Fault-Tolerant Bounded Flow Preservers

Shivam Bansal
Department of Computer Science and Engineering, IIT Delhi, India

Keerti Choudhary
Department of Computer Science and Engineering, IIT Delhi, India

Harkirat Dhanoa
Department of Computer Science and Engineering, IIT Delhi, India

Harsh Wardhan
Department of Electrical Engineering, IIT Delhi, India

Abstract

Given a directed graph $\mathcal{G} = (V, E)$ with n vertices, m edges and a designated source vertex $s \in V$, we consider the question of finding a sparse subgraph $\mathcal{H}$ of $\mathcal{G}$ that preserves the flow from s up to a given threshold λ even after failure of k edges. We refer to such subgraphs as (λ, k)-fault-tolerant bounded-flow-preserver ((λ, k)-FT-BFP). Formally, for any $F \subseteq E$ of at most k edges and any $v \in V$, the (s, v)-max-flow in $\mathcal{H} \setminus F$ is equal to (s, v)-max-flow in $\mathcal{G} \setminus F$, if the latter is bounded by λ, and at least λ otherwise. Our contributions are summarized as follows:

1. We provide a polynomial time algorithm that given any graph $\mathcal{G}$ constructs a (λ, k)-FT-BFP of $\mathcal{G}$ with at most $\lambda 2^k n$ edges.
2. We also prove a matching lower bound of $\Omega(\lambda 2^k n)$ on the size of (λ, k)-FT-BFP. In particular, we show that for every $\lambda, k, n \geqslant 1$, there exists an n-vertex directed graph whose optimal (λ, k)-FT-BFP contains $\Omega(\min\{2^k \lambda n, n^2\})$ edges.
3. Furthermore, we show that the problem of computing approximate (λ, k)-FT-BFP is NP-hard for any approximation ratio that is better than $O(\log(\lambda^{-1} n))$.

2012 ACM Subject Classification Theory of computation → Data structures design and analysis; Mathematics of computing → Graph algorithms

Keywords and phrases Fault-tolerant Data-structures, Max-flow, Bounded Flow Preservers

Digital Object Identifier 10.4230/LIPIcs.ISAAC.2024.9

Related Version *Full Version*: https://arxiv.org/abs/2404.16217

Funding *Keerti Choudhary*: The author is supported in part by Google India Algorithms Research grant 2021.

1 Introduction

We address the problem of computing single-source fault-tolerant bounded-flow-preservers for directed graphs. The objective is to construct a sparse subgraph that preserves the flow value up to a parameter λ from a given fixed source s, even after failure of up to k edges.

The following definition provides a precise characterization of this subgraph.

▶ **Definition 1.** *Let $\mathcal{G} = (V, E)$ be a directed graph with unit edge-capacities and $s \in V$ be a designated source vertex. A (λ, k)-Fault-Tolerant Bounded-Flow-Preserver ((λ, k)-FT-BFP) for $\mathcal{G}$ is a subgraph $\mathcal{H} = (V, E_{\mathcal{H}} \subseteq E)$ of $\mathcal{G}$ satisfying that for every $F \subseteq E$ of at most k edges, and every $t \in V$,*

$$\textit{MAX-FLOW}(s, t, \mathcal{H} - F) = \begin{cases} \textit{MAX-FLOW}(s, t, \mathcal{G} - F) & \text{if } \textit{MAX-FLOW}(s, t, \mathcal{G} - F) \leqslant \lambda, \\ \textit{At least } \lambda, & \textit{otherwise.} \end{cases}$$

For the special case of $\lambda = 1$, the problem is referred to as k-Fault-Tolerant Reachability Subgraph (k-FTRS) in the literature. Here the goal is to preserve reachability from s after k edge failures. Baswana et al. [4] showed that there exists a k-FTRS with at most $2^k n$ edges. Lokshtanov et al. [17] presented an algorithm for computing a (λ, k)-FT-BFP for directed graphs. Their algorithm runs in time $O(4^{k+\lambda}(k+\lambda)^2(m+n)\cdot m)$, and each vertex of the FT-BFP has in-degree at most $4^{k+\lambda}(k+\lambda)$. They also showed that a $(k+\lambda-1)$-FTRS of a graph $\mathcal{G}$ also serves as it's (λ, k)-FT-BFP. Using this result in conjunction with the algorithm from [4], they obtain an alternate construction of a (k, λ)-FT-BFP with at most $2^{k+\lambda}n$ edges. However, this bound is quadratic in n for any λ larger than $\log n$.

We consider the problem of obtaining a tight bound on (λ, k)-FT-BFP. Specifically, we aim to answer the following question:

Given a directed graph $\mathcal{G} = (V, E)$ with a source s, and a flow threshold $\lambda \geqslant \log n$, can we construct a sparse *(λ, k)-FT-BFP $\mathcal{H} = (V, E_H \subset E)$? If so, can we present graphs for which the construction turns out to be tight?*

In this paper, we affirmatively answer the question above. We provide construction for FT-BFP that has a linear dependence on λ.

1.1 Upper Bound Results and Applications

We prove the following:

▶ **Theorem 2.** *There exists an algorithm that for any directed graph $\mathcal{G}$ on n vertices and m edges, and any integers $\lambda, k \geqslant 1$, computes in $O(\lambda 2^k mn)$ time a (λ, k)-FT-BFP for $\mathcal{G}$ with at most $\lambda 2^k n$ edges.*

We also present an application of our FT-BFP construction in computing an all-pairs fault-tolerant λ-reachability oracle. We show that for any positive constants $\lambda, k \geqslant 1$, we can compute an oracle of $O(n^2)$ size that given any query vertex-pair $x, y \in V$ and any set F of k edge failures, reports (x, y)-λ-reachability in $\mathcal{G} \setminus F$ efficiently.

▶ **Theorem 3.** *Given any directed graph $\mathcal{G} = (V, E)$ on n vertices and any positive constants $\lambda, k \geqslant 1$, we can preprocess $\mathcal{G}$ in polynomial time to build an $O(n^2)$ size data structure that, given any query vertex-pair (x, y) and any set F of k edges, reports the (x, y) λ-reachability in $\mathcal{G} \setminus F$ in $O(n^{1+o(1)})$ time.*

1.2 Lower Bound and Hardness Results

We show that the extremal bound of $\lambda 2^k n$ obtained in Theorem 2 is tight. In particular, we prove existence of n-vertex graphs whose (λ, k)-FT-BFP must contain at least $\Omega(\min \lambda 2^k n, n^2)$ edges.

▶ **Theorem 4.** *For every $\lambda, k, n \geqslant 1$ satisfying $\lambda 2^k = O(n)$, there exists a construction of an n-vertex directed graph whose optimal (λ, k)-FT-BFP contains $\Omega(\lambda 2^k n)$ edges.*

While the lower-bound in above theorem proves that the bound of $\lambda 2^k n$ obtained in Theorem 2 is existentially tight, it does not address the problem of computing a sparsest (λ, k)-FT-BFP.

We next demonstrate the hardness of computing optimal (λ, k)-FT-BFP structures. We show that unless $P = NP$, there is no polynomial-time algorithm to obtain an $O(\log(\lambda^{-1} n))$-approximation to optimal (λ, k)-FT-BFP.

▶ **Theorem 5.** *For any $\lambda, k, n \geqslant 1$ satisfying $k = \Omega(\log(\lambda^{-1}n))$, the problem of computing an $O(\log(\lambda^{-1}n))$ approximation to optimal (λ, k)-FT-BFP for n vertex directed graphs is NP-hard.*

As a corollary, we obtain the following hardness result for the FTRS problem.

▶ **Corollary 6.** *For any $k, n \geqslant 1$ satisfying $k = \Omega(\log n)$, the problem of computing an $O(\log n)$ approximation to optimal k-FTRS for n vertex directed graphs is NP-hard.*

1.3 Existing Works

For undirected graphs, there exists a tight construction for (λ, k)-FT-BFP with $O((k+\lambda)\cdot n)$ edges that directly follows from edge connectivity certificate constructions provided by Nagamochi and Ibaraki [19].

A closely related problem to that of graph preservers is fault-tolerant reachability oracles. For dual failures, the work of [11] obtained an $O(n)$ size single source reachability oracle with constant query time for directed graphs. Brand and Saranurak [23], showed construction of an $\widetilde{O}(n^2)$ sized k-fault-tolerant all-pairs reachability oracle that has $O(k^\omega)$ query time, where ω is the constant of matrix multiplication.

Recently, Baswana et al. [2] considered the problem of constructing a sensitivity oracle for reporting the max-flow value for a single source-destination pair. They presented an $O(n^2)$ size data-structure that after failure of any two edges, reports the max-flow value of the surviving graph in constant time.

For the problem of computing the value of all-pairs max-flow up to λ in the static setting, Abboud et at. [1] obtained two deterministic algorithms that work for DAGs: a combinatorial algorithm which runs in $O(2^{O(\lambda^2)} \cdot mn)$ time, and another algorithm that can be faster on dense graphs which runs in $O((\lambda \log n)4^{\lambda+o(\lambda)} \cdot n^\omega)$ time.

Some other graph theoretic problems studied in the fault-tolerant model include computing distance preservers [12, 21, 20], depth-first-search tree [3], spanners [8, 13], approximate single source distance preservers [5, 22, 6], approximate distance oracles [14, 9], compact routing schemes [9, 7].

2 Preliminaries

Given a digraph $\mathcal{G} = (V, E)$ on $n = |V|$ vertices and $m = |E|$ edges with unit edge capacities, we first define some notations used throughout the paper.

- $\textsc{in}(v, \mathcal{G})$: The set of in-neighbours of v in $\mathcal{G}$.
- $\textsc{out}(v, \mathcal{G})$: The set of out-neighbours of v in $\mathcal{G}$.
- $\textsc{In-Edges}(v, \mathcal{G})$: The set of all incoming edges of v in $\mathcal{G}$.
- $\textsc{Out-Edges}(v, \mathcal{G})$: The set of all outgoing edges of v in $\mathcal{G}$.
- $\textsc{out}(A, \mathcal{G})$: The set of all those vertices in $V \backslash A$ having an incoming edge from some vertex of A in $\mathcal{G}$, where $A \subseteq V(\mathcal{G})$.
- $\mathcal{G}(A)$: The subgraph of $\mathcal{G}$ induced by the vertices lying in a subset A of V.
- $\mathcal{G} + (u, v)$: The graph obtained by adding an edge (u, v) to graph $\mathcal{G}$.
- $\mathcal{G} \setminus F$: The graph obtained by deleting the edges lying in a set F from graph $\mathcal{G}$.
- $\textsc{max-flow}(S, t, \mathcal{G})$: The value of the maximum flow in graph $\mathcal{G}$ from a source set S to a destination vertex t. When the set S comprises of a single vertex, say s, we represent it simply by $\textsc{max-flow}(s, t, \mathcal{G})$.
- $\textsc{path}[a, b, T]$: The path from node a to b in a tree T.

- $P[a,b]$: The subpath of path P lying between vertices a and b, where a precedes b on P.
- $P \circ Q$: The path formed by concatenating paths P and Q in $\mathcal{G}$. Here it is assumed that the last edge (or vertex) of P is the same as the first edge (or vertex) of Q.

We next define the concept of farthest min-cut that was introduced by Ford and Fulkerson in their pioneering work on flows and cuts [15]. Let S be a source set, and t be a destination vertex. Any (S,t)-cut C is a partition of the vertex set into two sets: $A(C)$ and $B(C)$, where $S \subseteq A(C)$ and $t \in B(C)$. An (S,t)-min-cut C^* is said to be the *farthest min-cut* if $A(C^*) \supsetneq A(C)$ for every (S,t)-min-cut C other than C^*. We denote the cut C^* by $\text{FMC}(S,t,\mathcal{G})$. Similar to farthest-min-cut, we can define the nearest min-cut. An (S,t)-min-cut C^* is said to be the *nearest min-cut* if $A(C^*) \subsetneq A(C)$ for every (S,t)-min-cut C other than C^*. We denote the cut C^* by $\text{NMC}(S,t,\mathcal{G})$.

Below we state a property of nearest and farthest (s,t)-min-cuts [15] showing that they can be computed efficiently.

▶ **Property 7.** *Let s be a source vertex, t be a destination vertex, and f be an s to t max-flow in graph $\mathcal{G}$. Let $\mathcal{G}_f$ denote the residual graph corresponding to flow f. Further let X be the set of vertices reachable from s in $\mathcal{G}_f$, and Y be the set of vertices having a path to t in $\mathcal{G}_f$. Then $NMC(s,t,\mathcal{G}) = (X, V \setminus X)$ and $FMC(s,t,\mathcal{G}) = (V \setminus Y, Y)$.*

3 Hardness of logarithmic approximation

We prove in this section the following hardness result for approximating optimal FT-BFP.

▶ **Theorem 8.** *For any $\lambda, k, n \geqslant 1$ satisfying $k = \Omega(\log(\lambda^{-1}n))$, the problem of computing an $O(\log(\lambda^{-1}n))$ approximate (λ,k)-FT-BFP for n vertex digraphs is NP-hard.*

We prove the above theorem by showing a reduction from the SET-COVER problem to the optimal FT-BFP.

▶ **Problem 9** ([18], Definition 1)**.** *The input to SET-COVER consists of base set U, $|U| = n$ and a family $\mathfrak{F} = (S_1, ..., S_m)$ of m subsets of U satisfying $\cup_{j=1}^{m} S_j = U$, $m \leqslant poly(n)$. The goal is to find as few sets $S_{i_1}, ..., S_{i_k}$ as possible that cover U, that is, $\cup_{j=1}^{k} S_{i_j} = U$*

▶ **Lemma 10** ([18], Theorem 2)**.** *For every $0 < \alpha < 1$ (exact) SAT on inputs of size n can be reduced in polynomial time to approximating SET-COVER to within $(1-\alpha)\ln N$ on inputs of size $N = n^{O(1/\alpha)}$.*

From Lemma 10, we can also deduce that it is NP-Complete to approximate SET-COVER up to a multiplicative factor of $c_1 \log \max(n,m)$ for some $c_1 > 0$ as $m \leqslant poly(n)$.

Transformation. Given a SET-COVER instance $\langle U, \mathfrak{F} \rangle$, we will construct a (λ,k)-FT-BFP instance $\langle \mathcal{G}, s \rangle$. The transformation is as follows (also see Figure 1).

1. Round up the number for elements in U to nearest power of 2 (let this be 2^u) by adding $2^u - |U|$ new elements to U and all these new elements to every set in $\mathfrak{F}$.
2. Initialize $\mathcal{G}$ to be the graph with $N+1$ vertices, namely, $s, v_1, \ldots, v_N$ where $N = 4\lambda(m+n)$.
3. Next construct the following subgraph $\mathcal{G}_i$, for each $i \in [1,\lambda]$.
   a. Construct a complete binary tree B_i rooted at a vertex r_i of height u and 2^u leaf nodes. The leaf nodes of B_i will correspond to elements in the universe U. From each leaf node x_i in B_i, add out-edges to two new vertices, namely, $\ell(x_i)$ and $r(x_i)$.

b. For each set $W \in \mathfrak{F}$, add a vertex $y_{i,W}$ to graph $\mathcal{G}_i$. Let Y_i denote the resulting set which consists of $|\mathfrak{F}|$ vertices. For each $x \in U$ and $W \in \mathfrak{F}$, add an edge from $\ell(x_i)$ to $y_{i,W}$ if and only if $x \in W$.

c. Add a set Z_i of $u+1$ additional vertices. For each leaf x_i in B_i, add an edge from $r(x_i)$ to each vertex in the set Z_i.

4. Finally, we add an edge from s to the roots $r_1, \ldots, r_\lambda$. Also for each $i \in [1, \lambda]$, we add an edge from each vertex in $Y_i \cup Z_i$ to each of the vertices $v_1, \ldots, v_N$.

We set $k = u + 1$ for this (λ, k)-FT-BFP instance.

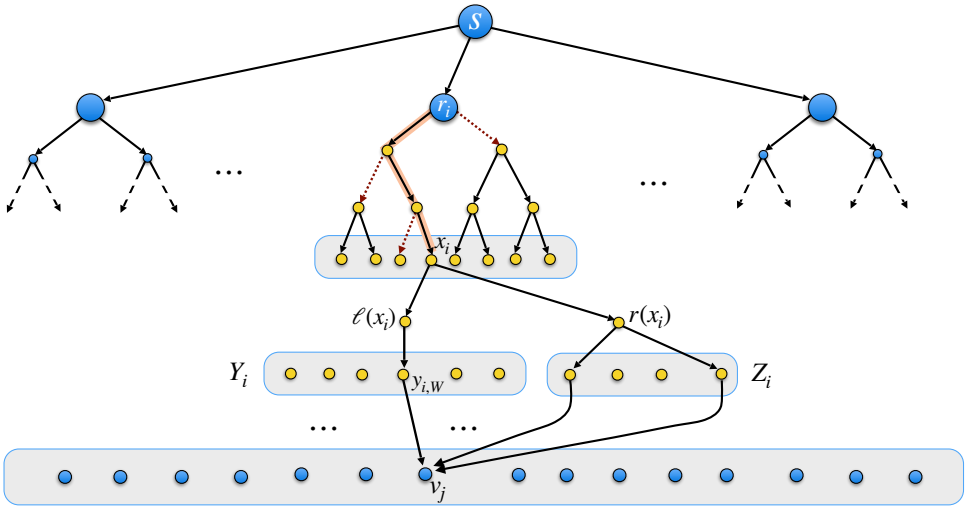

Figure 1 Depiction of a (λ, k)-FT-BFP instance obtained from a SET-COVER instance $\langle U, \mathfrak{F} \rangle$.

▶ **Lemma 11.** *Any (λ, k)-FT-BFP $\mathcal{H}$ of the graph instance $\langle \mathcal{G}, s \rangle$, can be used to construct a solution of the SET-COVER instance of size at most $\lambda^{-1}(\min_{j=1}^{N} |\text{IN}(v_j, \mathcal{H})|)$.*

Proof. Consider a vertex v_j in $\mathcal{H}$ that minimizes $|\text{IN}(v_j, \mathcal{H})|$. Consider the following candidate solutions

$$S_i = \{W \in \mathfrak{F} \mid (y_{i,W}, v_j) \in E(\mathcal{H})\}.$$

Out of the λ sets, namely $S_1, \ldots, S_\lambda$, let S_{i_0} be the set with least cardinality. The cardinality of S_{i_0} is at most $|\text{IN}(v_j, \mathcal{H})|/\lambda$ as minimum value is upper-bounded by the average value.

Now in order to prove that S_{i_0} is a valid solution, consider an element $x \in U$. Let P be the unique path from r_{i_0} to leaf node x_{i_0} in B_{i_0}, and let F_1 be the set of all those edges $(u, v) \in B_{i_0}$ such that $u \in P$ and v is the child of u not lying on P. Observe that x_{i_0} is the unique leaf in B_{i_0} that is reachable from s in $\mathcal{H} \setminus F_1$. Let F_2 be a singleton set comprising of the edge $(x_{i_0}, r(x_{i_0}))$. Consider the set $F = F_1 \cup F_2$ of size k. Since $\text{MAX-FLOW}(s, v_j, \mathcal{G} \setminus F) = \lambda$, there must exists a path, say Q, from s to v_j in $\mathcal{H} \setminus F$ passing through r_{i_0}. Such a path Q must pass through $\ell(x_{i_0})$ as well as a vertex in Y_{i_0}, say $y_{i_0,W}$. This implies that the edge $(y_{i_0,W}, v_j)$ lies in $\mathcal{H}$, and so by definition of S_{i_0}, the set W lies in S_{i_0}. Moreover W contains the element x as $(\ell(x_{i_0}), y_{i_0,W})$ is an edge in $\mathcal{G}$. This proves that element $x \in U$ is covered by S_{i_0}, and thus S_{i_0} is a valid solution to $\langle U, \mathfrak{F} \rangle$. ◀

▶ **Lemma 12.** *Any solution S of the SET-COVER instance $\langle U, \mathfrak{F} \rangle$, can be used to construct a solution $\mathcal{H}$ of (λ, k)-FT-BFP instance satisfying $|\text{IN}(v_j, \mathcal{H})| = \lambda(|S|+k)$, for each $j \in [1, N]$.*

Proof. Let S be a solution of the SET-COVER instance $\langle U, \mathfrak{F} \rangle$. Consider the sets

$$A_i = \{y_{i,W} \mid W \in S\} \cup Z_i, \text{ for } i \leqslant \lambda, \quad \text{and} \quad A = \bigcup_{i=1}^{\lambda} A_i.$$

We will show that

$$\mathcal{H} = \mathcal{G} \setminus \cup_{j=1}^{N} \textsc{In-Edges}(v_j) + \cup_{j=1}^{N} (A \times v_j).$$

is a (λ, k)-FT-BFP of $\mathcal{G}$.

Let us assume, to the contrary, that $\mathcal{H}$ is not a (λ, k)-FT-BFP of $\mathcal{G}$. Then there must exist an edge set F of size at most k and an index $j \in [1, N]$ satisfying $\textsc{max-flow}(s, v_j, \mathcal{G} \setminus F)$ is greater than $\textsc{max-flow}(s, v_j, \mathcal{H} \setminus F)$. Observe that each path from s to v_j must pass through a vertex r_i, for some $i \in [1, \lambda]$, and each r_i only allows a unit flow to pass through it.

Since $\textsc{max-flow}(s, v_j, \mathcal{G} \setminus F) > \textsc{max-flow}(s, v_j, \mathcal{H} \setminus F)$, there must exist an index $i \in [1, \lambda]$ satisfying that there exists a path from s to v_j in $\mathcal{G} \setminus F$ *passing through* r_i, but no such corresponding path exists in $\mathcal{H} \setminus F$.

Let $R = \{x_i^0, x_i^1, \ldots, x_i^\alpha\}$ be the set of leaf nodes in tree B_i *reachable* from s in $\mathcal{G} \setminus F$. There exist at least $\min(k+1, |R|)$ vertex-disjoint paths from R to v_j in $\mathcal{H}$, namely,

- $(\{x_i^0, \ell(x_i^0), y_{i,W}, v_j)$, where $W \in \mathfrak{F}$ is the set in S that contains the element $x^0 \in U$.
- $(\{x_i^c, r(x_i^c), z_i^c, v_j)$, for $c = 1$ to $\min(k, |R| - 1)$.

Thus even after k faults atleast one path from r_i to v_j will exist in $\mathcal{H} \setminus F$. This contradicts the assumption that there is no s to v_j path in $\mathcal{G} \setminus F$ passing through r_i. Hence, $\textsc{max-flow}(s, v_j, \mathcal{G} \setminus F)$ must be identical to $\textsc{max-flow}(s, v_j, \mathcal{H} \setminus F)$. ◀

The proof of Theorem 8 now directly follows from Lemma 10, Lemma 11, and Lemma 12, along with the fact that for every integer $n \geqslant 1$, there exist hard instances of the SET-COVER problem $(U, \mathfrak{F})$ satisfying $|U| = n$, where the size of the optimal solution is significantly larger than $\log |U|$.

4 Upper bound of $\lambda 2^k n$ Edges

In this section we will provide construction of a sparse (λ, k)-FT-BFP.

4.1 Locality Property for Flow Preservers

▶ **Lemma 13.** *Let $\mathcal{G} = (V, E)$ be a graph with a source $s \in V$, $\lambda \geqslant 1$ be an integer, and v be a fixed vertex in V. Let $\alpha = \min\big(\lambda, \textsc{max-flow}(s, v, \mathcal{G})\big)$. Let $\mathcal{E}_v$ be the set of in-edges of v corresponding to any arbitrary set of α-edge-disjoint paths from s to v in $\mathcal{G}$. Further, let $\mathcal{H}$ be a subgraph of $\mathcal{G}$ obtained by restricting the in-edges of the given node v to those present in $\mathcal{E}_v$. Then, for each $t \in V$, we have*

$$\textit{MAX-FLOW}(s, t, \mathcal{H}) \;\geqslant\; \min\big(\lambda, \textit{MAX-FLOW}(s, t, \mathcal{G})\big).$$

Proof. We first observe that $\alpha = \textsc{max-flow}(s, v, \mathcal{H})$. Indeed, by construction there are at least α edge-disjoint paths from s to v in $\mathcal{H}$, additionally, the in-degree of v in $\mathcal{H}$ is exactly α which proves that the (s, v)-max-flow in $\mathcal{H}$ can not be larger than α.

Now consider a vertex $t \in V$, and let $\beta = \textsc{max-flow}(s, t, \mathcal{H})$. Consider an (s, t)-min-cut (A, B) in $\mathcal{H}$. If $v \in A$ then, by construction of $\mathcal{H}$, the (s, t)-cut (A, B) has value β also in $\mathcal{G}$, so $\beta \geqslant \textsc{max-flow}(s, t, \mathcal{G})$ and we are done. Assume next $v \in B$. Then (A, B) is an (s, v)-cut of value β in $\mathcal{H}$. Since $\alpha = \textsc{max-flow}(s, v, \mathcal{H})$, we have $\beta \geqslant \alpha$. If $\alpha = \lambda$ we are done. We next study the **non-trivial case** of $\alpha = \textsc{max-flow}(s, v, \mathcal{G}) < \lambda$.

Let f be an (s,t)-max-flow in $\mathcal{H}$. Let us assume on contrary that $\beta < \text{MAX-FLOW}(s,t,\mathcal{G})$. Then the residual graph $\mathcal{G}_f$ must have an augmenting path, say P, containing some edges present in $\mathcal{G}$ but not in $\mathcal{H}$. Such edges must be all incoming to v. Thus, $P = P[s,w] \circ (w,v) \circ P[w,t]$ where $(w,v) \in E(G) \setminus E(H)$, and $P[s,w], P[v,t]$ are present in the residual graph $\mathcal{H}_f$. Adding P to f gives an (s,t)-flow of in $\mathcal{H} + (w,v)$, implying that

(i) $\text{MAX-FLOW}(s,t,\mathcal{H}+(w,v)) = \beta + 1$

(ii) $(w,v) \in A \times B$

(iii) (A,B) is an (s,t)-min-cut in $\mathcal{H} + (w,v)$

Let $\{Q_i \circ e_i \circ Q'_i\}_{i=1}^{\alpha}$ be α edge-disjoint s-to-v paths in $\mathcal{H}$, where the edge e_i of each such path is its last edge crossing the (s,v)-cut (A,B), so $V(Q'_i) \subseteq B$. Such exist as $\alpha = \text{MAX-FLOW}(s,v,\mathcal{H})$. Let $e_{\alpha+1}, \ldots, e_\beta$ be the other edges crossing (A,B) in $\mathcal{H}$. Let $e_0 = (w,v)$, crossing (A,B) by (ii). Let $\{P_j \circ e_j \circ P'_j\}_{j=0}^{\beta}$ be $\beta+1$ edge-disjoint s-to-t paths in $\mathcal{H} + (w,v)$, each crossing the cut (A,B) exactly once, at e_j, so $V(P_j) \subseteq A$. Such exist by (i) and (iii). Then, $\{P_0 \circ e_0\} \cup \{P_i \circ e_i \circ Q'_i\}_{i=1}^{\alpha}$ are $\alpha+1$ edge-disjoint s-to-v paths in $\mathcal{G}$, contradicting $\alpha = \text{MAX-FLOW}(s,v,\mathcal{G})$. ◀

In the next lemma we show that in order to compute a sparse (λ,k)-FT-BFP it suffices to focus on a single destination node.

▶ **Lemma 14** (Locality Lemma for Flow Preservers). *Let $\mathcal{A}$ be an algorithm that given any graph $\mathcal{G}$ and any vertex $v \in V(\mathcal{G})$, computes a (λ,k)-FT-BFP of $\mathcal{G}$ with at most $c_{\lambda,k}$ in-edges to v. Then using $\mathcal{A}$, one can construct for any n vertex digraph a (λ,k)-FT-BFP with at most $c_{\lambda,k} \cdot n$ edges.*

Proof. Consider a graph $\mathcal{G}$ with n vertices, namely, $v_1, \ldots, v_n$. We will provide a construction of (λ,k)-FT-BFP of $\mathcal{G}$ using black-box access to algorithm $\mathcal{A}$. We compute a sequence of graphs $\mathcal{G}_0, \mathcal{G}_1, \ldots, \mathcal{G}_n$ as follows:

1. Initialize $\mathcal{G}_0 = \mathcal{G}$.
2. For $i \geqslant 1$, compute $\mathcal{G}_i$ in two steps:
   a. First use $\mathcal{A}$ to compute a (λ,k)-FT-BFP of $\mathcal{G}_{i-1}$ in which the in-degree of v_i is bounded by $c_{\lambda,k}$, let this graph be $\mathcal{H}_{i-1}$.
   b. Obtain $\mathcal{G}_i$ from $\mathcal{G}_{i-1}$ by restricting the incoming edges of v_i to those present in $\mathcal{H}_{i-1}$.

It is easy to verify that the in-degree of each vertex in $\mathcal{G}_n$ is at most $c_{\lambda,k}$.

To show that $\mathcal{G}_n$ is a (λ,k)-FT-BFP of $\mathcal{G}$, it suffices to show that $\mathcal{G}_i$ is a (λ,k)-FT-BFP of $\mathcal{G}_{i-1}$, for each $i \geqslant 1$.

Let us fix an index i in the range $[1,n]$. Consider a set F of at most k edges in $\mathcal{G}_{i-1}$, and let

$$\alpha = \min\big(\lambda, \text{MAX-FLOW}(s, v_i, \mathcal{G}_{i-1} \setminus F)\big).$$

By construction, $\mathcal{H}_{i-1}$ is a (λ,k)-FT-BFP of $\mathcal{G}_{i-1}$, so there exists at least α edge-disjoint paths from s to v_i in the graph $\mathcal{H}_{i-1} \setminus F$. Let $\mathcal{E}_i$ be the set of in-edges of v_i corresponding to these α edge-disjoint paths. Observe that the edges in $\mathcal{E}_i$ lie in graph $\mathcal{G}_i \setminus F$. Moreover, graphs $\mathcal{G}_i \setminus F$ and $\mathcal{G}_{i-1} \setminus F$ differ only at in-edges of v_i. Therefore, by Lemma 13 it follows that for any vertex $t \in V(G)$, $\text{MAX-FLOW}(s,t,\mathcal{G}_i \setminus F) \geqslant \min\big(\lambda, \text{MAX-FLOW}(s,t,\mathcal{G}_{i-1} \setminus F)\big)$. This proves that $\mathcal{G}_i$ is a (λ,k)-FT-BFP of $\mathcal{G}_{i-1}$. ◀

4.2 Construction of an Improved FTRS

We present here an improved bound on the in-degree of a node t in an FTRS when the node t satisfies that (s,t)-max-flow in $\mathcal{G}$ is larger than one. In particular, we prove the following theorem.

▶ **Theorem 15.** *Let $\mathcal{G}$ be an n vertex, m edges directed graph with a designated source node s. Let t be a vertex satisfying $\textsc{max-flow}(s,t,\mathcal{G}) = f$, for some positive integer f. Then for every $k \geqslant 1$, we can compute in $O(2^k fm)$ time a $(k+f-1)$-FTRS for $\mathcal{G}$ in which the in-degree of node t is at most $2^k f$.*

Let us focus on a single destination node t. We first show that it suffices to provide construction of $(k+f-1)$-FTRS for a graph in which out-degree of each vertex other than s is bounded by 2. In order to prove this we will transform the graph $\mathcal{G} = (V, E)$ into another graph $\mathcal{H} = (V_H, E_H)$ satisfying that (i) the value of (s,t)-max-flow in graphs $\mathcal{G}$ and $\mathcal{H}$ is identical; (ii) the out-degree of every vertex in $\mathcal{H}$ other than s is bounded by two. The steps to transform $\mathcal{G}$ into graph $\mathcal{H}$ are as follows:

1. Initialize $\mathcal{H}$ to be the graph $\mathcal{G}$.
2. Split each edge $e = (x,y) \in E$ by inserting two new vertices $\ell_{x,y}$ and $r_{x,y}$ between the endpoints x and y, so that edge (x,y) is translated into the path $(x, \ell_{x,y}, r_{x,y}, y)$.
3. For every node $y \in V \setminus \{s,t\}$ if $x_1, \ldots, x_p$ are in-neighbours of y in $\mathcal{G}$ and $z_1, \ldots, z_q$ are out-neighbours of y in $\mathcal{G}$, then we replace vertex y (in current $\mathcal{H}$) by p binary trees as follows. First we remove node y from $\mathcal{H}$. Next for each $x_i \in \textsc{in}(y,\mathcal{G})$ insert a binary tree $B_{x_i,y}$ to graph $\mathcal{H}$ (along with new internal nodes and edges) whose root is $r_{x_i,y}$ and leaves are $\ell_{y,z_1}, \ldots, \ell_{y,z_q}$.

Notice that $\mathcal{H}$ has $O(mn)$ edges and vertices. Indeed for every vertex v (other than s and t) in $\mathcal{G}$, $|\textsc{in}(v,\mathcal{G})|$ binary trees have been added to $\mathcal{H}$, each of size $O(|\textsc{out}(v,\mathcal{G})|)$. So the number of edges and vertices in the transformed graph is $O(\sum_{v \in V} |\textsc{in}(v,\mathcal{G})| \cdot |\textsc{out}(v,\mathcal{G})|)) = O(mn)$. Also, observe that the out-degree of each vertex in $\mathcal{H}$ other than s bounded by two.

▶ **Lemma 16.** $\textsc{max-flow}(s,t,\mathcal{G}) = \textsc{max-flow}(s,t,\mathcal{H})$

Proof. We will show that each s to t path in $\mathcal{G}$ now corresponds to a unique s to t path in $\mathcal{H}$. Suppose there exists a path $(s = u_0, u_1, u_2, \ldots, u_k = t)$ in $\mathcal{G}$. Then we will have an equivalent path in $\mathcal{H}$ as

$$(s, \ell_{u_0,u_1}, r_{u_0,u_1}) \circ \textsc{path}(r_{u_0,u_1}, \ell_{u_1,u_2}, B_{u_0,u_1}) \circ (\ell_{u_1,u_2}, r_{u_1,u_2}) \circ \ell \cdots \circ \textsc{path}(r_{u_{k-2},u_{k-1}}, \ell_{u_{k-1},u_k}, B_{u_{k-1},u_k}) \circ \ell_{u_{k-1},u_k}, r_{u_{k-1},u_k}) \circ (r_{u_{k-1},u_k}, t)$$

where $\textsc{path}(r, \ell, B)$ denotes the path from r to ℓ using edges in binary tree B. Therefore, the (s,t)-max-flow values in graphs $\mathcal{G}$ and $\mathcal{H}$ are identical. ◀

We will now justify the significance of our transformation by providing a way to construct a $(k+f-1)$-FTRS of $\mathcal{G}$ if we know a $(k+f-1)$-FTRS for $\mathcal{H}$ such that the in-degree of t in both the FTRSs is identical.

▶ **Lemma 17.** *A $(k+f-1)$-FTRS for $\mathcal{G}$ can be constructed by knowing a $(k+f-1)$-FTRS of $\mathcal{H}$, that preserves the in-degree of node t.*

Proof. Let $\mathcal{H}^*$ be a $(k+f-1)$-FTRS of $\mathcal{H}$. We want to construct $\mathcal{G}^*$, a $(k+f-1)$-FTRS for $\mathcal{G}$ satisfying the condition that in-degree of t in graphs $\mathcal{G}^*$ and $\mathcal{H}^*$ is identical.

The construction of $\mathcal{G}^*$ is as follows: For each in-neighbour w of the vertex t in $\mathcal{G}$, include edge (w,t) in $\mathcal{G}^*$ if and only if edge $(r_{w,t},t)$ is present in $\mathcal{H}^*$. Thus, the in-degree of t in graphs $\mathcal{G}^*$ and $\mathcal{H}^*$ is identical. For vertices v other than t, we include all in-neighbours of v in $\mathcal{G}^*$.

We will now prove that $\mathcal{G}^*$ is a $(k+f-1)$-FTRS of $\mathcal{G}$. Consider any set F of at most k failed edges in $\mathcal{G}$. Define a set F_0 of failed edges in $\mathcal{H}$ by including edge $(\ell_{u,v}, r_{u,v})$ in F_0 for every $(u,v) \in F$. From the path correspondence above and the fact that $\mathcal{H}^*$ is a $(k+f-1)$-FTRS of $\mathcal{H}$, it is evident that for any $r \leqslant \lambda$, there are r-edge-disjoint paths from s to t in $\mathcal{G}^* \setminus F$ if and only if there are r-edge-disjoint paths from s to t in $\mathcal{H}^* \setminus F_0$. Therefore, $\mathcal{G}^*$ is a $(k+f-1)$-FTRS of $\mathcal{G}$. ◀

It was shown in [4] that if out-degree of s is one, and out-degree of all other vertices is bounded by two, then Algorithm 1 computes a k-FTRS for $\mathcal{G}$ in which in-degree of t is at most 2^k. We will prove in the next lemma that if $\textsc{max-flow}(s,t,\mathcal{G}) = f$, and out-degree of every vertex other than s is bounded by two, then Algorithm 1 in fact computes a $(k+f-1)$-FTRS for $\mathcal{G}$ in which the in-degree of t is at most $2^k f$.

▶ **Lemma 18.** *Let $\mathcal{G}$ be a directed graph satisfying that the out-degree of every vertex other than the designated source s is bounded by 2, and $k \geqslant 1$ be an integer parameter. Let $t \in V(\mathcal{G})$ satisfy $\textsc{max-flow}(s,t,\mathcal{G}) = f$, for some positive integer f. Then Algorithm 1 computes a $(k+f-1)$-FTRS for $\mathcal{G}$ in which the in-degree of node t is at most $2^k f$.*

Proof. Consider the following algorithm from [4] for computing k-FTRS that bounds in-degree of an input node t.

Algorithm 1 Algorithm for computing k-FTRS.

```
S_1 ← {s};
for i = 1 to k do
    C_i ← FMC(S_i, t, G);
    (A_i, B_i) ← Partition(C_i);
    S_{i+1} ← (A_i ∪ OUT(A_i, G)) \ {t};
end
f_0 ← max-flow from S_{k+1} to t;
E(t) ← Incoming edges of t present in E(f_0);
Return G* = (G \ IN-EDGES(t, G)) + E(t);
```

We will now show $\mathcal{G}^*$ is a $(k+f-1)$-FTRS of $\mathcal{G}$. Let F be any set of $k+f-1$ failed edges. If there exists a path R from s to t in $\mathcal{G} \setminus F$ then we shall prove the existence of a path $\hat{R}$ from s to t in $\mathcal{G}^* \setminus F$. Observe that R must pass through each (s,t)-cut C_i, for each $i \in [1,k]$, through an edge, say (u_i, v_i). If $v_i = t$ then $(u_i, v_i) \in \mathcal{E}(t)$ and thus R is intact in the graph $\mathcal{G}^*$. Now we need to prove for the case when the edge $(u_i, v_i) \notin \mathcal{E}(t)$.

To prove that a path $\hat{R}$ exists in $\mathcal{G}^*$, we will construct a sequence of auxiliary graphs as done in [4], say $\mathcal{H}_i$'s, for each $i \in [1, k+1]$, as follows:

$$\mathcal{H}_1 = \mathcal{G}, \qquad \mathcal{H}_i = \mathcal{G} + (s, v_1) + \ldots + (s, v_{i-1}), i \in [2, k+1].$$

From the induction proof of Lemma 18 of [4], we get $\textsc{max-flow}(s,t,\mathcal{H}_{i+1}) = 1 + \textsc{max-flow}(s,t,\mathcal{H}_i)$ and since $\textsc{max-flow}(s,t,\mathcal{H}_1) = \textsc{max-flow}(s,t,\mathcal{G}) = f$, we get that $\textsc{max-flow}(s,t,\mathcal{H}_{k+1}) = k+f$. Let $\mathcal{H}^* = (\mathcal{H}_{k+1} \setminus \textsc{In-Edges}(t)) + \mathcal{E}(t)$ i.e. the incoming

edges of t are restricted in $\mathcal{H}_{k+1}$ to those present in the set $\mathcal{E}(t)$. In Lemma 19 of [4] it is shown that $\text{MAX-FLOW}(s, t, \mathcal{H}^*) = \text{MAX-FLOW}(s, t, \mathcal{H}_{k+1}) = k + f$. Since the flow in $\mathcal{H}^*$ is greater than $|F|$ or the number of faults, we can directly use the Lemma 20 of [4] to see that there exists a path $\hat{R}$ in $\mathcal{G}^* \setminus F$.

The bound on the number of edges also follows from [4]. Lemma 21 of [4] states that $|C_{i+1}| \leqslant 2|C_i|$ where $C_{k+1} = FMC(S_{k+1}, t, \mathcal{G})$. Since $|C_1| = f$, we get the bound on $\mathcal{E}(t) = C_{k+1}$ as $2^k f$. Note that the proof of Lemma 21 of [4] assumes that every vertex has out-degree bounded by two but it can be shown that the Lemma will hold true even when the out-degree of all vertices except the source vertex is bounded by two by using the fact that in the proof of Lemma 21, $\text{OUT}(A_i)$ will never contain the source vertex for any i. ◀

4.3 Computing sparse (λ, k)-FT-BFP

In this subsection, we will show how to construct a (λ, k)-FT-BFP of $\mathcal{G}$ from a $(k + f - 1)$-FTRS of $\mathcal{G}$. We will start by introducing a lemma from [17], followed by additional lemmas that will help us to obtain a tight construction for FT-BFP.

▶ **Lemma 19** ([17]). *Let $\mathcal{G}$ be a directed graph with a designated source node s, and let $\mathcal{H}$ be a $(k + \lambda - 1)$-FTRS of $\mathcal{G}$. Then, $\mathcal{H}$ is also a (λ, k)-FT-BFP of $\mathcal{G}$.*

To strengthen the above lemma, we present a method for constructing a (λ, k)-FT-BFP from a $(\min\{f, \lambda\} + k - 1)$-FTRS, where f represents the maximum flow from the source node s to a destination node t in the graph.

▶ **Lemma 20.** *Let $\mathcal{G}$ be a directed graph with a designated source node s, and let t be a vertex satisfying $\text{MAX-FLOW}(s, t, \mathcal{G}) = f$, for some positive integer f. Then a $(\min\{f, \lambda\} + k - 1)$-FTRS of $\mathcal{G}$ that differs from $\mathcal{G}$ only at in-edges of t is a (λ, k)-FT-BFP for $\mathcal{G}$.*

Proof. Let $\mathcal{H}$ be a $(\min\{f, \lambda\} + k - 1)$-FTRS of $\mathcal{G}$ that deviates from $\mathcal{G}$ only at in-edges of t. It follows from Lemma 19 that the subgraph $\mathcal{H}$ is a $(\min\{f, \lambda\}, k)$-FT-BFP for $\mathcal{G}$.

The claim trivially holds true if $f \geqslant \lambda$, so let us consider the scenario $f < \lambda$. Consider a set F of at most k edge failures in $\mathcal{G}$, and let p be $\text{MAX-FLOW}(s, t, \mathcal{G} \setminus F)$. Since $p \leqslant f < \lambda$ and $\mathcal{H}$ is a (f, k)-FT-BFP, the max-flow from s to t in $\mathcal{H} \setminus F$ must be exactly p.

Since $\mathcal{G}$ and $\mathcal{H}$ only differs at in-edges of t, it follows from Lemma 13 that for each $v \in V(\mathcal{G})$, $\text{MAX-FLOW}(s, v, \mathcal{H} \setminus F) \geqslant \min(\lambda, \text{MAX-FLOW}(s, v, \mathcal{G} \setminus F))$. This proves that $\mathcal{H}$ is a (λ, k)-FT-BFP for $\mathcal{G}$. ◀

We now provide construction of a (λ, k)-FT-BFP that bounds the in-degree of a single destination node t.

▶ **Lemma 21.** *Let $\mathcal{G}$ be an n vertex, m edges directed graph with a designated source node s, and t be any arbitrary vertex in $\mathcal{G}$. Then for any $\lambda, k \geqslant 1$, we can compute in $O(\lambda 2^k m)$ time a (λ, k)-FT-BFP for $\mathcal{G}$ in which the in-degree of t is bounded above by $\lambda 2^k$.*

Proof. Let f be the value of (s, t)-max-flow in $\mathcal{G}$. We present a construction of a (λ, k)-FT-BFP, say $\mathcal{H}$, by considering the following two cases.

Case 1. max-flow$(s, t, \mathcal{G}) \geqslant \lambda + k$:
Let us start by taking a look at the scenario $f \geqslant \lambda + k$. In this case we can choose any $\lambda + k$ incoming edges of t which carry a flow of $\lambda + k$ from s to t and discard all other incoming edges of t to construct $\mathcal{H}$. The resulting graph $\mathcal{H}$ will be a (λ, k)-FT-BFP of $\mathcal{G}$ due to Lemma 20, and the in-degree of t in $\mathcal{H}$ will be $\lambda + k \leqslant \lambda 2^k$.

Case 2. max-flow$(s, t, \mathcal{G}) < \lambda + k$:
We next consider the case $f < \lambda + k$. In this case we use Theorem 15 to compute a $(\min\{f, \lambda\} + k - 1)$-FTRS of $\mathcal{G}$, say $\mathcal{H}_0$, such that the in-degree of t in $\mathcal{H}_0$ is at most $2^k \min\{f, \lambda\}$. We obtain the graph $\mathcal{H}$ from $\mathcal{G}$ by limiting the incoming edges of t to those present in $\mathcal{H}_0$. The resulting graph $\mathcal{H}$ will be a (λ, k)-FT-BFP of $\mathcal{G}$ due to Lemma 20. ◀

We conclude with the following theorem that directly follows by combining together Lemma 14 and Lemma 21.

▶ **Theorem 22.** *Let $\mathcal{G}$ be an n vertex, m edges directed graph with a designated source node s. Then for any $\lambda, k \geqslant 1$, we can compute in $O(\lambda 2^k mn)$ time a (λ, k)-FT-BFP for $\mathcal{G}$ with at most $\lambda 2^k n$ edges. Moreover, the in-degree of each vertex in this (λ, k)-FT-BFP is bounded above by $\lambda 2^k$.*

5 Matching Lower Bound

We shall now show that for each λ, k, n $(n \geqslant 3\lambda 2^{k+1})$, there exists a directed graph $\mathcal{G}$ with $O(n)$ vertices whose (λ, k)-FT-BFP must have $\Omega(2^k \lambda n)$ edges.

The construction of graph $\mathcal{G}$ is as follows. Let $B_1, \ldots, B_\lambda$ be vertex-disjoint complete binary trees of height k rooted at vertices $r_1, \ldots, r_k$, and let s be a new vertex have an edge to each of the r_i's. Let X denote the set of leaf nodes of these λ trees, and let Y be another set containing $n - (1 + \sum_{i=1}^{\lambda} |V(B_i)|)$ vertices. Note that $|Y| \geqslant n/3$. The graph $\mathcal{G}$ is obtained by adding an edge from each $x \in X$ to each $y \in Y$. In other words, $V(\mathcal{G}) = \{s\} \cup V(B_1) \cup \cdots \cup V(B_\lambda) \cup Y$ and $E(\mathcal{G}) = \{(s, r_i) \mid 1 \leqslant i \leqslant \lambda\} \cup E(B_1) \cup \cdots \cup (B_\lambda) \cup (X \times Y)$.

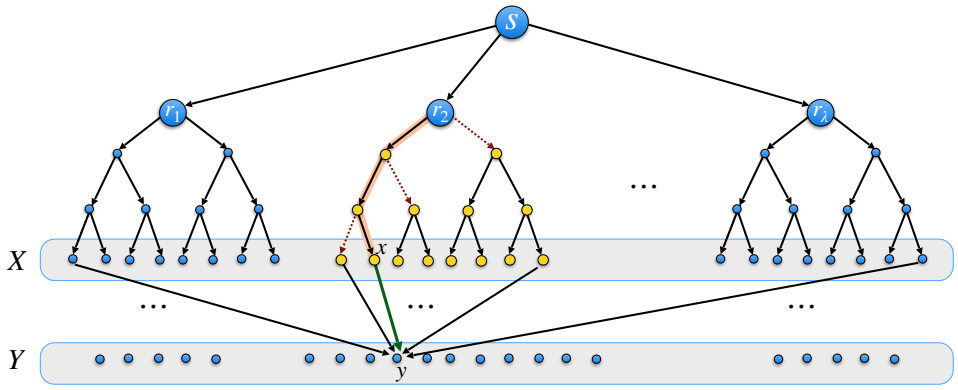

Figure 2 Depiction of lower bound on the size of (λ, k)-FT-BFP when $k = 3$.

We prove in the following lemma that any (λ, k)-FT-BFP of the above constructed graph contains at least $\Omega(2^k \lambda n)$ edges.

▶ **Lemma 23.** *Any (λ, k)-FT-BFP of $\mathcal{G}$ must contain $\Omega(2^k \lambda n)$ edges.*

Proof. It is easy to see that the out-edges of s, and the edges of each of the binary tree B_i's must be present in a (λ, k)-FT-BFP of $\mathcal{G}$. Thus, let us consider an edge $(x, y) \in X \times Y$, where x is the leaf node of some binary tree B_i.

Let P be the unique path from r_i to x in B_i, and let F be the set of all those edges $(u, v) \in B_i$ such that $u \in P$ and v is the child of u not lying on P. On failure of set F, there remains a unique path from s to y that passes through edge (s, r_i). Moreover, $\text{MAX-FLOW}(s, y, \mathcal{G} \setminus F) = \lambda$. So, any subgraph $\mathcal{H}$ of $\mathcal{G}$ not containing (x, y) edge would not be a (λ, k)-FT-BFP as on failure set F, $\mathcal{H}$ would not preserve (s, y)-max-flow.

Hence, any (λ, k)-FT-BFP of $\mathcal{G}$ contains at least $|X \times Y| = 2^k \lambda |Y| \geqslant 2^k \lambda n/3$ edges. ◀

6 Applications

In this section we present applications of FT-BFP structure.

6.1 Fault-tolerant All-Pairs λ-reachability oracle

Georgiadis et al. [16] showed that for any n vertex directed graph $\mathcal{G} = (V, E)$ we can compute 2-reachability information for all pairs of vertices in $O(n^\omega \log n)$ time, where ω is the matrix multiplication exponent. Abboud et at. [1] extended this result to all-pairs λ-reachability by presenting an algorithm that takes $O((\lambda \log n) 4^{\lambda + o(\lambda)} \cdot n^\omega)$ time. One of the interesting open questions is if for any constants $\lambda, k \geqslant 1$, we can compute an oracle that given any query vertex-pair $x, y \in V$ and any set F of k edge failures, reports (x, y)-λ-reachability in $\mathcal{G} \setminus F$ efficiently.

For any vertex $x \in V$, let $\mathcal{H}_x$ denote a (λ, k)-FT-BFP of $\mathcal{G}$ with x as the source. Our data structure simply stores the graph family $\{\mathcal{H}_x \mid x \in V\}$. Given any query vertex-pair (x, y) and any set F of k edges, we compute the (x, y)-max-flow in $\mathcal{H}_x$ by employing the max-flow algorithm of Chen et al. [10]. The time to compute the max-flow is $O(|E(\mathcal{H}_x)|^{1+o(1)})$, which is just $O(2^k \lambda n^{1+o(1)})$. Note that the total space used is bounded by $O(2^k \lambda n^2)$. Therefore, we have the following theorem.

▶ **Theorem 24.** *Given any directed graph $\mathcal{G} = (V, E)$ on n vertices, and any positive constants $\lambda, k \geqslant 1$, we can preprocess G in polynomial time to build an $O(n^2)$ size data structure that, given any query vertex-pair (x, y) and any set F of k edges, can determine the (x, y)-λ-reachability in $\mathcal{G} \setminus F$ in $O(n^{1+o(1)})$ time.*

6.2 FT-BFPs for graphs with non-unit capacities

We have shown till now that for any digraph $\mathcal{G}$ with unit capacities, one can compute a (λ, k)-FT-BFP with $O(2^k \lambda n)$ edges. We shall now show how to extend this result to a digraph with integer edge capacities such that flow values up to λ are preserved under bounded capacity decrement.

Let us first formalize the notion of FT-BFP under capacity decrement function.

▶ **Definition 25.** *Let $\mathcal{G} = (V, E, c)$ be a directed flow graph such that capacity of any edge is a positive integer, and let $s \in V$ be a designated source vertex. A subgraph $\mathcal{H} = (V, E_0 \subseteq E)$ of $\mathcal{G}$ is said to be a (λ, k)-Fault-Tolerant Bounded-Flow-Preserver if for any capacity decrement function $I : E(G) \to \mathbb{N}$ satisfying $\sum_{e \in E(G)} I(e) \leqslant k$, the following holds for the capacity function c^* defined as $c^*(e) = c(e) - I(e)$, for $e \in E$:*
For every $t \in V$,

$$\text{MAX-FLOW}(s, t, \mathcal{H}|c^*) = \begin{cases} \text{MAX-FLOW}(s, t, \mathcal{G}|c^*) & \text{if } \text{MAX-FLOW}(s, t, \mathcal{G}|c^*) \leqslant \lambda, \\ \text{At least } \lambda, & \text{otherwise;} \end{cases}$$

where, $\mathcal{H}|c^*$ and $\mathcal{G}|c^*$ are respectively the graphs $\mathcal{H}$ and $\mathcal{G}$ with capacity function c^*.

Let us now discuss the construction of (λ, k)-FT-BFPs. Let $\mathcal{G} = (V, E, c)$ be a digraph with integer edge capacities. We first transform $\mathcal{G}$ into a multigraph $\mathcal{G}^*$ by replacing an edge (x, y) of capacity $c(x, y)$ by exactly $c(x, y)$ copies of edge (x, y) of unit-capacity. Thus, for vertex $v \in V$, the s to v max-flow in graphs $\mathcal{G}$ and $\mathcal{G}^*$ are identical.

Now, let $\mathcal{H}^*$ be a (λ, k)-FT-BFP of multigraph $\mathcal{G}^*$. Then, a (λ, k)-FT-BFP of $\mathcal{G}$, say $\mathcal{H} = (V, E_0, c)$, can be obtained by simply retaining all those edges whose multiplicity in $\mathcal{H}^*$ is non-zero. The graph $\mathcal{H}$ will indeed be a (λ, k)-FT-BFP of $\mathcal{G}$ since a bounded capacity decrement in $\mathcal{G}$ corresponds to k-edge failures in $\mathcal{G}^*$.

References

1. Amir Abboud, Loukas Georgiadis, Giuseppe F. Italiano, Robert Krauthgamer, Nikos Parotsidis, Ohad Trabelsi, Przemyslaw Uznanski, and Daniel Wolleb-Graf. Faster algorithms for all-pairs bounded min-cuts. In Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi, editors, *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019*, volume 132 of *LIPIcs*, pages 7:1–7:15, 2019. doi:10.4230/LIPICS.ICALP.2019.7.
2. Surender Baswana, Koustav Bhanja, and Abhyuday Pandey. Minimum+1 (s, t)-cuts and dual edge sensitivity oracle. In Mikolaj Bojanczyk, Emanuela Merelli, and David P. Woodruff, editors, *49th International Colloquium on Automata, Languages, and Programming, ICALP 2022, July 4-8, 2022, Paris, France*, volume 229 of *LIPIcs*, pages 15:1–15:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.ICALP.2022.15.
3. Surender Baswana, Shreejit Ray Chaudhury, Keerti Choudhary, and Shahbaz Khan. Dynamic DFS in undirected graphs: Breaking the o(m) barrier. *SIAM J. Comput.*, 48(4):1335–1363, 2019. doi:10.1137/17M114306X.
4. Surender Baswana, Keerti Choudhary, and Liam Roditty. Fault-tolerant subgraph for single-source reachability: General and optimal. *SIAM Journal on Computing*, 47(1):80–95, 2018. doi:10.1137/16M1087643.
5. Surender Baswana and Neelesh Khanna. Approximate shortest paths avoiding a failed vertex: Near optimal data structures for undirected unweighted graphs. *Algorithmica*, 66(1):18–50, 2013. doi:10.1007/S00453-012-9621-Y.
6. Davide Bilò, Luciano Gualà, Stefano Leucci, and Guido Proietti. Multiple-edge-fault-tolerant approximate shortest-path trees. In *33rd Symposium on Theoretical Aspects of Computer Science, STACS 2016*, pages 18:1–18:14, 2016. doi:10.4230/LIPICS.STACS.2016.18.
7. Shiri Chechik. Fault-tolerant compact routing schemes for general graphs. *Inf. Comput.*, 222:36–44, 2013. doi:10.1016/J.IC.2012.10.009.
8. Shiri Chechik, Michael Langberg, David Peleg, and Liam Roditty. Fault-tolerant spanners for general graphs. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 435–444, 2009. doi:10.1145/1536414.1536475.
9. Shiri Chechik, Michael Langberg, David Peleg, and Liam Roditty. *f*-sensitivity distance oracles and routing schemes. In *18th Annual European Symposium on Algorithms - ESA (1)*, pages 84–96, 2010. doi:10.1007/978-3-642-15775-2_8.
10. Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 612–623. IEEE, 2022. doi:10.1109/FOCS54457.2022.00064.
11. Keerti Choudhary. An optimal dual fault tolerant reachability oracle. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016*, pages 130:1–130:13, 2016. doi:10.4230/LIPICS.ICALP.2016.130.
12. Camil Demetrescu, Mikkel Thorup, Rezaul Alam Chowdhury, and Vijaya Ramachandran. Oracles for distances avoiding a failed node or link. *SIAM J. Comput.*, 37(5):1299–1318, 2008. doi:10.1137/S0097539705429847.

13 Michael Dinitz and Robert Krauthgamer. Fault-tolerant spanners: better and simpler. In *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011*, pages 169–178, 2011. doi:10.1145/1993806.1993830.

14 Ran Duan and Seth Pettie. Dual-failure distance and connectivity oracles. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009*, pages 506–515, 2009. doi:10.1137/1.9781611973068.56.

15 D. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 2010.

16 Loukas Georgiadis, Daniel Graf, Giuseppe F. Italiano, Nikos Parotsidis, and Przemyslaw Uznanski. All-pairs 2-reachability in o(nˆw log n) time. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 74:1–74:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.74.

17 Daniel Lokshtanov, Pranabendu Misra, Saket Saurabh, and Meirav Zehavi. A brief note on single source fault tolerant reachability, 2019. arXiv:1904.08150.

18 Dana Moshkovitz. The projection games conjecture and the np-hardness of ln n-approximating set-cover. In Anupam Gupta, Klaus Jansen, José Rolim, and Rocco Servedio, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 276–287, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. doi:10.1007/978-3-642-32512-0_24.

19 Hiroshi Nagamochi and Toshihide Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(5&6):583–596, 1992. doi:10.1007/BF01758778.

20 Merav Parter. Dual failure resilient BFS structure. In *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015*, pages 481–490, 2015. doi:10.1145/2767386.2767408.

21 Merav Parter and David Peleg. Sparse fault-tolerant BFS trees. In *Algorithms - ESA 2013 - 21st Annual European Symposium, Proceedings*, pages 779–790, 2013. doi:10.1007/978-3-642-40450-4_66.

22 Merav Parter and David Peleg. Fault tolerant approximate BFS structures. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014*, pages 1073–1092, 2014. doi:10.1137/1.9781611973402.80.

23 Jan van den Brand and Thatchaphol Saranurak. Sensitive distance and reachability oracles for large batch updates. In *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 424–435, 2019. doi:10.1109/FOCS.2019.00034.