Fault Tolerant Max-Cut

Keren Censor-Hillel
Department of Computer Science, Technion, Haifa, Israel

Noa Marelly
Department of Computer Science, Technion, Haifa, Israel

Roy Schwartz
Department of Computer Science, Technion, Haifa, Israel

Tigran Tonoyan
Department of Computer Science, Technion, Haifa, Israel

Abstract

In this work, we initiate the study of fault tolerant Max-Cut, where given an edge-weighted undirected graph $G = (V, E)$, the goal is to find a cut $S \subseteq V$ that maximizes the total weight of edges that cross S even after an adversary removes k vertices from G. We consider two types of adversaries: an adaptive adversary that sees the outcome of the random coin tosses used by the algorithm, and an oblivious adversary that does not. For any constant number of failures k we present an approximation of $(0.878-\epsilon)$ against an adaptive adversary and of $\alpha_{GW} \approx 0.8786$ against an oblivious adversary (here α_{GW} is the approximation achieved by the random hyperplane algorithm of [Goemans-Williamson J. ACM '95]). Additionally, we present a hardness of approximation of α_{GW} against both types of adversaries, rendering our results (virtually) tight.

The non-linear nature of the fault tolerant objective makes the design and analysis of algorithms harder when compared to the classic Max-Cut. Hence, we employ approaches ranging from multi-objective optimization to LP duality and the ellipsoid algorithm to obtain our results.

2012 ACM Subject Classification Mathematics of computing → Approximation algorithms; Mathematics of computing → Graph algorithms

Keywords and phrases fault-tolerance, max-cut, approximation

Digital Object Identifier 10.4230/LIPIcs.ICALP.2021.46

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2105.01138` [19]

Funding This project has received funding from the European Research Council (ERC), under the European Unions Horizon 2020 research and innovation programme under grant agreement No 755839, and from Israel Science Foundation (ISF), under grant No 1336/16.

1 Introduction

In this work, we initiate the study of *fault tolerant Max-Cut*. In the classic Max-Cut problem, we are given an undirected graph $G = (V, E)$ equipped with non-negative edge weights $w : E \to \mathbb{R}_+$. The goal is to find a cut $S \subseteq V$ that maximizes the total weight of edges that cross S. Max-Cut is one of Karp's 21 NP-complete problems [37] and has been for close to three decades a case study for the introduction of new approaches both in the theory of algorithms and the complexity theory. Perhaps the two most prominent examples of the above are: (1) the random hyperplane rounding method of Goemans and Williamson for semi-definite programs [29], which yields an approximation of $\alpha_{GW} \approx 0.8786$ for Max-Cut; and (2) the Unique Games Conjecture of Khot [38]. The former has opened an entirely new area in the field of approximation algorithms with applications to a wide range of problems, *e.g.*, Max-DiCut [26,42,44], Max-Bisection [5,53], Max-Agreement [20,56], Max-2SAT [26,42],

48th International Colloquium on Automata, Languages, and Programming (ICALP 2021).
Editors: Nikhil Bansal, Emanuela Merelli, and James Worrell; Article No. 46; pp. 46:1–46:20
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Max-SAT [4, 7], and Cut Norm [2], to name a few. The latter has been a dominant method for proving hardness of approximation results in the last two decades, *e.g.*, the celebrated tight hardness for Max-Cut [39, 45], and Vertex Cover [40].

Motivated by large scale real life systems, fault tolerant algorithms seek to find a solution to a given optimization problem that is resilient to failures of some parts of the input. The above can be intuitively formulated as a two step process: (1) the algorithm finds a solution to the problem at hand; and (2) an adversary removes parts of the input. The goal of the algorithm is that no matter which part of the input the adversary removes, the remaining solution after removal still retains some desired properties despite the removal. Typically, the focus of fault tolerance has been network design problems, *e.g.*, BFS [33, 48, 50–52] and spanners [15–17, 25, 41, 47, 55]. Additional related algorithmic problems for which fault tolerant algorithms were studied include, *e.g.*, single source reachability [9, 10], connected dominating set [18, 59], and facility location [23, 32, 36, 57].

In this work, we initiate the study of fault tolerant Max-Cut, where the adversary can remove vertices from the graph (all edges touching the removed vertices are also deleted). Intuitively, fault tolerant Max-Cut can be seen as a two players game, in which one player (the algorithm) chooses a cut and the other player (the adversary) removes up to a prespecified number k of vertices. The algorithm desires to maximize the total weight of edges crossing the cut, while the adversary aims to minimize the total weight of edges crossing the cut.

We study two types of adversaries. The first is an *adaptive adversary* that chooses which k vertices to fail *after* seeing the cut the algorithm produces. Specifically, the adaptive adversary knows the input, how the algorithm operates, and if the algorithm is randomized, the adaptive adversary also knows the outcome of all random coin tosses used by the algorithm. The second type of adversary is an *oblivious adversary*. Similarly to the adaptive adversary, the oblivious adversary knows the input and how the algorithm operates. However, in contrast to the adaptive adversary, the oblivious adversary does not know the outcome of the random coin tosses used by the algorithm, in case the latter is randomized (equivalently, the oblivious adversary only knows the distribution over cuts the algorithm produces). Thus, the oblivious adversary is required to choose which k vertices to fail without the knowledge of which cut was sampled. To the best of our knowledge only adaptive adversaries were studied in the fault tolerance literature.

The Challenges. The fault tolerant Max-Cut problem differs considerably from classic Max-Cut for several reasons. First, the structure of the solutions may be different. Specifically, there are instances for which an optimal solution to fault tolerant Max-Cut is not an optimal solution to classic Max-Cut, and vice versa (refer to [19] for details). Furthermore, it might be the case that the ratio between the values of the optimal solutions is large or even unbounded.

Second, the application of known techniques (which can be successfully applied to Max-Cut) to fault tolerant Max-Cut imposes some obstacles that arise from the non-linear nature of the fault tolerant objective. For example, the random hyperplane rounding method of Goemans and Williamson cannot be analyzed in a straightforward manner as one is required to lower bound the expectation of the minimum value (over all possible actions of the adversary) of the cut the random hyperplane defines, as opposed to just the expected value of the cut the random hyperplane defines. Moreover, even analyzing the simplest known algorithm for Max-Cut, *i.e.*, choosing a uniform random cut, requires great care (refer to [19] for further details). Hence, the design and analysis of algorithms for fault tolerant Max-Cut requires some new insights into the problem.

1.1 Our Contributions

Adaptive Adversary. When focusing on an adaptive adversary, our main result is an (almost) tight approximation of $0.878-\epsilon$, for any constant number k of failures and unweighted graphs. This is summarized in the following theorem (it is important to note that the constant in the theorem is slightly smaller than the Goemans-Williamson approximation factor α_{GW}).

▶ **Theorem 1.1.** *For every constant $k > 0$ and $\epsilon > 0$, there is a polynomial time $(0.878 - \epsilon)$-approximation algorithm for fault tolerant Max-Cut on unweighted graphs against an adaptive adversary and k faults.*

Our algorithm is based on viewing fault tolerant Max-Cut against an adaptive adversary as a multi-objective optimization problem, where for every possible subset of k vertices the adversary can fail, one can define a different objective. The goal is to maximize the worst, *i.e.*, minimum, objective. This approach does not suffice, since all known results for the multi-objective variant of Max-Cut (formally known as Simultaneous Max-Cut [12, 13]) can handle only a constant number of objectives. In our case, even when a single failure is allowed, the number of objectives equals n. Hence, to overcome this difficulty, we incorporate local search into the above multi-objective approach to obtain the claimed result in Theorem 1.1.

Oblivious Adversary. When focusing on an oblivious adversary, our main result is a tight approximation of α_{GW} for any constant number k of failures. However, in contrast to the adaptive adversary setting, this result holds for general weighted graphs and achieves the α_{GW}-approximation guarantee exactly. This is summarized in the following theorem.

▶ **Theorem 1.2.** *For every constant $k > 0$, there is a polynomial time α_{GW}-approximation algorithm for fault tolerant Max-Cut on general weighted graphs against an oblivious adversary and k faults.*

The approach we adopt for approximating fault tolerant Max-Cut against an oblivious adversary significantly differs from the approach taken against an adaptive adversary. Surprisingly, our algorithm is based on an approximation-preserving reduction from fault tolerant Max-Cut to the classic Max-Cut problem. This reduction uses LP duality alongside the ellipsoid algorithm and is achieved by presenting a suitable approximate dual separation oracle for a configuration LP that encodes the distribution over cuts that the algorithm produces.

Hardness of Approximation. We prove that fault tolerant Max-Cut in unweighted graphs, against both adaptive and oblivious adversaries, cannot be approximated better than α_{GW} without breaking well-known hardness assumptions. It is important to note that this settles the approximability of the oblivious adversary setting (see Theorem 1.2 above), and almost settles the approximability of the adaptive adversary setting (see Theorem 1.1 above) as the constant in Theorem 1.1 is slightly smaller than α_{GW}.

▶ **Theorem 1.3.** *Assuming the Unique Games Conjecture and $NP \not\subseteq BPP$, there is no polynomial time $(\alpha_{GW} + \epsilon)$-approximation algorithm for fault tolerant Max-Cut in unweighted graphs, for any constant $\epsilon > 0$. This holds for both adaptive and oblivious adversaries.*

Simple Purely Combinatorial Algorithms. While Theorem 1.1 provides an (almost) tight result against an adaptive adversary, and Theorem 1.2 provides a tight result against an oblivious adversary, the techniques we employ yield algorithms which are polynomial but not simple. For example, the work of [12] for approximating Simultaneous Max-Cut, an important ingredient in the design of our algorithm against an adaptive adversary, is based on SDP hierarchies and the running time is exponential in the number of objectives. In contrast, the classic Max-Cut problem admits some very simple and fast heuristics, *e.g.*, choosing a random uniform cut. Thus, we also aim to study simple and purely combinatorial algorithms for fault tolerant Max-Cut.

We prove that fault tolerant Max-Cut does yield a simple purely combinatorial local search algorithm with a provable approximation guarantee against an adaptive adversary. Unfortunately, the classic local search for Max-Cut, that in each step moves a single vertex from one side of the cut to the other side, fails in the fault tolerant setting. Nonetheless, we prove that a local search that allows for a slightly richer family of local improvement steps suffices. This is summarized in the following theorem (refer to Section 3.2 for additional details).

▶ **Theorem 1.4.** *There is a purely combinatorial polynomial time* $1/2$*-approximation algorithm for fault tolerant Max-Cut on unweighted input graphs against an adaptive adversary and a single fault.*

We further study how a *uniform random cut* performs against both types of adversaries (deferred to the full version [19]), and prove that this performance depends on the type of the adversary. Specifically, for an oblivious adversary an approximation of $1/2$ is achieved, by a uniform random cut. However, this is not the case when considering an adaptive adversary, since we prove that a uniform random cut cannot achieve an approximation better than $1/4$.

1.2 Related Work

The weighted version of Max-Cut is one of Karp's NP-complete problems [37], and the unweighted version is also known to be NP-complete [27]. In general graphs, one cannot obtain an approximation factor better than $16/17$ for the undirected version, or better than $12/13$ for the directed version, unless $P = NP$ [34, 58]. The best known approximation for Max-Cut is the celebrated random hyperplane algorithm of Goemans and Williamson that obtains an approximation factor of roughly 0.8786 by rounding the natural semi-definite programming relaxation [29]. This is the best approximation that one can achieve, assuming the Unique Games Conjecture of Khot [39] and $P \neq NP$.

The problem of fault tolerant Max-Cut against an adaptive adversary that we introduce in this paper can be viewed as a special case of Simultaneous Max-Cut, in which the input is a collection of τ weighted graphs on the same vertex set and the goal is to partition the vertices into two parts, such that the size of the cut is large in every given graph. In a straightforward manner, our problem would imply $\tau = \binom{n}{k}$, which is unacceptable since the known approximations for Simultaneous Max-Cut are for a constant number of instances only [3, 12, 13]. Nonetheless, we do use the algorithm from [12] to obtain an algorithm that achieves an approximation of 0.878 for fault tolerant Max-Cut against an adaptive adversary. The state-of-the-art for Simultaneous Max-Cut is a polynomial 0.878-approximation for any constant number of input graphs [12], which is nearly optimal since assuming the Unique Games conjecture, Simultaneous Max-Cut cannot be approximated better than $(\alpha_{GW} - \delta)$ (where $\delta \geq 10^{-5}$) [11].

One more notion of resilience is that of robust submodular maximization, see, *e.g.*, [6, 46]. Given a submodular function f and, *e.g.*, a cardinality constraint k, a set A is robust against τ failures if $A = \arg\max_{A \subseteq V, |A| \leq k} \min_{Z \subseteq A, |Z| \leq \tau} f(A - Z)$, i.e., a subset of size at most k that achieves the maximal value after at most τ elements are removed from the solution.

Note that this notion of robustness differs from fault tolerance. The reason is that the failed elements are removed from the solution, as opposed to removed from the instance. Specifically, when considering the cut function of an undirected graph (which is submodular) the removal of a vertex from S (as in robust) differs from removing the same vertex from the graph (as in fault tolerant).

Due to the importance of coping with failures, the fault tolerance of many additional fundamental problems has been extensively studied. Prime examples are replacement paths [1, 21, 22, 30, 54], BFS trees [33, 48, 50–52], spanners [15–17, 25, 41, 47, 55], connected dominating sets [18, 59], and more [8–10, 14, 23, 32, 36, 57]

Fault tolerance was also studied in the distributed setting, such as for BFS trees [28], MST [28], and spanners [25, 49].

Paper Organization. Section 2 contains all required formal definitions and preliminary lemmas used throughout the paper. Section 3 deals with the adaptive adversary, whereas Section 4 deals with the oblivious adversary. In Section 5 we show a hardness of approximation result. Missing proofs and the analysis of a random cut appear in the full version [19].

2 Preliminaries

Graph Notations. We consider only edge-weighted graphs $G = (V, E, w)$ with positive integer weights w_e assigned to the edges $e \in E$. By *unweighted* graphs we mean graphs with $w_e = 1$, for all $e \in E$. A *cut* S in a graph $G = (V, E, w)$ is a subset of vertices $S \subseteq V$. We let $\delta(S, G) = \{e \in E : |e \cap S| = 1\}$ denote the set of all *crossing edges* of S in the graph G. The *size* or *weight* of a cut S, denoted by $C_{S,G}$, is the total weight of the crossing edges: $C_{S,G} = \sum_{e\in\delta(S,G)} w_e$. When G is clear from the context, we use C_S and $\delta(S)$.

For a set $F \subseteq V$ of vertices, the *degree* $d(F)$ of F is the total weight of edges adjacent to F: $d(F) = \sum_{e\in E:e\cap F\neq\emptyset} w_e$. For a subset $F \subseteq V$ and cut $S \subseteq V$, the *crossing degree* $d_S(F)$ of F is the total weight of edges adjacent to F that cross S: $d_S(F) = \sum_{e\in\delta(S):e\cap F\neq\emptyset} w_e$. We use $d(v)$ and $d_S(v)$, if $F = \{v\}$. We also let $n = |V|$, $m = |E|$, and $\Delta = \max_{v\in V} d(v)$. Finally, we let 2^V and $\binom{V}{k}$ denote the collection of all and all size-k subsets of V, respectively.

The Adaptive Adversary. We define the *k-FT value of a cut against an adaptive adversary* to be the minimal size of the cut, subsequent to a failure of any k vertices. Formally, for a cut S in a graph $G = (V, E, w)$ and a constant $k > 0$, the k-FT value of S is defined as $\varphi(S, k, G) = \min_{F\in\binom{V}{k}} C_{S-F,G-F}$.

▶ **Definition 2.1** (k-*AFTcut*). *Given an edge-weighted graph $G = (V, E, w)$ and a number $k \in \mathbb{N}$, a cut S is a* k-adaptive fault tolerant cut*, or k-AFTcut for short, if* $\varphi(S, k, G) = \max_{S'\subseteq V} \{\varphi(S', k, G)\}$.

We usually omit G and/or k from $\varphi(S, k, G)$ when G is clear from the context and $k = 1$. The *Max-Cut* problem, i.e., that of finding a cut with the largest size, corresponds to the special case $k = 0$, but will always be denoted by Max-Cut.

The Oblivious Adversary. We represent a randomized algorithm that finds a cut in a graph $G = (V, E, w)$ by a probability distribution $\mathcal{D}$ over all possible cuts 2^V. For a distribution $\mathcal{D}$ over cuts, we define the *k-FT value of $\mathcal{D}$* to be the minimal expected size of the cut, subsequent to the failure of any k vertices. Formally, for a graph $G = (V, E, w)$, a distribution $\mathcal{D}$ over cuts and a constant $k > 0$, we define the k-FT value of $\mathcal{D}$, denoted by $\mu(\mathcal{D}, k, G)$, as $\mu(\mathcal{D}, k, G) = \min_{F\in\binom{V}{k}} \mathbb{E}_{S\sim\mathcal{D}} [C_{S-F,G-F}]$.

▶ **Definition 2.2** (k-*OFTcut*). *Given an edge-weighted graph $G = (V, E, w)$ and a number $k \in \mathbb{N}$, a distribution $\mathcal{D}$ over all cuts 2^V is a k-oblivious fault tolerant cut, or k-OFTcut for short, if $\mu(\mathcal{D}, k, G) = \max_{\mathcal{D}'} \{\mu(\mathcal{D}', k, G)\}$.*

Note that here we assume the adversary chooses the set F of faults deterministically; it easily follows from the linearity of expectation that the adversary always has a deterministic best choice – a subset that has the largest expected crossing degree.

Greedy steps and stable cuts. We assume here that we are given an *unweighted* graph $G = (V, E)$. A key observation in our algorithms against an adaptive adversary is that any solution can be transformed into another one where each vertex contributes *many* of its edges to the cut. If a vertex contributes too little, we can just move it to the opposite side of the cut: while this could increase the crossing degree of some vertices (negative contribution to the FT value), it increases the cut size by more, giving a positive net contribution to the FT value. We prove this formally in Lemma 2.4, after some formal definitions.

For every $v \in V$ and $S \subseteq V$, let $S \oplus v$ denote the cut obtained from S by switching v to its opposite side, that is, $S \oplus v = S - v$, if $v \in S$, and $S \oplus v = S \cup \{v\}$, otherwise. Given a subset $S \subseteq V$, a constant $k \in \mathbb{N}$, and a vertex $v \in V$, we say that replacing S with $S \oplus v$, i.e., moving v to its opposite side w.r.t. S, is a *k-greedy step* if $d_S(v) \leq (d(v) - k)/2$. A cut S is *k-stable* if it has no k-greedy step, that is, for every $v \in V$, it holds that $d_S(v) > (d(v) - k)/2$. For $k = 1$, we use *stable* instead of 1-stable.

▶ **Observation 2.3.** *For every cut S and a vertex v, it holds that $C_{S\oplus v} - C_S = d_{S\oplus v}(v) - d_S(v)$.*

▶ **Lemma 2.4.** *Let $v \in V$ be a vertex, $S \subseteq V$ be a cut, and $k > 0$ be an integer, such that $d_S(v) \leq (d(v) - k)/2$; then $C_{S\oplus v} \geq C_S + k$, and $\varphi(S \oplus v, k) \geq \varphi(S, k)$.*

Proof. Assume, without loss of generality, that $v \in S$ (otherwise, we swap S and $V - S$). Observation 2.3 implies that $C_{S\oplus v} \geq C_S + k$, since $d_S(v) + k \leq d(v) - d_S(v) = d_{S\oplus v}(v)$.

For the second claim, we show that for every $F \in \binom{V}{k}$, $C_{S-F,G-F} \leq C_{S\oplus v-F,G-F}$. Assume that $v \notin F$, as otherwise $S - F = S \oplus v - F$, and the claim holds trivially. Recall that $C_{S\oplus v} \geq C_S + k$. In addition, $d_{S\oplus v}(F) \leq d_S(F) + k$, since for every $u \in F$, at most one crossing edge is added to the cut (the edge $\{u, v\}$). Putting those together, we have that: $C_{S-F,G-F} = C_S - d_S(F) \leq C_{S\oplus v} - d_{S\oplus v}(F) = C_{S\oplus v-F,G-F}$. Since this holds for every F, we have that $\varphi(S \oplus v, k) \geq \varphi(S, k)$. ◀

By repeatedly applying a k-greedy step to a cut, we keep increasing the cut value, while not decreasing the k-FT value; thus, after at most m greedy steps, we have a k-stable cut with a k-FT value at least as good as the original one. We let StabilizeCut(G,S,k) denote this procedure, which takes as input a graph G, a cut S in G, and a number k, then starting with S, repeatedly applies a (arbitrary) k-greedy step, while there is one, and returns the obtained k-stable cut. The following corollary follows from the reasoning above (the second claim follows by applying StabilizeCut to an optimal *k-AFTcut*).

▶ **Corollary 2.5.** *Let S be a cut in graph $G = (V, E)$, and let k be a positive integer. Let $S' = \text{StabilizeCut}(G, S, k)$. It holds that S' is k-stable, $C_{S'} \geq C_S$ and $\varphi(S', k) \geq \varphi(S, k)$. In particular, every unweighted graph $G = (V, E)$ has a k-stable optimal k-AFTcut.*

3 Fault Tolerance Against an Adaptive Adversary

3.1 A 0.878-Approximation for Multiple Faults

In this section, we give a $(0.878 - \epsilon)$-approximation algorithm for k-$AFTcut$ on unweighted graphs, for constants $k, \epsilon > 0$. A core tool that we use in our algorithm is an algorithm for the *Simultaneous Max-Cut* problem, where given several graphs defined over the same vertex set, the goal is to find a cut that is large for all graphs simultaneously. A 0.878-approximation algorithm for this problem with a *constant* number of graphs has been given in [12]. The algorithm is based on semidefinite programming techniques.

The main idea behind our algorithm is to separate a constant number of "heavy" (high-degree) vertices for which the following holds; given a cut which is large subsequent to any failure of k *heavy* vertices, the cut is large even if light (non-heavy) vertices fail as well. For such a heavy set, a good approximation for Simultaneous Max-Cut on the instances obtained by removing each possibility of k heavy vertices from G, should be a good approximation for k-$AFTcut$ on G. We give a greedy algorithm that selects the set of heavy vertices. We then consider two cases. We show that if the heavy vertices do not cover most of the edges in the graph (the "non-shallow" case), then an approximate solution for Simultaneous Max-Cut with respect to the heavy set gives an approximate solution for k-$AFTcut$. Otherwise (the "shallow" case), we identify a set of "super-heavy" vertices, which is shown to fail in any near-optimal solution. Therefore, finding a near-optimal solution for the original graph reduces to finding a near-optimal solution on the graph remaining by removing the "super-heavy" vertices. We show that it can be solved via brutforce, or by finding a good solution to Max-Cut (e.g., obtained via [29]). We prove the following theorem.

▶ **Theorem 1.1.** *For every constant $k > 0$ and $\epsilon > 0$, there is a polynomial time $(0.878 - \epsilon)$-approximation algorithm for fault tolerant Max-Cut on unweighted graphs against an adaptive adversary and k faults.*

Before proceeding to the algorithm, we introduce the Simultaneous Max-Cut framework.

▶ **Definition 3.1** (Simultaneous Max-Cut)**.** *Let V be a vertex set. We are given k edge-weighted graphs, $G_i = (V, E_i)$, $i = 1, \dots, k$, on the vertex set V, where the weights are normalized, so that $\sum_{e \in E_i} w_e = 1$, for each i. In the (Pareto)* Simultaneous Max-Cut *problem, given the graphs G_i together with thresholds $c_i \in [0, 1]$, the goal is to find a cut $S^* \subseteq V$ such that $C_{S^*,G_i} \geq c_i$, for every i. We say that an algorithm is an α-approximation algorithm for the problem if for every input G_i, c_i, $i = 1, \dots, k$, where there exists a cut S^* such that $C_{S^*,G_i} \geq c_i$ for every i, the algorithm returns a cut $\widetilde{S}$ such that $C_{\widetilde{S},G_i} \geq \alpha c_i$, for every i.*

▶ **Theorem 3.2.** *[12] For every constant $k \geq 1$ and parameter $n \geq 1$, there is a polynomial-in-n algorithm that computes an α_{SMC}-approximate solution to any Simultaneous Max-Cut instance with k weighted graphs on a vertex set of size n, in which all non-zero edge-weights are lower-bounded by $\exp(n^{-c})$, for constants k and c, and $\alpha_{SMC} = 0.878$.*

We apply the Simultaneous Max-Cut framework for unweighted graphs G_i. We let SimultaneousMC denote the algorithm that gets as input a constant number of unweighted graphs G_i, $i = 1, \dots, k$, and returns a cut $\widetilde{S}$ with the following property: for every cut S^* and number c such that $C_{S^*,G_i} \geq c$, for all i, it holds that $C_{\widetilde{S},G_i} \geq \alpha_{SMC} \cdot c$, for all i. This can be achieved by combining the algorithm given in Theorem 3.2 (by appropriately scaling the edge-weights and the thresholds) with a binary search on c.

Algorithm 1 $(\alpha_{SMC} - \epsilon)$-approximation for k-$AFTcut$.

Input: $G = (V, E)$, k, ϵ
Output: $(\alpha_{SMC} - \epsilon)$-approximation for k-$AFTcut$
$H \leftarrow$ HeavyVertices(G, k, ϵ)
$\widetilde{S} \leftarrow$ SimultaneousMC($\{G_{\neg F} : F \in \binom{H}{k}\}$)
if $(H, \widetilde{S})$ *is shallow* **then**
  return ShallowFTCut(G, H, $\widetilde{S}$, k, ϵ)
else
  return $\widetilde{S}$

In addition to the Simultaneous Max-Cut algorithm, we use the α_{GW}-approximation for Max-Cut due to Goemans and Williamson [29], for $\alpha_{GW} \approx 0.8786$. We use Goemans-Williamson (with input G) to denote this algorithm. Note that the actual value of the approximation factor α_{SMC} is slightly larger than 0.878 but is less than α_{GW}.

The Main Algorithm. The inputs to the algorithm (see the pseudocode in Algorithm 1) are an unweighted graph G, and parameters k (number of faults) and ϵ (precision). First, it computes the set H of heavy vertices via the subroutine HeavyVertices, then applies SimultaneousMC on a collection $\{G_{\neg F} : F \in \binom{H}{k}\}$ of subgraphs containing one subgraph for every failure of k heavy vertices. The following notation is used: for a subset $F \subseteq V$ of vertices, we let $G_{\neg F} = (V, E_{\neg F})$, where $E_{\neg F} = \{e \in E : e \cap F = \emptyset\}$. Note that in $G_{\neg F}$, we do not remove the vertices of F from the graph, as opposed to $G - F$, but only the edges adjacent to F.

The pair $(H, \widetilde{S})$ is *shallow* if all vertices in $V - H$ have degree at most $3k$, and there are k vertices in H whose removal reduces the weight of $\widetilde{S}$ below $3k^2/\epsilon$. To state this formally, let us introduce a notation that will be useful later too. For a cut $S \subseteq V$, we use $C_{S-k\times H}$ to denote the smallest size of the cut after the failure of any k vertices from H, i.e., $C_{S-k\times H} = \min\{C_{S-F,G-F} : F \in \binom{H}{k}\}$. Thus, $(H, \widetilde{S})$ is shallow if we have $\max_{v\in V-H} d(v) \leq 3k$ and $C_{\widetilde{S}-k\times H} < 3k^2/\epsilon$. If $(H, \widetilde{S})$ is not shallow, the algorithm simply returns $\widetilde{S}$. Otherwise, we recompute the cut via ShallowFTCut, using alternative methods.

The proof of Theorem 1.1 is split into two parts, addressing shallow and non-shallow cases separately. The running time is dominated by Simultaneous Max-Cut. Before giving further details, let us mention how the proof follows from the main lemmas addressing those cases.

Proof of Theorem 1.1. Let G be a graph and let S^* be an optimal k-$AFTcut$ on G. Let $\widetilde{S}$ be the output of Algorithm 1 on G, k, ϵ. We show that $\varphi(\widetilde{S}, k) \geq (\alpha_{SMC} - \epsilon) \cdot \varphi(S^*, k)$. Lemma 3.4 provides this for the non-shallow case, while Lemma 3.5 provides it in the shallow case. The algorithm is indeed polynomial, since the sub-routines are such, and the input to SimultaneousMC consists of $\binom{|H|}{k} = O(k/\epsilon)^k = O(1)$ subsets, where $|H| = O(k^2/\epsilon)$ is proven in Lemma 3.3. ◀

The selection of heavy vertices (Algorithm 2) is done by a simple greedy procedure, where we sequentially select vertices in the heavy set H in a non-increasing order by degree. The selection stops either when the remaining vertices $(V - H)$ have a small degree (at most $3k$) or when H has sufficiently many incident edges (used in Lemma 3.4). By Corollary 2.5, any cut can be transformed into one with a similar k-FT value, where every vertex v has crossing

Algorithm 2 HeavyVertices.

1 **Input:** $G = (V, E)$, k, ϵ
2 **Output:** $H \subseteq V$, the set of heavy vertices
3 Let $v_1, \ldots, v_n$ be an ordering of vertices by non-increasing degree
4 $\sigma \leftarrow 0$, $i \leftarrow 1$, $H \leftarrow \{v_1, \ldots, v_k\}$
5 **while** $d(v_{k+i}) > (\epsilon \cdot \alpha_{SMC}/k) \cdot \sigma$ **and** $d(v_{k+i}) > 3k$ **do**
6 $\quad \sigma \leftarrow \sigma + (d(v_{k+i}) - 3k)/4$
7 $\quad H \leftarrow H \cup \{v_{k+i}\}$
8 $\quad i \leftarrow i + 1$
9 **return** H

degree at least $(d(v) - k)/2$, and at least $(d(v) - 3k)/2$, after k failures. Thus, heavy vertices are guaranteed to contribute σ in the "stable version" of every cut. The degree constraint ensures that we do not select vertices that are unnecessary, according to this logic, which helps us keep the size of H bounded.

▶ **Lemma 3.3.** *Algorithm 2 terminates within $t = 4(3k^2 + k)/(\epsilon \cdot \alpha_{SMC})$ iterations. In particular, $|H| \leq t + k$.*

Proof. If $d(v_{k+t}) \leq 3k$, then by the condition in Line 5, the algorithm terminates before the t-th iteration; therefore, assume $d(v_{k+t}) > 3k$. For every $i \leq t$, after the i-th iteration, it holds that $\sigma_i = \sum_{j=1}^{i} (d(v_{k+j}) - 3k)/4$; thus, after t iterations,

$$\begin{aligned}\sigma_t &= \sum_{j=1}^{t} \frac{d(v_{k+j}) - 3k}{4} \geq t \cdot \frac{d(v_{k+t}) - 3k}{4} = \frac{3k^2 + k}{\epsilon \cdot \alpha_{SMC}} \cdot (d(v_{k+t}) - 3k) \\ &= \frac{k}{\epsilon \cdot \alpha_{SMC}} \cdot d(v_{k+t}) + \frac{3k^2}{\epsilon \cdot \alpha_{SMC}} \cdot d(v_{k+t}) - \frac{3k^2(3k+1)}{\epsilon \cdot \alpha_{SMC}} \geq \frac{k}{\epsilon \cdot \alpha_{SMC}} d(v_{k+t}) \ ,\end{aligned}$$

where in the first inequality, we use the fact that the vertices are processed in a non-increasing order of degrees, and in the last inequality, we use the assumption that $d(v_{k+t}) \geq 3k + 1$. It follows that $d(v_{k+t}) \leq (\epsilon \cdot \alpha_{SMC}/k) \cdot \sigma_t$, and using $d(v_{k+t+1}) \leq d(v_{k+t})$, we get that the algorithm terminates within the first t iterations, by the condition in Line 5. ◀

Non-Shallow Case. In this case, we have either $\max_{v \in V-H} d(v) > 3k$ or $C_{\widetilde{S}-k \times H} \geq 3k^2/\epsilon$. If the former holds, we see from Algorithm 2 that for all light vertices $v \notin H$, $d(v) \leq (\epsilon \cdot \alpha_{SMC}/k) \cdot \sigma$. As it was observed earlier, Corollary 2.5 implies that every cut can be turned into another one with no smaller FT value, such that H contributes σ edges to the cut, even after k failures. In such a cut, a failure of k light vertices would affect only an ϵ fraction of the cut. A similar reasoning applies in the other case, when $\max_{v \in V-H} d(v) \leq 3k$: here we have $C_{\widetilde{S}-k \times H} \geq 3k^2/\epsilon$, which leads to similar conclusions as above. Putting these together with the near-optimality of $\widetilde{S}$, we obtain the main lemma of the non-shallow case (see [19]).

▶ **Lemma 3.4.** *If $(H, \widetilde{S})$ is not shallow, then it holds that $\varphi(\widetilde{S}, k) \geq (\alpha_{SMC} - \epsilon)\varphi(S^*_{ft}, k)$, for an optimal k-AFTcut S^*_{ft}.*

Shallow Case. Recall that in this case, we have that $\max_{v\in V-H} d(v) \leq 3k$, and $C_{\widetilde{S}-k\times H} < 3k^2/\epsilon$. The subroutine ShallowFTCut (Algorithm 3) constructs another cut, $\hat{S}$, which we prove in Lemma 3.5 is a $(\alpha_{SMC} - \epsilon)$-approximation for *k-AFTcut*. A key role in this case is played by the following (possibly empty) set $\hat{H}$ of *super-heavy* vertices, where $\hat{H} = \left\{v \in V : (d(v) - 3k)/2 > C_{\widetilde{S}-k\times H}/\alpha_{SMC}\right\}$. We show that $\hat{H}$ is contained in *every worst-case failure set* of a cut where there is no k-greedy step of a vertex in $\hat{H}$. We let $G_R = G - \hat{H}$, and let m_R be the number of edges in G_R. Note that the degree of every $v \notin \hat{H}$ is bounded by some constant ℓ.

Algorithm 3 ShallowFTCut.

Input: $G = (V, E)$, H, $\widetilde{S}$, k, ϵ
Output: Cut $\hat{S} \subseteq V$
if $m_R < 2k\ell/(\alpha_{SMC}\epsilon)$ **then**
  for *every* $S' \subseteq V_R$ *(V_R is of constant size)* **do**
    Compute $\varphi(S', k - |\hat{H}|)$
  $\hat{S} \leftarrow \arg\max_{S'\subseteq V_R} \varphi(S', k - |\hat{H}|)$
else
  $\hat{S} \leftarrow$ Goemans-Williamson(G_R)
while $\exists v \in \hat{H}$ *such that* $d_{\hat{S}}(v) \leq (d(v) - k)/2$ **do**
  $\hat{S} \leftarrow \hat{S} \oplus v$
return $\hat{S}$

First, we compute a near-optimal $(k-|\hat{H}|)$-*AFTcut*, $\hat{S}$, on G_R, then add $\hat{H}$ and repeatedly apply k-greedy steps to the vertices in $\hat{H}$, while there are any. To compute a cut in G_R, we distinguish between two cases. If there are many edges, i.e., $m_R \geq ck\ell/\epsilon$, for a constant c, then we let $\hat{S}$ =Goemans-Williamson(G_R). This suffices, since the cut is of size at least $m_R/2 = \Omega(k\ell/\epsilon)$, and the degrees are bounded by ℓ , so failures do not affect the cut size significantly, and we get an $(\alpha_{GW} - \epsilon)$-approximation. If, on the other hand, there are few edges, i.e., $m_R < ck\ell/\epsilon$, then we can compute an optimal $(k - |\hat{H}|)$-*AFTcut* in G_R via brute-force. Also using that $\hat{H}$ belongs to every worst-cast failure set of a cut not having a k-greedy step of a vertex in $\hat{H}$ (including the one we constructed, and the optimal ones that exist by Corollary 2.5), we get the following main lemma (proved in the full version [19]).

▶ **Lemma 3.5.** *If $(H, \widetilde{S})$ is shallow, then it holds that $\varphi(\hat{S}, k) \geq (\alpha_{SMC} - \epsilon)\cdot\varphi(S^*_{ft}, k)$, for an optimal k-AFTcut S^*_{ft}.*

3.2 A Combinatorial 1/2-Approximation for a Single Fault

In the case of a single fault, we have the following result, that is, a simple and efficient 1/2-approximation for the case of a single fault. Moreover, we show that an FT value of $(m - \Delta)/2$ can be achieved, for $\Delta \geq 3$, while $m - \Delta$ is an (easy) upper bound.

▶ **Theorem 1.4.** *There is a purely combinatorial polynomial time* $1/2$*-approximation algorithm for fault tolerant Max-Cut on unweighted input graphs against an adaptive adversary and a single fault.*

In the discussion below, we call a vertex v *critical* for a cut S if $C_{S-v,G-v} = \varphi(S)$. It is well-known (and easy to show) that every stable cut is a 1/2-approximate Max-Cut. This even holds for *AFTcut*, with $\Delta = 2$ (see [19]). However, in general, while we know

Algorithm 4 Combinatorial 1/2-approximation for *AFTcut*.

Input: $G = (V, E)$
if $\Delta \leq 2$ **then return** $\textsc{StabilizeCut}(G, \emptyset, 1)$
$\widetilde{S} \leftarrow \emptyset$
while $\varphi(\widetilde{S}) < (m - \Delta)/2$ **do**
  if $\exists v, \varphi(\widetilde{S} \oplus v) \geq (m - \Delta)/2$ **then**
    $\widetilde{S} \leftarrow \widetilde{S} \oplus v$ `// type-0 step`
  else if $\exists v, d_{\widetilde{S}}(v) < d(v)/2$ **then**
    $\widetilde{S} \leftarrow \widetilde{S} \oplus v$ `// type-1 step`
  else if $\exists v, w, \left(d_{\widetilde{S}}(v) = \frac{d(v)}{2}\right.$ **and** $\varphi(\widetilde{S} \oplus v) \geq \varphi(\widetilde{S})$ **and** $\left.d_{\widetilde{S} \oplus v}(w) < \frac{d(w)}{2}\right)$ **then**
    $\widetilde{S} \leftarrow \widetilde{S} \oplus v$ `// build-up for another type-1 step`
  else
    $v \leftarrow$ a vertex such that $d_{\widetilde{S}}(v) = d(v)/2$ and $\varphi(\widetilde{S} \oplus v) > \varphi(\widetilde{S})$
    $\widetilde{S} \leftarrow \widetilde{S} \oplus v$ `// type-2 step`
return $\widetilde{S}$

that greedy steps (moving a vertex v with $d(v) < d_S(v)/2$) never decrease the FT value (Lemma 2.4), a stable cut can be a poor approximation for *AFTcut*. Consider, for example, a graph that consists of t triangles with a single common vertex u. Note that $d(u) = \Delta = 2t$, $d(v) = 2$, for every $v \neq u$, and $m = 3t$. The cut $S' = \{u\}$ is a stable cut, with $\varphi(S') = 0$. In order to transform S' into a 1/2-approximation, we have to decrease the crossing degree of the critical vertex u without decreasing the size of the cut. This can be done by moving a neighbor v of u from the opposite side of the cut, since $d_{S'}(v) = d(v)/2$.

In general, moving such vertex v (which we call a *neutral* move below) does not change the size of the cut, and decreases the crossing degree of u. Nevertheless, it does not always imply that the FT value increases, as there can be an additional critical vertex u' in S that is not affected, or that moving v creates a new critical vertex u'' with the same crossing degree as u.

Our algorithm (see Algorithm 4) is based on some key structural properties of stable cuts that we prove. Essentially, we show that any given cut S with FT value less than $(m - \Delta)/2$ either admits a *greedy step*, or a *neutral move followed by a greedy step*, or a *neutral move that increases the FT value*. Our algorithm is then a repeated application of such steps until the cut has the desired FT value; thus, it can be seen as a local search over two-move combinations, for maximizing the *sum* of the cut size and FT value.

Our key technical observation is that in a balanced cut S with an FT value less than $(m - \Delta)/2$, the critical vertex is unique. Moreover, letting $x_S(v) = d_S(v) - d(v)/2$ denote the *excess* contribution of a vertex v to the cut, it holds for the critical vertex u that $x_S(u) > \sum_{v \neq u} x_S(v) + \Delta - d(u)$, as proved in the following lemma.

▶ **Lemma 3.6.** *Let S be a stable cut in a graph $G = (V, E)$ such that $\varphi(S) < (m - \Delta)/2$. Then S has a unique critical vertex vertex u, and u satisfies*

$$d_S(u) > \sum_{v \neq u} x_S(v) + \Delta - d(u)/2 \ . \tag{1}$$

Moreover, u has a neighbor w in its opposite side of the cut, which satisfies $x_S(w) = 0$.

Proof. First, we show that S has a unique critical vertex. Since for every vertex v, $d_S(v) = d(v)/2 + x_S(v)$ and $\sum_{v\in V} d(v)/2 = m$, we get that

$$C_S = \frac{1}{2}\sum_{v\in V} d_S(v) = \frac{1}{2}\sum_{v\in V}\left(\frac{d(v)}{2} + x_S(v)\right) = \frac{m}{2} + \sum_{v\in V}\frac{x_S(v)}{2} \ . \tag{2}$$

Let u be a critical vertex, and assume, without loss of generality, that $u \in S$ (otherwise we swap S and $V - S$). On one hand, we have $\varphi(S) = C_S - d_S(u) = m/2 + \sum_{v\in V} x_S(v)/2 - d_S(u)$, and on the other hand, we have $\varphi(S) < (m - \Delta)/2$, which together imply:

$$m/2 + \sum_{v\in V} x_S(v)/2 - d_S(u) < (m - \Delta)/2 \ .$$

After a rearrangement, the latter implies (1). Using $d_S(u) = d(u)/2 + x_S(u)$ in (1) and simplifying, we get $x_S(u) > \sum_{v\neq u} x_S(v) + \Delta - d(u) \geq \sum_{v\neq u} x_S(v)$. Since u is an arbitrary critical vertex, this implies that u is the only critical vertex of S.

Next, let us show that there is a neighbor $w \in V - S$ of u (recall that $u \in S$) with $x_S(w) = 0$. Assume to the contrary that for every $v \notin S$ such that $\{u, v\} \in E$, it holds that $x_S(v) \geq 1/2$ (recall that S is stable, and hence $x_S(v)$ is a non-negative integer multiple of $1/2$). Using (2), this implies:

$$\begin{aligned} C_S = \frac{m}{2} + \sum_{v\in V}\frac{x_S(v)}{2} &\geq \frac{m}{2} + \frac{1}{2}\left(x_S(u) + \sum_{v:\{u,v\}\in\delta(S)} x_S(v)\right) \\ &\geq \frac{m}{2} + \frac{1}{2}\left(x_S(u) + \frac{d_S(u)}{2}\right) \geq \frac{m}{2} + x_S(u) \ , \end{aligned}$$

where we use $d_S(u) = |\{v : \{u, v\} \in \delta(S)\}|$ in the second inequality, and $d_S(u) = d(u)/2 + x_S(u) \geq 2x_S(u)$, in the third one. Since u is the critical vertex of S, this gives that

$$\varphi(S) = C_S - d_S(u) \geq \frac{m}{2} + x_S(u) - d_S(u) = \frac{m}{2} - \frac{d(u)}{2} \geq (m - \Delta)/2 \ ,$$

in contradiction to $\varphi(S) < (m - \Delta)/2$. This completes the proof. ◀

Note that in a stable cut S, $x_S(v)$ is a non-negative multiple of $1/2$, for all v. In most typical cases (e.g., when $d(u) < \Delta$, or when there are not too few nodes v with $x_S(v) > 0$), the inequality from the lemma quickly gives us the properties we claimed. However, covering all cases turns out to be quite tedious. The complete analysis can be found in the full version [19].

4 Fault Tolerance Against an Oblivious Adversary

We give an algorithm that approximates the fault tolerant Max-Cut against the oblivious adversary with (constant) k faults within an α_{GW}-approximation factor. The main idea is to frame the problem as a linear program (LP) with an exponential number of variables, then reduce the number of variables using a solution of its dual (with an exponential number of constraints but a polynomial number of variables). The dual is approximately solved by the ellipsoid algorithm together with an approximate separation oracle that is given by a Max-Cut algorithm. A similar approach has been used, e.g. in [35], for an unrelated problem.

▶ **Theorem 1.2.** *For every constant $k > 0$, there is a polynomial time α_{GW}-approximation algorithm for fault tolerant Max-Cut on general weighted graphs against an oblivious adversary and k faults.*

For simplicity, we present the algorithm for a single fault. In the full version [19], we show how to extend it to any constant number k of faults. The *OFTcut* problem can be formulated as the following LP, $(Primal_1)$, with an exponential number of variables.

$$\begin{aligned}
\max\quad & \sum_{S\subseteq V} P_S \cdot \sum_{e\in\delta(S)} w_e - Z && && (Primal_1)\\
\text{s.t.}\quad & \sum_{S\subseteq V} P_S \cdot \sum_{v:\{u,v\}\in\delta(S)} w_{\{u,v\}} \le Z && \forall u \in V && (3)\\
& \sum_{S\subseteq V} P_S \le 1 && && (4)\\
& 0 \le P_S && \forall S \subseteq V && (5)
\end{aligned}$$

The variable P_S represents the probability assigned to the cut $S \subseteq V$. The variable Z represents the expected weight that the adversary removes from the graph. Constraints (4-5) make P_S a probability distribution. In (3), for each vertex u, we bound by Z the expected weight that is removed from the cut when u fails. To see that the left hand side is indeed the expected removed weight, note that it equals $\sum_{S\subseteq V} P_S \cdot d_S(u)$.

Consider the dual problem of the LP above, $(Dual_1)$:

$$\begin{aligned}
\min\quad & Y && && (Dual_1)\\
\text{s.t.}\quad & \sum_{\{u,v\}\in\delta(S)} w_{\{u,v\}} - \sum_{u\in V} X_u \sum_{v:\{u,v\}\in\delta(S)} w_{\{u,v\}} \le Y && \forall S \subseteq V && (6)\\
& \sum_{u\in V} X_u \le 1 && && (7)\\
& 0 \le X_u && \forall u \in V && (8)
\end{aligned}$$

The dual LP captures the following problem: The adversary picks a distribution over the vertices, and the algorithm picks a cut (depending on the choice of the adversary). The goal of the adversary is to choose its distribution (without knowing the cut choice of the algorithm) so as to minimize the expected cut size after a random failure from its distribution.

The dual LP $(Dual_1)$ has an exponential number of constraints but only $|V|+1$ variables. Such LPs can be solved efficiently via the *ellipsoid method* [31], given an efficient *separation oracle*. The latter is an algorithm that given an assignment of values to the variables of the LP, reports a violated constraint if the assignment is infeasible, or otherwise reports that it is feasible. For the particular case of $(Dual_1)$, the ellipsoid algorithm can be viewed as a binary search over the values of Y, such that in each stage (fixed Y), a black-box procedure does a polynomial number of queries to a given separation oracle, and either reports the first solution $\{X_u\}_{u\in V}$ it finds such that $\{X_u\}_{u\in V}, Y$ is feasible according to the oracle, or reports that there is no such solution.

Let us see what a separation oracle looks like in our case. For given values $\{X_u\}_{u\in V}, Y$, let $G' = (V', E', w')$ be the graph with weights $w'_{\{u,v\}} = (1-X_u-X_v)w_{\{u,v\}}$. With this notation, constraint (6) becomes $C_{S,G'} \le Y$. In order to see if a given assignment of variables is feasible, it thus suffices to find a maximum weight cut S^* in G' and test if $C_{S^*,G'} \le Y$. Since Max-Cut is hard to solve exactly, we use an *approximate separation oracle*. Given $\{X_u\}_{u \in V}, Y$, it immediately returns the constraint (7), if it is violated, and otherwise computes a cut S_{ALG} in G' using a derandomized variant of the Goemans-Williamson algorithm [43], which we denote by Derandomized-Goemans-Williamson. Given an assignment $\{X_u\}_{u \in V}, Y$ to the variables in the LP, the oracle either outputs `feasible` or a violated constraint. If the size of the cut is larger than Y, it returns the violated constraint (6) corresponding to S_{ALG}, otherwise it reports that the solution is feasible (see Algorithm 5).

Algorithm 5 Approximate separation oracle.

Input: $\{X_u\}_{u \in V}, Y, G$
if $\sum\limits_{u \in V} X_u > 1$ **then**
  return violated constraint $\sum\limits_{u \in V} X_u \leq 1$
$S_{ALG} \leftarrow$ Derandomized-Goemans-Williamson(G')
if $C_{S_{ALG}, G'} > Y$ **then**
  return violated constraint for subset S_{ALG}
else
  return `feasible`

We show (Lemma 4.1) that if $\{X_u\}_{u \in V}, Y$ is feasible, then the oracle reports that it is feasible, and otherwise, it either reports a violated constraint, or incorrectly reports that it is feasible, in which case, however, $\{X_u\}_{u \in V}, Y/\alpha_{GW}$ is feasible.

▶ **Lemma 4.1.** *Given an assignment $\{X_u\}_{u \in V}, Y$ to the variables in ($Dual_1$) as input to the separation oracle in Algorithm 5, it holds that:*
1. *if the assignment is feasible, then the oracle returns* `feasible`*,*
2. *if the assignment is infeasible, then either the oracle outputs a violated constraint, or reports* `feasible`*, in which case $\{X_u\}_{u \in V}, Y/\alpha_{GW}$ is feasible.*

Proof. Let $\{X_u\}_{u \in V}, Y$ be an assignment to the variables of ($Dual_1$). If it is feasible, then it holds that $\sum\limits_{u \in V} X_u \leq 1$, and in addition every $S \subseteq V$ satisfies $C_{S,G'} \leq Y$, therefore the oracle returns `feasible`.

If the assignment is infeasible, there are two cases. If $\sum\limits_{u \in V} X_u > 1$, the oracle returns this violated constraint. Otherwise, there is a subset $S' \subseteq V$ such that $C_{S',G'} > Y$. Let S^* be an optimal solution for Max-Cut on G', and note that $C_{S^*,G'} > Y$. If $\alpha_{GW} \cdot C_{S^*,G'} > Y$, then we also have that $C_{S_{ALG},G'} > Y$ (since S_{ALG} is an α_{GW}-approximate Max-Cut), and the oracle returns the violated constraint for S_{ALG}. Otherwise, $C_{S^*,G'} \leq Y/\alpha_{GW}$. Since S^* is an optimal solution for Max-Cut on G', it follows that for every $S \subseteq V$, it holds that $C_{S,G'} \leq Y/\alpha_{GW}$, i.e., the solution $\{X_u\}_{u \in V}, Y/\alpha_{GW}$ is feasible. ◀

It is not hard to see that the application of the ellipsoid algorithm on ($Dual_1$) takes a polynomial time (i.e., at most as much time as it would take with an exact separation oracle), since our approximate oracle is (possibly) incorrect only on *the last* call from the ellipsoid algorithm (for a given Y), when it incorrectly reports a solution as feasible.

The output of the ellipsoid algorithm/binary search is an assignment $\{X_u\}_{u \in V}, Y$ to the variables of ($Dual_1$) such that $\{X_u\}_{u \in V}, Y$ is feasible according to the oracle, while $Y - \epsilon$ is infeasible with every assignment to the X variables, where ϵ is the precision of the binary search. As observed above, we have that $\{X_u\}_{u \in V}, Y/\alpha_{GW}$ is feasible, and it follows that if

Y^* is the optimal value of $(Dual_1)$, then $Y - \epsilon \le Y^* \le Y/\alpha_{GW}$. Since the ellipsoid algorithm queries the oracle a polynomial number of times, there is a set $\mathcal{H} \subseteq 2^V$ of a polynomial number of cuts S, for which constraint (6) is queried. Consider a modified variant of $(Dual_1)$, called $(Dual_2)$, where only constraints of cuts in $\mathcal{H}$ are present. Let Y_2^* be the optimal value of $(Dual_2)$. Note that $Y_2^* \le Y^*$. Note also that the ellipsoid algorithm returns exactly the same solution $\{X_u\}_{u\in V}, Y$, when executed on $(Dual_1)$ and $(Dual_2)$ (since our algorithm is deterministic, and only constraints in $\mathcal{H}$ are queried); hence, we have $Y - \epsilon \le Y_2^*$. Finally, let us consider the primal LP corresponding to $(Dual_2)$ it is obtained from $(Primal_1)$ by removing variables P_S with $S \notin \mathcal{H}$ (i.e., setting $P_S = 0$).

The new primal has polynomially many constraints and variables, so can be solved in polynomial time. From the arguments above, we have that its optimal value Y_2^* satisfies $Y - \epsilon \le Y_2^* \le Y^* \le Y/\alpha_{GW}$. Recalling that Y^* is the optimal value for the original LP, we see that Y_2^* is a α_{GW}-approximation (with any polynomial precision ϵ).

5 Hardness of Approximation

In this section we show that assuming the Unique Games Conjecture, one cannot approximate $AFTcut$ and $OFTcut$ within a factor greater than α_{GW}. Formally, we prove the following:

▶ **Theorem 1.3.** *Assuming the Unique Games Conjecture and $NP \not\subseteq BPP$, there is no polynomial time $(\alpha_{GW} + \epsilon)$-approximation algorithm for fault tolerant Max-Cut in unweighted graphs, for any constant $\epsilon > 0$. This holds for both adaptive and oblivious adversaries.*

In both cases, given an unweighted instance G of Max-Cut, we construct an unweighted graph G', as follows: we take the disjoint union of G with a star with $n = |V|$ leaves and a center u^*, and add an edge joining u^* to an arbitrary vertex $v_1 \in V$. This completes the construction of G'. Clearly, this is a polynomial construction.

We show for each kind of adversary how to translate a given (approximate) solution to $AFTcut$ or $OFTcut$ in G' into a solution to Max-Cut in G, which would imply the corresponding inapproximability results, using the fact that Max-Cut is hard to approximate within a factor better than α_{GW} [39]. We only present the proof for $OFTcut$, leaving $AFTcut$ to the full version. We use the following simple observation.

▶ **Observation 5.1.** *Let $S \subseteq V$ be a cut in G, and $S' = S \cup \{u^*\}$. It holds that in G', u^* is a critical vertex of S', i.e., $\varphi(S') = C_{S'-u^*,G'-u^*}$. For every cut $S'' \subseteq V'$, we have $C_{S''-u^*,G'} = C_{S''\cap V,G}$.*

The proof follows from the fact that for every vertex $v \in V'$, $d_{S'}(v) \le n \le d_{S'}(u^*)$, and that all edges in $G' - u^*$ belong to G.

We show first that the optimal values for $OFTcut$ in G' and Max-Cut in G are equal.

▶ **Lemma 5.2.** *Let $\mathcal{D}^*$ be the distribution of an optimal $OFTcut$ in G', and S^*_{mc} be an optimal Max-Cut in G. It holds that $\mu(\mathcal{D}^*, G') = C_{S^*_{mc},G}$.*

Proof. Let $\widetilde{S} = S^*_{mc} \cup \{u^*\}$, and let $\mathcal{D}$ be the distribution that assigns probability 1 to $\widetilde{S}$ and probability 0 to all other cuts. By Observation 5.1, u^* is a critical vertex, hence for every vertex $v \in G'$, we have $\mathop{\mathbb{E}}_{S\sim\mathcal{D}}[C_{S-u^*,G'-u^*}] = C_{\widetilde{S}-u^*,G'-u^*} \le C_{\widetilde{S}-v,G'-v} = \mathop{\mathbb{E}}_{S\sim\mathcal{D}}[C_{S-v,G'-v}]$. Using Observation 5.1 again, we have $\mu(\mathcal{D}^*, G') \ge \mu(\mathcal{D}, G') \ge C_{\widetilde{S}-u^*,G'-u^*} = C_{S^*_{mc},G}$. Next, by Observation 5.1, we have $C_{S-u^*,G'-u^*} = C_{S\cap V,G} \le C_{S^*_{mc},G}$, for every cut S from the support of $\mathcal{D}^*$, which implies that $\mu(\mathcal{D}^*, G') \le \mathop{\mathbb{E}}_{S\sim\mathcal{D}^*}[C_{S-u^*,G'-u^*}] \le C_{S^*_{mc},G}$. This completes the proof. ◀

Proof of Theorem 1.3 for an Oblivious Adversary. Assume, for a contradiction, that we have an α-approximation algorithm for $OFTcut$, for $\alpha = \alpha_{GW} + \epsilon > \alpha_{GW}$. We design a randomized approximation algorithm for Max-Cut. Let G be an input to Max-Cut. Construct the graph G' as described above. Let $\mathcal{D}$ be the distribution of an α-approximate $OFTcut$ in G'. By Lemma 5.2, we have $\mathop{\mathbb{E}}_{S\sim\mathcal{D}}[C_{S-u^*,G'-u^*}] \geq \mu(\mathcal{D}, G') \geq \alpha \cdot C_{S^*_{mc},G}$, where S^*_{mc} is a Max-Cut in G. By Observation 5.1, it holds that $C_{S_{ft}-u^*,G'-u^*} = C_{S_{ft}\cap V,G} \leq C_{S^*_{mc}}$ for every cut S_{ft} in the support of $\mathcal{D}$. Letting $p = \mathbb{P}\left[C_{S-u^*,G'-u^*} \geq (\alpha - \epsilon/2)C_{S^*_{mc}}\right]$, we have

$$\alpha \cdot C_{S^*_{mc},G} \leq \mathop{\mathbb{E}}_{S\sim\mathcal{D}}[C_{S-u^*,G'-u^*}] \leq p \cdot C_{S^*_{mc},G} + (1-p) \cdot (\alpha - \epsilon/2)C_{S^*_{mc},G} \ ,$$

implying that $p \geq \epsilon/2$. Thus, for a random cut S_{ft} sampled from $\mathcal{D}$, it holds that $S_{ft} \cap V$ is an $(\alpha - \epsilon/2)$-approximation to Max-Cut, with probability $\epsilon/2$, where $\alpha - \epsilon/2 > \alpha_{GW}$. This contradicts to our assumption about the Unique Games Conjecture and $NP \not\subseteq BPP$. ◀

6 Discussion

Our work leaves several open questions regarding fault tolerant Max-Cut. An immediate question is to bridge the (rather small) gap between our approximation of $(0.8780 - \epsilon)$ and our hardness of α_{GW} for k-$AFTcut$.

The central bottleneck is that Simultaneous Max-Cut, a main ingredient in our algorithm, has hardness of approximation that is slightly below α_{GW} and equals $(\alpha_{GW} - \delta)$ (where $\delta \geq 10^{-5}$) [11]. Thus, either one finds a different algorithm for k-$AFTcut$ that does not rely on Simultaneous Max-Cut and achieves an approximation of α_{GW}, or one can extend the hardness result of [11] to k-$AFTcut$ and thus rule out an approximation of α_{GW} for k-$AFTcut$. Another question is what approximation factors can be obtained for $AFTcut$ on general weighted graphs.

Another interesting question is how to deal with a non-constant number of faults, for both of the adversaries. Since the number of all possible cases of failure is not polynomial, a new approach may be needed. There are techniques that are used to deal with a non-constant number of faults, e.g., failure sampling, that is presented in [24]. It would be interesting to see whether these techniques can be used for fault tolerant Max-Cut as well.

One more important and intriguing open question is what happens in other fault tolerant problems when an oblivious adversary is considered. We are unaware of previous algorithms for an oblivious adversary in the fault-tolerance literature. Since an oblivious adversary is arguably more realistic in its nature, and since it is likely that one can get improved algorithms for this case, pursuing this line of research could be crucial for many additional fundamental problems involving fault tolerance.

References

1 Noga Alon, Shiri Chechik, and Sarel Cohen. Deterministic combinatorial replacement paths and distance sensitivity oracles. 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, Patras, Greece*, volume 132 of *LIPIcs*, pages 12:1–12:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ICALP.2019.12.

2 Noga Alon and Assaf Naor. Approximating the cut-norm via grothendieck's inequality. *SIAM J. Comput.*, 35(4):787–803, 2006.

3 Eric Angel, Evripidis Bampis, and Laurent Gourvès. Approximation algorithms for the bi-criteria weighted MAX-CUT problem. *Discret. Appl. Math.*, 154(12):1685–1692, 2006. doi:10.1016/j.dam.2006.02.008.

4 Takao Asano and David P Williamson. Improved approximation algorithms for MAX SAT. *Journal of Algorithms*, 42(1):173–202, 2002.

5 Per Austrin, Siavosh Benabbas, and Konstantinos Georgiou. Better balance by being biased: A 0.8776-approximation for Max Bisection. *ACM Trans. Algorithms*, 13(1):2:1–2:27, 2016.

6 Dmitrii Avdiukhin, Slobodan Mitrovic, Grigory Yaroslavtsev, and Samson Zhou. Adversarially robust submodular maximization under knapsack constraints. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 148–156. ACM, 2019. doi:10.1145/3292500.3330911.

7 Adi Avidor, Ido Berkovitch, and Uri Zwick. Improved approximation algorithms for MAX NAE-SAT and MAX SAT. In Thomas Erlebach and Giuseppe Persiano, editors, *Approximation and Online Algorithms, Third International Workshop, WAOA 2005, Palma de Mallorca, Spain, October 6-7, 2005, Revised Papers*, volume 3879 of *Lecture Notes in Computer Science*, pages 27–40. Springer, 2005. doi:10.1007/11671411_3.

8 Surender Baswana, Keerti Choudhary, Moazzam Hussain, and Liam Roditty. Approximate single source fault tolerant shortest path. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1901–1915. SIAM, 2018. doi:10.1137/1.9781611975031.124.

9 Surender Baswana, Keerti Choudhary, and Liam Roditty. Fault tolerant reachability for directed graphs. In Yoram Moses, editor, *Distributed Computing - 29th International Symposium, DISC 2015, Tokyo, Japan, October 7-9, 2015, Proceedings*, volume 9363 of *Lecture Notes in Computer Science*, pages 528–543. Springer, 2015. doi:10.1007/978-3-662-48653-5_35.

10 Surender Baswana, Keerti Choudhary, and Liam Roditty. Fault tolerant subgraph for single source reachability: generic and optimal. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 509–518. ACM, 2016. doi:10.1145/2897518.2897648.

11 Amey Bhangale and Subhash Khot. Simultaneous max-cut is harder to approximate than max-cut. In Shubhangi Saraf, editor, *35th Computational Complexity Conference, CCC 2020, July 28-31, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 169 of *LIPIcs*, pages 9:1–9:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.CCC.2020.9.

12 Amey Bhangale, Subhash Khot, Swastik Kopparty, Sushant Sachdeva, and Devanathan Thiruvenkatachari. Near-optimal approximation algorithm for simultaneous max-cut. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1407–1425. SIAM, 2018. doi:10.1137/1.9781611975031.93.

13 Amey Bhangale, Swastik Kopparty, and Sushant Sachdeva. Simultaneous approximation of constraint satisfaction problems. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 193–205. Springer, 2015. doi:10.1007/978-3-662-47672-7_16.

14 Davide Bilò, Luciano Gualà, Stefano Leucci, and Guido Proietti. Fault-tolerant approximate shortest-path trees. *Algorithmica*, 80(12):3437–3460, 2018. doi:10.1007/s00453-017-0396-z.

15 Greg Bodwin, Michael Dinitz, Merav Parter, and Virginia Vassilevska Williams. Optimal vertex fault tolerant spanners (for fixed stretch). In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1884–1900. SIAM, 2018. doi:10.1137/1.9781611975031.123.

16 Greg Bodwin, Fabrizio Grandoni, Merav Parter, and Virginia Vassilevska Williams. Preserving distances in very faulty graphs. 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 73:1–73:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.73.

17 Greg Bodwin and Shyamal Patel. A trivial yet optimal solution to vertex fault tolerant spanners. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 541–543. ACM, 2019. doi:10.1145/3293611.3331588.

18 Austin Buchanan, Je Sang Sung, Sergiy Butenko, and Eduardo L. Pasiliao. An integer programming approach for fault-tolerant connected dominating sets. *INFORMS J. Comput.*, 27(1):178–188, 2015. doi:10.1287/ijoc.2014.0619.

19 Keren Censor-Hillel, Noa Marelly, Roy Schwartz, and Tigran Tonoyan. Fault tolerant max-cut. *CoRR*, arXiv:2105.01138, 2021. arXiv:2105.01138.

20 Moses Charikar, Venkatesan Guruswami, and Anthony Wirth. Clustering with qualitative information. *J. Comput. Syst. Sci.*, 71(3):360–383, 2005.

21 Shiri Chechik and Sarel Cohen. Near optimal algorithms for the single source replacement paths problem. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 2090–2109. SIAM, 2019. doi:10.1137/1.9781611975482.126.

22 Shiri Chechik and Ofer Magen. Near optimal algorithm for the directed single source replacement paths problem. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 81:1–81:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.81.

23 Shiri Chechik and David Peleg. Robust fault tolerant uncapacitated facility location. *Theor. Comput. Sci.*, 543:9–23, 2014. doi:10.1016/j.tcs.2014.05.013.

24 Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 323–332. ACM, 2011. doi:10.1145/1993636.1993680.

25 Michael Dinitz and Robert Krauthgamer. Fault-tolerant spanners: better and simpler. In Cyril Gavoille and Pierre Fraigniaud, editors, *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011, San Jose, CA, USA, June 6-8, 2011*, pages 169–178. ACM, 2011. doi:10.1145/1993806.1993830.

26 Uriel Feige and Michel Goemans. Approximating the value of two power proof systems, with applications to Max 2-SAT and Max DiCut. In *istcs*, page 0182. IEEE, 1995.

27 M. R. Garey, David S. Johnson, and Larry J. Stockmeyer. Some simplified np-complete graph problems. *Theor. Comput. Sci.*, 1(3):237–267, 1976. doi:10.1016/0304-3975(76)90059-1.

28 Mohsen Ghaffari and Merav Parter. Near-optimal distributed algorithms for fault-tolerant tree structures. In Christian Scheideler and Seth Gilbert, editors, *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA 2016, Asilomar State Beach/Pacific Grove, CA, USA, July 11-13, 2016*, pages 387–396. ACM, 2016. doi:10.1145/2935764.2935795.

29 Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42(6):1115–1145, 1995. doi:10.1145/227683.227684.

30 Fabrizio Grandoni and Virginia Vassilevska Williams. Improved distance sensitivity oracles via fast single-source replacement paths. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 748–757. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.17.

31 Martin Grötschel, László Lovász, and Alexander Schrijver. *The Ellipsoid Method*, pages 64–101. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993. doi:10.1007/978-3-642-78240-4_4.

32 Sudipto Guha, Adam Meyerson, and Kamesh Munagala. A constant factor approximation algorithm for the fault-tolerant facility location problem. *J. Algorithms*, 48(2):429–440, 2003. doi:10.1016/S0196-6774(03)00056-7.

33 Manoj Gupta and Shahbaz Khan. Multiple source dual fault tolerant BFS trees. 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 127:1–127:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.127.

34 Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. doi:10.1145/502090.502098.

35 Kamal Jain, Mohammad Mahdian, and Mohammad R. Salavatipour. Packing steiner trees. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, January 12-14, 2003, Baltimore, Maryland, USA*, pages 266–274. ACM/SIAM, 2003. URL: http://dl.acm.org/citation.cfm?id=644108.644154.

36 Kamal Jain and Vijay V. Vazirani. An approximation algorithm for the fault tolerant metric facility location problem. *Algorithmica*, 38(3):433–439, 2004. doi:10.1007/s00453-003-1070-1.

37 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. doi:10.1007/978-1-4684-2001-2_9.

38 Subhash Khot. On the power of unique 2-prover 1-round games. In John H. Reif, editor, *Proceedings on 34th Annual ACM Symposium on Theory of Computing, May 19-21, 2002, Montréal, Québec, Canada*, pages 767–775. ACM, 2002. doi:10.1145/509907.510017.

39 Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, 2007. doi:10.1137/S0097539705447372.

40 Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-ϵ. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. doi:10.1016/j.jcss.2007.06.019.

41 Christos Levcopoulos, Giri Narasimhan, and Michiel H. M. Smid. Improved algorithms for constructing fault-tolerant spanners. *Algorithmica*, 32(1):144–156, 2002. doi:10.1007/s00453-001-0075-x.

42 Michael Lewin, Dror Livnat, and Uri Zwick. Improved rounding techniques for the MAX 2-sat and MAX DI-CUT problems. In William J. Cook and Andreas S. Schulz, editors, *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*, volume 2337 of *Lecture Notes in Computer Science*, pages 67–82. Springer, 2002. doi:10.1007/3-540-47867-1_6.

43 Sanjeev Mahajan and H. Ramesh. Derandomizing approximation algorithms based on semidefinite programming. *SIAM J. Comput.*, 28(5):1641–1663, 1999. doi:10.1137/S0097539796309326.

44 Shiro Matuura and Tomomi Matsui. 63-approximation algorithm for MAX DICUT. In Michel X. Goemans, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques, 4th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2001 and 5th International Workshop on Randomization and Approximation Techniques in Computer Science, RANDOM 2001 Berkeley, CA, USA, August 18-20, 2001, Proceedings*, volume 2129 of *Lecture Notes in Computer Science*, pages 138–146. Springer, 2001. doi:10.1007/3-540-44666-4_17.

45 Elchanan Mossel, Ryan O'Donnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. In *Foundations of Computer Science, (FOCS)*, pages 21–30, 2005.

46 James B. Orlin, Andreas S. Schulz, and Rajan Udwani. Robust monotone submodular function maximization. In Quentin Louveaux and Martin Skutella, editors, *Integer Programming and Combinatorial Optimization - 18th International Conference, IPCO 2016, Liège, Belgium, June 1-3, 2016, Proceedings*, volume 9682 of *Lecture Notes in Computer Science*, pages 312–324. Springer, 2016. doi:10.1007/978-3-319-33461-5_26.

47 Merav Parter. Vertex fault tolerant additive spanners. In Fabian Kuhn, editor, *Distributed Computing - 28th International Symposium, DISC 2014, Austin, TX, USA, October 12-15, 2014. Proceedings*, volume 8784 of *Lecture Notes in Computer Science*, pages 167–181. Springer, 2014. doi:10.1007/978-3-662-45174-8_12.

48 Merav Parter. Dual failure resilient BFS structure. In Chryssis Georgiou and Paul G. Spirakis, editors, *Proceedings of the 2015 ACM Symposium on Principles of Distributed Computing, PODC 2015, Donostia-San Sebastián, Spain, July 21 - 23, 2015*, pages 481–490. ACM, 2015. doi:10.1145/2767386.2767408.

49 Merav Parter. Distributed constructions of dual-failure fault-tolerant distance preservers. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPIcs*, pages 21:1–21:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.DISC.2020.21.

50 Merav Parter and David Peleg. Sparse fault-tolerant BFS trees. *CoRR*, abs/1302.5401, 2013. arXiv:1302.5401.

51 Merav Parter and David Peleg. Fault tolerant approximate BFS structures. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1073–1092. SIAM, 2014. doi:10.1137/1.9781611973402.80.

52 Merav Parter and David Peleg. Fault tolerant BFS structures: A reinforcement-backup tradeoff. In Guy E. Blelloch and Kunal Agrawal, editors, *Proceedings of the 27th ACM on Symposium on Parallelism in Algorithms and Architectures, SPAA 2015, Portland, OR, USA, June 13-15, 2015*, pages 264–273. ACM, 2015. doi:10.1145/2755573.2755590.

53 Prasad Raghavendra and Ning Tan. Approximating csps with global cardinality constraints using SDP hierarchies. In *Symposium on Discrete Algorithms (SODA)*, pages 373–387. SIAM, 2012.

54 Liam Roditty and Uri Zwick. Replacement paths and k simple shortest paths in unweighted directed graphs. *ACM Trans. Algorithms*, 8(4):33:1–33:11, 2012. doi:10.1145/2344422.2344423.

55 Shay Solomon. From hierarchical partitions to hierarchical covers: optimal fault-tolerant spanners for doubling metrics. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 363–372. ACM, 2014. doi:10.1145/2591796.2591864.

56 Chaitanya Swamy. Correlation clustering: Maximizing agreements via semidefinite programming. In *Symposium on Discrete Algorithms*, SODA, pages 526–527, 2004.

57 Chaitanya Swamy and David B. Shmoys. Fault-tolerant facility location. *ACM Trans. Algorithms*, 4(4):51:1–51:27, 2008. doi:10.1145/1383369.1383382.

58 Luca Trevisan, Gregory B. Sorkin, Madhu Sudan, and David P. Williamson. Gadgets, approximation, and linear programming. *SIAM J. Comput.*, 29(6):2074–2097, 2000. doi:10.1137/S0097539797328847.

59 Jiao Zhou, Zhao Zhang, Shaojie Tang, Xiaohui Huang, and Ding-Zhu Du. Breaking the $O(\ln n)$ barrier: An enhanced approximation algorithm for fault-tolerant minimum weight connected dominating set. *INFORMS J. Comput.*, 30(2):225–235, 2018. doi:10.1287/ijoc.2017.0775.