Adversarially-Robust Gossip Algorithms for Approximate Quantile and Mean Computations

Bernhard Haeupler
INSAIT & Sofia University "St. Kliment Ohridski", Bulgaria
ETH Zürich, Switzerland

Marc Kaufmann
ETH Zürich, Switzerland

Raghu Raman Ravi
ETH Zürich, Switzerland

Ulysse Schaller
ETH Zürich, Switzerland

Abstract

This paper presents gossip algorithms for aggregation tasks that demonstrate both robustness to adversarial corruptions of any order of magnitude and optimality across a substantial range of these corruption levels.

Gossip algorithms distribute information in a scalable and efficient way by having random pairs of nodes exchange small messages. Value aggregation problems are of particular interest in this setting, as they occur frequently in practice, and many elegant algorithms have been proposed for computing aggregates and statistics such as averages and quantiles. An important and well-studied advantage of gossip algorithms is their robustness to message delays, network churn, and unreliable message transmissions. However, these crucial robustness guarantees only hold if all nodes follow the protocol and no messages are corrupted.

In this paper, we remedy this by providing a framework to model both adversarial participants and message corruptions in gossip-style communications by allowing an adversary to control a small fraction of the nodes or corrupt messages arbitrarily. Despite this very powerful and general corruption model, we show that robust gossip algorithms can be designed for many important aggregation problems. Our algorithms guarantee that almost all nodes converge to an approximately correct answer with optimal efficiency and essentially as fast as without corruptions.

The design of adversarially-robust gossip algorithms poses completely new challenges. Despite this, our algorithms remain very simple variations of known non-robust algorithms with often only subtle changes to avoid non-compliant nodes gaining too much influence over outcomes. While our algorithms remain simple, their analysis is much more complex and often requires a completely different approach than the non-adversarial setting.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Gossip Algorithms, Distributed Computing, Adversarial Robustness

Digital Object Identifier 10.4230/LIPIcs.ITCS.2026.74

Related Version *Full Version*: `https://arxiv.org/abs/2502.15320`

Funding *Bernhard Haeupler*: Partially funded by the Ministry of Education and Science of Bulgaria's support for INSAIT as part of the Bulgarian National Roadmap for Research Infrastructure and through the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (ERC grant agreement 949272).
Marc Kaufmann: Supported by the Swiss National Science Foundation [grant number 200021_192079].
Raghu Raman Ravi: Supported by the Swiss National Science Foundation [grant number 0003390].
Ulysse Schaller: Supported by the Swiss National Science Foundation [grant number 200021_192079].

1 Introduction

Distributed computing provides a resource-efficient framework for scalable algorithms that can handle the vast amounts of data produced by the increasingly interconnected and digitalized world. Designing fast and reliable algorithms to compute aggregates, such as averages, quantiles, maxima, or minima, is one active area of research in this field. These algorithms are especially vital in the context of sensor networks and P2P networks [9, 16, 18, 20].

A promising class of distributed algorithms are gossip algorithms, which are typically simple, fast, scalable, and robust to errors. Due to this, many gossip algorithms have been developed for the aggregation of information in the real world. Their communication protocol is strikingly simple. In these algorithms, communication between nodes occurs in (synchronous) rounds – in each round, every node selects a communication partner uniformly at random from the set of all nodes and chooses to either push or pull a message from its partner. These messages are usually small – $O(\log n)$ bits, where n is the total number of nodes.

Push-based and pull-based protocols each have their own unique advantages and shortcomings. In [14], the authors gave a simple, fast gossip algorithm to compute the mean and built on this to compute other aggregates, including arbitrary quantiles. The main algorithm to compute the mean is a simple yet elegant push-based protocol where each node pushes half of its current value to a random node in every round. Since such a push-protocol preserves the total sum of the values, it is not hard to see why this algorithm computes the mean, and it can be shown that it converges in $O(\log n)$ rounds with high probability. Improved algorithms that reduce the total number of messages sent were given in [3] and [13]. In [10], the authors gave a faster, elegant, and in fact optimal algorithm to compute exact quantiles in $O(\log n)$ rounds with high probability, even in the presence of random faults. Their main algorithm is a pull-based tournament-style algorithm, where each node repeatedly updates its value to the median of the values of three randomly chosen nodes. This computes an approximate median extremely quickly – in $O(\log\log n + \log\frac{1}{\varepsilon})$ rounds for an ε-approximation – and can be bootstrapped to compute exact quantiles in $O(\log n)$ rounds.

One obstacle to the real-world deployment of these algorithms is the presence of message corruptions, be they malicious or benign. In the case of some algorithms, such as the median finding algorithm in [10], they are shown to be robust to random failures, that is, where nodes fail to perform their (pull- or push-) operation. The much more challenging problem, however, lies in the design of algorithms that are also robust to – potentially adversarial – message corruption. This arguably also greatly increases their applicability in the real world. Indeed, some algorithms break completely in the presence of even a few adversarial faults in each round. This is the case for the push-protocol in [14], where if even a single node reports a value that is abnormally large, all nodes would converge to an average that is much larger than the true average. Moreover, the proof techniques used there, such as the mass preservation property, completely break down in the presence of adversarial corruption. However, a pull-based approach like the one used in [10] naturally limits the number of corrupted messages, making it a powerful paradigm to address adversarial message faults. Our algorithms additionally ensure that every node sends the same value in a given round, regardless of how many nodes pull from it in that round. This approach simplifies the problem of adversarial corruption and makes the algorithm itself simpler.

To ensure that our algorithms are suitable for realistic settings, we strive to model faults in the most general manner possible. Some fundamental questions that arise include the amount of (computational) power that a potential adversary may have or how many messages may be affected by corruptions. We need to strike a balance and allow enough power to

retain a meaningful adversary, but not so much that it stifles all reasonable attempts at node communication and eliminates nontrivial results. In the following, we briefly sketch the setting, concentrating for convenience on the example of an adversary, but the model readily covers nonmalicious message corruptions.

We propose an adversary that can influence a β fraction of the nodes in each round, which we call a *β-strong adversary*, see section 3 for a more formal definition. The adversary can corrupt any message pulled from the nodes under its influence in this round and can arbitrarily change their contents. We do not make any assumptions about the computational power of the adversary. This model covers the special case where the same βn nodes are controlled by the adversary throughout all rounds (in other words, the case where there are βn adversarial nodes).

With the introduction of such adversarial faults, it is not reasonable to ask the nodes to produce exact answers. Thus, the best that we can hope for is an approximate solution, which is exactly the type of problem we investigate. In the *ε-approximate quantile problem*, the goal is to compute a value whose rank is in the interval $[(\phi - \varepsilon)n, (\phi + \varepsilon)n]$. We will start by tackling the special case $\phi = 1/2$ of computing an ε-approximate median, and show afterwards how the problem of computing an approximation of arbitrary quantiles can be reduced to that case. We then turn to the problem of finding the (approximate) mean, which is not as robust to outliers as the median. Thus, computing the mean, even approximately, in the presence of adversarial faults is difficult, as they can sabotage any algorithm by injecting extreme outliers into the computation. The best one can hope for this *ε-approximate mean problem* is therefore to find a value that is at most an additive εM away from the true mean, where the values are assumed to be restricted to the interval $[0, M]$.

The scenario with βn adversarial nodes tells us that we cannot hope for an approximation factor ε smaller than β. Indeed, one could simply have all the adversarial nodes reliably report an extremely large or an extremely small value, which would shift the quantiles of the "good nodes" by β, and the same scenario can be applied to the mean approximation problem as well. Hence, all our results have $\varepsilon \geq \beta$ as an (implicit) assumption. In some applications – and given that a β fraction of nodes might never output a correct answer anyway – it might be acceptable that some other nodes also hold an incorrect answer, provided this number is small enough, especially if this accelerates the algorithm in return. To investigate the possibility of trading off speed with correctness, we analyze our algorithms from the perspective of parametrized correctness, which is new in this setting. Our gossip algorithms are said to have *correctness level* γ if at most γn nodes are storing an incorrect value (e.g. a value with rank outside $[1/2 - \varepsilon, 1/2 + \varepsilon]$ for the ε-approximate median problem) upon termination. Note that we can always assume that $\gamma \geq \frac{1}{2n}$, since $\gamma < \frac{1}{n}$ already implies that there must be no incorrect node.

Paper Organization

The remainder of the paper is organized as follows. We give an overview of our results in section 2, followed by a description of our model and an intuitive description of our three algorithms – complemented by their pseudocode – in section 3. Section 4 contains a brief discussion of related work. Then, we proceed to provide a proof sketch of our main theorems in order: section 5 is dedicated to our median algorithm and the proof of Theorem 1; section 6 concerns itself with our quantile shifting algorithm and the proof of Theorem 2; section 7 deals with our mean algorithm. Then, section 8 contains the discussion of our lower bounds on the round complexity, including a proof of Theorem 4. Finally, in the appendix, we first collect standard concentration results and prove some technical inequalities which are used throughout the paper, then give a complete proof of all our main results.

2 Results

In the remainder of the paper, we say that an event $\mathcal{E}$ occurs *with high probability (w.h.p.)* if $\mathbb{P}[\mathcal{E}] = 1 - n^{-\Omega(1)}$. Our first main result gives a fast and robust algorithm for estimating the median. The pseudocode can be found in Algorithm 1 and is complemented by the runtime and correctness guarantees provided in our first theorem.

▶ **Theorem 1.** *For any $\varepsilon(n), \beta(n), \gamma(n) \in (0,1)$ satisfying $\beta \leq \frac{\varepsilon}{14}$ and $\varepsilon = \Omega\left(\frac{1}{n^{0.0019}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that solves the ε-approximate median problem in the presence of a β-strong adversary with correctness level γ with high probability in $O\left(\log\frac{1}{\varepsilon} + \log\log\frac{1}{\gamma} + \frac{\log(1/\gamma)}{\log(1/\beta)}\right)$ rounds.*

Next, we investigate the problem of approximate quantiles, where the objective is to compute an ε-approximate ϕ-quantile, i.e. the $\lceil\phi\rceil$-th smallest value. Notice that it suffices to first transform the values of the nodes such that an ε-approximate ϕ-quantile in the original setting is the same as an ε-approximate median in this new setting. Our next main result allows for adversarially robust quantile shifting, which along with our previous result yields an adversarially robust algorithm to approximate quantiles. The pseudocode of the quantile shifting algorithm is provided in Algorithm 2, the corresponding runtime guarantees are given in the theorem below.

▶ **Theorem 2.** *For any $\varepsilon(n), \beta(n) \in (0,1)$ satisfying $\varepsilon \leq \frac{1}{6}$, $\beta \leq \frac{\varepsilon^{2.5}}{16}$, and $\varepsilon = \Omega\left(\frac{(\log n)^{1/5}}{n^{1/5}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that reduces any ε-approximate quantile problem into an ε-approximate median problem in the presence of a β-strong adversary with high probability in $O\left(\log\frac{1}{\varepsilon}\right)$ rounds.*

Next, we investigate the approximate mean problem with M-bounded values. Here, every node is initially given a value in the range $[0, M]$ (where M is polynomially bounded in n), and the problem is to find the mean of the values up to an additive error of εM. In this setting, we provide the following adversarially robust result for computing approximate means. The pseudocode for our algorithm is given in Algorithm 3, its runtime and correctness guarantees are stated in the subsequent theorem.

▶ **Theorem 3.** *For any $\varepsilon(n), \beta(n), \gamma(n) \in (0,1)$ satisfying $\beta \leq \left(\frac{\varepsilon}{100}\right)^{2.5}$ and $\varepsilon = \Omega\left(\frac{(\log n)^{6/5}}{n^{1/5}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that solves the ε-approximate mean problem with M-bounded values in the presence of a β-strong adversary with correctness level γ with high probability in $O\left(\log\frac{1}{\varepsilon} + \log\frac{1}{\gamma+\beta} + \frac{\log(1/\gamma)}{\log(1/\beta)}\right)$ rounds.*

We note that the approximation here is weaker as it depends on the promised bound on M. The reason for this is simple: the adversary can always claim to have a value that is very large, and this would inevitably skew the computed average to a very large value. Thus, the best one can hope for is to give an error bound as a function of a bound on the possible communicated values.

Note that in the special case where all nodes have values in $\{0,1\}$, the above yields an approximate *counting algorithm* with an additive error of at most εn.

Finally, we prove the following theorem, which can be used to lower-bound the round complexity of gossip algorithms in the presence of adversarial nodes.

▶ **Theorem 4.** *Let $\beta(n), \gamma(n) \in (0,1)$. Then, in the presence of a β-strong adversary, for any gossip algorithm that runs for less than $\frac{\log(1/2\gamma)}{\log(1/\beta)}$ rounds, with probability at least $1/10$ more than a γ fraction of the nodes have only received corrupted messages (if any).*

We also extend the $\Omega\big(\log\log n+\log\frac{1}{\varepsilon}\big)$ lower bound given in [10] to an $\Omega\big(\log\log\frac{1}{\gamma}+\log\frac{1}{\varepsilon}\big)$ lower bound in our setting, see Proposition 5 in section 8. These results prove that our median and quantile algorithms have optimal round complexity and the mean algorithm has almost optimal round complexity. We note that our theorems require varying upper bounds of β in terms of ε. As mentioned in the introduction, it is not possible to hope for ε-approximation with $\beta > \varepsilon$, as in particular all adversarial nodes might always output values that are completely out of the range of the correct values. Our analysis here is likely not tight and we did not optimize these bounds since our algorithms already cover a substantial range of values. The same holds for the upper bound on ε in Theorem 2. For lower bounds of ε, note that any $\varepsilon < \frac{1}{2n}$ would yield an exact algorithm, which is out of reach due to the presence of the adversary. Hence a lower bound which is inversely polynomial in n follows naturally, where again we did not optimize for the precise exponent, and simply stated what was required for our analysis in order to prove concentration of the expected behavior of the algorithms. However, the key reason why we do not view this restriction as major is a different one: For many applications, approximation factors that are a small constant fraction are completely sufficient. Moreover, this is also the parameter range in which our algorithms are extremely fast. More concretely, when both ε and γ are constant, our algorithms are guaranteed to terminate in constant time, no matter how large n is.

We summarize our results in Table 1.

Table 1 Comparison of the running times of the three algorithms with their lower bounds.

Algorithm	Time Complexity	Lower Bound
Approximate Median	$O\left(\log\frac{1}{\varepsilon}+\log\log\frac{1}{\gamma}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$	$\Omega\left(\log\frac{1}{\varepsilon}+\log\log\frac{1}{\gamma}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$
Approximate Quantiles	$O\left(\log\frac{1}{\varepsilon}+\log\log\frac{1}{\gamma}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$	$\Omega\left(\log\frac{1}{\varepsilon}+\log\log\frac{1}{\gamma}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$
Approximate Mean	$O\left(\log\frac{1}{\varepsilon}+\log\frac{1}{\gamma+\beta}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$	$\Omega\left(\log\frac{1}{\varepsilon}+\log\log\frac{1}{\gamma}+\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$

3 Technical Overview

We consider a network of n nodes that are all connected to each other (one can think of it as a complete graph on n vertices). For all the problems that we study, each node is initially assigned a value that is polynomially bounded in n. Communication occurs in synchronous rounds, during each of which every node contacts some node chosen uniformly at random (including itself) and then either pushes or pulls a message – of size $O(\log n)$ bits – from the chosen node. As mentioned before, all our algorithms use the pull operation only, and the message sent out by a pulled node is the same, no matter which (and how many) nodes are pulling from it.

In every round, the messages sent out by at most βn of the nodes can become corrupted. More precisely, we assume that these corruptions can be adversarial in the following sense: At the beginning of each round, the adversary can choose at most βn nodes so that any message pulled from or pushed to these nodes can be entirely and arbitrarily modified by the adversary, that is, the message can be changed to whatever the adversary wants. Crucially, the adversary does not know beforehand the random choice of partners of the nodes in this round. We do not assume any additional restrictions on the adversary. For example, the adversary may have unbounded computational power and know the entire history of the algorithm at any given moment of time.

Before describing our new adversarially-robust algorithms, we briefly describe the state-of-the-art in the non-adversarial setting. The approximate median find algorithm given by [10] uses a pull-based tournament-style protocol. Here, every node repeatedly updates its value to the median of three randomly chosen node values. The effect is that after $O\big(\log\log n + \log\frac{1}{\varepsilon}\big)$ rounds, every node has communicated directly or indirectly with $\log n/\varepsilon^2$ other nodes. Notice that the problem becomes straightforward with large message sizes, as every node can collect all the information about every node it has directly or indirectly heard from, and $\log n/\varepsilon^2$ many uniformly randomly chosen node values are sufficient to prove the concentration of the median value for this node. As the authors show, we do not need to store all such information, and we can make do with just maintaining the median. Their algorithm then concludes with a majority vote over a constant number of nodes to allow for a union bound.

Our algorithms retain the simplicity of these previous algorithms with minor changes while being robust to adversarial corruptions. Some of the crucial changes include giving a more nuanced termination condition which takes into account β and γ, and deriving tailored bounds on expressions involving binomial coefficients to bound the effect of the adversaries on each round. We also introduce an extended second phase – now collecting values from $O\big(\frac{\log(1/\gamma)}{\log(1/\beta)}\big)$ many nodes – to ensure that at most a γ fraction of the nodes are storing an incorrect value upon termination, as desired.

More specifically, for the approximate median computation, as outlined above, we first let the nodes conduct a 3-tournament, that is, in the first phase, each node pulls three node values uniformly at random and updates its own value to the median of the three. In the second, additional, phase, the algorithm then uniformly samples a number of nodes and outputs their median. The pseudocode for our approximate median algorithm is given in Algorithm 1.

To compute approximate quantiles ϕ, [10] gives another tournament-style algorithm which reduces the problem to that of finding an approximate median. Our algorithm is based on their ingeniously simple algorithm. Its basic principle can be described in one sentence: In each round, each node pulls two node values uniformly at random and updates its value to be the minimum (if $\phi < 1/2$) or maximum (if $\phi > 1/2$) of the two. Our algorithm here requires a more refined tracking of its own progress and knowing when to terminate, taking into account the magnitude of the potential corruption, which we measure in terms of β. This also greatly affects the required analysis. The pseudocode for our quantile shifting algorithm is given in Algorithm 2 (for the case $\phi < 1/2$).

To compute the approximate mean, [14] gives the elegant push-sum algorithm, which works roughly as follows: In each round, every node pushes half of its value to another node which is chosen uniformly at random. After $O\big(\log n + \log\frac{1}{\varepsilon}\big)$ rounds, the algorithm converges to an approximate solution. The authors use a crucial property of this push-based protocol, mass preservation, to show that the sample variance of the values decreases exponentially, while the sample mean remains more or less unchanged.

As discussed before, the mass preservation property breaks down in the presence of adversarial corruption. Additionally, push-based protocols are not ideal as adversarially corrupted nodes gain a significant amount of influence by pushing faulty values to an arbitrary number of nodes (unless we restrict their power). Moreover, pull-based approaches generally work better than push-based ones in the presence of adversarial faults, as they inherently restrict the influence that adversarial corruptions can have. We adopt a novel approach and design a tournament-style algorithm, but use, however, a similar proof technique as in [14], despite the absence of mass preservation. The algorithm itself is very simple – each node repeatedly updates its value to the average of two node values chosen uniformly at

random. Finally, we also end with a majority vote, as in the approximate median algorithm, to ensure that at most a γ fraction of the nodes output a wrong answer. The pseudocode of our adversarially robust mean approximation algorithm can be found in Algorithm 3.

We highlight the trade-off between the tournament-style aggregation in the first phase and the sequential collection of values for the majority vote in the second phase. Though the former is more efficient, it also gives the adversary more (exponentially more in fact) influence over the values.

There are a variety of directions for follow-up work. In terms of other aggregation tasks, beyond computation of median, other quantiles, and mean, the main open problem which may robustly be solved in our setting is the computation of the *mode* (since there is no hope for computing the minimum or maximum value in the presence of adversaries). We also think it would be interesting to investigate the model in the asynchronous setting or relax the assumption that communications occur on an underlying complete graph.

4 Related Work

The Gossip Protocol is also known as the Epidemic Protocol, as these algorithms were first developed to mimic the spread of epidemics [4]. One of the first gossip algorithms studied was rumor spreading or randomized broadcast [12, 19], where in [12] the authors give an algorithm for rumor spreading in $O(\log n)$ rounds and $O(n \log\log n)$ messages.

In [14], the authors study a gossip protocol to compute aggregates such as sums and counts in $O(\log n)$ rounds. Furthermore, they also develop algorithms for random sampling and quantile computation (also known as randomized selection), the latter of which can be computed in $O(\log^2 n)$ rounds. The work in [10] improves that and gives an algorithm to compute exact quantiles in $O(\log n)$ rounds and approximate quantiles in $O\big(\log\log n + \log\frac{1}{\varepsilon}\big)$ rounds. The problem of computing quantiles has also been studied in the centralized setting [2] and the distributed setting [15]. Other problems such as computing the mode [8] (also known as plurality consensus), have also been studied in the gossip setting.

In both [1] and [12], the authors investigate a gossip-based rumor spreading in the presence of adaptive failures, but the adversary here only has the power to crash certain messages and cannot send altered messages (which is much more powerful). To the best of our knowledge, this is the first time a gossip model has been investigated for general aggregation problems in the presence of adversarial nodes. We note that for the special case of median computation, restricting the adversarial influence to $\beta \leq \frac{1}{n^{1/2}}$, the authors of [5] derive a runtime bound of $O(\log n \log\log n)$ for the computation of an approximate median from their approach to the stabilizing consensus problem. In this special case, at the cost of an additional factor of $O(\log\log n)$ compared to our runtime, they get an approximation factor of $O\big(\frac{(\log n)^{1/2}}{n^{1/2}}\big)$, thus yielding a tighter approximation than provided by our Theorem 1, which stipulates that $\varepsilon = \Omega(\frac{1}{n^{0.0019}})$. We note that we did not try to optimize the exponent in this lower bound on ε. We note that Byzantine-robust gossip algorithms have been widely studied in the context of decentralized machine learning [11], [6], [7].

In the deterministic setting, the approximate median computation in the presence of byzantine nodes has been studied [21]. The algorithm presented there was shown to produce an approximation factor of at most $\frac{\beta}{2}$ and runs in time $O(\beta n)$, as long as $\beta < \frac{1}{3}$. The latter requirement on the fraction of adversarial nodes was proven to be tight. This was later generalized to approximate the k -th largest values in the presence of Byzantine nodes [17].

5 Approximate Median

In this section, we sketch the ideas behind our median approximation algorithm and the proof of its correctness and runtime guarantees stated in Theorem 1.

▶ **Theorem 1.** *For any $\varepsilon(n), \beta(n), \gamma(n) \in (0,1)$ satisfying $\beta \leq \frac{\varepsilon}{14}$ and $\varepsilon = \Omega\left(\frac{1}{n^{0.0019}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that solves the ε-approximate median problem in the presence of a β-strong adversary with correctness level γ with high probability in $O\left(\log\frac{1}{\varepsilon} + \log\log\frac{1}{\gamma} + \frac{\log(1/\gamma)}{\log(1/\beta)}\right)$ rounds.*

The algorithm, which draws on ideas from [10], is described in Algorithm 1, and proceeds in two phases. In the first phase, every iteration consists of 3 rounds. More concretely, in every iteration, each node pulls the value of three random nodes and updates its value to the median of these three values. In the second phase, every node obtains K many independent samples and outputs the median of these K values.

Algorithm 1 3-TOURNAMENT(v).

1 $\delta \leftarrow \left(\frac{30\log n}{n}\right)^{1/3}$
2 $\gamma' \leftarrow \max(\delta, \beta, \min(\frac{\gamma}{4}, \frac{\varepsilon}{14}))$
3 $t \leftarrow \lceil \log_{(157/156)}(\frac{1}{3(\varepsilon-\beta)})\rceil + \lceil \log_2(\log_{9/8}(\frac{1}{\gamma'}))\rceil + 2$
`// Phase 1`
4 **for** $i = 1$ *to* t **do**
5 Select 3 nodes $u_1(v), u_2(v), u_3(v)$ uniformly at random
6 $x_v \leftarrow \text{median}(x_{u_1(v)}, x_{u_2(v)}, x_{u_3(v)})$
7 **end**
`// Phase 2`
8 **if** $\gamma' > \frac{\gamma}{4}$ **then**
9 Sample $K = \lceil 8 \cdot \log_{20\gamma'}\left(\frac{\gamma}{4}\right)\rceil$ nodes uniformly at random and set x_v to the median value of these K nodes.
10 **end**
11 Output x_v

We consider the sets of nodes whose quantiles lie in $\left[0, \frac{1}{2}-\varepsilon\right)$, $\left[\frac{1}{2}-\varepsilon, \frac{1}{2}+\varepsilon\right]$ and $\left(\frac{1}{2}+\varepsilon, 1\right]$ at the end of iteration i in line 7 of Algorithm 1, and call these sets L_i, M_i, H_i respectively. Additionally, we also define l_i to be such that $l_0 := \frac{1}{2} - \varepsilon$ and $l_{i+1} := \overline{B}_{3,1}(l_i + \beta) = 3(l_i+\beta)^2 - 2(l_i+\beta)^3$.[1] These values capture the expected values of $\frac{|L_i|}{n}$ as we will show later. We remark that H_i and L_i behave symmetrically, hence without loss of generality, we focus on bounding $|L_i|$ in the following proof. In the end, we use this symmetry to infer matching bounds for $|H_i|$.

Proof Sketch. We first show that, at the end of the first phase, that is, after t rounds (line 8 of the pseudocode of Algorithm 1), l_t is small enough (smaller than γ' as defined in line 2 of Algorithm 1). The proof is a straightforward induction based on the definition of l_i, but we utilize tailored bounds on the expressions (involving binomial coefficients) that arise from the recursive definition of l_i.

[1] $\overline{B}_{3,1}(l_i+\beta)$ is defined as $1 - B_{3,1}(l_i+\beta)$ with $B_{n,k}(p)$ the Cumulative Distribution Function of the binomial distribution with parameters n, k, p. For details, see the Appendix.

Next, we show that $\mathbb{E}\left[\frac{|L_{i+1}|}{n} \mid L_i\right] \leq \overline{B}_{3,1}\left(\frac{|L_i|}{n} + \beta\right)$. This follows from an analysis of the main loop of the algorithm. Then, using Chernoff's bounds, we show that $\frac{|L_{i+1}|}{n}$ is concentrated around its expectation.

Notice that l_i is defined recursively in such a way as to mirror the evolution of $\frac{|L_i|}{n}$ in a single step. In fact, we make this connection more explicit by showing that for all $0 \leq i \leq t$, $|L_i|/n$ is approximately equal to l_i in expectation and with high probability. Thus, we use the bound on l_t to conclude that after the first phase, at most a γ' fraction of the nodes have a wrong answer.

Finally, the second phase amplifies the number of correct nodes in each step and precisely runs the number of steps needed to reduce this fraction of bad nodes from γ' to γ. ◀

6 Shifting Quantiles

In this section, we give some intuition underlying our quantile shifting algorithm and the proof of its correctness and runtime guarantees given in Theorem 2.

▶ **Theorem 2.** *For any $\varepsilon(n), \beta(n) \in (0,1)$ satisfying $\varepsilon \leq \frac{1}{6}$, $\beta \leq \frac{\varepsilon^{2.5}}{16}$, and $\varepsilon = \Omega\left(\frac{(\log n)^{1/5}}{n^{1/5}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that reduces any ε-approximate quantile problem into an ε-approximate median problem in the presence of a β-strong adversary with high probability in $O\left(\log \frac{1}{\varepsilon}\right)$ rounds.*

Note that we can assume $\phi \leq 1/2$ without loss of generality, as the algorithm and analysis are exactly symmetric for the other case.

The pseudocode for our algorithm is described in Algorithm 2. We consider the sets of nodes whose quantiles lie in $[0, \phi-\varepsilon), [\phi-\varepsilon, \phi+\varepsilon]$ and $(\phi+\varepsilon, 1]$ at the end of iteration i of the while loop and call these sets L_i, M_i, H_i respectively. We want to show that $|L_i|$ and $|H_i|$ decrease rapidly as the algorithm progresses.

Algorithm 2 2-TOURNAMENT(v).

```
h'_0 ← 1 − (φ + ε)
i ← 0
T ← 1/2 − 21ε/16
while h'_i > T do
    h'_{i+1} ← (h'_i − β)^2
    δ ← min(1, (h'_i − T)/(h'_i − h'_{i+1}))
    With probability δ do
        Select 2 nodes u_1(v), u_2(v) uniformly at random
        x_v ← min(x_{u_1(v)}, x_{u_2(v)})
    Otherwise do
        Select a node u_1(v) uniformly at random
        x_v ← x_{u_1(v)}
    i ← i + 1
end
```

The values h'_i in Algorithm 2 determine when the algorithm terminates, and will also be helpful for the analysis. We define another quantity recursively as

$$h_0 \coloneqq h'_0, \qquad h_{i+1} \coloneqq (h_i + \beta)^2,$$

which we will use for the analysis as well. As will be shown later, the values h_i, h'_i capture the expected values of $\frac{|H_i|}{n}$.

Proof Sketch. Let t be the total number of iterations of the while-loop executed by Algorithm 2. We first show that t is bounded by $\mathcal{O}\left(\log \frac{1}{\varepsilon}\right)$ as required. Notice that h_i, h'_i are defined so that they start out the same ($h_0 = h'_0$) and drift apart very slowly. We make this explicit by showing that after t steps, they are within about a $(1+\varepsilon)$ factor of each other. Next, we show that $\left(\frac{|H_i|}{n} - \beta\right)^2 \le \mathbb{E}\left[\frac{|H_{i+1}|}{n} \mid H_i\right] \le \left(\frac{|H_i|}{n} + \beta\right)^2$. This follows from an analysis of the main loop of the algorithm.

As in the median find algorithm, notice that the recursive definition of h'_i, h_i mirrors the evolution of $\frac{|H_i|}{n}$. In fact, we make this explicit by giving an upper and lower bound of $\frac{|H_i|}{n}$ in terms of h_i, h'_i, respectively. This is the most technically involved part of the proof, utilizing tedious but straightforward bounds that we tailored for this analysis. Consequently, from the termination condition $h'_t \le T \coloneqq \frac{1}{2} - \frac{21\varepsilon}{16}$, we deduce that $\frac{|H_t|}{n}$ is around $\frac{1}{2} - \varepsilon$.

The final part of the proof involves showing that $|M_i|$ does not change significantly during the entire algorithm and stays around its initial value of 2ε. This follows from an analysis of the expected value of $|M_{i+1}|$ given M_i and then using Chernoff's bounds. This would imply that $\frac{|L_t|}{n}$ is also around $\frac{1}{2} - \varepsilon$, which concludes our proof. ◀

7 Approximate Mean

In this section, we give a proof sketch for the runtime and correctness guarantees of our adversarially robust approximate mean algorithm stated in Theorem 3.

▶ **Theorem 3.** *For any $\varepsilon(n), \beta(n), \gamma(n) \in (0,1)$ satisfying $\beta \le \left(\frac{\varepsilon}{100}\right)^{2.5}$ and $\varepsilon = \Omega\left(\frac{(\log n)^{6/5}}{n^{1/5}}\right)$, there exists a gossip algorithm using messages of size $O(\log n)$ that solves the ε-approximate mean problem with M-bounded values in the presence of a β-strong adversary with correctness level γ with high probability in $O\left(\log \frac{1}{\varepsilon} + \log \frac{1}{\gamma+\beta} + \frac{\log(1/\gamma)}{\log(1/\beta)}\right)$ rounds.*

The pseudocode is described in Algorithm 3. In the first phase of the algorithm, in each round, every node u samples two nodes and updates its value to their arithmetic mean. Samples of node value that exceed the acceptable value interval $[0, M]$ are cut to the interval limits. In the second phase, the algorithm globally samples a set of nodes uniformly at random and outputs the median value of these nodes. Let $x_v(t)$ denote the value x_v stored in the node v at the end of iteration t. Additionally, let us denote by V the set of all nodes.

To help analyze the progress of the algorithm, we define the potential function.

$$\Phi(t) \coloneqq \sum_{u \ne v \in V} (x_u(t) - x_v(t))^2,$$

where the sum covers all subsets of V of size 2 (that is, we do not double-count the pairs). Notice that this potential captures the variation in the values stored in the nodes. As the algorithm progresses, we expect the potential to decrease to a small value as the values stored in the nodes converge to the average. We also define the quantity $\psi(t) \coloneqq \sum_{u \in V} x_u(t)$ as the sum of the node values. Notice that $\frac{\psi(0)}{n}$ is the true average that we want to approximate.

Algorithm 3 Pull-Avg(v).

$\delta \leftarrow \frac{(\log n)^{1/2}}{n^{1/2}}$
$\eta \leftarrow \max(\beta, \delta, \min(\gamma^5, \left(\frac{\varepsilon}{100}\right)^{2.5}))$
$T \leftarrow \lceil \log_{9/5} \frac{1}{\eta} \rceil$
// Phase 1
for $t = 1$ *to* T **do**
  Select 2 nodes $u_1(v), u_2(v)$ uniformly at random
  If $x_{u_1(v)}, x_{u_1(v)} \notin [0, M]$, set them to the closest value in $[0, M]$
  $x_v \leftarrow \frac{x_{u_1(v)} + x_{u_2(v)}}{2}$
end
// Phase 2
if $\eta > \min(\gamma^5, \left(\frac{\varepsilon}{100}\right)^{2.5}))$ **then**
  Sample $K = \max\left(100, \lceil 40 \log_{32\beta}\left(\frac{\gamma}{2}\right)\rceil\right)$ nodes uniformly at random and set x_v to the median value of these K nodes.
end
Output x_v

Proof Sketch. Intuitively, the 2-tournament algorithm must decrease the variation of the node values in each step as each node updates its value to the average of two other node values. Through a more precise analysis, we show that $\Phi(t)$ approximately halves in expectation in each step. Thus, by using Azuma's inequality, we can obtain concentration on this expectation and show that at the end of the T steps it has been reduced to less than about a β fraction of its original value.

On the other hand, in each step, the influence of the adversary is limited to a β fraction of the nodes. This suffices to limit the change in $\psi(t)$ in a single step to about βnM in expectation. Using Azuma's inequality again, we can bound the total change in $\psi(t)$ after T steps by about εnM.

If the average of the values does not change much but the variance decreases substantially, we can conclude that most of the values must be close to the average. This is exactly what we make next explicit, showing that the number of nodes that can have a wrong answer at the end of the first phase is bounded by a polynomial in β.

Finally, as in the median find algorithm, the second phase amplifies the number of correct nodes in each step and precisely runs the number of steps needed to reduce this fraction of bad nodes to γ. ◀

8 Lower Bounds on Round Complexity

In this section, we prove Theorem 4, which yields a general tool for calculating the lower bounds for the round complexity of gossip algorithms in the presence of adversaries. In essence, the theorem quantifies a lower bound on the number of rounds needed for all but a γ fraction of the nodes to produce a correct answer, in the presence of βn adversaries. Note that a similar lower bound was proven for the non-adversarial setting in [10], but its proof does not extend readily to our case because our parametrized notion of correctness allows a γ fraction of nodes to be incorrect in addition to the βn adversarial nodes. However, what remains true is that even with the slack provided by γ, a correct algorithm cannot terminate in fewer than $\Omega\left(\frac{\log(1/\gamma)}{\log(1/\beta)}\right)$ rounds, since up to this point, more than a γ fraction of nodes will

have communicated only with bad nodes. Therefore, they will have received no accurate information whatsoever and could not possibly output the correct result. This is formalized in Theorem 4.

▶ **Theorem 4.** *Let $\beta(n), \gamma(n) \in (0,1)$. Then, in the presence of a β-strong adversary, for any gossip algorithm that runs for less than $\frac{\log(1/2\gamma)}{\log(1/\beta)}$ rounds, with probability at least $1/10$ more than a γ fraction of the nodes have only received corrupted messages (if any).*

Proof. First, suppose that $2\gamma > \beta$. Then, notice that $\frac{\log(1/2\gamma)}{\log(1/\beta)} < 1$ and hence the claim holds trivially. Thus, we can now assume that $2\gamma \leq \beta$.

Consider a single good node v. In one round, the probability of communicating with an adversarially affected node is β. Thus, after $t < \frac{\log(1/2\gamma)}{\log(1/\beta)}$ rounds, the probability that it has communicated only with adversarial nodes is at least $(\beta)^t > 2\gamma$. Thus, in expectation after t rounds, there is at least a 2γ fraction of such nodes.

Now, notice that the communication partner chosen by an arbitrary node in an arbitrary round is completely independent of what is chosen by any other node in any round. Thus, by the Chernoff bounds, the probability that there is at least a γ fraction of the nodes that have only communicated with bad nodes during the first $t < \frac{\log(1/2\gamma)}{\log(1/\beta)}$ rounds is at least

$$1 - \exp\left(-\frac{2\gamma n}{2^2 \cdot 2}\right) \geq 1 - e^{-1/8} > \frac{1}{10},$$

where we assumed without loss of generality that $\gamma \geq \frac{1}{2n}$. ◀

Notice that we immediately get a $\Omega\big(\frac{\log(1/\gamma)}{\log(1/\beta)}\big)$ lower bound for the tasks that we study in this paper. This is because any node that has only communicated with adversarial nodes cannot reliably output a correct answer.

Similarly, one can adapt the proof of [10, Theorem 1.3] to show a $\Omega\left(\log\log\frac{1}{\gamma} + \log\frac{1}{\varepsilon}\right)$ lower bound for these problems. We formalize this statement in Proposition 5.

▶ **Proposition 5.** *For any $\varepsilon \in (\frac{10\log n}{n}, \frac{1}{6})$ and any $\gamma \in (\frac{40e\log n}{n}, \frac{1}{2})$, any gossip algorithm (even with unlimited message size) that uses less than $\log_2\log_{4e}\frac{1}{\gamma} + \log_4\frac{1}{6\varepsilon}$ rounds fails to solve either of the ε-approximate median or (bounded) mean problem with probability at least $1/3$.*

Together, Theorem 4 and Proposition 5 show that our algorithm for the approximate median is optimal. Moreover, our algorithm for the approximate mean is also almost optimal with only an extra $O(\log\frac{1}{\gamma+\beta})$ factor and is optimal whenever ε is polynomially smaller than γ or β.

References

1 Dan Alistarh, Seth Gilbert, Rachid Guerraoui, and Morteza Zadimoghaddam. How efficient can gossip be? (on the cost of resilient information exchange). In Samson Abramsky, Cyril Gavoille, Claude Kirchner, Friedhelm Meyer auf der Heide, and Paul G. Spirakis, editors, *Automata, Languages and Programming*, pages 115–126, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. doi:10.1007/978-3-642-14162-1_10.

2 Manuel Blum, Robert W. Floyd, Vaughan Pratt, Ronald L. Rivest, and Robert E. Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973. doi:10.1016/S0022-0000(73)80033-9.

3 Jen-Yeu Chen and Gopal Pandurangan. Almost-optimal gossip-based aggregate computation. *SIAM Journal on Computing*, 41(3):455–483, 2012. doi:10.1137/100793104.

4 Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the Sixth Annual ACM Symposium on Principles of Distributed Computing*, PODC '87, pages 1–12, New York, NY, USA, 1987. Association for Computing Machinery. doi:10.1145/41840.41841.

5 Benjamin Doerr, Leslie Ann Goldberg, Lorenz Minder, Thomas Sauerwald, and Christian Scheideler. Stabilizing consensus with the power of two choices. In *Proceedings of the Twenty-Third Annual ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '11, pages 149–158, New York, NY, USA, 2011. Association for Computing Machinery. doi:10.1145/1989493.1989516.

6 Renaud Gaucher, Aymeric Dieuleveut, and Hadrien Hendrikx. Byzantine-robust gossip: Insights from a dual approach, 2025. doi:10.48550/arXiv.2405.03449.

7 Renaud Gaucher, Aymeric Dieuleveut, and Hadrien Hendrikx. Unified breakdown analysis for byzantine robust gossip, 2025. arXiv:2410.10418.

8 Mohsen Ghaffari and Merav Parter. A polylogarithmic gossip algorithm for plurality consensus. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 117–126, New York, NY, USA, 2016. Association for Computing Machinery. doi:10.1145/2933057.2933097.

9 Michael B. Greenwald and Sanjeev Khanna. Power-conserving computation of order-statistics over sensor networks. In *Proceedings of the Twenty-Third ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '04, pages 275–285, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1055558.1055597.

10 Bernhard Haeupler, Jeet Mohapatra, and Hsin-Hao Su. Optimal gossip algorithms for exact and approximate quantile computations. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pages 179–188, 2018. doi:10.1145/3212734.3212770.

11 Lie He, Sai Praneeth Karimireddy, and Martin Jaggi. Byzantine-robust decentralized learning via clippedgossip, 2023. arXiv:2202.01545.

12 R. Karp, C. Schindelhauer, S. Shenker, and B. Vocking. Randomized rumor spreading. In *Proceedings 41st Annual Symposium on Foundations of Computer Science*, pages 565–574, 2000. doi:10.1109/SFCS.2000.892324.

13 Srinivas Kashyap, Supratim Deb, K. V. M. Naidu, Rajeev Rastogi, and Anand Srinivasan. Efficient gossip-based aggregate computation. In *Proceedings of the Twenty-Fifth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '06, pages 308–317, New York, NY, USA, 2006. Association for Computing Machinery. doi:10.1145/1142351.1142395.

14 David Kempe, Alin Dobra, and Johannes Gehrke. Gossip-based computation of aggregate information. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, pages 482–491. IEEE, 2003. doi:10.1109/SFCS.2003.1238221.

15 Fabian Kuhn, Thomas Locher, and Rogert Wattenhofer. Tight bounds for distributed selection. In *Proceedings of the Nineteenth Annual ACM Symposium on Parallel Algorithms and Architectures*, SPAA '07, pages 145–153, New York, NY, USA, 2007. Association for Computing Machinery. doi:10.1145/1248377.1248401.

16 Samuel Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146, 2003. doi:10.1145/844128.844142.

17 Darya Melnyk and Roger Wattenhofer. Byzantine agreement with interval validity. In *2018 IEEE 37th Symposium on Reliable Distributed Systems (SRDS)*, pages 251–260, 2018. doi:10.1109/SRDS.2018.00036.

18 Boaz Patt-Shamir. A note on efficient aggregate queries in sensor networks. In *Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing*, PODC '04, pages 283–289, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1011767.1011809.

19 Boris Pittel. On spreading a rumor. *SIAM Journal on Applied Mathematics*, 47(1):213–223, 1987. URL: http://www.jstor.org/stable/2101696.

20 Nisheeth Shrivastava, Chiranjeeb Buragohain, Divyakant Agrawal, and Subhash Suri. Medians and beyond: new aggregation techniques for sensor networks. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, SenSys '04, pages 239–249, New York, NY, USA, 2004. Association for Computing Machinery. doi:10.1145/1031495.1031524.

21 David Stolz and Roger Wattenhofer. Byzantine agreement with median validity. In Emmanuelle Anceaume, Christian Cachin, and Maria Gradinariu Potop-Butucaru, editors, *19th International Conference on Principles of Distributed Systems, OPODIS 2015, December 14-17, 2015, Rennes, France*, volume 46 of *LIPIcs*, pages 22:1–22:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2015. doi:10.4230/LIPIcs.OPODIS.2015.22.