph G. A distance query (s,t) requires finding, for a given pair of vertices s and t in V, the distance (namely, the length of the shortest path) between u and v in G. The query protocol of an oracle $\mathcal{S}$ correctly answers distance queries on G. In a *fault tolerant distance oracle*, the query may include also a set F of failed edges or vertices (or both), and the oracle $\mathcal{S}$ must return, in response to a query (s,t,F), the distance between s and t in $G' = G \setminus F$. Such a structure is sometimes called an *F-sensitivity distance oracle*. The focus is on both fast preprocessing time, fast query time and low space. It has been shown in [9] that given a directed weighted graph G of size n, it is possible to construct in time $\tilde{O}(mn^2)$ a 1-sensitivity fault tolerant distance oracle of size $O(n^2 \log n)$ capable of answering distance queries in $O(1)$ time in the presence of a single failed edge or vertex. The preprocessing time was recently improved to $\tilde{O}(mn)$, with unchanged size and query time [4]. A 2-sensitivity fault tolerant distance oracle of size $O(n^2 \log^3 n)$, capable of answering 2-sensitivity queries in $O(\log n)$ time, was presented in [11].

Recently, distance sensitivity oracles have been considered for weighted and directed graphs in the *single source* setting [13]. Specifically, Grandoni and Williams considered the problem of *single-source replacement paths* where one aims to compute the collection of all replacement paths for a given source node s, and proposed an efficient randomized algorithm that does so in $\widetilde{O}(APSP(n,M))$

where $APSP(n, M)$ is the time required to compute all-pairs-shortest-paths in a weighted graph with integer weights $[-M, M]$.

A relaxed variant of distance oracles, in which distance queries are answered by *approximate* distance estimates instead of *exact* ones, was introduced in [26], where it was shown how to construct, for a given weighted undirected n-vertex graph G, an approximate distance oracle of size $O(n^{1+1/k})$ capable of answering distance queries in $O(k)$ time, where the *stretch* (multiplicative approximation factor) of the returned distances is at most $2k-1$. An f-sensitivity approximate distance oracle $\mathcal{S}$ was presented in [5]. For an integer parameter $k \geq 1$, the size of $\mathcal{S}$ is $O(kn^{1+\frac{8(f+1)}{k+2(f+1)}} \log(nW))$, where W is the weight of the heaviest edge in G, the stretch of the returned distance is $2k-1$, and the query time is $O(|F| \cdot \log^2 n \cdot \log\log n \cdot \log\log d)$, where d is the distance between s and t in $G \setminus F$. A fault-tolerant label-based $(1+\epsilon)$-approximate distance oracle for the family of graphs with doubling dimension bounded by α is presented in [2]. Our final example concerns fault tolerant routing schemes. A fault-tolerant routing protocol is a distributed algorithm that, for any set of failed edges F, enables any source vertex $\hat{s}$ to route a message to any destination vertex $\hat{d}$ along a shortest or near-shortest path in the surviving network $G \setminus F$ in an efficient manner (and without knowing F in advance). Compact routing schemes are considered in [1,7,19,22,25]. Fault-tolerant routing schemes are considered in [5].

2 Preliminaries

Notation. Given a graph $G = (V, E)$ and a source node s, let $T_0(s) \subseteq G$ be a shortest paths (or BFS) tree rooted at s. For a source node set $S \subseteq V$, let $T_0(S) = \bigcup_{s \in S} T_0(s)$ be a union of the single source BFS trees. Let $\pi(s, v, T)$ be the $s-v$ shortest-path in tree T, when the tree $T = T_0(s)$, we may omit it and simply write $\pi(s, v)$. Let $\Gamma(v, G)$ be the set of v neighbors in G. Let $E(v, G) = \{(u, v) \in E(G)\}$ be the set of edges incident to v in the graph G and let $\mathtt{deg}(v, G) = |E(v, G)|$ denote the degree of node v in G. When the graph G is clear from the context, we may omit it and simply write $\mathtt{deg}(v)$. Let $\mathtt{depth}(s, v) = \mathrm{dist}(s, v, G)$ denote the *depth* of v in the BFS tree $T_0(s)$. When the source node s is clear from the context, we may omit it and simply write $\mathtt{depth}(v)$. Let $\mathtt{Depth}(s) = \max_{u \in V}\{\mathtt{depth}(s, u)\}$ be the *depth* of $T_0(s)$. For a subgraph $G' = (V', E') \subseteq G$ (where $V' \subseteq V$ and $E' \subseteq E$) and a pair of nodes $u, v \in V$, let $\mathrm{dist}(u, v, G')$ denote the shortest-path distance in edges between u and v in G'. For a path $P = [v_1, \ldots, v_k]$, let $\mathtt{LastE}(P)$ be the last edge of path P. Let $|P|$ denote the length of the path and $P[v_i, v_j]$ be the subpath of P from v_i to v_j. For paths P_1 and P_2, $P_1 \circ P_2$ denote the path obtained by concatenating P_2 to P_1. Assuming an edge weight function $W : E(G) \to \mathbb{R}^+$, let $SP(s, v_i, G, W)$ be the set of $s-v_i$ shortest-paths in G according to the edge weights of W. Throughout, the edges of these paths are considered to be directed away from the source node s. Given an $s-v$ path P and an edge $e = (x, y) \in P$, let $\mathrm{dist}(s, e, P)$ be the distance (in edges) between s and e on P. In addition, for an edge $e = (x, y) \in T_0(s)$, define $\text{dist}(s, e) = i$ if $\texttt{depth}(x) = i - 1$ and $\texttt{depth}(y) = i$.

Definition 1. *A graph T^* is an edge (resp., vertex)* FT-BFS *tree for G with respect to a source node $s \in V$, iff for every edge $f \in E(G)$ (resp., vertex $f \in V$) and for every $v \in V$,* $\text{dist}(s, v, T^* \setminus \{f\}) = \text{dist}(s, v, G \setminus \{f\})$.

A graph T^* is an edge (resp., vertex) FT-MBFS *tree for G with respect to source set $S \subseteq V$, iff for every edge $f \in E(G)$ (resp., vertex $f \in V$) and for every $s \in S$ and $v \in V$,* $\text{dist}(s, v, T^* \setminus \{f\}) = \text{dist}(s, v, G \setminus \{f\})$.

For simplicity, we refer to edge FT-BFS (resp., edge FT-MBFS) trees simply by FT-BFS (resp., FT-MBFS) trees. Throughout, we focus on edge fault, yet the entire analysis extends trivially to the case of vertex fault as well.

Like other papers in this field [14,4], throughout, we assume without loss of generality that the shortest paths are unique since we can always add small perturbations to break any ties. Let W be a weight assignment that captures these symbolic perturbations.

The Minimum FT-BFS Problem. Denote the set of solutions for the instance (G, s) by $\mathcal{T}(s, G) = \{\widehat{T} \subseteq G \mid \widehat{T}$ is an FT-BFS tree w.r.t. $s\}$. Let $\texttt{Cost}^*(s, G) = \min\{|E(\widehat{T})| \mid \widehat{T} \in \mathcal{T}(s, G)\}$ be the minimum number of edges in any FT-BFS subgraph of G. These definitions naturally extend to the multi-source case where we are given a source set $S \subseteq V$ of size σ. Then
$\mathcal{T}(S, G) = \{\widehat{T} \subseteq G \mid \widehat{T}$ is a FT-MBFS with respect to $S\}$ and $\texttt{Cost}^*(S, G) = \min\{|E(\widehat{T})| \mid \widehat{T} \in \mathcal{T}(S, G)\}$.

In the *Minimum* FT-BFS problem we are given a graph G and a source node s and the goal is to compute an FT-BFS $\widehat{T} \in \mathcal{T}(s, G)$ of minimum size, i.e., such that $|E(\widehat{T})| = \texttt{Cost}^*(s, G)$. Similarly, in the *Minimum* FT-MBFS problem we are given a graph G and a source node set S and the goal is to compute an FT-MBFS $\widehat{T} \in \mathcal{T}(S, G)$ of minimum size i.e., such that $|E(\widehat{T})| = \texttt{Cost}^*(S, G)$. We begin by establishing hardness (for missing proofs see full version [18]).

Theorem 1. *The Minimum* FT-BFS *problem is NP-complete and cannot be approximated to within a factor $c \log n$ for some constant $c > 0$ unless $\mathcal{NP} \subseteq \mathcal{TIME}(n^{poly \log(n)})$.*

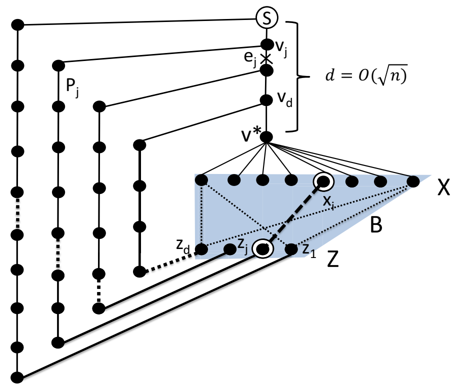
3 Lower Bounds

We now present a lower bound for the case of a single source.

Theorem 2. *There exists an n-vertex graph $G(V, E)$ and a source node $s \in V$ such that any* FT-BFS *tree rooted at s has $\Omega(n^{3/2})$ edges, i.e.,* $\texttt{Cost}^*(s, G) = \Omega(n^{3/2})$.

Proof: Let us first describe the structure of $G = (V, E)$. Set $d = \lfloor \sqrt{n}/2 \rfloor$.

The graph consists of four main components. The first is a path $\pi = [s = v_1, \ldots, v_{d+1} = v^*]$ of length d. The second component consists of a node set $Z = \{z_1, \ldots, z_d\}$ and a collection of d disjoint paths of deceasing length, $P_1, \ldots, P_d$, where $P_j = [v_j = p_1^j, \ldots, z_j = p_{t_j}^j]$ connects v_j with z_j and its length is $t_j = |P_j| = 6 + 2(d - j)$, for every $j \in 1, \cdots, d$. Altogether, the set of nodes in these paths, $Q = \bigcup_{j=1}^{d} V(P_j)$, is of size $|Q| = d^2 + 7d$.



The third component is a set of nodes X of size $n - (d^2 + 7d)$, all connected to the terminal node v^*. The last component is a complete bipartite graph $B = (X, Z, \hat{E})$ connecting X to Z. Overall, $V = X \cup Q$ and $E = \hat{E} \cup E(\pi) \cup \bigcup_{j=1}^{d} E(P_j)$. Note that $n/4 \leq |Q| \leq n/2$ for sufficiently large n. Consequently, $|X| = n - |Q| \geq n/2$, and $|\hat{E}| = |Q| \cdot |X| \geq n^{3/2}/4$. A BFS tree T_0 rooted at s for this G (illustrated by the solid edges in the figure) is given by

$$E(T_0) \;=\; \{(x_i, z_i) \mid i \in \{1, \ldots, d\}\} \cup \bigcup_{j=1}^{d} E(P_j) \setminus \{(p^j_{\ell_j}, p^j_{\ell_j - 1})\},$$

where $\ell_j = t_j - (d - j)$ for every $j \in \{1, \ldots, d\}$. We now show that every `FT-BFS` tree $T' \in \mathcal{T}(s, G)$ must contain all the edges of B, namely, the edges $e_{i,j} = (x_i, z_j)$ for every $i \in \{1, \ldots, |X|\}$ and $j \in \{1, \ldots, d\}$ (the dashed edges in the figure). Assume, towards contradiction, that there exists a $T' \in \mathcal{T}(s, G)$ that does not contain $e_{i,j}$ (the bold dashed edge (x_i, z_j) in the figure). Note that upon the failure of the edge $e_j = (v_j, v_{j+1}) \in \pi$, the unique $s - x_i$ shortest path connecting s and x_i in $G \setminus \{e_j\}$ is $P'_j = \pi[v_1, v_j] \circ P_j \circ [z_j, x_i]$, and all other alternatives are strictly longer. Since $e_{i,j} \notin T'$, also $P'_j \nsubseteq T'$, and therefore $\text{dist}(s, x_i, G \setminus \{e_j\}) < \text{dist}(s, x_i, T' \setminus \{e_j\})$, in contradiction to the fact that T' is an `FT-BFS` tree. It follows that every `FT-BFS` tree T' must contain at least $|\hat{E}| = \Omega(n^{3/2})$ edges. The theorem follows. ∎

We next consider an intermediate setting where it is necessary to construct a fault-tolerant subgraph `FT-MBFS` containing several `FT-BFS` trees in parallel, one for each source $s \in S$, for some $S \subseteq V$. In the full version [18], we establish the following.

Theorem 3. *There exists an n-vertex graph $G(V, E)$ and a source set $S \subseteq V$ of cardinality σ, such that any* `FT-MBFS` *tree from the source set S has $\Omega(\sqrt{\sigma} \cdot n^{3/2})$ edges, i.e.,* $\texttt{Cost}^*(S, G) = \Omega(\sqrt{\sigma} \cdot n^{3/2})$.

4 Upper Bounds

Single Source. In this section we consider the case of FT-BFS trees and establish the following.

Theorem 4. *There exists a polynomial time algorithm that for every n-vertex graph G and source node s constructs an* FT-BFS *tree rooted at s with $O(n \cdot \min\{\texttt{Depth}(s), \sqrt{n}\})$ edges.*

To prove the theorem, we first describe a simple algorithm for the problem and then prove its correctness and analyze the size of the resulting FT-BFS tree. Using the sparsity lemma of [24] and the tools of [13], one can provide a randomized construction for an FT-BFS tree with $O(n^{3/2} \log n)$ edges with high probability. In contrast, the simple algorithm presented here is *deterministic* and achieves an FT-BFS tree with $O(n^{3/2})$ edges, matching exactly the lower bound established in Sec. 3. We note that known time-efficient (and rather involved) algorithms for constructing replacement paths and distance sensitivity oracles (cf., [14,24,4,28,13]) can be modified to construct sparse FT-BFS and FT-MBFS trees by breaking shortest path ties properly and maintaining the successors of the computed replacement paths. Since our focus here is on the size of the resulting FT-BFS trees, and not on optimizing the running time, we introduce the construction using a simple but slow ($O(nm + n^2 \log n)$ round) algorithm. In the analysis section we then show that as long as the collection of the single-source replacement paths are computed in a way that breaks shortest path ties properly, the total number of edges in this collection is bounded by $O(n^{3/2})$.

The Algorithm. Recall that W is a weight assignment that guarantees the uniqueness of the shortest paths, by introducing some symbolic perturbation to the edge lengths. Let $T_0 = BFS(s, G)$ be the BFS tree rooted at s in G, computed according to the weight assignment W. For every $e_j \in T_0$, let $T_0(e_j)$ be the BFS tree rooted at s in $G \setminus \{e_j\}$. Then the final FT-BFS tree is given by $T^*(s) = T_0 \cup \bigcup_{e_j \in T_0} T_0(e_j)$. The correctness is immediate by construction.

Observation 5. *$T^*(s)$ is an* FT-BFS *tree.*

It remains to bound the size of $T^*(s)$.

Size Analysis. We first provide some notation. For a path P, let $\texttt{Cost}(P) = \sum_{e \in P} W(e)$ be the weighted cost of P, i.e., the sum of its edge weights. An edge $e \in G$ is defined as *new* if $e \notin E(T_0)$. For every $v_i \in V$ and $e_j \in T_0$, let $P^*_{i,j} = \pi(s, v_i, T_0(e_j)) \in SP(s, v_i, G \setminus \{e_j\}, W)$ be the optimal *replacement path* of s and v_i upon the failure of $e_j \in T_0$. Let $\texttt{New}(P) = E(P) \setminus E(T_0)$ and

$$\texttt{New}(v_i) \;=\; \{\texttt{LastE}(P^*_{i,j}) \mid e_j \in T_0\} \setminus E(T_0)$$

be the set of v_i new edges appearing as the last edge in the replacement paths $P^*_{i,j}$ of v_i and $e_j \in T_0$. It is convenient to view the edges of $T_0(e_j)$ as directed away from s. We then have that

$$T^*(s) = T_0 \cup \bigcup_{v_i \in V \setminus \{s\}} \texttt{New}(v_i).$$

I.e., the set of new edges that participate in the final `FT-BFS` tree $T^*(s)$ are those that appear as a last edge in some replacement path.

We now upper bound the size of the `FT-BFS` tree $T^*(s)$. Our goal is to prove that $\texttt{New}(v_i)$ contains at most $O(\sqrt{n})$ edges for every $v_i \in V$. The following observation is crucial in this context.

Observation 6. *If* $\texttt{LastE}(P^*_{i,j}) \notin E(T_0)$*, then* $e_j \in \pi(s, v_i)$*.*

Obs. 6 also yields the following.

Corollary 1. *(1)* $\texttt{New}(v_i) = \{\texttt{LastE}(P^*_{i,j}) \mid e_j \in \pi(s, v_i)\} \setminus E(T_0)$ *and (2)* $|\texttt{New}(v_i)| \le \min\{\texttt{depth}(v_i), \texttt{deg}(v_i)\}$.

This holds since the edges of $\texttt{New}(v_i)$ are coming from at most $\texttt{depth}(v_i)$ replacement paths $P^*_{i,j}$ (one for every $e_j \in \pi(s, v_i)$), and each such path contributes at most one edge incident to v_i.

For the reminder of the analysis, let us focus on one specific node $u = v_i$ and let $\pi = \pi(s, u)$, $N = |\texttt{New}(u)|$. For every edge $e_k \in \texttt{New}(u)$, we define the following parameters. Let $f(e_k) \in \pi$ be the failed edge such that $e_k \in T_0(f(e_k))$ appears in the replacement path $P_k = \pi(s, u, T')$ for $T' = T_0(f(e_k))$. (Note that e_k might appear as the last edge on the path $\pi(s, u, T_0(e'))$ for several edges $e' \in \pi$; in this case, one such e' is chosen arbitrarily).

Let b_k be the *last* divergence point of P_k and π, i.e., the last vertex on the replacement path P_k that belongs to $V(\pi) \setminus \{u\}$. Since $\texttt{LastE}(P_k) \notin E(T_0)$, it holds that b_k is not the neighbor of u in P_k.

Let $\texttt{New}(u) = \{e_1, \ldots, e_N\}$ be sorted in non-decreasing order of the distance between b_k and u, $\text{dist}(b_k, u, \pi) = |\pi(b_k, u)|$. I.e.,

$$\text{dist}(b_1, u, \pi) \le \text{dist}(b_2, u, \pi) \ldots \le \text{dist}(b_N, u, \pi). \tag{1}$$

We consider the set of truncated paths $P'_k = P_k[b_k, u]$ and show that these paths are vertex-disjoint except for the last common endpoint u. We then use this fact to bound the number of these paths, hence bound the number N of new edges. The following observation follows immediately by the definition of b_k.

Observation 7. $(V(P'_k) \cap V(\pi)) \setminus \{b_k, u\} = \emptyset$*.*

Lemma 1. $\big(V(P'_i) \cap V(P'_j)\big) \setminus \{u\} = \emptyset$ *for every* $i, j \in \{1, \ldots, N\}$*,* $i \neq j$*.*

Proof: Assume towards contradiction that there exist $i \neq j$, and a node

$$u' \in \big(V(P'_i) \cap V(P'_j)\big) \setminus \{u\}$$

in the intersection. Since $\texttt{LastE}(P'_i) \neq \texttt{LastE}(P'_j)$, by Obs. 7 we have that $P'_i, P'_j \subseteq G \setminus E(\pi)$. The faulty edges $f(e_i), f(e_j)$ belong to $E(\pi)$. Hence there are two distinct $u' - u$ shortest paths in $G \setminus \{f(e_i), f(e_j)\}$. By the optimality of P'_i in $T_0(f(e_i))$, (i.e., $P_i \in SP(s, u, G \setminus \{f(e_i)\}, W)$), we have that $\texttt{Cost}(P'_i[u', u]) < \texttt{Cost}(P'_j[u', u])$. In addition, by the optimality of P'_j in $T_0(f(e_j))$, (i.e., $P_j \in SP(s, u, G \setminus \{f(e_j)\}, W)$), we have that $\texttt{Cost}(P'_j[u', u]) < \texttt{Cost}(P'_i[u', u])$. Contradiction. ∎

We are now ready to prove our key lemma.

Lemma 2. $|\texttt{New}(u)| = O(n^{1/2})$ *for every* $u \in V$.

Proof: Assume towards contradiction that $N = |\texttt{New}(u)| > \sqrt{2n}$. By Lemma 1, we have that $b_1, \ldots, b_N$ are distinct and by definition they all appear on the path π. Therefore, by the ordering of the P'_k, we have that the inequalities of Eq. (1) are strict, i.e., $\text{dist}(b_1, u, \pi) < \text{dist}(b_2, u, \pi) < \ldots < \text{dist}(b_N, u, \pi)$. Since $b_1 \neq u$ (by definition), we also have that $\text{dist}(b_1, u, \pi) \geq 1$. We Conclude that

$$\text{dist}(b_k, u, \pi) = |\pi(b_k, u)| \geq k \ . \tag{2}$$

Next, note that each P'_k is a replacement $b_k - u$ path and hence it cannot be shorter than $\pi(b_k, u)$, implying that $|P'_k| \geq |\pi(b_k, u)|$. Combining with Eq. (2), we have that

$$|P'_k| \geq k \quad \text{for every} \quad k \in \{1, \ldots, N\} \ . \tag{3}$$

Since by Lemma 1, the paths P'_k are vertex disjoint (except for the common vertex u), we have that

$$\left| \bigcup_{k=1}^{N} (V(P'_k) \setminus \{u\}) \right| = \sum_{k=1}^{N} |V(P'_k) \setminus \{u\}| \ \geq \ \sum_{k=1}^{N} (k-1) \ > \ n,$$

where the first inequality follows by Eq. (3) and the last by the assumption that $N > \sqrt{2n}$. Since there are n nodes in G, we end with contradiction. ∎

Multiple Sources. For the case of multiple sources, in the full version [18], we establish the following upper bound.

Theorem 8. *There exists a polynomial time algorithm that for every n-vertex graph $G = (V, E)$ and source set $S \subseteq V$ of size $|S| = \sigma$ constructs an* `FT-MBFS` *tree $T^*(S)$ from each source $s_i \in S$, with a total number of $n \cdot \min\{\sum_{s_i \in S} \texttt{depth}(s_i), O(\sqrt{\sigma n})\}$ edges.*

We note that both our lower and upper bound analysis naturally extend to the case of directed and edge weighted graphs with integer weights in the range $[-M, M]$ by paying an extra factor of $O(\sqrt{M})$ in the size of the `FT-MBFS` trees.

5 $O(\log n)$-Approximation for FT-MBFS Trees

In Sec. 4, we presented an algorithm that for every graph G and source s constructs an `FT-BFS` tree $\widehat{T} \in \mathcal{T}(s, G)$ with $O(n^{3/2})$ edges. In Sec. 3, we showed that there exist graphs G and $s \in V(G)$ for which $\texttt{Cost}^*(s, G) = \Omega(n^{3/2})$, establishing tightness of our algorithm in the worst-case. Yet, there are also inputs (G', s') for which the algorithm of Sec. 4, as well as algorithms based on the analysis of [13] and [24], might still produce an `FT-BFS` $\widehat{T} \in \mathcal{T}(s', G')$ which is denser by a factor of $\Omega(\sqrt{n})$ than the size of the optimal `FT-BFS` tree, i.e., such that $|E(\widehat{T})| \geq \Omega(\sqrt{n}) \cdot \texttt{Cost}^*(s', G')$. For an illustration of such a case see [18]. Clearly, a universally optimal algorithm is unlikely given the hardness of approximation result of Thm. 1. Yet the gap can be narrowed down. The goal of this section is to present an $O(\log n)$ approximation algorithm for the Minimum FT-BFS Problem (hence also to its special case, the Minimum FT-BFS Problem, where $|S| = 1$).

Theorem 9. *There exists a polynomial time algorithm that for every n-vertex graph G and source node set $S \subseteq V$ constructs an* FT-MBFS *tree $\widehat{T} \in \mathcal{T}(S, G)$ such that $|E(\widehat{T})| \leq O(\log n) \cdot \texttt{Cost}^*(S, G)$.*

To prove the theorem, we first describe the algorithm and then bound the number of edges. Let $\texttt{ApproxSetCover}(\mathfrak{F}, U)$ be an $O(\log n)$ approximation algorithm for the Set-Cover problem, which given a collection of sets $\mathfrak{F} = \{S_1, \ldots, S_M\}$ that covers a universe $U = \{u_1, \ldots, u_N\}$ of size N, returns a cover $\mathfrak{F}' \subseteq \mathfrak{F}$ that is larger by at most $O(\log N)$ than any other $\mathfrak{F}'' \subseteq \mathfrak{F}$ that covers U (cf. [27]).

The Algorithm. Starting with $\widehat{T} = \emptyset$, the algorithm adds edges to $\widehat{T}$ until it becomes an FT-MBFS tree.

Set an arbitrary order on the vertices $V(G) = \{v_1, \ldots, v_n\}$ and on the edges $E^+ = E(G) \cup \{e_0\} = \{e_0, \ldots, e_m\}$ where e_0 is a new fictitious edge whose role will be explained later on. For every node $v_i \in V$, define

$$U_i = \{\langle s_k, e_j \rangle \ \mid s_k \in S \setminus \{v_i\}, e_j \in E^+\}.$$

The algorithm consists of n rounds, where in round i it considers v_i. Let $\Gamma(v_i, G) = \{u_1, \ldots, u_{d_i}\}$ be the set of neighbors of v_i in some arbitrary order, where $d_i = \texttt{deg}(v_i, G)$. For every neighbor u_j, define a set $S_{i,j} \subseteq U_i$ containing certain source-edge pairs $\langle s_k, e_\ell \rangle \in U_i$. Informally, a set $S_{i,j}$ contains the pair $\langle s_k, e_\ell \rangle$ iff there exists an $s_k - v_i$ shortest path in $G \setminus \{e_\ell\}$ that goes through the neighbor u_j of v_i. Note that $S_{i,j}$ contains the pair $\langle s_k, e_0 \rangle$ iff there exists an $s_k - v_i$ shortest-path in $G \setminus \{e_0\} = G$ that goes through u_j. I.e., the fictitious edge e_0 is meant to capture the case where no fault occurs, and thus we take care of true shortest-paths in G. Formally, every pair $\langle s_k, e_\ell \rangle \in U_i$ is included in every set $S_{i,j}$ satisfying that

$$\text{dist}(s_k, u_j, G \setminus \{e_\ell\}) = \text{dist}(s_k, v_i, G \setminus \{e_\ell\}) - 1. \quad (4)$$

Let $\mathfrak{F}_i = \{S_{i,1}, \ldots, S_{i,d_i}\}$. The edges of v_i that are added to $\widehat{T}$ in round i are now selected by using algorithm ApproxSetCover to generate an approximate solution for the set cover problem on the collection $\mathfrak{F} = \{S_{i,j} \ \mid \ u_j \in \Gamma(v_i, G)\}$. Let $\mathfrak{F}'_i = \texttt{ApproxSetCover}(\mathfrak{F}_i, U_i)$. For every $S_{i,j} \in \mathfrak{F}'_i$, add the edge (u_j, v_i) to $\widehat{T}$. In [18], we prove the correctness of this algorithm and establish Thm. 9.

Acknowledgment. We are grateful to Gilad Braunschvig, Alon Brutzkus, Adam Sealfon, Oren Weimann and the anonymous reviewers for helpful comments.

References

1. Awerbuch, B., Bar-Noy, A., Linial, N., Peleg, D.: Compact distributed data structures for adaptive network routing. In: STOC, pp. 230–240 (1989)

2. Abraham, I., Chechik, S., Gavoille, C., Peleg, D.: Forbidden-Set Distance Labels for Graphs of Bounded Doubling Dimension. In: PODC, pp. 192–200 (2010)
3. Baswana, S., Sen, S.: Approximate distance oracles for unweighted graphs in expected $O(n^2)$ time. ACM Trans. Algorithms 2(4), 557–577 (2006)
4. Bernstein, A., Karger, D.: A nearly optimal oracle for avoiding failed vertices and edges. In: STOC, pp. 101–110 (2009)
5. Chechik, S., Langberg, M., Peleg, D., Roditty, L.: f-sensitivity distance oracles and routing schemes. Algorithmica, 861–882 (2012)
6. Chechik, S., Langberg, M., Peleg, D., Roditty, L.: Fault-tolerant spanners for general graphs. In: STOC, pp. 435–444 (2009)
7. Chechik, S.: Fault-Tolerant Compact Routing Schemes for General Graphs. In: Aceto, L., Henzinger, M., Sgall, J. (eds.) ICALP 2011, Part II. LNCS, vol. 6756, pp. 101–112. Springer, Heidelberg (2011)
8. Czumaj, A., Zhao, H.: Fault-tolerant geometric spanners. Discrete & Computational Geometry 32 (2003)
9. Demetrescu, C., Thorup, M., Chowdhury, R., Ramachandran, V.: Oracles for distances avoiding a failed node or link. SIAM J. Computing 37, 1299–1318 (2008)
10. Dinitz, M., Krauthgamer, R.: Fault-tolerant spanners: better and simpler. In: PODC, pp. 169–178 (2011)
11. Duan, R., Pettie, S.: Dual-failure distance and connectivity oracles. In: SODA (2009)
12. Feige, U.: A Threshold of ln n for Approximating Set Cover. J. ACM, 634–652 (1998)
13. Grandoni, F., Williams, V.V.: Improved Distance Sensitivity Oracles via Fast Single-Source Replacement Paths. In: FOCS (2012)
14. Hershberger, J., Subhash, S.: Vickrey prices and shortest paths: What is an edge worth? In: FOCS (2001)
15. Hershberger, J., Subhash, S., Bhosle, A.: On the difficulty of some shortest path problems. In: Alt, H., Habib, M. (eds.) STACS 2003. LNCS, vol. 2607, pp. 343–354. Springer, Heidelberg (2003)
16. Levcopoulos, C., Narasimhan, G., Smid, M.: Efficient algorithms for constructing fault-tolerant geometric spanners. In: STOC, pp. 186–195 (1998)
17. Lukovszki, T.: New results on fault tolerant geometric spanners. In: Dehne, F., Gupta, A., Sack, J.-R., Tamassia, R. (eds.) WADS 1999. LNCS, vol. 1663, pp. 193–204. Springer, Heidelberg (1999)
18. Parter, P., Peleg, D.: Sparse Fault-Tolerant BFS Trees (2013), `http://arxiv.org/abs/1302.5401`
19. Peleg, D.: Distributed Computing: A Locality-Sensitive Approach. SIAM (2000)
20. Peleg, D., Schäffer, A.A.: Graph spanners. J. Graph Theory 13, 99–116 (1989)
21. Peleg, D., Ullman, J.D.: An optimal synchronizer for the hypercube. SIAM J. Computing 18(2), 740–747 (1989)
22. Peleg, D., Upfal, E.: A trade-off between space and efficiency for routing tables. J. ACM 36, 510–530 (1989)
23. Roditty, L., Thorup, M., Zwick, U.: Deterministic constructions of approximate distance oracles and spanners. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) ICALP 2005. LNCS, vol. 3580, pp. 261–272. Springer, Heidelberg (2005)
24. Roditty, L., Zwick, U.: Replacement paths and k simple shortest paths in unweighted directed graphs. ACM Trans. Algorithms (2012)
25. Thorup, M., Zwick, U.: Compact routing schemes. In: SPAA, pp. 1–10 (2001)
26. Thorup, M., Zwick, U.: Approximate distance oracles. J. ACM 52, 1–24 (2005)
27. Vazirani, V.: Approximation Algorithms. Georgia Inst. Tech. (1997)
28. Weimann, O., Yuster, R.: Replacement paths via fast matrix multiplication. In: FOCS (2010)