Sublinear Edge Fault Tolerant Spanners for Hypergraphs

Jialin He, Nicholas Popescu, Chunjiang Zhu

Abstract

We initiate the study on fault-tolerant spanners in hypergraphs and develop fast algorithms for their constructions. A fault-tolerant (FT) spanner preserves approximate distances under network failures, often used in applications like network design and distributed systems. While classic (fault-free) spanners are believed to be easily extended to hypergraphs such as by the method of associated graphs [1], we reveal that this is not the case in the fault-tolerant setting: simple methods can only get a linear size in the maximum number of faults f. In contrast, all known optimal size of FT spanners are sublinear in f. Inspired by the FT clustering technique in [2], we propose a clustering based algorithm that achieves an improved sublinear size bound. For an n-node m-edge hypergraph with rank r and a sketch parameter k, our algorithm constructs edge FT (EFT) hyperspanners of stretch $2k-1$ and size $O(k^2 f^{1-1/(rk)} n^{1+1/k} \log n)$ with high probability in time $\widetilde{O}(mr^3 + fn)$ ($\widetilde{O}$ hides polylogarithmic factors). We also establish a lower bound of $\Omega(f^{1-1/r-1/rk} n^{1+1/k-o(1)})$ edges for EFT hyperspanners, which leaves a gap of $\text{poly}(k) f^{1/r}$. Finally, we provide an algorithm for constructing additive EFT hyperspanners by combining multiplicative EFT hyperspanners with additive hyperspanners. We believe that our work will spark interest in developing optimal FT spanners for hypergraphs.

1 Introduction

Graph spanners are sparse subgraphs that approximately preserve pairwise distances [3]. A (*multiplicative*) α-spanner ensures that all distances in the spanner are stretched by at most a factor α, while an *additive* β-spanner allows distances to increase by an additive surplus β. It was well-established that for an integer $k > 1$, every undirected graph on n vertices has a $(2k-1)$-spanner of size (number of edges) $O(n^{1+1/k})$ [4, 5]. Spanners have found wide applications in network design, distributed algorithms, and approximation algorithms, such as routing [6], synchronizers [7], distance oracles [5], and preconditioning of linear systems [8]. While prior work focuses on graphs, higher order relationships in many real-world systems such as biological pathways, social communities, and acquaintance networks are more naturally modeled as *hypergraphs*, where a hyperedge may contain more than two vertices. Extending spanner theory to hypergraphs is a natural and important direction. The scarcity of literature seems to suggest the extension is trivial: one can convert a hypergraph into an associated graph [1] by clique expansion (i.e., replacing each hyperedge by a clique on its vertices). Then any spanner in the associated graph can be translated to a hyperspanner in the hypergraph. Since we have not realized any proof, we provide a formal proof in Section 2.1, including a distinction between associated multi-graphs and simple associated graphs, where the latter are obtained by keeping only the lightest edge for parallel edges between two vertices.

However, the problem of constructing hyperspanners in the *fault-tolerant* setting becomes non-trivial and more interesting. Many real-world networks are inherently prone to failures of vertices and edges. Fault-tolerant (FT) spanners are robust spanners that still function in the presence of faults. Vertex FT (VFT) and edge FT (EFT) spanners are guaranteed to contain a spanner

in $G \setminus F$ for any possible small subset $F \subset V$ and $F \subset E$, respectively. Directly applying the associated graph idea to construct VFT hyperspanners does not work, since it is unclear how to set the number of faults. Considering a hyperedge with three vertices v_1, v_2, v_3 corresponding to a clique/triangle in the associated graph, a vertex fault v_1 in the hypergraph would remove the hyperedge, which corresponds to the removal of all three edges in the clique. However, in the associated graph a vertex fault v_1 would NOT remove the edge (v_2, v_3), which is inconsistent with vertex faults in hypergraphs. Constructing EFT hyperspanners via the associated graph idea is valid. But since a hyperedge fault corresponds to at most r^2 edge faults in the associated graph (where r is the rank of the hypergraph), the number of faults allowed for the EFT spanner in the associated graph has to be fr^2, instead of only f. The size of EFT spanners in multi-graphs was settled to $\Theta(fn^{1+1/k})$ by [9]. Then the size of EFT hyperspanners is $O(fr^2n^{1+1/k})$. In fact, we show that adapting the peel-off method of [10] to hypergraphs results in $O(fn^{1+1/k})$-sized hyperspanners. Since the optimal sizes of EFT and VFT spanners are sublinear in the number of faults f, *can we achieve a sublinear size for EFT hyperspanners?* In this paper, we provide an affirmative answer accompanied with a lower bound on the size of EFT hyperspanners.

Contributions. In this paper, we initiate the study of fault-tolerant spanners for hypergraphs. We first provide a formal proof on the approach of associated graphs for fault-free hyperspanners, carefully distinguishing the setting of multi-graphs and simple associated graphs. We then show a simple application of the associated graphs and an adaptation of the peel-off method [10] get size $O(fr^2n^{1+1/k})$ and $O(fn^{1+1/k})$ respectively, falling much behind the partially optimal size with sublinearity in the number of faults in the graph counterpart [11]. We analyze that the greedy(-like) algorithm widely used in the graph setting is inherently challenging in hypergraphs. We then propose a clustering-based algorithm based on Parter [2] and obtain a sublinear size $O(k^2f^{1-1/(rk)}n^{1+1/k}\log n)$ in fast time $\widetilde{O}(mr^3 + fn)$. Furthermore, we accompany the results with a size lower bound $\Omega(f^{1-1/r-1/rk}n^{1+1/k-o(1)})$, showing a separation from known graph bounds. Finally, we show EFT additive hyperspanners can be constructed from taking the union of EFT multiplicative hyperspanners and fault-free additive hyperspanners.

Theorem 1. *For an n-vertex m-hyperedge hypergraph of rank r and a sketch parameter $k > 1$, there is a randomized algorithm constructing a $(2k-1)$-hyperspanner of size $O(k^2f^{1-1/(rk)}n^{1+1/k}\log n)$ in $\widetilde{O}(mr^3 + fn)$ time.*

Theorem 2. *For integers $k \geq 1, f \geq 1, r \geq 2$, there exist infinite families of undirected unweighted n-node r-uniform hypergraph with $\Omega(f^{1-1/r-1/rk}n^{1+1/k-o(1)})$ hyperedges for which any f-EFT $(2k-1)$-spanner must contain all the hyperedges of H.*

Theorem 3. *For a hypergraph H of rank r, an algorithm for constructing f-EFT μ-hyperspanner S_1 and an algorithm for constructing additive hyperspanner S_2 of surplus α, there is an algorithm that constructs an additive f-EFT hyperspanner of surplus $fr(2\alpha+(\mu-1)W_{s,t})+\alpha$ and size $|S_1 \cup S_2|$, where $W_{s,t}$ is the maximum hyperedge weight on the shortest path between s and t in H after the failure event.*

Related Work. FT graph spanners have been subject to intensive study through works like [9, 12–15] which push for optimal size bounds and stretch factors under different constraints. Naturally, there exists a tradeoff between the maximum size bound of an FT spanner and the amount of stretch for approximating distances. The question of optimal-sized VFT spanners has been settled, first in [16] (for a fixed sketch) and then in [17], $O(f^{1-1/k}n^{1+1/k})$ for stretch $2k-1$. For EFT spanners, the best known size lower bound is $\Omega(f^{1/2-1/(2k)}n^{1+1/k} + fn)$ [16]. For odd k, a matching upper bound was obtained, up to polynomial in k. For even k, the upper bound is

$O(k^2 f^{1/2} n^{1+1/k} + fkn)$, leaving a gap of $k^2 f^{1/(2k)}$. Recent efforts have been devoted to optimal-time constructions of optimized-sized FT spanners [2,14,18]. [18] first developed a polynomial-time algorithm at the expense of a factor of k in the size, which was further removed by [14]. Optimal-time $\widetilde{O}(m)$ was achieved by [2] based on a novel FT clustering approach. Adapting from the FT clustering, our method inherits its local computation, which is friendly to parallel and distributed computing. Unlike graph spanners, the literature on hyperspanners is scarce, with only a few known prior work [19–21]. [19, 21] adopted the associated graph idea to compute hyperspanners for spectral sparsfication while [20] empirically studied how to minimize the number of hyperedges when mapping spanner edges in an associated graph to hyperedges.

2 Constructing Multiplicative EFT Hyperspanners

We start as a warm up by discussing the applications of associated graphs and the peel-off method in constructing hyperspanners, and then present our main algorithm for the construction of sublinear EFT hyperspanners.

2.1 Warm up: Associated Graph, Peel-off, and Greedy Algorithms

We denote $\delta_H(u, v)$ the shortest distance between u and v in H. After formally defining EFT hyperspanners below, we show two simple but useful lemmas in general (fault-free) hyperspanner construction.

Definition 1. *For a hypergraph $G = (V, E)$ and a subhypergraph $S(V, E' \subseteq E)$, S is an f-edge fault tolerant (EFT) multiplicative t-hyperspanner (additive β-hyperspanner) if $\forall u, v \in V(G)$, $\forall F \subseteq E(G)$, $|F| \leq f$, $\delta_{S\setminus F}(u, v) \leq t \cdot \delta_{G\setminus F}(u, v)$ ($\delta_{S\setminus F}(u, v) \leq \delta_{G\setminus F}(u, v) + \beta$, respectively).*

For a hypergraph $H(V, F, z)$, the *associated graph* $G(V, E, w)$ of H is a multi-graph obtained by replacing each hyperedge $h \in F$ with a clique $C(h)$ with $P(|h|, 2)$ edges of weights $z(h)$. A *simple associated graph* G' of H is to remove all parallel edges but the lightest edge (i.e., the edge of minimum weight with ties broken arbitrarily) between every pair of vertices in H's associated graph G. Although the associated graph idea has been mentioned in some prior work [19, 21], we have not seen a proof and they do not distinguish multi-graphs and simple associated graphs, limiting its applicability. We provide the proof in the Appendix.

Lemma 1. *Let G be an associated graph of a hypergraph H with a function $f : E(G) \to E(H)$ mapping each edge in G to its corresponding hyperedge in H. If G' is an α-spanner of G for $\alpha > 1$, then the sub-hypergraph H' of H with the set of hyperedges $E(H') = \{f(e') \mid e' \in E(G)\}$ is an α-hyperspanner of H. This also works when G is a simple associated graph of H.*

The idea also holds for additive hyperspanners.

Corollary 1. *Let G be an associated graph of a hypergraph H with a function $f : E(G) \to E(H)$ mapping each edge in G to its corresponding hyperedge in H. If G' is an additive β-spanner of G for $\beta > 0$, then the sub-hypergraph H' of H with the set of hyperedges $E(H') = \{f(e') \mid e' \in E(G)\}$ is an additive β-spanner of H. This also works when G is a simple associated graph of H.*

With Lemma 1 and Corollary 1, we can translate the results for any multiplicative and additive spanners, whether they are for *multi-graphs or simple graphs*, into corresponding hyperspanners of the same stretch/additive surplus. The size of the spanners also serve as an upper bound on the size of hyperspanners, although it is possible multiple spanner edges are mapped to one hyperedge, resulting in a smaller size. Then we have the following result for hypergraph constructions.

Lemma 2. *Assuming the Erdös Girth conjecture, every hypergraph has a $(2k-1)$-hyperspanner of size $\Theta(n^{1+1/k})$ for $k \geq 1$.*

It is viable to use the associated graph idea to construct EFT hyperspanners. But since the failure of a hyperedge results in r^2 edge failures in the associated graph, we have to set the number of faults to fr^2. Because the size of EFT spanners in multi-graphs is $\Theta(fn^{1+1/k})$ [9], the size of EFT hyperspanners from this method is $O(fr^2n^{1+1/k})$.

As an alternative, the peel-off method [10] constructs f-EFT spanners in $f+1$ iterations. In each iteration, it computes the spanner of the current graph and then deletes the spanner edges from the current graph. Since the spanner edges in $f+1$ iterations are non-overlapping, their union contains $f+1$ parallel paths for every edge not included in the union, providing resilience to at most f faulty edges. It is easy to see this method can be adapted for hyperspanner constructions and the proof is in the Appendix. Equipped with Lemma 2, however, this method results in a linear size $O(fn^{1+1/k})$ in the number of faults, lagging behind the optimal VFT and partially optimal EFT spanners, both with a sublinearity in f.

Recent optimality results on FT spanners heavily use the greedy method [11, 16, 18]. Rooted at the greedy algorithm for optimal spanners [22], it checks for each edge whether there exists a fault set such that the stretch after the failure event does not hold. Many of these results use the blocking set to bound the size, which is a set of edge pairs such that for every cycle of small length, there must be an edge pair from the set on the cycle. The greedy method, if naively adapted to the hypergraph setting, has technical curdles in bounding the size unfortunately. Specifically, for a small cycle, its last-added hyperedge may not be added due to the considered vertex pair on the cycle, but a different vertex pair for another cycle. It remains open to bound the output size of the greedy algorithm for constructing FT hyperspanners.

2.2 Sublinear EFT Hyperspanners

Instead of extending the greedy algorithm, we are able to adapt Parter's VFT clustering and analysis [2] to EFT hyperspanners. The adaptation requires careful considerations to address the complexity of hypergraphs, including the vertex-disjointness on head hyperedges to maintain the independence requirement in Lemma 6 and maintenance of different vertex pairs in a hyperedge. We also simplify the analysis for hypergraphs e.g., excluding the shortcut operation. Essentially, we maintain overlapping clustering where each vertex can belong to $\Omega(f)$ clusters. Similar to [2], the algorithm contains k iterations. In each iteration, we gradually examine remaining hyperedges and add some of them to increase the radius of clusters by one.

Inspired by an edge-centric perspective description [23] for algorithm in [2], we define the status for each triple u, v, h, vertices u and v in a hyperedge h: keep (kp), safely discard (sd), or postpone to subsequent iterations (pp). $status_i(u, v, h) = kp$ if we decide to add h in the hyperspanner when examining vertex pair u, v in iteration i; $status_i(u, v, h) = sd$ if we have found sufficently many edge-disjoint paths between u, v of weight at most $(2i-1)w(h)$; and $status_i(u, v, h) = pp$ otherwise. We also generalize the status to each hyperedge h: $status_i(h) = kp$ if $\exists u, v \in h, status_i(u, v, h) = kp$ (which in turn changes the status of all pairs in h to kp); $status_i(h) = sd$ if $\forall u, v \in h, status_i(u, v, h) = sd$; $status_i(h) = pp$ if $\forall u, v \in h, status_i(u, v, h) = sd$ or pp. At the beginning, all hyperedges and their triples are initialized as pp.

We maintain a clustering structure across k iterations, where the cluster centers Z_i in iteration $0 < i < k$ is sampled from Z_{i-1} with probability $p = f^{1/rk}/n^{1/k}$ (since we allow $f = \Theta(n^r)$), $Z_0 = V$, and $Z_k = \emptyset$. For each vertex $v \in V_{i-1}$, we aim to build a collection of $\Omega(f)$ edge-disjoint $Z_i - v$ paths $Q_i(v)$, where vertices in the paths' first hyperedges $head(\cdot)$, the ones containing a

center in Z_i, are vertex-disjoint. The $\Omega(f)$ edge-disjoint paths from v to Z_i ensure that there are a sufficient number of paths that can reach the centers, resilient to faulty hyperedges. As a key adaptation, the vertex-disjoint requirement is for the size analysis and will be elaborated later. The collection of paths $Q_i(v)$ is then served as input to the next iteration.

In each iteration i, we are given the collections of paths Q_{i-1}, vertices V_{i-1}, the collections R_{i-1} of all hyperedges with status *pp*, cluster centers Z_{i-1}, and current hyperspanner H_{i-1}. We first get a random sample $S_{i-1}(v)$ of $O(\log n)$ paths from $Q_{i-1}(v)$ (Line 6) and then for each vertex in V_{i-1}, process its hyperedges in R_{i-1} in the sorted order of increasing weight (Line 11). For each triple u, v, h of status *pp*, if there exists a path $P \in S_{i-1}(u)$ whose hyperedges are disjoint with those in $P_{i-1}(v)$ and head hyperedge is vertex-disjoint with head hyperedges of paths in $P_{i-1}(v)$, then we add the concatenation of h and P to $P_{i-1}(v)$ and set h to status *kp*. We also adopt an early-stopping strategy as follows: a path in $Q_{i-1}(v)$ is called *sampled* if its head hyperedge has at least one cluster center in Z_i, i.e., sampled from Z_{i-1}. A vertex $v \in V_{i-1}$ is added to V_i if its $Q_{i-1}(v)$ contains $K_f = 12(k+r)f$ sampled paths. When adding a path to $Q_{i-1}(v)$, we also check whether $Z_i \cap \text{head}(P) \neq \emptyset$, maintain the counter n_v of sampled paths accordingly (Line 17), and stop processing subsequent hyperedges of v when we already have K_f sampled paths to save computations (Line 23). Since we process hyperedges of v in the sorted order of increasing weight, we are adding K_f sampled paths with lightest weight on the last hyperedge. When examining triple u, v, h of status *pp*, if we cannot find an eligible path, then we are safe to discard the triple (Line 21). We can prove that h is not needed for vertex pair u, v. Finally, we include paths $P_{i-1}(v)$ for each $v \in V_{i-1}$ into H_{i-1} to get H_i.

We first show two properties of the paths in the clustering structure.

Lemma 3. *(a) For any $v \in V_i$, the weight of hyperedges along the path $P \in Q_i(v)$ (towards v) are monotone increasing. (b) For all $h \in R_i$ and $u, v \in h$, if $status_i(u, v, h) = pp$, then $u, v \in V_i$ and $w(h) > Q_i(u), Q_i(v)$.*

Proof. We prove the two properties together by induction on iteration i. When $i = 0$, Property (a) holds clearly since $Q_0(v) = \emptyset$. Property (b) also holds because both $Q_0(v)$ and $Q_0(u)$ are empty. Assume both properties hold for iteration $i-1$. In iteration i, since $v \in V_i$, for any new formed path $P \in P_{i-1}(v)$, by construction, it is a concatenation of a hyperedge $h \in E_{i-1}(v) \in R_{i-1}$ and a path $P' \in Q_{i-1}(u)$, where $u \in V_{i-1}$ and $status_{i-1}(u, v, h) = pp$. By the inductive hypothesis of Property (b), we get $w(h) > Q_{i-1}(u)$. By the inductive hypothesis of Property (a) on u, the weight of hyperedges along the path $P' \in Q_{i-1}(u)$ is monotone increasing. Then the weight of hyperedges along the concatenation $h \circ P' = P$ is also monotone increasing. Because $Q_i(v) \subseteq P_{i-1}(v)$, Property (a) holds for iteration i.

Since $h \in R_i$, by definition, $h \notin H_i$, then the if-test in Line 13 cannot be passed. In addition, the else-part of the if-test cannot be executed as otherwise, the status of all triples with h would be *sd*, contradicting with $status_i(u, v, h) = pp$. Therefore, it must be that h is not processed in the for-loop of Line 11 in both u and v's turn. Taking v as an example, we have $|Q_i(v)| = K_f$ and $v \in V_i$. Because $E_{i-1}(v)$ is sorted by increasing weight, $w(h) > w(LastEdge(P))$ for $P \in P_{i-1}(v)$. Since the weight of hyperedges for paths in $P_{i-1}(v)$ are monotone increasing as proved above, $LastEdge(P)$ is the heaviest hyperedge in the path P. Therefore, we have $w(h) > Q_i(v)$. By symmetry, we get $u, v \in V_i$ and $w(h) > Q_i(u), Q_i(v)$, completing the proof of Property (b). □

We show that for a hyperedge $h \in R_{i-1} \backslash R_i$, if $h \notin P_{i-1}(v)$, then h can be safely discarded, which means for every $u, v \in h$ and faulty hyperedges F of size at most f, it holds that $\delta_{H_i \backslash F}(u, v) \leq (2k-1)w(h)$. Define $P_{i-1}(v, h)$ be the current $P_{i-1}(v)$ before examining h.

Algorithm 1 iCompute

Input: $Q_{i-1}, V_{i-1}, R_{i-1}, Z_{i-1}, H_{i-1}$
Output: Q_i, V_i, R_i, Z_i, H_i

if $i \leq k-1$ **then**
  $Z_i \leftarrow Z_{i-1}[f^{1/rk}/n^{1/k}]$
else
  $Z_i \leftarrow \emptyset$
for all $v \in V_{i-1}$ **do**
  $S_{i-1}(v) \leftarrow$ a random sample of $O(\log n)$ paths from $Q_{i-1}(v)$
for all $v \in V_{i-1}$ **do**
  $n_v \leftarrow 0$
  $P_{i-1}(v) \leftarrow Q_{i-1}(v)$
  $E_{i-1}(v) \leftarrow \{h \in R_{i-1} \mid v \in h\}$ sorted by increasing weight
  for all $h \in E_{i-1}(v)$ **do**
    for all $u \in h$ **such that** getStatus$(u, v, h) = pp$ **do**
      if there exists $P \in S_{i-1}(u)$ such that $E(P) \cap E(P_{i-1}(v)) = \emptyset$ and $V(head(P)) \cap V(head(P_{i-1}(v))) = \emptyset$ **then**
        $P_{i-1}(v) \leftarrow P_{i-1}(v) \cup (h \circ P)$
        setStatusKp(h)
        if $Z_i \cap \text{head}(P) \neq \emptyset$ **then**
          $n_v \leftarrow n_v + 1$
          $Q_i(v) \leftarrow Q_i(v) \cup (h \circ P)$
        break
      else
        setStatusSd(u, v, h)
    if $n_v = K_f = 12(k+r)f$ **then**
      break
$V_i \leftarrow \{v \in V_{i-1} \mid |Q_i(v)| = K_f = 12(k+r)f\}$
$Q_i \leftarrow \{Q_i(v) \mid v \in V_i\}$
$H_i \leftarrow \bigcup_{v \in V_{i-1}} P_{i-1}(v) \cup H_{i-1}$
$R_i \leftarrow \{h \notin H_i \mid$ getStatus$(h) = pp\}$
return H_i

Lemma 4. *For a hyperedge $h \in R_{i-1} \backslash R_i, u, v \in h$ with $status_{i-1}(u, v, h) = pp$, if $h \notin P_{i-1}(v)$, then at least half of the paths in $Q_{i-1}(u)$ share a hyperedge or have head hyperedge sharing vertices with those of $P_{i-1}(v, h)$ w.h.p.*

Proof. By contradiction, assume at least half of the paths in $Q_{i-1}(u)$ don't share a hyperedge with $P_{i-1}(v, h)$ and head vertices don't intersect *w.h.p.* Since we use uniform random sampling for $Q_{i-1}(u)$ to get $S_{i-1}(u)$, we have at most $1/2$ probability to sample a path that share a hyperedge or have head hyperedge sharing vertices with those of $P_{i-1}(v, h)$. Then, we have a probability of at most $1/2^{O(\log n)} = 1/n^c$ (for $c > 1$) that all the sampled paths share a hyperedge or have head hyperedge sharing vertices with those of $P_{i-1}(v, h)$. Hence, there exists a path $P \in S_{i-1}(u)$ that doesn't share a hyperedge and doesn't have head hyperedge sharing vertices with those of $P_{i-1}(v, h)$ *w.h.p.* In the algorithm, we will add the path $h \circ P$ to $P_{i-1}(v)$, which violates the assumption that $h \notin P_{i-1}(v)$. □

Similar to the key structural lemma in [2], we show there are $2f + 1$ paths between u and v for triple u, v, h with status *sd*.

Lemma 5. *For h, u, v defined in Lemma 4, there exist $2f+1$ distinct pairs of $(Q_j, P_j) \subset Q_{i-1}(u) \times$*

$P_{i-1}(v,h)$ such that $E(Q_j) \cap E(P_j) \neq \emptyset$ or $V(head(Q_j)) \cap V(head(P_j)) \neq \emptyset$ between u,v with total weight at most $(2i-1)w(h)$. All Q_j are edge-disjoint and all P_j are edge-disjoint.

Proof. By Lemma 4, we know there exist at least $\frac{1}{2}$ paths of $Q_{i-1}(u)$ that share a hyperedge or have head hyperedge sharing vertices with those of $P_{i-1}(v,h)$. We want to find $2f+1$ distinct pairs of $(Q_j, P_j) \subset Q_{i-1}(u) \times P_{i-1}(v,h)$ such that $E(Q_j) \cap E(P_j) \neq \emptyset$ or $V(head(Q_j)) \cap V(head(P_j)) \neq \emptyset$.

Greedily pick the (Q_j, P_j) pairs in $2f+1$ iterations. Let $A_1 = \{Q \mid Q \in Q_{i-1}(u), E(Q) \cap E(P_{i-1}(v,h)) \neq \emptyset \vee V(head(Q)) \cap V(head(P_{i-1}(v))) \neq \emptyset\}$, $B_1 = P_{i-1}(v,h)$. In each iteration, given A_j, B_j, first randomly pick $Q_j \in A_j$, and then randomly pick $P_j \in B_j$ that shares a hyperedge or has head hyperedge sharing vertices with that of Q_j to form (Q_j, P_j). Then let $B_{j+1} = B_j \backslash \{P_j\}$, $A_{j+1} = \{Q \in A_j \mid E(P_j) \cap E(Q) = \emptyset \wedge V(head(P_j)) \cap V(head(Q)) = \emptyset\}$. To ensure we can find a valid pair in each iteration j, we need to show (1) $|A_j| > 0$ and (2) for every $Q \in A_j$, $E(B_j) \cap E(Q) \neq \emptyset \vee V(head(B_j)) \cap V(head(Q)) \neq \emptyset$.

Since $|P| \leq k$ for $P \in P_{i-1}(v,h)$, P can share at most k hyperedges with A_1. Because paths in A_1 are edge-disjoint, P can share hyperedges with at most k paths in A_1. Since the head of P_j contains at most r vertices, and all the heads in A_1 are vertex-disjoint, P_j can share vertices in head hyperedges with at most r paths in A_1. Thus, P_j can share hyperedges or share vertices in head hyperedges with at most $k+r = c$ paths in A_1. So $|A_j| \geq |A_1| - (k+r)j \geq 4(k+r)f - (k+r)(2f-1) = (k+r)(2f-1) > 0$.

We prove (2) by induction. The base case holds by the definition of A_1 and B_1. Assume (2) holds for j. Let P_j be the path we delete in B_j. By the way we maintain A_{j+1} and B_{j+1}, it is easy to see that $\forall Q \in A_{j+1}$, $E(B_{j+1}) \cap E(Q) \neq \emptyset \vee V(head(B_{j+1})) \cap V(head(Q)) \neq \emptyset$.

Since all paths in $Q_i(u)$ are edge-disjoint and all paths in $P_i(v,h)$ are edge-disjoint, then all Q_j are edge-disjoint and all P_j are edge-disjoint. Because $status_{i-1}(u,v,h) = pp$, by Lemma 3(b), we have $w(h) > Q_{i-1}(u)$. According to the fact that $P_{i-1}(v,h)$ is the data structure before processing h and the monotone increasing property in Lemma 3(a), we get $w(h) > LastEdge(P_{i-1}(v,h)) \geq P_{i-1}(v,h)$. Because $|Q_j| \leq i-1$ and $|P_j| \leq i$, it holds that $w(Q_j) + w(P_j) \leq (2i-1)w(h)$. Therefore, there exist $2f+1$ paths between u, v of weight at most $(2i-1)w(h)$. □

Unlike that each pair of paths shares vertices ($V(Q_j) \cap V(P_j) \neq \emptyset$) in [2], we require that Q_j and P_j share hyperedges or their head hyperedges share vertices, both of which result in an edge-disjoint path between u and v. With these, we are now ready to prove the stretch bound.

Theorem 4 (Stretch)**.** *The spanner H' returned by Algorithm 3 is an f-EFT $(2k-1)$-hyperspanner of H.*

Proof. For any two vertices u, v in H, let π be their shortest path in $H \backslash F$. We need to show that there exists a path connecting them of weight at most $(2k-1)w(\pi)$ in $H' \backslash F$. For every hyperedge $h \in \pi$ missing from H', we show for every $x, y \in h$, we can find a path in $H' \backslash F$ of weight at most $(2k-1)w(h)$ between x and y. To see this, since $R_0 = E(H)$ and $R_k = \emptyset$, there must exist a $j \in [k]$ such that $h \in R_{j-1} \backslash R_j$. By Lemma 5, we build $2f+1$ paths between x and y. Because all Q_j are edge-disjoint and all P_j are edge-disjoint, a hyperedge can appear in at most two pairs of paths. Given a fault set F of size $|F| \leq f$, we must have at least one pair of paths that is unaffected by F. Thus, we can always have a path between x and y of weight at most $(2i-1)w(h)$. Therefore, for every $x, y \in h$ and $h \notin H'$, we can obtain a path of weight $(2k-1)w(\pi)$ by replacing each missing hyperedge by its corresponding path in H', completing the proof. □

For the size bound, we will first bound the number of hyperedges added in each iteration. Referring to Line 26, this depends on the size of $P_{i-1}(v)$. The following lemma helps achieve the bound.

Lemma 6. *Fix a vertex $v \in V_{i-1}$, and let $P_{i-1}(v) = \{P_1, ..., P_l\}$ sorted in increasing weights of their last hyperedges. If $|P_{i-1}(v)| > 16(k+r)f^{1-1/(rk)}n^{1/k}\log n$, then w.h.p, the number of sampled paths in iteration i, $Y = |\{P \in P_{i-1}(v) \mid head(P) \cap Z_i \neq \emptyset\}|$ must have $Y > K_f$.*

Proof. By construction, a path $P_j \in P_{i-1}(v)$ is sampled if its head hyperedge h_j contains a center $z \in Z_i$. Let $c_{i-1}^{h_j} = Z_{i-1} \cap h_j$ be the set of centers from Z_{i-1} in h_j. Since $P_j \in P_{i-1}(v)$, we must have in iteration $i-1$, $|c_{i-1}^{h_j}| \geq 1$. To get h_j sampled in iteration i, we need $|c_i^{h_j}| \geq 1$. Since each center is sampled independently with probability p, $\Pr(|c_i^{h_j}| \geq 1) = 1 - Pr(|c_i^{h_j}| = 0) = 1 - (1-p)^{|c_{i-1}^{h_j}|}$. Let X_j be the indicator of whether $|c_i^{h_j}| \geq 1$ for each path $P_j \in P_{i-1}(v)$. Since head hyperedges of the paths in $P_{i-1}(v)$ are vertex disjoint, the random variables X_j for $1 \leq j \leq l$ are i.i.d. Then we can use Chernoff bound to analyze the probability of $Y = \sum X_j > K_f = 12(k+r)f$.

By definition,

$$E[X_j] = Pr(|c_i^{h_j}| \geq 1) = 1 - (1-p)^{|c_{i-1}^{h_j}|} \geq p.$$

Here the last inequality holds since $|c_{i-1}^{h_j}| \geq 1$.

$$E[Y] = \sum_{j=1}^{|P_{i-1}(v)|} E[X_j] \geq |P_{i-1}(v)| \cdot p > 16(k+r)f \cdot \log n.$$

The last inequality follows since $|P_{i-1}(v)| > 16(k+r) \cdot f^{1-1/rk}n^{1/k} \cdot \log n$ and $p = f^{1/rk}n^{-1/k}$.

By Chernoff bound, $\Pr(Y < (1-\epsilon)E[Y]) \leq exp((-\epsilon^2/2)E[Y]) = \frac{1}{n^{8(k+r)f\epsilon^2}}$. Because $k, r \geq 2, f \geq 1$, define $\epsilon \leq \frac{1}{4}$, then $\frac{1}{n^{8(k+r)f\epsilon^2}} \leq \frac{1}{n^2}$. Thus, *w.h.p*, $Y \geq \frac{3}{4}E[Y] = 12(k+r)f \cdot \log n > K_f = 12(k+r)f$.

We now prove that even in the last iteration k, where we directly assign $Z_k = \emptyset$ without sampling, the statement still holds. Since $|Z_{k-1}| = O(np^{k-1}\log n) = O((k+r)f/p \cdot \log n)$ with high probability, and head hyperedges of the paths in $P_{k-1}(v)$ are vertex disjoint, we have $O((k+r)f/p \cdot \log n)$ paths to add in the last iteration for each vertex. □

Theorem 5 (Size Bound)**.** *The spanner H' has size $O(k(k+r)f^{1-1/(rk)}n^{1+1/k}\log n)$ w.h.p.*

Proof. To bound the output H' in the final iteration k, we focus on each iteration i and bound the size of $H_i \backslash H_{i-1}$, i.e., $(\bigcup_{v \in V_{i-1}} P_{i-1}(v)) \backslash H_{i-1}$. Since every such path is formed by a concatenation of a path from $Q_{i-1}(v)$ and a hyperedge in R_{i-1}, and $Q_{i-1}(v)$ is already in H_{i-1} (as $Q_{i-1}(v) \subset P_{i-2}(v)$), we only need to bound how many hyperedges in R_{i-1} are being added, that is, $|P_{i-1}(v)|$.

In our algorithm, we have the number of sampled paths $Y = n_k \leq K_f$ ($n_k = K_f$ for vertices in V_i and $n_k < K_f$ for vertices not in V_i). According to the contrapositive of Lemma 6, we get that $|P_{i-1}(v)| \leq 16(k+r)f^{1-1/(rk)}n^{1/k}\log n$. Then we add $O((k+r)f^{1-1/(rk)}n^{1+1/k}\log n)$ edges in each iteration i. Therefore, summing over k iterations gives us $O(k(k+r)f^{1-1/(rk)}n^{1+1/k}\log n)$ edges. □

Running Time Analysis We show that Algorithm 1 takes $\widetilde{O}(mr^3 + fn)$ time. First, sampling cluster centers takes $(|Z_i|) = \widetilde{O}(n)$ time and sampling paths S_{i-1} also takes $\widetilde{O}(n)$ time. For each vertex v, we loop through its associated hyperedges h in the outer for-loop and each vertex u of h (with status *pp*) in the inner for-loop, which result in $deg(v) \cdot r$ iterations. Then the total number of iterations for all vertices is mr^2. When storing the status of all triples in $O(mr^2)$ space, checking and setting statuses only incur constant time. In each iteration, we check each of the $O(\log n)$ paths in $S_{i-1}(u)$ whether its hyperedges intersect with $P_{i-1}(v)$ and its head hyperedge has vertices

sharing with those of $P_{i-1}(v)$. Since each path has at most k hyperedges and a hyperedge has size at most r, this takes $O(k+r)$ time. Similarly, checking the intersection of the head hyperedge and Z_i takes $O(r)$ time. In addition, we process each vertex and construct $\Omega(f)$ paths in Q_i, which takes $O(nf)$ time. Therefore, Algorithm 1 incurs $\widetilde{O}(mr^2(k+r)+fn)$. Summing over k iterations, the construction of EFT hyperspanners (Algorithm 3) incurs $\widetilde{O}(mr^2k(k+r)+fkn)$ time. Considering k is typically $O(\log n)$, the runtime can be simplified as $\widetilde{O}(mr^3+fn)$.

3 Lower Bound

Inspired by the lower bound technique of [16], we devise a lower bound for *f-EFT* $(2k-1)$-hyperspanners. We assume the following conjecture by Spiro and Verstraëte from [24]:

Conjecture 1. *For all $l \geq 3$, $r \geq 2$ and $k = \lfloor l/2 \rfloor$, the maximum number of hyperedges in r-uniform hypergraphs with girth at least $l+1$ is $n^{1+1/k-o(1)}$.*

Proof. (**Theorem 2**) Consider an unweighted r-uniform hypergraph H with girth at least $2k+2$. According to Spiro and Verstraëte's Conjecture, it contains at least $n^{1+1/k-o(1)}$ edges. We create a new hypergraph H' with $V(H') = V(H) \times [t]$ for $t = f^{1/r}$, i.e., each vertex u in H has t copies $(u,1),(u,2),\cdots,(u,t)$ in H'. Then for each hyperedge $\{u_1,u_2,\cdots,u_r\} \in H$, we create a hyperedge $\{(u_1,i_1),(u_2,i_2),\cdots,(u_r,i_r)\}$ for every $i_1,i_2,\cdots,i_r \in [t]^r$, i.e., $E(H') = \{\{(u_1,i_1),(u_2,i_2),\cdots,(u_r,i_r)\} \mid \{u_1,u_2,\cdots,u_r\} \in H, i_1,i_2,\cdots,i_r \in [t]^r\}$.

We prove that H' is the only f-EFT $(2k-1)$-hyperspanner of itself. Let S be a subgraph of H' missing some edge $h = \{(u_1,i_1),(u_2,i_2)\cdots,(u_l,i_l)\}$. We then let the fault set F be all hyperedges of H' created for the hyperedge $\{u_1,u_2,\cdots,u_r\} \in H$ except the missing hyperedge h. Formally, $F = \{(u_1,i'_1),(u_2,i'_2)\cdots,(u_l,i'_l) \mid \exists j \in [r], i_j \neq i'_j\}$. Note that h is the only hyperedge connecting any pair of its vertices (u_j,i_j) and (u_k,i_k), as otherwise there would be a cycle of length 2. Let P be the shortest path between (u_j,i_j) and (u_k,i_k) in $S \setminus F$ after the failure event. P cannot contain h since $h \notin S$. P cannot contain other hyperedges created for $\{u_1,u_2,\cdots,u_r\}$ since they are in the fault set F. Therefore, the distance of P must be at least $2k+1$, because the girth is at least $2k+2$. In contrast, h guarantees that the distance between (u_j,i_j) and (u_k,i_k) in $H \setminus F$ is 1. Hence, S cannot be an f-EFT $(2k-1)$-hyperspanner of H' and the only f-EFT $(2k-1)$-hyperspanner of H' is itself.

The size of H' is

$$\begin{aligned}
|E(H')| &= t^r|E(H)| = \Omega(t^r|V(H)|^{1+1/k-o(1)}) \\
&= \Omega(t^r(|V(H')|/t)^{1+1/k-o(1)}) \\
&= \Omega(t^{r-1-1/k}n^{1+1/k-o(1)}) \\
&= \Omega(f^{1-1/r-1/(rk)}n^{1+1/k-o(1)}).
\end{aligned}$$

Here the second equality follows from Conjecture 1. □

When $r=2$ in the graph setting, our lower bound can degenerate to the known lower bound for EFT graph spanners [16]. Since graphs are a special case of hypergraphs, a size lower bound for VFT hyperspanners $\Omega(f^{1-1/k}n^{1+1/k})$ follows from the graph case.

4 Additive EFT Hyperspanners

We provide an approach to constructing additive EFT spanners in hypergraphs, complementing the multiplicative results. Specifically, we show the connection between multiplicative and additive

EFT spanners in simple graphs [25] can be generalized to hypergraphs. Similar to the multiplicative case, one can also apply the approach of associated graphs to additive EFT hyperspanners, but the resulting surplus would contain the factor fr^2 (number of faults). Our algorithm adapted from [25] only includes a factor of fr, reducing a term r in the surplus. Additive EFT hyperspanners can be constructed by taking the union of multiplicative EFT hyperspanners and additive fault-free hyperspanners. The corresponding result in simple graphs was shown in [25] and we show a careful adaptation suffices to make the translation hold in hypergraphs as well. The main adaptation is to change the way of grouping vertex pairs into different classes. In simple graphs, this is based on the first faulty edges on the shortest path between a vertex pair in the spanner, with the direction of traversal considered, resulting in $2f$ total number of classes. In hypergraphs, the classification is based on vertex-hyperedge pairs, i.e., the first faulty hyperedge and its first vertex on the shortest path in the hyperspanner. The total number of classes fr is then used to get the desired surplus. With the established connection, we can use the multiplicative EFT hyperspanners in Theorem 1 and additive hyperspanners obtained from any known additive spanners plugged into Corollary 1. For example, in a hypergraph of rank $r = 3$ and maximum weight 1, combining an $(f = 2)$-EFT 3-hyperspanner and additive 2-hyperspanner yields an additive 2-EFT hyperspanner of surplus 38. The full details can be found in the Appendix.

5 Conclusions and Future Work

We provide a comprehensive study on multiple constructions of edge fault tolerant spanners in hypergraphs, revealing that achieving sublinear size is nontrivial. In fast time $\widetilde{O}(mr^3+fn)$, sublinear-sized EFT hyperspanners can be constructed by our clustering-based algorithm, improving previous linear size bounds. In developing additive EFT hyperspanners, our focus was on establishing the connections between multiplicative and additive EFT hyperspanners, instead of checking the adaptability of individual methods. Therefore, there might be methods resulting in improved additive surplus (after adaptation), such as [26]. Since this is only the first work for FT hyperspanners, there are quite a few open questions: 1) it is interesting to close the gap with the size lower bound $\Omega(f^{1-1/r-1/rk}n^{1+1/k-o(1)})$. In particular, can we get an improved size bound from the greedy algorithm, since it has contributed to many optimality results in FT spanners? 2) How to construct VFT hyperspanners of sublinear size and what is the size lower bound for general r? Note that the size lower bound $\Omega(f^{1-1/k}n^{1+1/k})$ for VFT spanners is different from the best known size lower bound $\Omega(f^{1/2-1/(2k)}n^{1+1/k}+fn)$ for EFT spanners. 3) Following FT spanner research, after settling the optimality of the size, it is an interesting direction to develop optimal-time algorithms for fast constructions.

References

[1] N. Bansal, O. Svensson, and L. Trevisan, "New notions and constructions of sparsification for graphs and hypergraphs," in *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2019, pp. 910–928.

[2] M. Parter, "Nearly optimal vertex fault-tolerant spanners in optimal time: sequential, distributed, and parallel," in *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, ser. STOC 2022. New York, NY, USA: Association for Computing Machinery, 2022, p. 1080–1092. [Online]. Available: https://doi.org/10.1145/3519935.3520047

[3] D. Peleg and A. Schaffer, "Graph spanners," *Journal of Graph Theory*, vol. 13, no. 1, pp. 99–116, 1989.

[4] I. Althofer, G. Das, D. Dobkin, D. Joseph, and J. Soares, "On sparse spanners of weighted graphs," *Discrete Computational Geometry*, vol. 9, pp. 81–100, 1993.

[5] M. Thorup and U. Zwick, "Approximate distance oracles," *Journal of the ACM*, vol. 52, no. 1, pp. 1–24, 2005.

[6] D. Peleg and E. Upfal, "A trade-off between space and efficiency for routing tables," *Journal of the ACM (JACM)*, vol. 36, no. 3, pp. 510–530, 1989.

[7] B. Awerbuch and D. Peleg, "Network synchronization with polylogarithmic overhead," in *Proceedings [1990] 31st Annual Symposium on Foundations of Computer Science*. IEEE, 1990, pp. 514–522.

[8] M. Elkin, Y. Emek, D. A. Spielman, and S.-H. Teng, "Lower-stretch spanning trees," *SIAM Journal on Computing*, vol. 38, no. 2, pp. 608–628, 2008.

[9] A. Petruschka, S. Sapir, and E. Tzalik, "Color fault-tolerant spanners," 2023. [Online]. Available: https://arxiv.org/abs/2311.08868

[10] S. Chechik, M. Langberg, D. Peleg, and L. Roditty, "Fault-tolerant spanners for general graphs," *SIAM Journal on Computing*, vol. 39, no. 7, pp. 3403–3423, 2010.

[11] G. Bodwin, M. Dinitz, and C. Robelle, "Partially optimal edge fault-tolerant spanners," arXiv preprint arXiv:2102.11360, 2021.

[12] S. Chechik, M. Langberg, D. Peleg, and L. Roditty, "Fault-tolerant spanners for general graphs," in *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing (STOC)*, 2009.

[13] M. Dinitz and R. Krauthgamer, "Fault-tolerant spanners: better and simpler," in *Proceedings of ACM PODC Conference*, 2011, pp. 169–178.

[14] G. Bodwin, M. Dinitz, and C. Robelle, "Optimal vertex fault-tolerant spanners in polynomial time," in *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2021.

[15] G. Bodwin, M. Dinitz, A. Koranteng, and L. Wang, "Light edge fault tolerant graph spanners," 2025. [Online]. Available: https://arxiv.org/abs/2502.10890

[16] G. Bodwin, M. Dinitz, M. Parter, and V. V. Williams, "Optimal vertex fault tolerant spanners (for fixed stretch)," in *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2018, pp. 1884–1900.

[17] G. Bodwin and S. Patel, "A trivial yet optimal solution to vertex fault tolerant spanners," in *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, ser. PODC '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 541–543. [Online]. Available: https://doi.org/10.1145/3293611.3331588

[18] M. Dinitz and C. Robelle, "Efficient and simple algorithms for fault-tolerant spanners," in *Proceedings of the 39th Symposium on Principles of Distributed Computing*, 2020, pp. 493–500.

[19] K. Oko, S. Sakaue, and S.-i. Tanigawa, "Nearly tight spectral sparsification of directed hypergraphs," in *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023, pp. 94–1.

[20] M. Pathak, Y. Sabharwal, and N. Gupta, "Spanners in hypergraphs," in *2024 IEEE 31st International Conference on High Performance Computing, Data and Analytics Workshop (HiPCW)*. IEEE, 2024.

[21] G. Goranci and A. Momeni, "Fully dynamic spectral sparsification of hypergraphs," *arXiv preprint arXiv:2502.01421*, 2025.

[22] I. Althöfer, G. Das, D. P. Dobkin, D. Joseph, and J. Soares, "On sparse spanners of weighted graphs," *Discrete & Computational Geometry*, vol. 9, no. 1, pp. 81–100, 1993.

[23] M. Parter, A. Petruschka, S. Sapir, and E. Tzalik, "Parks and recreation: Color fault-tolerant spanners made local," in *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2025, pp. 4061–4094.

[24] S. Spiro and J. Verstraëte, "Counting hypergraphs with large girth," *Journal of Graph Theory*, vol. 100, no. 3, pp. 543–558, 2022.

[25] G. Braunschvig, S. Chechik, and D. Peleg, "Fault tolerant additive spanners," in *Graph-Theoretic Concepts in Computer Science*, ser. Lecture Notes in Computer Science, M. C. Golumbic, M. Stern, A. Levy, and G. Morgenstern, Eds. Berlin, Heidelberg: Springer, 2012, vol. 7551, pp. 328–339. [Online]. Available: https://doi.org/10.1007/978-3-642-34611-8_22

[26] D. Bilò, F. Grandoni, L. Gualà, S. Leucci, and G. Proietti, "Improved purely additive fault-tolerant spanners," in *Algorithms-ESA 2015: 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*. Springer, 2015, pp. 167–178.

[27] R. Ahmed, G. Bodwin, F. Darabi Sahneh, S. Kobourov, and R. Spence, "Weighted Additive Spanners," in *International Workshop on Graph-Theoretic Concepts in Computer Science*, 2020, pp. 401–413.

[28] A. La and H. Le, "New weighted additive spanners," *arXiv preprint arXiv:2408.14638*, 2024.

[29] M. Elkin, Y. Gitlitz, and O. Neiman, "Improved weighted additive spanners," *Distributed Computing*, vol. 36, no. 3, pp. 385–394, 2023.

A Adaptation of the Peel-off Method [10]

Algorithm 2 Adapted Peel-off Algorithm [10] for f-EFT $(2k-1)$-Hyperspanner

Input: $H = (V, E)$
$E' \leftarrow \emptyset$
for $i = 1$ to $f + 1$ **do**
    $(V, E'_i) \leftarrow spanner(H \backslash E', k)$
    $E' = E' \cup E'_i$
return $H' \leftarrow (V, E')$

First, we prove the correctness of the algorithm.

Lemma 7. *Algorithm 2 returns a f-EFT k-hyperspanner H' of H.*

Proof. For an arbitrary $F \subseteq E(H), |F| < f$, for any pair of vertices $u, v \in V(H)$, let π be the minimum path of u, v in $H \backslash F$. Hence, we want to show that for any hyperedge and $h_i \in \pi$, for any $u_i, v_i \in h_i, dist_{H' \backslash F}(u_i, v_i) < (2k-1)w(h_i)$.

If $h_i \in E(H') \backslash F$, then we are good because the stretch threshold trivially holds. If not, it means that for any $u_i, v_i \in h_i$, we can find $f+1$ alternative path such that the length is smaller than $(2k-1)w(h_i)$. Specifically, it is possible that $u_i, v_i \in h_i^0, .., h_i^n$ where $w(h_i^0) \leq w(h_i^1) \leq \cdots \leq w(h_i^n)$ and assume $h_i^i = h_i$, but we could simply treat each hyperedge as a u_i, v_i path. Notice that spanner in each iteration are edge disjoint. If for all $f+1$ iterations, $w(h_i)$ doesn't appear as the minimum u_i, v_i path length, then in each iteration j we have a u_i, v_i path that satisfy $dist_{H' \backslash E'}(u_i, v_i) < (2k-1) \times dist_{H \backslash E'}(u_i, v_i) < (2k-1)w(h_i)$. And because h_i is not included in the spanner, there's no concern that we skip the comparison to $(2k-1)w(h_i)$ as $h_i \in H \backslash E'$. And if it appears in iteration j, then in the previous iterations the same argument hold that we can find $j-1$ u_i, v_i-path within the threshold. And from iteration j, since h_i is not included in the spanner, it means that we still can find u_i, v_i-path satisfy $dist_{H' \backslash E'}(u_i, v_i) < w(h_i)$ until the end.

Since $|F| \leq f, H' \backslash F$ can fail at most f u_i, v_i-path, which means we always have one path unaffected and within the threshold. Therefore, $dist_{H' \backslash F}(u, v) = \sum_{(u_i, v_i) \in h_i \in \pi} dist(u_i, v_i) < \sum_{h_i \in \pi} (2k-1)w(h_i) = (2k-1) \times dist_{H \backslash F}(u, v)$. □

Then, we show the size of the spanner.

Lemma 8. *The spanner size returned by Algorithm 2 is $O(fn^{1+\frac{1}{k}})$.*

Proof. Lemma 2 states that there exists a (2k-1)-hyperspanner of size $O(n^{1+1/k})$. Since the algorithm takes the union of f such hyperspanner, the output has size $O(fn^{1+1/k})$. □

B Missing Proof and Algorithm

Algorithm 3 f-EFT $(2k-1)$-Hyperspanner

Input: $H(V, E)$
Output: H'
$Q_0 \leftarrow \emptyset$; $V_0 \leftarrow V$; $R_0 \leftarrow E$; $Z_0 \leftarrow V$; $status \leftarrow \{\}$; $H_0 \leftarrow \emptyset$

Helper functions:
function getStatus(u, v, h)
    if $status[h] = kp$ **then**
        return kp
    else if $(u, v) \in status[h]$ or $(v, u) \in status[h]$ **then**
        return sd
    else
        return pp

function setStatusSd(u, v, h)
    add (u, v) to $status[h]$

function setStatusKp(h)
    $status[h] \leftarrow kp$

for $i \in [1, k]$ **do**
    $(Q_i, V_i, R_i, Z_i, H_i) \leftarrow$ iCompute$(Q_{i-1}, V_{i-1}, R_{i-1}, Z_{i-1}, H_{i-1})$
return $H' \leftarrow H_k$

Lemma 9. *Let G be an associated graph of a hypergraph H with a function $f : E(G) \to E(H)$ mapping each edge in G to its corresponding hyperedge in H. If G' is an α-spanner of G for $\alpha > 1$, then the sub-hypergraph H' of H with the set of hyperedges $E(H') = \{f(e') \mid e' \in E(G)\}$ is an α-hyperspanner of H. This also works when G is a simple associated graph of H.*

Proof. For any pair of vertices u, v in G, by the definition of spanner, we have a path P' in G' of distance at most $\alpha \cdot d_G(u, v)$. Let P be a path obtained by replacing each edge e' in P' with its corresponding hyperedge $f(e')$. It is easy to see that P is a path between u and v in H' of distance at most $\alpha \cdot d_G(u, v) = \alpha \cdot d_H(u, v)$. When G is a simple associated graph, we still have $d_G(u, v) = d_H(u, v)$ since removed edges must not be on any shortest path in the original associated multi-graph. This completes the proof. □

Lemma 10. *Assuming the Erdös Girth conjecture, every hypergraph has a $(2k-1)$-hyperspanner of size $\Theta(n^{1+1/k})$ for $k \geq 1$.*

Proof. By applying the algorithm of Althöfer et al. [4] and Lemma 1, we get the upper bound $O(n^{1+\frac{1}{k}})$. Since graphs are a special case of hypergraphs, the lower bound $\Omega(n^{1+\frac{1}{k}})$ follows from the Erdös Girth Conjecture. □

C EFT Additive Hyperspanners

In the section, we use our multiplicative EFT hyperspanner construction to build an additive hyperspanner of a weighted graph. The following techniques are adapted from the approach in [25].

While [25] considered unweighted graphs, we perform an extension to weighted setting and alter some proofs, resulting in different bounds from their results.

C.1 Preliminaries

Let $H = (V, E, w)$ be a weighted hypergraph. Denote $SP(u, v, H)$ as the shortest path between nodes u and v in H and $\delta_H(u, v)$ as its distance. For a path P, let the position of v on the path be $p(v)$. Namely, if v is the first vertex, then $p(v) = 0$. For a pair of vertices (v, u), we say that it is ordered before another pair if the pair $(p(v), p(u))$ comes before the other in lexicographic order. Denote $P[x, y]$ as the subpath of P from x to y. Since we are in the weighted setting, $|P|$ denotes the sum of edge weights along the path P. We let $H \setminus F$ denote the graph $H' = (V, E \setminus F, w)$ for a fault set $F \subseteq E$. We let f be the size of the maximum fault set and $\tilde{f}$ as the actual amount of edges that fail.

C.2 Algorithm

The algorithm proceeds by initializing an additive hyperspanner and a fault tolerant multiplicative hyperspanner. Then, it returns the union of these two hyperspanners, S. The size of S is $|S_1 \cup S_2|$, which depends on the specific constructions chosen in Lines 1, 2.

Algorithm 4 EFT-Additive Hyperspanner

Input: $H = (V, E, w)$
Output: $S = (V, E', w)$
1: $S_1 = Span(+, \alpha, H)$
2: $S_2 = Span(\times, \mu, H)$
3: $S = S_1 \cup S_2$
4: **return** S

The function Span$(+, \alpha, H)$ constructs a weighted additive hyperspanner for H with surplus α. By Corollary 1, one can translate all prior results for weighted spanners in graphs into an algorithm for the function. Candidate weighted spanners include $+2W_{max}$ spanner with $O(n^{3/2})$ edges [27], $+4W_{max}$ spanner with $\widetilde{O}(n^{7/5})$ edges [27], $+6W_{max}$ spanner with $\widetilde{O}(n^{4/3})$ edges [28], and $+(6+\epsilon)W_{u,v}$ spanner with $O(n^{4/3}/\epsilon)$ edges [29], where W_{max} is the maximum edge weight in the input graph and $W_{u,v}$ is the maximum edge weight on the shortest path between u and v. The function Span$(\times, \mu, H)$ constructs an f-EFT μ-multiplicative hyperspanner of the weighted hypergraph H, which can be our proposed algorithm in Section 2.

C.3 Correctness

Consider a source vertex and target vertex pair $(s, t) \subseteq V$ and a set of $\tilde{f}$ hyperedge faults $F = \{e_1, e_2, \ldots e_{\tilde{f}}\}$. Let $P = SP(s, t, H \setminus F)$ be the shortest path from s to t in the input hypergraph H after the failure event. Our goal is to prove that the hypergraph S returned by Algorithm 4 is an EFT additive hyperspanner of H up to a constant: $\alpha' = O(\tilde{f}r(\alpha + \mu W_{s,t}))$. In the context of EFT additive hyperspanners, the local weight $W_{s,t}$ is the maximum edge weight on the shortest path $SP(s, t, H \setminus F)$.

We categorize vertex pairs (u, v) on P into classes as follows: Suppose that e is the first faulty hyperedge on the path $SP(u, v, S)$, and that this hyperedge is first reached by vertex s. Then, this pair is of class (s, e). If the bypass does not use any faulty hyperedge, we say that it is of class Φ.

We first derive useful distance guarantees for shortest distances on some vertex pairs on P.

Lemma 11. *If the pair $(u, v) \subseteq P$ is of class Φ, then:*

$$\delta_{S\setminus F}(u, v) \leq |P[u, v]| + \alpha.$$

Proof.

$$\begin{aligned}\delta_{S\setminus F}(u, v) = \delta_S(u, v) &\leq \delta_H(u, v) + \alpha \\ &\leq \delta_{H\setminus F}(u, v) + \alpha.\end{aligned}$$

□

Lemma 12. *Let x_1, x_2 and y_1, y_2 be two pairs of vertices on path P of the same class (v, e) and $p(x_1) < p(x_2) \leq p(y_1) < p(y_2)$. Then,*

$$\delta_{S\setminus F}(x_1, y_1) \leq |P[x_1, y_1]| + 2\alpha$$

Proof. Consider the bypass $B_x = SP(x_1, x_2, S)$ and $B_y = SP(y_1, y_2, S)$. Consider the subpaths: $B_1 = B_x[x_1, v]$, $B_2 = B_x[v, x_2]$, $B_3 = B_x[y_1, v]$, $P_1 = P[x_1, x_2]$, $P_2 = P[x_2, y_1]$ (See Figure 1). By definition of the class (v, e), the paths B_1, B_3, which are prefix paths leading to the faulty hyperedge, do not contain any faults. Therefore,

$$\delta_{S\setminus F}(x_1, y_1) \leq |B_1| + |B_3| \tag{1}$$

Since S contains S_1, and S_1 is an additive hyperspanner of H, $\delta_S(w_1, w_2) \leq |Q| + \alpha$ for any two nodes w_1, w_2 and any path Q from w_1 to w_2 in H. In particular,

$$|B_1| + |B_2| + w(e) \leq |P_1| + \alpha \tag{2}$$

and

$$|B_3| \leq w(e) + |B_2| + |P_2| + \alpha \tag{3}$$

Using Equations 1, 2, and 3, we get

$$\begin{aligned}\delta_{S\setminus F}(x_1, y_1) &\leq |B_1| + |B_3| \\ &\leq |B_1| + w(e) + |B_2| + |P_2| + \alpha \\ &\leq |P_1| + |P_2| + 2\alpha \\ &= |P[x_1, y_1]| + 2\alpha\end{aligned}$$

□

We then consider the first vertex pair (x_1, x_2) and the last pair (y_1, y_2) in the same class (except Φ) and bound the distance $\delta_{S\setminus F}(x_1, y_1)$.

Lemma 13. *Let (x_1, x_2) be the first pair in lexicographic order of a class other than Φ and let this class be (v, e). Let (y_1, y_2) be the last pair in P of class (v, e). Then,*

$$\delta_{S\setminus F}(x_1, y_1) \leq |P[x_1, y_1]| + 2\alpha$$

Proof. In the first case, $p(y_1) < p(x_2)$. Then, (x_1, y_1) is of class Φ and by Lemma 11,

$$\delta_{S\setminus F}(x_1, y_1) \leq |P[x_1, y_1]| + \alpha \leq |P[x_1, y_1]| + 2\alpha$$

In the second case, we have $p(y_1) \geq p(x_2)$. Thus, by Lemma 12,

$$\delta_{S\setminus F}(x_1, y_1) \leq |P[x_1, y_1]| + 2\alpha$$

□

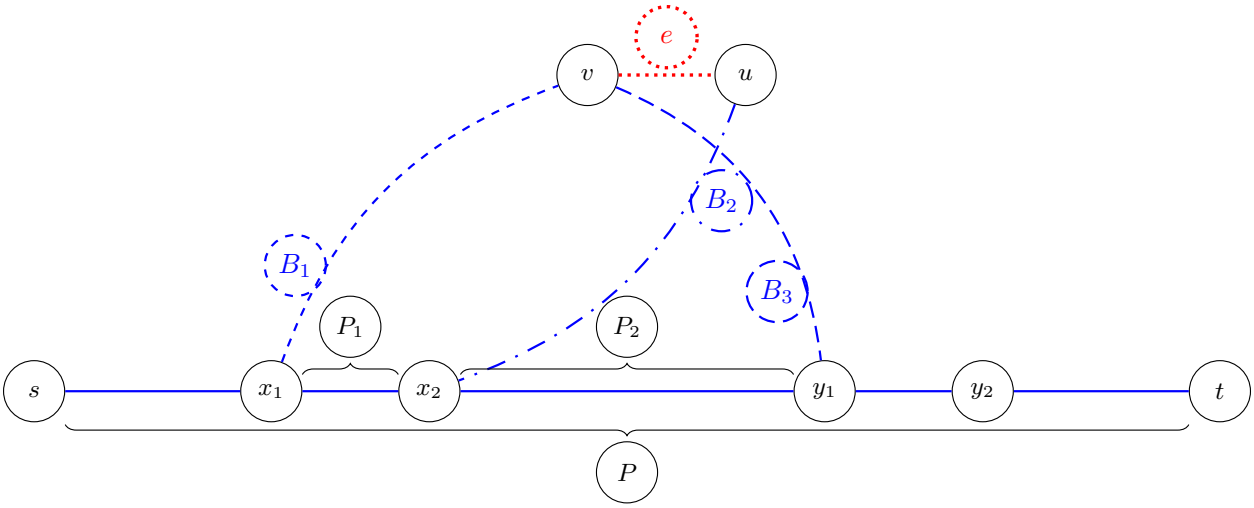

Figure 1: Bypasses of class (v, e). $P = SP(s, t, H \setminus F)$, $B_x = SP(x_1, x_2, S)$ and $B_y = SP(y_1, y_2, S)$. Hyperedge e containing v and u belongs to the faulty set.

Next, we derive the important distance $\delta_{S\setminus F}(s, s_1)$ that can be used for induction.

Lemma 14. *Let (x_1, x_2) be the first pair in lexicographic order of class (v, e) and (y_1, y_2) be the last pair of this class. Let s_1 be the neighbor of y_1 on path $P[y_1, y_2]$. Then, either $\delta_{S\setminus F}(s, s_1) \leq |P[s, s_1]| + 2\alpha + (\mu - 1)W_{s,t}$ or $\delta_{S\setminus F}(s, t) \leq |P| + \alpha$.*

Proof. If the class of (s, t) is Φ, then the bypass from s to t in S contains no faults. So,

$$\delta_{S\setminus F}(s, t) = \delta_S(s, t) \leq |P| + \alpha$$

Now, suppose (s, t) is not of class Φ. Then, $x_1 = s$ since otherwise $p(x_1) > p(s)$ which is in contradiction with the (x_1, x_2) bypass being the first non Φ class pair. By Lemma 13,

$$\delta_{S\setminus F}(s, y_1) \leq |P[s, y_1]| + 2\alpha \tag{4}$$

Since S contains S_2 which is a $f-$EFT μ multiplicative hyperspanner, we have that

$$\delta_{S\setminus F}(y_1, s_1) \leq |P[y_1, s_1]| \cdot \mu = |P[y_1, s_1]| + (\mu - 1)|P[y_1, s_1]| \leq |P[y_1, s_1]| + (\mu - 1)W_{s,t} \tag{5}$$

Combining Equations 4 and 5, we get

$$\begin{aligned}
\delta_{S\setminus F}(s, s_1) &\leq \delta_{S\setminus F}(s, y_1) + \delta_{S\setminus F}(y_1, s_1) \\
&\leq |P[s, y_1]| + |P[y_1, s_1]| + 2\alpha + (\mu - 1)W_{s,t} \\
&= |P[s, s_1]| + 2\alpha + (\mu - 1)W_{s,t}
\end{aligned}$$

□

Lemma 15. *Let N be the number of classes on $SP(s, t, H \setminus F)$. Then,*

$$\delta_{S\setminus F}(s, t) \leq \delta_{H\setminus F}(s, t) + N(2\alpha + (\mu - 1)W_{s,t}) + \alpha$$

Proof. We proceed by induction. For the base case $N = 0$, we have that (s, t) is in class Φ and our claim holds by Lemma 14. Now, suppose our statement is true for $n < N$. By Lemma 14, either,

$$\delta_{S\setminus F}(s,t) \leq |P| + \alpha$$

or

$$\delta_{S\setminus F}(s,s_1) \leq |P[s,s_1]| + 2\alpha + (\mu-1)W_{s,t}$$

Note that $P[s_1, t]$ does not contain any pair of class (v, u), therefore, the number of classes on $P[s_1, t]$ is $< N$ and since $P[s_1, t]$ is the shortest path from $s_1 \to t$ in $H \setminus F$, the induction hypothesis holds. Then,

$$\begin{aligned}
\delta_{S\setminus F}(s,t) &\leq \delta_{S\setminus F}(s,s_1) + \delta_{S\setminus F}(s_1,t) \\
&\leq \delta_{H\setminus F}(s,s_1) + \delta_{H\setminus F}(s_1,t) + (2\alpha + (\mu-1)W_{s,t}) + \\
&(N-1)(2\alpha + (\mu-1)W_{s,t}) + \alpha \\
&= \delta_{H\setminus F}(s,t) + N(2\alpha + (\mu-1)W_{s,t}) + \alpha
\end{aligned}$$

□

Since the number of classes is at most fr, the output S is an f-EFT additive hyperspanner of H with surplus $fr(2\alpha + (\mu-1)W_{s,t}) + \alpha$ and size $|E(S_1 \cup S_2)|$.