Distributed Computing Column

Stefan Schmid
Aalborg University, Denmark
schmiste@cs.aau.dk

Fault-Tolerant Logical Network Structures

Merav Parter (MIT)
parter@mit.edu[1]

Abstract

The undisrupted operation of logical structures and services is a crucial requirement in modern day communication networks. As the vertices and edges of the network may occasionally fail or malfunction, it is desirable to make those structures robust against failures.

Fault-Tolerant (FT) Logical Network Structures are low cost high *resilient* structures, constructed on top of a given network, that satisfy certain desirable performance requirements concerning, e.g., connectivity, distance or capacity.

In this survey, we review some *recent* results for designing FT network structures. We classify the existing construction algorithms into four classes and discuss the settings in which a given approach might become useful. We hope that this would pave the way towards the development of a more generalized theory for the design of fault-tolerant networks.

1 Introduction

The design of FT logical network structures can be viewed as a subarea of network design in which robustness against faults is taken into considerations. In network design, the general theme is to compute a *logical* structure, on top of a *given* physical network, that possesses some desirable properties. Examples include *graph structures* such as shortest-path trees, spanners, Minimum Spanning Trees (MSTs); *clustered representations* such as partitions and decompositions; and *data structures* such as distance oracles and routing schemes. All these extensively studied structures fall into the following network design framework: given a graph G and a *requirement predicate* $\rho(\mathcal{S}, G)$—e.g., related to connectivity, flows, distances—compute the "cheapest" logical structure $\mathcal{S}$ that satisfies the predicate $\rho(\mathcal{S}, G)$.

In fault-tolerant network design, the objective is to compute a cheap logical structure that is also resilient against faults. Network design has various applications, especially in distributed computing and network algorithms. In these settings, computer networks are modeled by a graph in which the vertices represent processors and the edges represent the communication links. An important aspect

[1]Supported in part by AFOSR FA9550-13-1-0042, NSF Award 0939370-CCF, CCF-1461559, and by the Rothschild and Fulbright Postdoctoral Fellowships.

of distributed systems that is not captured by the above network design framework is the possibility of *failure*. By necessity, the optimal solutions for most classical network design tasks are lean and intentionally avoid redundancy, and hence are not robust against failures, and a deletion of even a single edge may lead to complete loss of functionality. For example, in the cases where the optimal solution is a tree—e.g., a minimum spanning tree or a shortest path tree—a failure of an edge disconnects the tree, and its functionality is lost. To tolerate failures, the optimal solutions for network design tasks should be augmented with additional edges from the base graph G, which will provide some redundancy and flexibility.

To recap, *fault-tolerant network design* is concerned with the following question:

> *How we can save a logical structure from being disconnected and losing functionality by a failing edge/vertex, while at the same time keep the total size of the structure small?*

Roughly speaking, a logical structure is said to be *fault-tolerant* if it satisfies the requirement predicate even when some of its elements (e.g., vertices, edges) fail or malfunction. In this survey we focus on a notion of fault tolerance where for a given bound f on the number of faults, the surviving part of the structure $\mathcal{S} \setminus F$ is required to satisfy the predicate $\rho(\mathcal{S} \setminus F, G \setminus F)$ with respect to the *surviving part* of the original network $G \setminus F$ for *every* faulty-set F of at most f edges (or vertices). This notion was termed *competitive fault tolerance* by Chechik and Peleg [16] and it is discussed further in Sec. 5.

Our Focus. Fault-tolerant network design has become a broad and rich area and as such it is full of variants and dimensions. This survey is focused on the specific context of *constructing logical structures in fault-prone physical networks*. Within this framework, we further limit ourself to the following settings; Our fault model is adversarial, meaning that the structure should maintain their functionality against *every* possible fault setting. Moreover, we focus on logical structures that are *subgraphs*, usually denoted by $H \subseteq G$.

We mainly consider a notion of fault-tolerance where the basic structure (e.g., as obtained by computing an optimal solution for the standard non-resilient setting) is augmented by adding edges to it so that after the failure of some of the network's vertices or edges, the surviving part of the structure is still operational. As this augmentation carries certain costs, it is desirable to minimize the number of the added edges. In particular, our primary cost measure would be the *size* of the resulting subgraphs and the computation time would be of secondary importance.

On the Recent History of Fault-Tolerant Network Structures. In this survey, we focus on a recent line of studies concerning fault-tolerant network *structures*.

In 1998, Levcopoulos, Narasimhan, and Smid [33] introduced the notion of fault tolerance in the context of *geometric* spanners (as will be explained later). Fault-tolerant spanners for *general* graphs were studied later by Chechik at el. [15] in 2009. This last work has led to a line of further work, all aiming at designing *sparse* fault tolerant structures on top of a general input graph, for predicates that are mostly distance related. This survey will give a short overview of these studies.

It is worth noting that the virtue of fault tolerance network structures can be traced back to the ancient graph theoretical notion of edge-connectivity[2]. Indeed, edge connectivity is perhaps the most basic measure of the reliability of the network. Therefore, fault-tolerant networks are usually designed by making them highly connected (which usually requires more than just plain connectivity) [2].

Fault tolerant network design is also related to the area of *survivable network design* introduced in 1969 by Steiglitz, Weiner and Kleitman [51]. In this area, one is concerned with designing low-cost subgraphs that preserve certain connectivity requirements. We provide a more detailed description of this area, as well as a comparison with fault tolerant network design, in Sec. 5.

Two other related problems which have been studied since the early 80's are as follows: the *most vital edges* problems [4], which is defined as computing the k edges whose removal from the structure has the highest effect on the functionality of the structure, and the *sensitivity analysis* problem [53], which focuses on the extent to which a given edge's weight can be perturbed without changing the optimal solution. Both of these problems have been studied in the context of shortest paths and minimum spanning trees (MST's).

We start, in Sec. 2, by presenting a collection of the major problems and structures that have been studied within this framework. In Sec. 3, we classify the current algorithmic construction techniques into four classes. In Sec. 4, we turn to discuss the lower bounds and describe graph families for which introducing fault tolerance requires a high cost. Finally, in Sec. 5, we discuss alternative notions and models of fault tolerance.

2 Introducing Different Fault-Tolerant Graph Structures

2.1 Connectivity

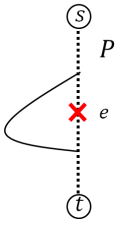
Definition. Fault-tolerant connectivity structures are subgraphs $H \subseteq G$ that provide the same connectivity guarantees as in the original graph for every failure

[2]The minimum number of edges whose failing disconnects the graph.

of f edges or vertices. Formally, given a graph G and a bound f on the number of edge failures, a subgraph $H \subseteq G$ is an *f-FT-connected subgraph* if for every pair $s, t \in V$ and every set of at most f edges F, it holds that if s and t are connected in $G \setminus F$, then they are also connected in $H \setminus F$. The minimum f-FT-connectivity problem concerns the computation of the minimum size f-FT-connected subgraph. These definitions naturally extend to vertex faults.

State of the Art. The *minimum f-FT-connectivity* problem was introduced by Nardelli, Stege, and Widmayer [36] in the geometric setting. Particularly, they considered the single failure case in the Euclidean setting and showed that this problem is NP-hard. They also provided a 4-approximation algorithm (i.e., for the size of the subgraph) that runs in time $O(n \log n)$. Chechik and Peleg [16] considered the general graph setting and provided an $O(\log n)$-approximation algorithm for any constant number f of faults.

The minimum f-FT-connectivity problem is closely related to the area of survivable network design and particularly to the *minimum-cost survivable network* problem [52]. In this problem, one is given a weighted n-vertex graph along with connectivity requirements $r_{i,j}$ representing the number of vertex disjoint paths between vertices i and j in the output subgraph. The goal is to compute the minimum cost subgraph that satisfies the connectivity requirements.



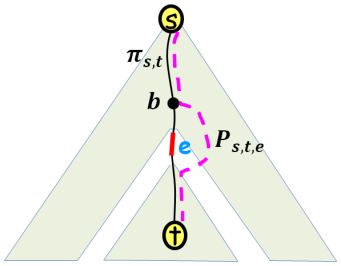
However, we note that these problems are not equivalent, even for the single pair case. To see that, consider the figure to the right. In this case, there are two $s - t$ paths which are non vertex-disjoint, and hence the vertex-connectivity of s and t is 1. Assuming that the path P is cheaper than the other path, the solution for the minimum-cost survivable network problem for $r_{s,t} = 1$ is simply P. This is not a valid solution for the minimum 1-FT-connectivity problem since when e fails, the path P gets disconnected, while there is still an $s - t$ path in $G \setminus \{e\}$.

Directed Connectivity: Fault-Tolerant Reachibility Structures. In the *fault-tolerant reachibility* problem, one is given a directed graph $G = (V, E)$, a source vertex s and an arbitrary directed tree T rooted at s, and the goal is to compute a set of edges $E' \subseteq E \setminus T$ of minimum size such that for every failure of a vertex $v \in V$, the set of vertices reacheable from s in $T \cup E' \setminus \{v\}$ is the same as the set of vertices reachable from s in $G \setminus \{v\}$. These structures have been introduced by Baswana, Choudhary and Roditty [6], who showed an optimal construction with $n - 1$ edges in $O(m \log n)$ time for any given reachability tree T. An $O(m)$-time algorithm was recently given by Georgiadi and Tarjan in [24].

2.2 Replacement Paths

Definition. The replacement path can be viewed as an analogous counterpart of shortest path for the *faulty* setting. Given a graph $G = (V, E)$ and a vertex pair s, t, denote by $\pi_{s,t}$ the shortest path between s and t in G. When an edge e fails, the shortest path $\pi_{s,t}$ survives when $e \notin \pi_{s,t}$. However, when $e \in \pi_{s,t}$, the $s - t$ shortest path in $G \setminus \{e\}$ should be recomputed. The *replacement path* $P_{s,t,e}$ is the shortest path between s and t that avoids e. When drawn on top of the shortest path $\pi_{s,t}$, the replacement path $P_{s,t,e}$ admits a rather convenient form[3], consisting of three segments: a prefix of the shortest-path $\pi_{s,t}$ up to some vertex $b \in \pi_{s,t}$ occurring before the failing edge e, followed by a "detour" avoiding the path $\pi_{s,t}$ (and in particular the failing edge e), and terminating with a suffix of $\pi_{s,t}$.

For a given pair of vertices s and t, the *replacement path* problem asks for computing as fast as possible the collection of all replacement paths $\mathcal{P}_{s,t} = \{P_{s,t,e} \mid e \in \pi_{s,t}\}$. A brute force approach simply treats each replacement path computation independently. However, the clean decomposition of the replacement path has led to the development of algorithms that compute the collection $\mathcal{P}_{s,v}$ efficiently, as discussed next.



State of the Art. In a seminal work, Malik et al. [35] gave an algorithm for solving the replacement path problem with time complexity of $\widetilde{O}(m)$ for undirected graphs on m edges. For directed graphs with arbitrary edge weights, the best bound is $O(mn \cdot n^2 \log\log n)$ due to Gotthilf and Lewenstein, where n and m is the number of graph's vertices and edges respectively. Weimann and Yuster [55] were the first to apply fast matrix multiplication techniques to the problem and obtained a randomized algorithm that runs in time $O(Mn^{2.584})$ for directed graphs with integer weights in $[-M, M]$. These bounds were later improved by Vassilevska-Williams in [56] to $\widetilde{O}(Mn^{\omega})$ where ω is the exponent of matrix multiplication. For a nearly optimal maintenance of replacement-paths in data structures, see [7, 19].

It is worth noting that replacement paths are useful also in other contexts beyond short paths, such as in biological sequence alignment [14], in the computation of Vickrey Prices [29], and in finding the k shortest simple paths between two nodes [48].

[3] upon a proper construction, e.g., breaking shortest-path ties in a consistent manner.

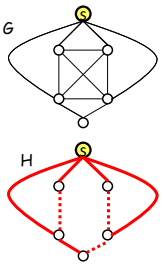
2.3 Fault-Tolerant BFS Structures

Definition. Given a graph G and a source vertex s, a *fault-tolerant BFS* structure (or FT-BFS for short) is a subgraph $H \subseteq G$ containing a BFS tree in $G \setminus \{e\}$ for every failing edge $e \in E(G)$[4].

In other words, FT-BFS structure H with respect to the source s, is required to satisfy the following:

$$\text{dist}(s, t, H \setminus \{e\}) = \text{dist}(s, t, G \setminus \{e\}), \quad \text{for every } \ t \in V \text{ and } e \in E(G).$$

For an illustration see the figure to the right, where the dashed edges are edges added to the BFS tree (solid edges) to make it robust against a single edge failure.



State of the Art. The notion of FT-BFS structures, first introduced in [38], is closely related to the problem of constructing *replacement paths* and in particular to its *single source* variant, the *single-source replacement paths* problem, studied in [28]. 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. Note that the collection $\mathcal{P}_s$ is in fact a FT-BFS structure H and in fact these two notions are equivalent. Whereas the typical *cost* measure when studying replacement paths is the *time complexity*, the main objective when constructing FT-BFS structures is to minimize the *size* of the resulting structure that contains the collection $\mathcal{P}_s$ of all replacement paths given a source node s.

It is shown in [38] that FT-BFS structures require $\Theta(n^{3/2})$ edges, and this number of edges is sufficient. For the dual failure case, Parter showed in [42] a construction with $\Theta(n^{5/3})$ edges; the case of multiple faults for $f \geq 3$ remains open. Table 1 summarizes the cost bounds for FT-BFS and FT-spanners structures.

Table 1: Bounds for f-FT-BFS structures resilient against f edge faults

Number of Faults	Upper Bound	Lower Bound
1	$O(n^{3/2})$ [38]	$\Omega(n^{3/2})$ [38]
2	$O(n^{5/3})$ [38]	$\Omega(n^{5/3})$ [42]
f	?	$\Omega(n^{2-1/(f+1)})$ [42]

The rather dense lower bounds of FT-structures make one wonder how such a price can be avoided. One option is to relax the predicate requirement, for

[4]This definition can be extended to multiple edge or vertex faults.

example by resorting to approximate distances. This approach was taken first by Baswana and Khanna in [5] which provided a $(1+\epsilon)$-approximation for the single source case with $\widetilde{O}(n/\epsilon^3)$ edges. For additional constructions of approximate fault-tolerant structures, see [8, 9, 11, 39].

2.4 Fault-Tolerant Minimum Spanning Tree (FT-MST) Structures

Definition. For a weighted graph $G = (V, E, W)$, the MST represents the cheapest way to connect the network. To tolerate failures, the MST tree should be augmented with as few edges as possible. The *fault-tolerant MST* problem asks for constructing the cheapest subgraph (i.e., with respect to total edge weights) $H \subseteq G$ that contains an MST tree in $G \setminus \{e\}$ for every $e \in E$.

State of the Art. The problem of computing the minimum FT-MST is closely related to the MST sensitivity analysis problem which was introduced by Tarjan in a seminal work [53] in the early 80's. This problem is defined as follows: given a graph G and minimum spanning tree $T = MST(G)$, decide how much each individual edge weight can be perturbed without invalidating the identity of T (i.e., that T is still an MST for the preturbed graph). Tarjan showed that the MST sensitivity analysis can be perform in $O(m\alpha(m,n))$ time, where m in the number of edges, n is the number of vertices, and α is the inverse-Ackermann function. This was later improved by Pettie to an $O(m \log(\alpha(m,n)))$ time algorithm [47].

An FT-MST structure with at most $2(n-1)$ edges can be computed within the same time complexity as that of the MST sensitivity problem. If the weights of the edges are non-unique, Nardelli, Stege and Widmayer showed that the minimum FT-MST problem is NP-hard already for the single failure case in the Euclidean setting [36]. For this setting, they also give a 4-approximation algorithm that runs in time $O(n \log n)$. This problem has been later studied by Chechik and Peleg [16] in the *general* graph setting where they showed an $O(\log n)$-approximation algorithm for an arbitrary constant number f of edge faults.

2.5 Fault-Tolerant Spanners

Definition. Graph spanners are sparse subgraphs that preserve the distances between all pairs up to some multiplicative or additive stretch (cf. [43, 45, 46]). Formally, a multiplicative k-spanner $H \subseteq G$ satisfies that $\text{dist}(s,t,H) \leq k \cdot \text{dist}(s,t,G)$. Introducing fault tolerance, the subgraph H is a fault-tolerant k-spanner if for every $s,t \in V \times V$ and $e \in E(G)$, it holds that

$$\text{dist}(s,t,H \setminus \{e\}) \leq k \cdot \text{dist}(s,t,G \setminus \{e\}) \,. \tag{1}$$

A similar definition applies to f-edge (resp. vertex) fault-tolerant k-spanners which tolerate at most f edge or vertex failures. *Additive* fault-tolerant spanners are defined analogously with the only exception that the stretch on the distances is additive rather than multiplicative.

State of the Art. The notion of fault-tolerant spanners was introduced by Levcopoulos at el. [33] for the geometric setting. They presented an efficient algorithm that given a set S of n points in $\mathbb{R}^d$, constructs an f-vertex fault-tolerant geometric $(1+\epsilon)$-spanner for S, that is, a sparse graph H satisfying that for every faulty set $F \subseteq S$ of size f and any pair of points $u, v \in S \setminus F$, $\text{dist}(u, v, H \setminus F) \leq (1+\epsilon)|uv|$, where $|uv|$ is the Euclidean distance between u and v. A fault-tolerant geometric spanner with optimal maximum degree and total weight was presented in [18].

Later, Chechik et al. [15] provided the first construction of sparse fault-tolerant spanners for *general graphs*. They obtain the following bounds: for the case of f-edge faults and stretch $k' = 2k-1$, they construct an f-edge FT k'-spanner with $O(f \cdot n^{1+1/k})$ edges. Recall that the standard k'-spanner (in the fault free setting) requires $O(n^{1+1/k})$ edges. Hence introducing fault tolerance against up to f edge faults increases the size of the structure by at most an f factor. For the case of f vertex faults, they provide a construction with $\widetilde{O}(f^2 k^{f+1} n^{1+1/k})$ edges. This result was later improved by Dinitz and Krauthgamer [21] to $\widetilde{O}(f^{2-1/k} \cdot n^{1+1/k})$ edges; a factor of $\widetilde{O}(f^{2-1/k})$ more edges compared to the standard k'-spanner.

Additive fault-tolerant spanners that are resilient against *edge* failures were defined and studied by Braunschvig, Chechik and Peleg [13], establishing the following general result. For a given n-vertex graph G, let H_1 be an ordinary β-additive spanner for G and H_2 be a fault-tolerant α multiplicative spanner for G resilient against up to f edge faults. Then $H = H_1 \cup H_2$ is a $\beta(f)$-additive fault-tolerant spanner for G, for up to f edge faults, for $\beta(f) = O(f(\alpha + \beta))$. Additive fault-tolerant spanners that are resilient against *single vertex* fault has been studied in [40]. For additional results on FT-spanners and FT-geometric spanners, see [1, 12, 22, 23, 34].

On the Difference to FT-BFS Structures. Note that both FT-BFS structures and FT-spanners are fault-tolerant distance preserving structures, yet they differ on two important features. First, FT-BFS structures are "single source" as they require to maintain the distances from a given source. In contrast, FT-spanners are "all pairs" structures as they aim to preserve the distances between any pair of vertices. In this sense, the FT-BFS requirement seems *easier* to achieve (where "easier" means requires the addition of fewer edges). On the other hand, FT-BFS structures insist on maintaining the *exact* distance whereas FT-spanners allows a multiplicative slack. In this sense, the FT-BFS requirement is *harder* to achieve. As will be revealed later, it turns out that the insistence on exact distances plays a more dominant role and makes the FT-BFS problem significantly harder (i.e.,

FT-BFS structures are much denser than FT-spanners), despite the fact that they only concern a "single source" solution.

The Cost of Introducing Fault Tolerance. The cost of adding fault-tolerance can be measured by the ratio between the number of edges of the FT-structure and the number of edges of the non-resilient structure. This measure depends on the strength of the requirement predicate. Whereas for weak predicates such as connectivity the cost is linear in the number of faults, the cost becomes poly-logarithmic in the graph size n when considering vertex faults and requiring a constant *multiplicative* stretch for the pairwise distances. Strengthening the requirement to maintaining *additive* approximate distances increases the cost significantly, i.e., by factor n^{ϵ} for some $\epsilon \in (0,1)$. Finally, when insisting on *exact* distances, the structures become rather dense already for the single source and single edge failure case, i.e, FT-BFS structures require $\Omega(n^{3/2})$ edges – an increase of factor $\Omega(\sqrt{n})$compared to BFS trees. For a schematic illustration, see Fig. 1.

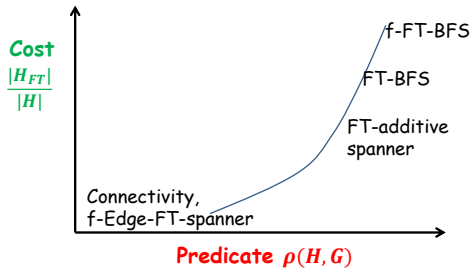


Figure 1: A schematic illustration of the cost of introducing fault tolerance, measured by the ratio between the size bounds of the fault-tolerant structure and the standard (non resilient) structure.

3 Recipes for Algorithms and Upper Bounds

We consider the following general setup: given a graph G, a predicate $\rho(H,G)$ and a bound f on the number of faults, the goal is to output the minimum size subgraph $H \subseteq G$ satisfying $\rho(H \setminus F, G \setminus F)$ for every $F \subseteq G$, $|F| \leq f$. The naïve approach would be to consider the collection of all $\binom{|E|}{f}$ subsets of f faults $F \subseteq E$, and take the union of all subgraphs that satisfy the predicate in $G \setminus F$, for different failing sets F.

Algorithm BruteForceFT(G, ρ)

- $H \leftarrow \emptyset$.
- For every $F \subseteq E_f$
  - Compute H_F satisfying $\rho(H_F, G \setminus F)$.
  - $H \leftarrow H \cup H_F$.

Although the correctness of the output structure is immediate, its size might be quite large as it consists of a union of $O(n^{2f})$ structures, and hence we might end up taking the entire graph. We next describe four approaches in this area that lead to significantly better size bounds.

3.1 The Iterative Approach

In contrast to the $O(n^{2f})$ iterations of Alg. BruteForceFT, the iterative approach consists of only $f + 1$ iterations and hence the size of the output subgraph is increased by factor $O(f)$ compared to its non-resilient counterpart. The main essence of this approach is to compute a collection of $(f + 1)$ *edge*-disjoint solutions for the standard (non-resilient) network design task at hand. Starting with the empty subgraph H, in each iteration i, a subgraph H_i that satisfies the predicate in $G \setminus H$ is computed and added to H. The pseudo code is given below.

Algorithm IterativeFT(G, ρ)

- $H \leftarrow \emptyset$, $G' \leftarrow G$.
- For $i = 1, \ldots, f + 1$:
  - Compute H_i satisfying $\rho(H_i, G')$.
  - $H \leftarrow H \cup H_i$.
  - $G' \leftarrow G' \setminus H_i$.

This approach has been applied to construct fault-tolerant connectivity structures and fault-tolerant spanners that are resilient to f*-edge* faults, as we described next.

Sample Problem: Edge Fault-Tolerant Spanners. Recall that an f-edge FT k-spanner is a subgraph H satisfying Eq. (1) (see Subsec. 2.5). Chechik et al. [15] employed the iterative approach for constructing f-edge FT k'-spanner with $O(f \cdot n^{1+1/k})$ edges for $k' = 2k - 1$. Starting with $H = \emptyset$, in each iteration i, a k'-spanner H_i is constructed in $G \setminus H$ and added to H. Hence, the output subgraph is simply a collection of $(f + 1)$ edge-disjoint k'-spanners, each of which contains $O(n^{1+1/k})$ edges. Whereas the size bound is immediate, the correctness analysis requires some arguments. Consider a pair s, t and a failing set F of at most f

edges and let $P_{s,t,F}$ be the $s-t$ shortest path in $G \setminus F$. It is then required to show that there exists an $s-t$ path in $H \setminus F$ whose length is at most $k' \cdot |P_{s,t,F}|$. Since the stretch is multiplicative, it is sufficient to show that for every edge (u,v) on the path $P_{s,t,F}$ there exists a $u-v$ path of length at most k' in $H \setminus F$. Consider such an edge $(u,v) \in P_{s,t,F}$. If this edge was added into H, the claim is immediate because then $\text{dist}(u,v,H \setminus F) = 1$. However, if (u,v) was not added into the subgraph H, it follows that in each of the $(f+1)$ edge-disjoint k'-spanners that were added to H, there was a $u-v$ path of length at most k' (otherwise, they would add the (u,v) edge to the spanner). Since there are $(f+1)$ edge-disjoint paths in H, each of length $\leq k'$ and at most f failing edges, one of these paths survived the failure of F. We therefore have that $\text{dist}(u,v,H \setminus F) \leq k'$, as required.

When to apply? The main benefit of the iterative approach is its simplicity and the sparsity of the output structures. This approach supports only edge failures and cannot be applied to handle vertex faults. In addition, the iterative approach is useful in cases where the predicate is *edge-sufficient*, i.e., it is sufficient to satisfy the predicate for every two *neighboring* vertices rather than all pairs. Indeed, it can be easily verified that both the multiplicative spanner and the connectivity predicates are edge sufficient.

3.2 The Edge Swapping Approach

The edge swapping approach is one of the most useful methods for introducing fault tolerance into *tree* structures. Consider an optimal tree structure for some given requirement predicate (e.g., MST). When one of the tree edges e fails, the tree is decomposed into two components. The edge swapping approach computes the replacement edge, or the *swap* edge e', that reconnects the tree in the best possible manner, depending on the objective in hand. Regardless of the exact requirement predicate, the output structure of these algorithms consists of at most $2(n-1)$ edges: the $n-1$ edges of the original tree plus (at most) $n-1$ swap edges, one per each of the tree edges. Within this setting there are two main objectives: the first objective concerns the size of the output structure (2-approximation is trivial but maybe one can do better). The second objective concerns the *time* efficiency of the computation.

Algorithm SwapEdgeFT(G, ρ)

- $H \leftarrow T$.
- For every $e \subseteq T$
  - Find the best edge e' satisfying $\rho((T \setminus \{e\}) \cup \{e'\}, G)$.
  - $H \leftarrow H \cup \{e\}$.

Sample Problem: FT-MST. For the definition of FT-MST, see Subsec. 2.5. For a given weighted graph $G = (V, E, W)$ and an MST tree $T \subseteq G$, the best swap edge e' for a tree edge $e \in T$ is the minimum weight edge that reconnects the two disconnected components of $T \setminus \{e\}$ (i.e., the minimum edge that crosses the cut induced by the removal of the edge e in T). It can be easily verified that the new tree $T_e = (T \setminus \{e\}) \cup \{e'\}$ is a valid MST for the remaining graph $G \setminus \{e\}$. Hence, an FT-MST structure can be computed by adding the best swap edge for each of the tree edges into the structure. See Fig. 2 for an illustration of this approach.

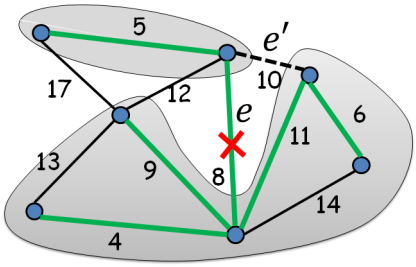


Figure 2: A weighted graph $G = (V, E, W)$. The MST tree T edges are marked in green. When the edge e fails, the tree disconnects into two components. To recover the MST, the lightest edge e' connecting the two components is added to the structure. The edge e' is the *best swap edge* to e since $(T \setminus \{e\}) \cup \{e'\}$ is an MST in $G \setminus \{e\}$.

When to Apply? The edge swapping approach is useful in settings where the optimal non-resilient structure is a tree and the requirnment is to make it robust against a *single edge* fault. Note that this approach does not apply for *every* predicate even if the optimal structure is a tree. For example, an FT-BFS structure cannot be obtained using this technique. Interestingly, Bilò at el. [8] showed that an *approximate* version of FT-BFS[5] can actually be obtained by computing the best swap edges for a proper definition of the objective function. The efficient computation of best swap edges has been studied for minimum diameter spanning tree [26], minimum average distances tree [50] and tree spanners [10].

3.3 The Sampling Approach: Handling Vertex Faults

The sampling approach for FT-network design was first introduced by Weimann and Yuster in [55] to compute efficiently the collection of all replacement paths. It is inspired by the color-coding technique of Alon, Yuster and Zwick [3]. The basic idea of this approach is to sample (many) nodes to act as a faulty set, and then apply the standard algorithm to compute a subgraph that satisfies the predicate on

[5] where the sourcewise distances are preserved up to multiplicative stretch 3.

what remains. These sampling experiments are repeated for polynomially (in the number of faults f) many iterations and the output subgraph is obtained by taking the union over all experiments. The core idea behind this approach is that the sampling of the faulty set in each experiment oversamples nodes, hence instead of computing a structure in a subgraph $G \setminus F$ for $|F| \leq f$, the subgraph is computed in $G \setminus F'$ for a much larger faulty set F', i.e., $|F'| = O(1 - 1/\operatorname{poly}(f)) \cdot n$. This allows one to satisfy many fault sets of size f within a *single* experiment. Alg. SamplingFT(G, ρ) describes the general procedure of this approach.

Algorithm SamplingFT(G, ρ)

- $H \leftarrow \emptyset$.
- For every $i = 1, \ldots, \operatorname{poly}(f)$
  - Add v to F_i with prbability $p_f = 1 - 1/\operatorname{poly}(f)$.
  - Let H_i be the subgraph satisfying $\rho(H_i, G \setminus F_i)$.
  - $H \leftarrow H \cup H_i$.

Sample Problem: Fault-Tolerant (Vertex) Spanners. The sampling approach was used by Dinitz and Krauthgamer [21] for constructing fault-tolerant spanners that are resilient against f *vertex* faults (see Subsec. 2.5). Let Spanner(G, k) be any algorithm for computing k-spanner H for the graph G.

Algorithm VertexFTSpanner(G, ρ)

- $H \leftarrow \emptyset$.
- For every $i = 1, \ldots, O(f^3 \cdot \log n)$
  - Sample faulty set F_i by adding every $v \in V$ to F_i independently with probability $p_f = 1 - 1/f$.
  - $H_i \leftarrow$ Spanner$(G \setminus F_i, k)$.
  - $H \leftarrow H \cup H_i$.

The crucial part of this technique is the correctness argument; the size analysis follows immediately. Consider an $s - t$ pair and a faulty set of F vertices and let $P_{s,t,F}$ be the shortest $s-t$ path in $G \setminus F$. It is sufficient to show that $\text{dist}(u, v, H \setminus F) \leq k$ for every *edge* $e = (u, v) \in P_{s,t,F}$.

We say that iteration i in Alg. VertexFTSpanner(G, ρ) is *good* for the edge $(u, v) \in P_{s,t,F}$, if the sampled faulty set F_i contains both u and v but does not contain any vertex from f. Note that if iteration i is good then the k-spanner H_i constructed in $G \setminus F$ satisfies that $\text{dist}(u, v, H_i) \leq k$ and hence also $\text{dist}(u, v, H \setminus F) \leq k$. Since every vertex v is sampled into F_i independently with probability $p = 1 - 1/f$, the probability that iteration i is good is $p^f \cdot (1 - p)^2 = 1/(e \cdot f^2) \sim 1/f^2$. Since there are $O(f^3 \cdot \log n)$ iterations and $\binom{n}{f} \cdot n^2$ triples u, v, F, the probability that there is no good iteration for one of the triples is at most $n^{f+2} \cdot (1 - 1/f^2)^{c \cdot f^3 \cdot \log n} = O(1/n^{c'})$ for some constant $c' \geq 1$.

When to Apply? The sampling approach works when one can partition the collection of $\binom{n}{f}$ possible sets of (vertex) failures into a much smaller number of groups – and to protect all the failure possibilities in the same group using one basic (non-resilient) structure that is computed in $G \setminus F'$ where F' is the union over all failure possibilities in this group. For example, in the FT-spanner problem, the task boils down to satisfing the desired stretch on the edges (rather than for each vertex pair). Then, to provide a good stretch for an edge (u, v) in the presence of a faulty set F, it is sufficient to require that the much larger faulty set F_i does not contain a failed vertex from F and also that it does not contain the edge endpoints u and v. This approach can also be applicable for edge faults, see [55].

3.4 The Structural Approach

The structural approach is based on Algorithm BruteForceFT(G, ρ) which simply adds into the output subgraph the union of (non-resilient) solutions over all possible fault sets. Whereas the correctness of the algorithm is immediate, the main challenge is to prove that the output structure is *small*. To do that, one needs to zoom in on the structure of the computed subgraphs and to show that they satisfy

some simplifying structural properties. The key intuition that makes this approach successful in certain cases is rooted in the fact that there is a large dependency (e.g., mutual edges) between subgraphs that correspond to different faulty sets F (i.e., subgraphs that are solutions for $G \setminus F$ for different fault sets F), as they are all related to the same base graph G and the number f of faults is mostly bounded by a constant.

Sample Problem: FT-Spanners and FT-BFS. To recall the definitions of FT-Spanners and FT-BFS structures, see Subsec. 2.5 and 2.3, respectively. To construct f-vertex fault-tolerant k'-spanners, for $k' = 2k - 1$, Chechik et al. [15] applied the spanner construction of Thorup and Zwick [49] to every possible fault-set, eventually taking the union of all of these spanners. They showed that the union of these $O(n^f)$ spanners increases the size bound (of the standard spanners) only by an $O(f^2 \cdot k^f)$ factor.

FT-BFS structures are constructed via a similar naïve idea; however, the argumentation is clearly very different. Parter and Peleg [38] showed that one can obtain an FT-BFS structure with $O(n^{3/2})$ edges, simply by doing the following; take the union of m BFS structures, one for subgraph $G \setminus \{e\}$, for each $e \in E$, while breaking ties in a consistent manner.

Limitation of the Structural Approach. The main advantage of the structural approach is the simplicity of the construction algorithm. The size analysis of these simple looking algorithms is unfortunately rather involved. For example, the structural approach of [38] for single failure FT-BFS was extended recently in [42] to the dual failure case. When concerning replacement-paths, there is a sharp qualitative difference between a single failure and two or more failures and hence the extension of the structural approach to multiple faults is far from trivial.

4 Lower Bound Constructions

Unlike the algorithmic side, for which we have many techniques by now, somewhat less is known about lower bounds. To the best of our knowledge, the first lower bound for FT-structures (i.e., one that is stronger than the standard lower bounds for the fault free setting) was given for FT-BFS structures. (In the setting of data structures, other lower bounds were known such as [19], but in this survey we focus on subgraphs).

In this section we thus describe the lower bound construction for FT-BFS structures, which turns out to be the basis for further subsequent constructions as described later on. See Subsec. 2.3 for the definition of FT-BFS structures and recall from Sec. 3.4, that there exists an upper bound of $O(n^{3/2})$ edges on their size. We now sketch the matching lower bound construction [38].

The core of our lower bound graph G is a complete bipartite graph $B(X, Z)$ whose top layer X consists of $\Theta(n)$ vertices and whose bottom layer Z consists of $\Theta(\sqrt{n})$ vertices, hence the graph B consists of $\Theta(n^{3/2})$ edges. Eventually, it will be shown that every FT-BFS structure H for G with respect to a suitably chosen source vertex s must include *each* of the edges of B. This is done by adding a path P rooted at s of length $\Theta(\sqrt{n})$. The last vertex of the path P is connected to each of the vertices in X and every vertex in P (except for the last) is connected to a vertex in Z by a sequence of paths that are vertex disjoint and have monotonically increasing lengths. See Fig. 3 for an illustration.

To see why every FT-BFS structure with respect to the source s must contain all the edges of the bipartite graph B, consider a given edge (x_j, z_i) in B. When the edge e_i on the path P fails (see Fig. 3), the shortest path from s to x_j cannot use the bypasses below the failing edge, although they are shorter than those above it, since they are disconnected from s by the failing. Hence, the $s-x_j$ path must use a bypass that starts *above* the failing edge. As these bypasses are strictly increasing in their lengths, the shortest $s - x_j$ path in $G \setminus \{e_i\}$ takes the first one which uses the edge (z_i, x_j). Since this edge is missing in H, any alternative path in $H \setminus \{e_i\}$ is strictly longer, which contradicts the fact the H is a legal FT-BFS structure (i.e., $\text{dist}(s, x_j, H \setminus \{e_i\}) > \text{dist}(s, x_j, G \setminus \{e_i\})$).

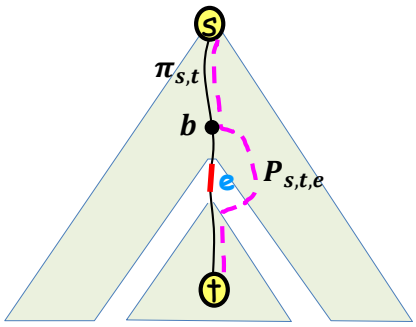


Figure 3: Lower bound graph $G = (V, E)$ for FT-BFS structures. For a subgraph H not containing the edge (x_j, z_i), we get that $\text{dist}(s, x_j, H \setminus \{e_i\}) > \text{dist}(s, x_j, G \setminus \{e_i\})$. Hence all edges of the bipartite graph $B(X, Z)$ must be taken into H.

This basic construction has been extended in [42] to multiple f-faults, to *additive* approximate distances [41] and to other models of fault tolerance, discussed in Sec. 5, cf. [41]. Despite these results, the understanding lower bounds for fault

tolerant structures is still lacking. See Sec. 6, for a concrete open question in this area.

5 Other Notions for Fault-Tolerance

In this section, we review different models for fault tolerance that have been studied in the literature.

Rigid and Competitive Fault Tolerance. As pointed out by Chechik and Peleg [16], fault tolerance can be formalized in two ways. One interpretation of fault tolerance, called *rigid* fault tolerance, requires that the surviving part of the structure $\mathcal{S}$ after the failing of f edges (or vetices) F would satisfy the predicate $\rho(\mathcal{S} \setminus F, G)$ in the *original* network. In the second interpretation, *competitive* fault tolerance, the surviving part of the structure $\mathcal{S} \setminus F$ is required to satisfy the predicate $\rho(\mathcal{S} \setminus F, G \setminus F)$ with respect to the *surviving part* of the original network.

Clearly, rigid fault tolerance is a much stronger requirement than the competitive one. However, in most cases, it is usually asking for more than what possibly can be obtained. For example, consider the predicate of connectivity, where $\rho_{con}(H, G) = 1$ iff the subgraph H is connected. For an 1-edge connected input graph G, one cannot construct a fault-tolerant connected subgraph in the rigid sense, even when taking $H = G$. Rigid fault tolerance can still be obtained in certain cases, such as in the *geometric* setting [36]. Although not realistic in most scenarios, the rigid notion is an important aspect to have in mind when *designing* the underlying networks. In (the more common) case where the underlying network is already *given* and it is required to compute a logical structure on top of that, one has to resort to the notion of competitive fault tolerance which can always be attained.

Survivable Network Design. Steiglitz, Weiner and Kleitman [51] initiated the study of *survivable network design* by defining the minimum cost survivable network problem. Recall that in this problem, one is given a complete weighted n-vertex graph and a collection of demands $r_{i,j}$, each of which requires to have $r_{i,j}$ vertex-disjoint paths between vertices i and j. The goal is to compute a network of minimum total edge weights that satisfies the connectivity demands $r_{i,j}$ for each pair i, j. One of the most extensively studied problems in this area is the *minimum k-connected subgraph* problem which requires to compute the minimum collection of edges (in size or total edge weights) that has vertex or edge connectivity k.

The main difference between fault tolerant network design and survivable network design is the following: in FT-network design, the functionality of the *surviving* part of the subgraph H is compared to that of the surviving part of the original network G, for every failing set F. In other words, it is required that

$H \setminus F$ "behaves" as $G \setminus F$ for *every* set of failing edges F. In contrast, in survivable network design, there is one set of (connectivity related) constraints that the subgraph H should satisfy. Roughly speaking, survivable network design requires to maintain the same "connectivity level" between a given pair s and t—e.g., that the number of vertices, or edges, that disconnect s and t in the subgraph H is the same as that in G—but that still does not guarantee that for *every* failing set F, the "connectivity level" between s and t in $H \setminus F$ is the same as that of $G \setminus F$.

To get a better flavor of this distinction, let us consider two representative problems: the *minimum FT-MST subgraph* problem [16] and the *minimum biconnected subgraph* problem [31, 32]; the former is from the area of FT-network design and the latter is from survivable network design. These two problems may look similar at first glance, however they are not equivalent. In the problem of the *minimum FT-MST subgraph*, it is required to compute a minimum weighted subgraph H that contains an MST tree in $G \setminus \{e\}$ for every failing edge e. In the *minimum biconnected subgraph*, it is required to compute the minimum subgraph (i.e., minimum of the total edge weights) that is biconnected. Note that a solution for the minimum biconnected subgraph is not necessarily a solution for the minimum FT-MST problem and vice versa. This is because survivable network design optimizes for the size of the output subgraph without requiring to obtain the same functionality as in $G \setminus F$. In particular, to obtain a minimum size structure, in the minimum biconnected subgraph, we might end up taking edges that are heavier if they satisfy many cuts. On the other hand, in the FT-MST problem, for every cut induced by a failing set, we must take the *lightest* edge that crosses the cut, even if the output structure H ends up being heavier.

The Fault Model. Two main fault models have been considered in the literature: adversarial and random. In the adversarial model, the failure edges are chosen in an adversarial manner. Hence, the designed structure should be resilient against the failure of *any* set of f edges (or vertices) where the bound f on the number of faults is given as input.

In contrast, in the *random* fault model, edges fail according to some predefined probability distribution and the output structure should maintain its functionality despite these faults with good probability.

In the *network reliability* problem, one is given an n-vertex graph G with m edges, each edge has probability p of failing. The goal is to compute the probability that G becomes disconnected under random, independent edge failures. Karger showed in [30] a fully polynomial randomized approximation scheme (FPRAS) for this problem.

A somewhat more related problem to our setting has been studied by Chechik, Emek, Patt-Shamir and Peleg [17]: given a connected graph G and a failure probability p_e for each edge e in G, it is required to find a sparse backbone that approx-

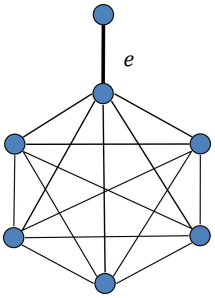
imate the reliability of G, where the reliability of a graph is the probability that the graph remains connected when every edge is removed independently with probability p_e. Their main result was the construction of subgraph with $O(n \log n)$ edges whose reliability is $(1 - n^{\Omega(1)})$ times that of G. For a more detailed description on the random fault model, see [25].

Cost Measures. Fault-tolerant network design mostly concerns two types of cost measures: *computation time* and *size* of the output subgraph (in the weighted case, the size is the total weight of subgraph's edges). The motivation for having time efficient constructions is clear. A typical motivation for having the *size* as the primary complexity measure 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 requirement predicate even when edges fails. 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.

Additional FT-Mechanisms: Reinforcement vrs. Backup. So far, we mainly focused on a notion of fault-tolerance where the structure is augmented by adding to it various edges. However, fault tolerance can be introduced in a different way.

In [41], Parter and Peleg introduced a mixed model that combines two orthogonal protection mechanisms: (a) *backup*, namely, augmenting the structure with many (redundant) low-cost but fault-prone components, and (b) *reinforcement*, namely, acquiring high-cost but fault-resistant components (that would never fail). Such a mixed model changes the way fault-tolerant systems are viewed. In the conventional view, the fault tolerance of a network is a function merely of its topology (i.e, the way the vertices are connected). By allowing one to introduce into the structure components with different levels of resilience, the fault tolerance of the network becomes a complex function of the quality level of its individual components and the way they interact.

To illustrate this point, consider for example an n-vertex network consisting of a single vertex s connected via a single edge e to an $(n-1)$-vertex clique. The edge connectivity of this network is 1, as the removal of e disconnects the graph. Hence the conservative approach of keeping all existing edges leaves this network with a low level of survivability. In contrast, in a mixed model allowing also reinforcements, it is sufficient to reinforce a single edge, namely, e, in order to obtain a high level of survivability, even by purchasing only a fraction of the edges of the clique.



For bypassing the high cost of FT-structures as well as to study the tradeoff between backup in reinforcement, Parter and Peleg [41] consid-

ered the concrete problem of designing a (b, r) fault-tolerant BFS structures. In these subgraphs there are two types of edges: $r(n)$ reinforced edges, which are assumed to never fail, and $b(n)$ fault prone backup edges such that subsequent to the failure of one of the backup edges e the surviving part of H still contains a BFS tree of $G \setminus \{e\}$; see Fig. 4(a) for an illustration. The following tradeoff between $r(n)$ and $b(n)$ has been established: For every $\epsilon \in (0, 1/2]$, if $r(n) = \widetilde{\Theta}(n^{1-\epsilon})$, then $b(n) = \widetilde{\Theta}(n^{1+\epsilon})$. This tradeoff is tight up to polylogarithmic factors, see Fig. 4(b) for an illustration.

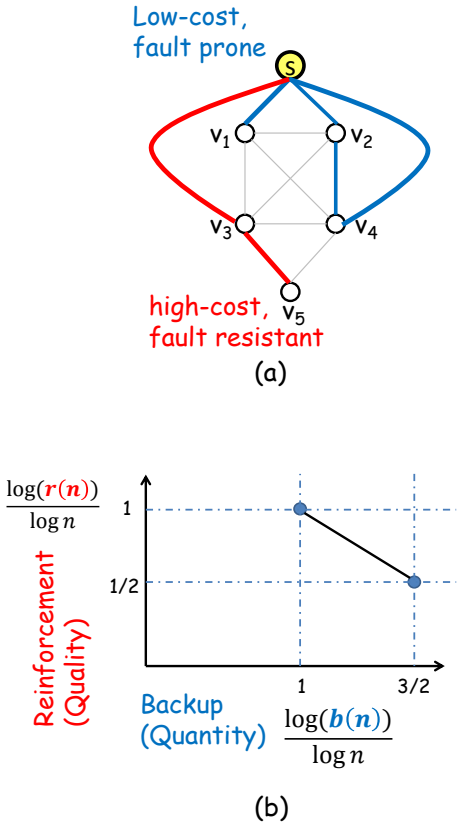


Figure 4: (a) Shown here is a (b, r) FT-BFS structure. The red edges are fault resistance and the blue edges are backup edges. The distances from s are preserved as in the surviving part of the original network when a single blue edge fails. (b) A schematic illustration of the tradeoff between backup and reinforcement. For example, when $r(n) = \Theta(n^{1/2})$, the number of backup edges needed is still $b(n) = \Theta(n^{3/2})$. To see this, consider the lower bound construction of Fig. 3: reinforcing half of the edges on the $s - u$ shortest path still requires $\Theta(n^{3/2})$ backup edges to protect against the failure of an edge the other half.

6 Discussion and Open Problems

The study of FT-network design has experienced much development in recent years. Yet, when considering the many thousands of papers on network design tasks, there is still much work to be done to establish a comprehensive picture for FT-network design.

So far, most studies considered the centralized computation model. However, since FT-structures have a wide range of applications in distributed systems, it is important to devise for them efficient construction algorithms in the *distributed* setting [43].

In the centralized setting, tightening the bounds for the problems mentioned in this survey is a clear direction. An important open problem concerns the construction of FT-BFS structures with $o(n^2)$ edges that are resilient against f edge faults, for $f \geq 3$. Beyond that, future research may develop in the following directions.

Approximation algorithms for FT-network design. As the lower bounds section suggests, some of our FT upper bounds are *optimal* as they match the worst-case lower bounds. Yet, these existentially-optimal structures might be rather dense with respect to the best that can be achieved for the *specific* input graph. In particular, the $\Theta(n^{3/2})$ upper bound for FT-BFS structure might still be far from optimal for certain instances, see [37]. This motivates the study of FT-network design from the combinatorial optimization point of view, attempting at providing the sparsest possible subgraph that can be obtained for the given input graph.

For example, in the optimization version of FT-BFS structures, denoted as the Minimum f-FT-BFS problem, one is given a graph $G = (V, E)$, a constant integer $f \geq 1$, and a source vertex $s \in V$. It is then required to construct an FT-BFS subgraph H of minimum size (i.e., number of edges) resilient against the failure of f edges. By establishing a connection to the Set Cover problem [54], $\Theta(\log n)$ approximation algorithms can be provided [38, 42].

Optimization algorithms for FT-spanners with small stretch $k = 2, 3, 4$ have been studied by Dinitz and Krauthgamer in [21, 23]. The main challenge in this context is to obtain approximation ratios which are *independent* of the number of faults f. This was obtained in [21] by Dinitz and Krauthgamer for $k = 2$. Recently, Dinitz and Zhang [22] obtained this for $k \in \{3, 4\}$, at the cost of turning to bi-criteria approximation.

Note that having an α-approximation algorithm for an FT-structure does not reveal any upper bound on the number of edges that are actually needed to be taken into the structure. Hence, it is important to study the construction of FT-structures from both the approximation guarantee and the existential optimality points of views.

Lower Bounds. In contrast to the richness in approaches to designing upper

bounds and algorithms, our knowledge on the lower bound sides is much narrower. One of the most basic FT-structures that calls for a lower bound construction is *fault-tolerant additive spanners*, namely, subraphs that provide an additive approximation for all pairs distances in presence of up to single (or more) faults. Whereas an upper bound construction of $O(n^{5/3})$ edges was provided in [40] for the single vertex case, no lower bound of $\Omega(n^{3/2+\epsilon})$ for any $\epsilon > 0$ is known.

Beyond Distances. So far, the main predicate requirements considered in FT design tasks are related to distances and connectivity. Future study may consider other graph functions, such as cuts, flows, electrical resistance, etc.

Acknowledgment. I am very grateful to David Peleg for introducing me into the world of fault tolerant network design, for many helpful discussions and reviewing this survey. I would also like to thank Mohsen Ghaffari for fruitful discussions and for his great contribution in reviewing this survey.

References

[1] M.A. Abam, M. de Berg, M. Farshi and J. Gudmundsson. Region-fault-tolerant geometric spanners. *Discrete & Computational Geometry*, 556–582, 2009.

[2] M. Ajtai, N. Alon, J. Bruck, R. Cypher, C.T. Ho, M. Naor and E. Szemerédi. Fault-tolerant graphs, perfect hash functions and disjoint paths. In Proc. *FOCS*, 693–702, 1992.

[3] N. Alon, R. Yuster and U. Zwick. Color-Coding. *J. ACM*, 844–856, 1995.

[4] M.O. Ball, B.L. Golden and R.V. Vohra. Finding the most vital arcs in a network. *Operations Research Letters*, 8(2):73–76, 1989.

[5] S. Baswana and N. Khanna. Approximate Shortest Paths Avoiding a Failed Vertex: Optimal Size Data Structures for Unweighted Graph. In Proc. *STACS*, 513–524, 2010.

[6] S. Baswana, K. Choudhary and L. Roditty. Fault-tolerant Reachability for Directed Graphs. In Proc. *DISC*, 528–543, 2015.

[7] A. Bernstein and D. Karger. A nearly optimal oracle for avoiding failed vertices and edges. In Proc. *STOC*, 101–110, 2009.

[8] D. Bilò, L. Gualà, S. Leucci and G. Proietti. Fault-Tolerant Approximate Shortest-Path Trees. In Proc. *ESA*, 137–148, 2014.

[9] D. Bilò, F. Grandoni, L. Gualà, S. Leucci and G. Proietti. Path-Fault-Tolerant Approximate Shortest-Path Trees. In Proc. *SIROCCO*, 224–238, 2014.

[10] D. Bilò, F. Colella, L. Gualà, S. Leucci and G. Proietti. A Faster Computation of All the Best Swap Edges of a Tree Spanner. In Proc. *SIROCCO*, 239–253, 2015.

[11] D. Bilò, F. Grandoni, L. Gualà, S. Leucci and G. Proietti. Improved Purely Additive Fault-Tolerant Spanners. In Proc. *ESA*, 167–178, 2015.

[12] P. Bose, V. Dujmovic P. Morin and S. Michiel. Robust geometric spanners. *SIAM J. Computing*, 42(4):1720–1736, 2013.

[13] G. Braunschvig, S. Chechik, D. Peleg and A. Sealfon. Fault tolerant additive and (μ, α)-spanners. In *Theor. Comput. Sci.*, 580:94–100, 2015.

[14] T.H. Byers and M.S. Waterman. Determining All Optimal and Near-Optimal Solutions when Solving Shortest Path Problems by Dynamic Programming. *Operations Research*, 32(6): 1381–1384, 1984.

[15] S. Chechik, M. Langberg, D. Peleg, and L. Roditty. Fault-tolerant spanners for general graphs. In Proc. *STACS*, 435–444, 2009.

[16] S. Chechik and D. Peleg. Rigid and competitive fault tolerance for logical information structures in networks. In Proc. *IEEEI*, 2010.

[17] S. Chechik, E. Yuval, B. Patt-Shamir and D. Peleg. Sparse reliable graph backbones. In Proc. *ICALP*, 261–272, 2010.

[18] A. Czumaj and H. Zhao. Fault-tolerant geometric spanners. *Discrete & Computational Geometry*, 32, 2003.

[19] C. Demetrescu, M. Thorup, R. Chowdhury, and V. Ramachandran. Oracles for distances avoiding a failed node or link. *SIAM J. Computing*, 37:1299–1318, 2008.

[20] M. Dinitz and R. Krauthgamer. Directed spanners via flow-based linear programs. In Proc. *STOC*, 2011, 323–332.

[21] M. Dinitz and R. Krauthgamer. Fault-tolerant spanners: better and simpler. In Proc. *PODC*, 2011, 169-178.

[22] M. Dinitz and Z. Zhang. Approximating Low-Stretch Spanners. In Proc. *SODA*, 2016, 821–840.

[23] R. Duan and S. Pettie. Dual-failure distance and connectivity oracles. In Proc. *SODA*, 2009.

[24] L. Georgiadis and R.E. Tarjan. A Note on fault-tolerant Reachability for Directed Graphs. *arXiv preprint arXiv:1511.07741*.

[25] I.B. Gertsbakh and Y. Shpungin. Models of network reliability: analysis, combinatorics, and Monte Carlo. *CRC Press*, 2009.

[26] B. Gfeller, N. Santoro, and P. Widmayer. A Distributed Algorithm for Finding All Best Swap Edges of a Minimum Diameter Spanning Tree. *Dependable and Secure Computing, IEEE Transactions on*, 1–12, 2011.

[27] Z. Gotthilf, N. Santoro, and M. Lewenstein. Improved algorithms for the k simple shortest paths and the replacement paths problems. *Information Processing Letter*, 109(7):352–355, 2009.

[28] F. Grandoni and V.V Williams. Improved Distance Sensitivity Oracles via Fast Single-Source Replacement Paths. In Proc. *FOCS*, 2012.

[29] J. Hershberger and S. Subhash. Vickrey prices and shortest paths: What is an edge worth? In Proc. *FOCS*, 2001.

[30] D.R Karger. A randomized fully polynomial time approximation scheme for the all-terminal network reliability problems. *SIAM review*, 43(3):499–522, 2001.

[31] S. Khuller and U. Vishkin. Biconnectivity approximations and graph carvings. *J. ACM*, 214–235, 1994.

[32] G. Kortsarz and Z. Nutov. Approximating minimum cost connectivity problems. *Approximation Algorithms and Metahueristics*, 2007.

[33] C. Levcopoulos, G. Narasimhan, and M. Smid. Efficient algorithms for constructing fault-tolerant geometric spanners. In Proc. *STOC*, 186–195, 1998.

[34] T. Lukovszki. New results of fault-tolerant geometric spanners. In Proc. *WADS*, 193–204, 1999.

[35] K. Malik, A.K. Mittal, S.K. Gupta The k most vital arcs in the shortest path problem *Operations Research Letters*, 8(4):223–227, 1989.

[36] E. Nardelli, U. Stege, and P. Widmayer. Low-cost Fault-tolerant Spanning Graphs for Point Sets in the Euclidean Plane. Technical report, 1997.

[37] P. Parter and D. Peleg. Sparse Fault-Tolerant BFS Trees. http://arxiv.org/abs/1302.5401, 2013.

[38] M. Parter and D. Peleg. Sparse fault-tolerant BFS Trees. In Proc. *ESA*, 779–790, 2013.

[39] M. Parter and David Peleg. Sparse fault-tolerant approximate BFS Trees. In Proc. *SODA*, 1073–10922014.

[40] M. Parter. Vertex fault-tolerant Additive Spanners. In Proc. *DISC*, 167–181, 2014.

[41] M. Parter and David Peleg. Fault-Tolerant BFS Structures: A Reinforcement-Backup Tradeoff. In Proc. *SPAA*, 264–273, 2015.

[42] M. Parter. Dual Failure Resilient BFS Structure. In Proc. *PODC*, 2015.

[43] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, 2000.

[44] D. Peleg. As good as it gets: Competitive fault tolerance in network structures. In Proc. *SSS*, 35–46, 2009.

[45] D. Peleg and A.A. Schäffer. Graph spanners. *J. Graph Theory*, 13:99–116, 1989.

[46] D. Peleg and J.D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Computing*, 18(2):740–747, 1989.

[47] S. Pettie. Sensitivity analysis of minimum spanning trees in sub-inverse-Ackermann time. *Algorithms and Computation*, 964–973, 2005.

[48] L. Roditty and U. Zwick. Replacement paths and k simple shortest paths in unweighted directed graphs. *ACM Trans. Algorithms* ,2012.

[49] M. Thorup and U. Zwick. Approximate distance oracles. *J. ACM*, 52:1–24, 2005.

[50] A.D. Salvo and P. Guido. Swapping a failing edge of a shortest paths tree by minimizing the average stretch factor. *Theoretical Computer Science*, 2007, 23–33.

[51] K. Steiglitz, P. Weiner and D.J Kleitman. The design of minimum-cost survivable networks. *Circuit Theory, IEEE Transactions on*, 1969, 16(4):455–460.

[52] M. Stoer. *Design of survivable networks*. Lecture notes in mathematics, vol. 1531, Springer-Verlag, 1992.

[53] R.E. Tarjan. Sensitivity analysis of minimum spanning trees and shortest path trees problems. *Information Processing Letters*, 1982, 30–33.

[54] V. Vazirani. Approximation Algorithms. *Georgia Inst. Tech.*, 1997.

[55] O. Weimann and R. Yuster. Replacement paths via fast matrix multiplication. In Proc. *FOCS*, 2010.

[56] V.V. Williams. Faster replacement paths. In Proc. *SODA*, 1337–1346, 2011.