Byzantine Agreement in Polynomial Time with Near-Optimal Resilience*

Shang-En Huang
sehuang@umich.edu
University of Michigan
Ann Arbor, MI, USA

Seth Pettie
pettie@umich.edu
University of Michigan
Ann Arbor, MI, USA

Leqi Zhu
lezhu@umich.edu
University of Michigan
Ann Arbor, MI, USA

ABSTRACT

It has been known since the early 1980s that Byzantine Agreement in the full information, asynchronous model is impossible to solve deterministically against even one crash fault [FLP 1985], but that it *can* be solved with probability 1 [Ben-Or 1983], even against an adversary that controls the scheduling of all messages and *corrupts* up to $f < n/3$ players [Bracha 1987]. The main downside of [Ben-Or 1983, Bracha 1987] is that they terminate with $2^{\Theta(n)}$ latency in expectation whenever $f = \Theta(n)$.

King and Saia [KS 2016, KS 2018] developed a polynomial protocol (polynomial latency, polynomial local computation) that is resilient to $f < (1.14 \times 10^{-9})n$ Byzantine faults. The new idea in their protocol is to detect—and *blacklist*—coalitions of likely-bad players by analyzing the deviations of random variables generated by those players over many rounds.

In this work we design a simple collective coin-flipping protocol such that if any coalition of faulty players repeatedly does *not* follow protocol, then they will eventually be detected by one of two simple statistical tests. Using this coin-flipping protocol, we solve Byzantine Agreement in polynomial latency, even in the presence of up to $f < n/4$ Byzantine faults. This comes close to the $f < n/3$ upper bound on the maximum number of faults [LSP 1982, BT 1985, FLM 1986].

CCS CONCEPTS

• **Theory of computation** → **Distributed algorithms**; **Probabilistic computation**.

KEYWORDS

asynchronous message passing, Byzantine agreement, randomized algorithms, coin-flipping games

ACM Reference Format:
Shang-En Huang, Seth Pettie, and Leqi Zhu. 2022. Byzantine Agreement in Polynomial Time with Near-Optimal Resilience. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC '22), June 20–24, 2022, Rome, Italy*. ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3519935.3520015

*This work was supported by NSF grant CCF-1815316.

STOC '22, June 20–24, 2022, Rome, Italy

ACM ISBN 978-1-4503-9264-8/22/06…$15.00
https://doi.org/10.1145/3519935.3520015

1 INTRODUCTION

The field of *forensic accounting* is concerned with the detection of *fraud* in financial transactions, or more generally, finding evidence of fraud, malfeasance, or fabrication in data sets. Some examples include detecting faked digital images [13], suspicious reports of election data [41] and political fundraising [23], fraudulent COVID numbers,[1] and manipulated economic data [1, 30, 46] via Newcomb-Benford's law [31], detecting fabricated data sets[2] in social science research [44, 45], or detecting match-fixing in sumo wrestling [20].

Theoretical computer science has a strong tradition of embracing a fundamentally *adversarial* view of the universe that borders on being outright paranoid. Therefore it is somewhat surprising that TCS is, as a whole, credulous when it comes to adversarial manipulation of data and transactions. In other words, fraud detection does not play a significant part in most algorithm design, *even in multi-party models that explicitly posit the existence of malicious parties.*

To our knowledge, the only work in TCS that has explicitly adopted a forensic accounting mindset is King and Saia's [34, 35] breakthrough in Byzantine Agreement in the most challenging model: the full-information (no crypto) asynchronous model against an adaptive adversary. In this problem there are n players, each with initial input bits in $\{-1, 1\}$, up to f of which may fail (i.e., be *adaptively corrupted* by the adversary) and behave arbitrarily. They must each **decide** on a bit in $\{-1, 1\}$ subject to:

Agreement: All non-corrupt players **decide** the same value v.
Validity: If v is the value **decided** by non-corrupt players, v was initially held by some non-corrupt player.

See Section 2.1 for details of the model. Prior to King and Saia's work [34, 35], it was known from Bracha [14] (see also Ben-Or [9]) that the problem could be solved with probability 1 in $2^{\Theta(n)}$ time in expectation even if $f < n/3$ players fail, that $f < n/3$ cannot be improved [15, 21, 36], and by Fischer, Lynch, and Patterson's impossibility result [22], that no *deterministic* protocol exists even against a single crash failure.

King and Saia [34] reduce the problem to a certain coin-flipping game, in which all players—good and adversarial—attempt to generate a (global) unbiased coin flip and agree on its outcome. Coin flipping games have been studied extensively under adversarial manipulation (see Section 1.2), but the emphasis is always on bounding the power of the adversarial players to *bias* the coin flip in their desired direction. King and Saia recognized that the primary *long term* advantage of the adversary is *anonymity*. In other words, it

[1] https://theprint.in/opinion/benfords-law-detects-data-fudging-so-we-ran-it-through-indian-states-covid-numbers/673085/
[2] http://datacolada.org/98

Table 1: Asynchronous Byzantine Agreement in the full information model against an adaptive adversary.

Citation		Byzantine Faults (f)	Expected Latency / Computation Per Round
Fischer, Lynch, Patterson	1983	$f \geq 1$	impossible deterministically
[15, 21, 36]	1982	$f \geq n/3$	impossible, even with randomization
Ben-Or	1983	$f < n/5$	$\exp(n)$ / $\text{poly}(n)$
		$f < O(\sqrt{n})$	$O(1)$ / $\text{poly}(n)$
Bracha	1984	$f < n/3$	$\exp(n)$ / $\text{poly}(n)$
King & Saia	2016	$f < n/400$	$\text{poly}(n)$ / $\exp(n)$
		$f < n/(1.14^{-1} \times 10^9)$	$\text{poly}(n)$ / $\text{poly}(n)$
new	2021	$f < n/4$	$\text{poly}(n)$ / $\text{poly}(n)$

can bias the outcome of coin flips at will, in the short term, but its advantage simply evaporates if good players can merely *identify* who the adversarial players are, by detecting likely fraud via a statistical analysis of their transactions. Good players can *blacklist* (ignore) the adversarial players, removing their influence over the game. If a sufficient number of fraudulent players are blacklisted, collective coin-flipping by a set of good players becomes easy.

The journal version of King and Saia's work [34] presents two methods for blacklisting players, which leads to different fault tolerance levels. The first protocol has a polynomial round complexity and requires a polynomial amount of local computation; it is claimed to be resilient to $f < (4.25 \times 10^{-7})n$ Byzantine faults. The second protocol is tolerant to $f < n/400$ Byzantine faults, but requires exponential local computation. In response to some issues raised by Melynyk, Wang, and Wattenhofer (see Melynyk's Ph.D. thesis [38, Ch. 6]), King and Saia [35] released a corrigendum, reducing the tolerance of the first protocol to $f < (1.14 \times 10^{-9})n$.

1.1 New Results

In this paper we solve Byzantine Agreement in the full-information, asynchronous model against an adaptive adversary, by adopting the same forensic accounting paradigm of King and Saia [34]. We design a coin-flipping protocol and two simple statistical tests such that if the Byzantine players continually foil attempts to flip a fair coin, they will be detected in a polynomial number of rounds by at least one of the tests, so long as $f < n/4$. (The tests measure individual deviation in l_2 norm and pair-wise correlation.) Our analysis is tight inasmuch as these two particular tests may not detect anything when $f \geq n/4$.

One factor contributing to the low resiliency of King and Saia's protocols [34, 35] is that two good players may blacklist different sets of players, making it easier for the adversary to induce disagreements on the outcome of the shared coin flip. A technical innovation in our protocol is a method to drastically reduce the level of disagreement between the views of good players. First, we use a *fractional* blacklisting scheme. Second, to ensure better consistency across good players, we extend King and Saia's [34] *Blackboard* to an *Iterated Blackboard* primitive that drastically reduces good players' disagreements of the historical transaction record by allowing retroactive corrections to the record.

1.2 Related Work

The approach of King and Saia [34] was foreshadowed several years earlier by Lewko [37], who showed that protocols broadly similar to Ben-Or and Bracha must have exponential latency. The key assumption is that messages are taken at face value, without taking into account the *identity* of the sender, nor the *history* of the sender's messages.

Byzantine agreement has been studied in synchronous and asynchronous models, against computationally bounded or unbounded adversaries, and with adaptive or non-adaptive adversaries. (In particular, a special case of the problem that restricts attention to crash failures, called *consensus*, has been very extensively studied.) We refer the reader to [5–7, 11, 18, 33] for some key results and surveys of the literature. A result that is fairly close to ours is that of Kapron et al. [29]. They proved that against a *non-adaptive* adversary (all corruptions made in advance) Byzantine agreement can be solved asynchronously, against $f < n/(3+\epsilon)$ faults.

Collective coin flipping has an illustrious history in computer science, as it is a key concept in cryptography, distributed computing, and analysis of boolean functions. The problem was apparently first raised by Blum [12], who asked how two mutually untrusted parties could flip a shared coin over the telephone. His solution used cryptography. See [8, 16, 17, 19, 25, 26, 39] for some recent work on coin flipping using cryptography.

Ben-Or and Linial [10] initiated a study of *full information* protocols for coin-flipping. The players broadcast messages one-by-one in a specific order, and the final coin flip is a function of these messages. The goal is to minimze the *influence* of a coalition of k bad players, which is, roughly speaking, the amount by which they can bias the outcome towards *heads* or *tails*. Ben-Or and Linial's [10] protocol limits $k < n^{\log_3 2}$ bad players to influence $O(k/n)$. Saks [43] and Ajtai and Linial [2] improved it to $O(k/n)$ influence with up to $k = O(n/\log n)$ players, and Alon and Naor [3] achieved optimum $O(k/n)$ influence for k even linear in n. The message size in these protocols is typically more than a single bit. If only single-bit messages are allowed and each player speaks once, the problem is equivalent to bounding the influence of variables in a boolean function [28]. Russel, Saks, and Zuckerman [42] considered parallel coin-flipping protocols. The proved that any protocol that uses 1-bit messages and is resilient to linear-size coalitions must use $\Omega(\log^* n)$ rounds.

Aspnes [4] considered a sequential coin-flipping game where n coins are flipped sequentially and the outcomes broadcast, but up to t of these may be *suppressed* by the adversary. Regardless of which function is used to map the coin-flip sequence to a shared coin, the adversary can bias it whenever $t = \Omega(\sqrt{n})$. Very recently Haitner and Karidi-Heller [24] resolved the complexity of Ben-Or-Linial-type sequential coin flipping games against an *adaptive* adversary, that can corrupt players at will, as information is revealed. They proved that *any* such shared coin can be fixed to a desired outcome with probability $1 - o(1)$ by adaptively corrupting $\tilde{O}(\sqrt{n})$ parties.

1.3 Organization

In Section 2 we review the model, the reliable broadcast primitive, and Bracha's Byzantine agreement protocol, and introduce the *Iterated Blackboard* primitive, which generalizes [32, 34].

In Section 3 we begin with a simplified iterated coin-flipping game and then proceed to study a more complicated iterated coin-flipping game that can be implemented in the asynchronous distributed model and used within Bracha's algorithm.

The full paper [27] contains proofs from Section 2 on reliable broadcast and the iterated blackboard. It also contains some proofs showing that a certain fractional matching algorithm has a Lipschitz property.

2 PRELIMINARIES

2.1 The Model

There are n *processes*, $p_1, \ldots, p_n$, and $2n^2$ *message buffers*, $\text{In}_{j\to i}$ and $\text{Out}_{i\to j}$ for all $i, j \in [n]$. All processes are initially *good* (they obey the protocol) and the adversary may dynamically *corrupt* up to f processes. A *bad*/*corrupted* process is under complete control of the adversary and may behave arbitrarily. The adversary controls the pace at which progress is made by scheduling two types of *events*.

- A compute(i) event lets p_i process all messages in the buffers $\text{In}_{j\to i}$, deposit new messages in $\text{Out}_{i\to j}$, and change state.
- A deliver(i, j) event removes a message from $\text{Out}_{i\to j}$ and inserts it into $\text{In}_{i\to j}$.

Note that the adversary may choose a malicious order of events, but cannot, for example, misdeliver or forge messages. The adversary must eventually deliver every message, and schedule compute(i) events infinitely often.

The adversary is computationally unbounded and is aware, at all times, of the internal state of all processes. Thus, cryptography is not helpful, but randomness potentially is, since the adversary cannot predict the outcome of future coin flips.

In this model, the *communication time* or *latency* is defined w.r.t. a hypothetical execution in which all local computation occurs instantaneously and all messages have latency in $[0, \Delta]$. The latency of the algorithm is L if all non-corrupt processes finish by time $L\Delta$. Note that in this hypothetical, Δ is unknown and cannot influence the execution of the algorithm.

2.1.1 Reliable Broadcast. The goal of Reliable-Broadcast is to simulate a broadcast channel using the underlying point-to-point message passing system. In Byzantine Agreement protocols, each process initiates a series of Reliable-Broadcasts. Call $m_{p,\ell}$ the ℓth message broadcast by process p.

Theorem 1. *If a good process p initiates the Reliable-Broadcast of $m_{p,\ell}$, then all good processes q eventually* accept *$m_{p,\ell}$. Now suppose a bad process p does so and some good q* accepts *$m_{p,\ell}$. Then all other good q' will eventually accept $m_{p,\ell}$, and no good q' will accept any other $m'_{p,\ell} \neq m_{p,\ell}$. Moreover, all good processes accept $m_{p,\ell-1}$ before $m_{p,\ell}$, if $\ell > 1$.*

The property that $m_{p,\ell}$ is only accepted after $m_{p,\ell-1}$ is accepted is sometimes called FIFO broadcast. This property is explicitly used in the Iterated-Blackboard algorithm outlined in Section 2.2. See [27, Appendix] for a proof of Theorem 1.

Algorithm 1 Reliable-Broadcast(p, ℓ)

1: **if** $\ell > 1$ **then wait** until $m_{p,\ell-1}$ has been accepted.
2: **if** I am process p **then** generate $m_{p,\ell}$ and send $(\text{init}, m_{p,\ell})$ to all processes.
3: **wait** until receipt of one $(\text{init}, m_{p,\ell})$ message from p, or more than $(n+f)/2$ $(\text{echo}, m_{p,\ell})$ messages, or $f+1$ $(\text{ready}, m_{p,\ell})$ messages.
send $(\text{echo}, m_{p,\ell})$ to all processes.
4: **wait** until the receipt of $> (n+f)/2$ $(\text{echo}, m_{p,\ell})$ messages or $f+1$ $(\text{ready}, m_{p,\ell})$ messages.
send $(\text{ready}, m_{p,\ell})$ to all processes.
5: **wait** until receipt of $2f+1$ $(\text{ready}, m_{p,\ell})$ messages.
accept $m_{p,\ell}$.

2.1.2 Validation and Bracha's Protocol. Consider a protocol Π of the following form. In each round r , each process reliably broadcasts its *state* to all processes, waits until it has accepted at least $n - f$ *validated* messages from round r, then processes all validated messages, changes its state, and advances to round $r + 1$. A good process *validates* a round-r state (message) $s_{q,r}$ accepted from another process q only if (i) it has validated the state $s_{q,r-1}$ of q at round $r - 1$, and (ii) it has accepted $n - f$ messages that, if they *were* received by a correct q, would cause it to transition from $s_{q,r-1}$ to $s_{q,r}$. The key property of validation (introduced by [14]) is:

Lemma 2. *A good process p validates the message of another process q in an admissible execution α of Π if and only if there is an execution β of Π in which q is a good process and the state of every other good process (including p) is the same in α and β (with respect to their validated messages).*

To recap, reliable broadcast prevents the adversary from sending conflicting messages to different parties (i.e., it is forced to participate as if the communication medium were a broadcast channel) and the validation mechanism forces its internal state transitions to be consistent with the protocol. Its remaining power is limited to (i) substituting deterministic outcomes for coin flips in bad processes, (ii) dynamic corruption of good processes, and (iii) malicious scheduling.

Bracha's protocol improves the resilience of Ben-Or's protocol to the optimum $f < n/3$. Each process p initially holds a value $v_p \in \{-1, 1\}$. It repeats the same steps until it **decides** a value $v \in \{-1, 1\}$ (Line 8). As we will see, if some process **decides** v, all good processes will **decide** v in this or the following iteration. Thus, good processes continue to participate in the protocol until

all other good processes have executed Line 8. Here $\mathrm{sgn}(x) = 1$ if $x \geq 0$ and -1 if $x < 0$.

Algorithm 2 Bracha-Agreement() *from the perspective of process p*

Require: $v_p \in \{-1, 1\}$.

1: **loop**
2: reliably broadcast v_p and **wait** until $n - f$ messages are validated from some processes S.
set $v_p := \mathrm{sgn}(\sum_{q \in S} v_q)$.
3: reliably broadcast v_p and **wait** until $n - f$ messages are validated.
if more than $n/2$ messages have some value v **then** set $v_p := (\mathrm{dec}, v)$.
4: reliably broadcast v_p and **wait** until $n - f$ messages are validated.
let x_p be the number of (dec, v) messages validated by p.
5: **if** $x_p \geq 1$ **then**
6: set $v_p := v$.
7: **if** $x_p \geq f + 1$ **then**
8: **decide** v.
9: **if** $x_p = 0$ **then**
10: $v_p :=$ Coin-Flip(). ▷ Returns value in $\{-1, 1\}$.

Correctness. Suppose that at the beginning of an iteration, there is a set of at least $(n + f + 1)/2$ good processes who agree on a value $v \in \{-1, 1\}$.[3] It follows that in Line 2, every process hears from at least $(n + f + 1)/2 - f > (n - f)/2$ of these good processes, i.e., a strict majority in any set of $n - f$. Thus, every good process broadcasts v in Line 3, and due to the validation mechanism, any bad process that wishes to participate in Line 3 *also* must broadcast v. Thus, every good process p will eventually validate $n - f > n/2$ votes for v and set $v_p := (\mathrm{dec}, v)$ indicating it is prepared to decide v in this iteration. By the same reasoning, every good process p will set $x_p := n - f \geq f + 1$ and **decide** v in Line 8.

It is impossible for p to validate two messages (dec, v) and (dec, v') in Line 4 with $v \neq v'$. To validate such messages, p would need to receive strictly greater than $n/2$ "v" and "v'" messages in Line 3, meaning some process successfully broadcast two distinct messages with the same timestamp. By Theorem 1 this is impossible.

Now suppose that in some iteration p **decides** v in Line 8. This means that p validated $n - f$ messages in Line 4 and set $x_p \geq f + 1$. Every other good process q must have validated at least $n - 2f$ of the messages that p validated, and therefore set $x_q \geq 1$, forcing it to set $v_q := v$ in Line 6. Thus, at the beginning of the next iteration $n - f$ good processes agree on the value v and all **decide** v (Line 8) in that iteration.[4]

The preceding paragraphs establish correctness. Turning to efficiency, consider any iteration in which no process **decides** v in Line 8. We can partition the good population into G_6 and G_{10}, depending on whether they execute Line 6 (setting $v_p := v$) or Line 10. If a sufficiently large number of calls to Coin-Flip() made by G_{10}-processes returns v (specifically, $(n + f + 1)/2 - |G_6|$) then by the argument above, all processes will **decide** v (Line 8) in the next iteration. Call this happy event $\mathcal{E}$. If $G_6 = \emptyset$ then both values of v are acceptable, which just increases the likelihood of $\mathcal{E}$.

Bracha [14] and Ben-Or [9] implement Coin-Flip by each process privately flipping an independent, unbiased coin. Thus, for any $f < n/3$, $\Pr(\mathcal{E}) \geq 2^{-(n-f-1)}$ and the expected number of iterations is at most $2^{\Theta(n)}$. If there were a mechanism to implement Coin-Flip as a roughly unbiased *shared* coin (all processes in G_{10} see the same value; see Rabin [40] and Toueg [47]), then $\Pr(\mathcal{E})$ is constant and we only need $O(1)$ iterations in expectation. Efficient collective coin-flipping is therefore the heart of the Byzantine Agreement problem in this model.

2.2 The Iterated Blackboard Model

King and Saia [34] implemented a Coin-Flip() routine using a *blackboard* primitive, which weakens the power of the scheduling adversary to give drastically different views to different processes.[5] Their blackboard protocol is resilient to $f < n/4$ faults. Kimmett [32] simplified and improved this protocol to tolerate $f < n/3$ faults. In this section, we describe a useful extension of the Kimmett-King-Saia style blackboard that *further reduces* the kinds of disagreements that good processes can have.

In the original model [32, 34], a *blackboard* is an $m \times n$ matrix BB, initially all blank ($\perp$), such that column $\mathrm{BB}(\cdot, i)$ is only written to by process i. Via reliable broadcasts, process i attempts to sequentially write non-$\perp$ values to $\mathrm{BB}(r, i)$, $r \in [m]$. The scheduling power of the adversary allows it to control the rate at which different processes write values. Because there could be up to f crash-faults, no process can count on BB containing more than $n - f$ *complete* columns (those i for which $\mathrm{BB}(m, i) \neq \perp$). The final BB-matrix may therefore contain up to f *partial* columns.

The main guarantee of [32, 34] is that every process p has a mostly accurate *view* $\mathrm{BB}^{(p)}$ that agrees with the "true" blackboard BB in all but at most f locations. In particular, the last non-$\perp$ entry of each partial column in BB may still be $\perp$ in $\mathrm{BB}^{(p)}$. If we were to generate a sequence of blackboards with [32, 34], the views from two processes could differ by f locations in *each* blackboard.

An *iterated blackboard* is an endless series $\mathrm{BB} = (\mathrm{BB}_1, \mathrm{BB}_2, \ldots)$ of $m \times n$ blackboards, such that process i only attempts to write its column in BB_t once it completes participation in BB_{t-1}. After p finalizes BB_t, p obtains a view of the full history $\mathrm{BB}^{(p,t)} = (\mathrm{BB}_1^{(p,t)}, \ldots, \mathrm{BB}_t^{(p,t)})$ that differs from $(\mathrm{BB}_1, \ldots, \mathrm{BB}_t)$ in f locations *in total*. As a consequence, $\mathrm{BB}^{(p,t-1)}$ may not be identical to the first $t - 1$ matrices of $\mathrm{BB}^{(p,t)}$, i.e., p could record "retroactive" updates to previous matrices while it is actively participating in the construction of BB_t.

See [27, Appendix] for proof of Theorem 3.

[3] Note that this is always numerically possible since $(n + f + 1)/2 \leq n - f$ with equality if $f = (n - 1)/3$.

[4] Bracha [14] sets the thresholds in Line 5 and 7 to be $f + 1$ and $2f + 1$. The idea was to guarantee that if $x_p \geq f + 1$ then at least one *good* process sent p a (dec, v) message. However, because of the validation mechanism this is not important. A corrupt process can *try* to send a (dec, v) message but it will not be validated unless v does, in fact, have a strict majority ($> n/2$) of messages sent in Line 3.

[5] For example, in Line 2 of Bracha-Agreement, the scheduling adversary can show p *any* $n - f$ messages S, and therefore have significant control over the value of $\mathrm{sgn}(\sum_{q \in S} v_q)$.

Theorem 3. *There is a protocol for n processes to generate an iterated blackboard BB that is resilient to $f < n/3$ Byzantine failures. For $t \geq 1$, the following properties hold:*

(1) *Upon completion of the matrix BB_t, each column consists of a prefix of non-$\perp$ values and a suffix of all-$\perp$ values. Let $\mathrm{last}(i) = (t', r)$ be the position of the last value written by process i, i.e., $\mathsf{BB}_{t'}(r, i) \neq \perp$ and if $t' < t$ then i has not written to any cells of BB_t. When BB_t is complete, it has at least $n - f$ full columns and up to f partial columns.*

(2) *For each t, each process p forms a history*

$$\mathsf{BB}^{(p,t)} = (\mathsf{BB}_1^{(p,t)}, \ldots, \mathsf{BB}_t^{(p,t)})$$

such that for every $t' \in [t]$, $i \in [n]$, $r \in [m]$,

$$\mathsf{BB}_{t'}^{(p,t)}(r, i) \begin{cases} = \mathsf{BB}_{t'}(r, i) & \text{if } \mathrm{last}(i) \neq (t', r) \\ \in \{\mathsf{BB}_{t'}(r, i), \perp\} & \text{otherwise} \end{cases}$$

(3) *If q writes any non-$\perp$ value to BB_{t+1}, then by the time any process p fixes $\mathsf{BB}^{(p,t+1)}$, p will be aware of q's view $\mathsf{BB}^{(q,t)}$ of the history up to blackboard t.*

3 ITERATED COIN FLIPPING GAMES

We begin in Section 3.1 with a simplified coin-flipping game and extend it in Section 3.2 to the real coin-flipping game we use to implement Coin-Flip() in Bracha-Agreement. In the real coin-flipping game we assign *weights* to the processes, which is a measure of trustworthiness. Section 3.3 explains how the weights are updated and Section 3.4 bounds numerical inconsistencies in different processors views.

3.1 A Simplified Game

In this game there are n players partitioned into $n - f$ *good* players G and $f = n/(3+\epsilon)$ *bad* players B, for some small $\epsilon > 0$. The good players are unaware of the partition (G, B). The game is played up to T times in succession according to the following rules. Let $t \in [T]$ be the current iteration.

- The adversary privately picks an *adversarial direction* $\sigma(t) \in \{-1, 1\}$.[6]
- Each good player $i \in G$ picks $X_i(t) \in \{-1, 1\}$ uniformly at random. The bad players see these values then generate their values $\{X_i(t)\}_{i \in B}$, each in $\{-1, 1\}$, as they like.
- If the *outcome* of the coin flip, $\mathrm{sgn}(\sum_{i \in [n]} X_i(t))$, is equal to $\sigma(t)$, the game continues to iteration $t + 1$.

From the good players' perspective, the nominal goal of this game is to eventually achieve the outcome $\mathrm{sgn}(\sum_{i \in [n]} X_i(t)) \neq \sigma(t)$, but the adversary can easily foil this goal if $T = \mathrm{poly}(n)$. We consider a secondary goal: namely to *identify* bad players based solely on the historical data $\{X_i(t)\}_{i,t}$. This turns out to be a tricky problem, but we can identify a *pair* of processes, at least one of which is bad, w.h.p.

[6] In the context of Bracha-Agreement, σ would be $-v$, where v is the value set by processes executing Line 6.

Lemma 4. *Suppose the game does not end after T iterations. If $T = \tilde{\Theta}((n/\epsilon)^2)$, then the pair $(i, j) \in [n]^2$, $i \neq j$, maximizing*

$$\langle X_i, X_j \rangle = \sum_{t=1}^{T} X_i(t) X_j(t)$$

has $B \cap \{i, j\} \neq \emptyset$.

Proof. If $i, j \in G$ are good, by a Chernoff-Hoeffding bound $\langle X_i, X_j \rangle \leq \beta = \tilde{O}(\sqrt{T})$ with high probability, thus every pair whose inner product exceeds β must contain at least one bad process. We now argue that there exists an $i^\star, j^\star \in B$ such that $\langle X_{i^\star}, X_{j^\star} \rangle$ exceeds β. Observe that

$$\sum_{(i \neq j) \in B^2} X_i(t) X_j(t) = \left(\sum_{i \in B} X_i(t) \right)^2 - \sum_{i \in B} (X_i(t))^2 = \left(\sum_{i \in B} X_i(t) \right)^2 - f. \tag{1}$$

Let $S(t) = \sum_{i \in G} X_i(t)$ be the sum of the good processes in iteration t. The bad players force the sign of the sum to be $\sigma(t)$, i.e., $\mathrm{sgn}(S(t) + \sum_{i \in B} X_i(t))\sigma(t) = 1$. Thus,

$$\left(\sum_{i \in B} X_i(t) \right)^2 \geq \begin{cases} (S(t))^2 & \text{if } \mathrm{sgn}(S(t)) \neq \sigma(t) \\ 0 & \text{otherwise} \end{cases} = (\max\{0, -\sigma(t) S(t)\})^2. \tag{2}$$

Let $Z(t) = (\max\{0, -\sigma(t)S(t)\})^2$. By a Chernoff-Hoeffding bound, w.h.p. $Z(t) \leq \gamma = \tilde{O}(n)$ for every t. Moreover, since the distribution of $S(t)$ is symmetric around the origin,

$$\mathbf{E}[Z(t)] \geq \frac{1}{2} \mathbf{E}[S(t)^2 \mid -\sigma(t) S(t) \geq 0] = \frac{1}{2}(n - f). \tag{3}$$

Thus, by linearity of expectation and Chernoff-Hoeffding, we have, w.h.p.,

$$\sum_{t=1}^{T} Z(t) \geq \frac{1}{2} T(n - f) - \gamma \cdot \tilde{O}(\sqrt{T}) = \frac{1}{2} T(n - f) - \tilde{O}(n\sqrt{T}). \tag{4}$$

Combining Eqns. (1), (2), and (4), we have, w.h.p.,

$$\sum_{(i \neq j) \in B^2} \langle X_i, X_j \rangle = \sum_{t \in [T]} \left(\left(\sum_{i \in B} X_i(t) \right)^2 - f \right) \geq \sum_{t \in [T]} (Z(t) - f) \geq \frac{1}{2} T(n - 3f) - \tilde{O}(n\sqrt{T}) = \epsilon n T/2 - \tilde{O}(n\sqrt{T}). \tag{5}$$

We lower bound the average correlation score within B by dividing Eqn. (5) by the $f(f-1)$ distinct pairs $i, j \in B^2$. Using the fact that $f = n/(3+\epsilon)$, we have

$$\max_{i^*, j^* \in B, i^* \neq j^*} \langle X_{i^*}, X_{j^*} \rangle \geq \frac{1}{f(f-1)} \left(\epsilon n T/2 - \tilde{O}(n\sqrt{T}) \right) \geq \frac{(3+\epsilon)\epsilon}{2(f-1)} T - \tilde{O}(\sqrt{T}/n)$$

Note that the $\tilde{O}(\sqrt{T}/n)$ term is negligible and that $\frac{(3+\epsilon)\epsilon}{2(f-1)} T \gg \beta = \tilde{O}(\sqrt{T})$ whenever $T = \tilde{\Theta}((n/\epsilon)^2)$. □

3.2 The Real Coin-Flipping Game

In this section we describe a protocol for calling Coin-Flip() iteratively in the context of Bracha-Agreement. It is based on a coin-flipping game that differs from the simplified game of Section 3.1 in several respects, most of which stem from the power of the adversarial scheduler to give good players slightly different views of reality. The differences are as follows.

- The bad players are *not* fixed in advance, but may be corrupted at various times.
- Rather than picking $X_i(t) \in \{-1, 1\}$, the processes generate an iterated blackboard BB where each write is a value in $\{-1, 1\}$, chosen uniformly at random *if the writing process is good*. Each blackboard BB_t has n columns and $m = \Theta(n/\epsilon^2)$ rows. When BB_t is complete, let $X_i(t)$ be the sum of all non-$\perp$ values in column $\mathsf{BB}_t(\cdot, i)$. Every player's view of reality is slightly different. $X_i^{(p)}(t)$ refers to p's most up-to-date view of $X_i(t)$, which is initially the sum of column $\mathsf{BB}_t^{(p,t)}(\cdot, i)$. By Theorem 3, $\sum_{i\in[n]} \left|X_i^{(p)}(t) - X_i(t)\right| \le f$ for any p, t.
- Each process i has a *weight* $w_i \in [0, 1]$, initially 1, which is non-increasing over time. At all times, the processes maintain *complete agreement* on the weights of the actively participating processes, i.e., those who broadcast coin flips. This is accomplished as follows. By Theorem 3(3), if any process q writes to BB_t, every other process p learns $\mathsf{BB}^{(q,t-1)}$ by the time they finish computing BB_t. Based on the history $\mathsf{BB}^{(q,t-1)}$, p can locally compute the weight vector $(w_i^{(q)})_{i\in[n]}$ of q. However, due to different views of the history, $(w_i^{(q)})_{i\in[n]}$ may be slightly different than $(w_i^{(p)})_{i\in[n]}$. We reconcile these views by defining the weight of each participating process based on its *own* view of history, i.e.

$$w_i = \begin{cases} w_i^{(i)} & \text{if } w_i^{(i)} > w_{\min} \\ 0 & \text{otherwise.} \end{cases}$$

  In other words, w_i is drawn from the weight vector computed by process i. Thus, by Theorem 3(3), the weight w_i of any process participating in BB_t is *common knowledge*. (It is fine that the weights of non-participating processes remain uncertain.) For technical reasons, a weight is rounded down to 0 if it is less than a small threshold, $w_{\min} = \sqrt{n \ln n}/T$, where T is defined below.
- In iteration t, process p sets its own output of Coin-Flip() to be $\operatorname{sgn}\left(\sum_{i\in[n]} w_i X_i^{(p)}(t)\right)$. If this quantity is $-\sigma(t)$ for every good process p, the game ends "naturally." (In the next iteration of Bracha's algorithm, all processes will **decide** on a common value.)
- The iterations are partitioned into $O(f)$ *epochs*, each with $T = \Theta(n^2 \ln^3 n/\epsilon^2)$ iterations, where the goal of each epoch is to either end the game naturally or gather enough statistical evidence to reduce the weight of some processes before the next epoch begins. This can be seen as *fractional blacklisting*.
- Because the scheduling adversary can avoid delivering messages from f good processes, the resiliency of the protocol drops to $f = n/(4+\epsilon)$. Any positive $\epsilon > 0$ suffices, so we can tolerate f as high as $(n-1)/4$. In some places we simplify calculations by assuming $\epsilon \le 1/2$.

Throughout c is an arbitrarily large constant. All "with high probability" bounds hold with probability $1 - n^{-\Omega(c)}$. Since each process flips at most m coins in each iteration, by a Chernoff-Hoeffding bound we have

$$|X_i(t)| \le \sqrt{cm \ln n} \stackrel{\text{def}}{=} X_{\max}$$

holds for all i, t, with high probability. To simplify some arguments we will actually enforce this bound deterministically. If $X_i(t)$ is not in the interval $[-X_{\max}, X_{\max}]$, map it to the nearest value of $\pm X_{\max}$.

With high probability, the weight updates always respect Invariant 1, which says that the total weight-reduction of good players is at most the weight reduction of bad players, up to an additive error of $\epsilon^2 f/8$. This error term arises from the fact that we are integrating slightly inconsistent weight vectors $(w_i^{(p)})$ for each p to yield (w_i). With the assumption $\epsilon \le 1/2$, Invariant 1 implies that the total weight of good processes is always $\Omega(n)$.

Invariant 1. *Let G and B denote the set of good and bad processes at any given time. Then,*

$$\sum_{i\in G}(1 - w_i) \le \sum_{i\in B}(1 - w_i) + \epsilon^2 f/8.$$

Whereas pairwise correlations alone suffice to detect bad players in the simplified game, the bad players can win the real coin-flipping game without being detected by this particular test. As we will see, this can only be accomplished if $\{X_i(t)\}_{t\in[T]}$ differs significantly from a binomial distribution, for some $i \in B$. Thus, in the real game we measure individual deviations in the l_2-norm in addition to pairwise correlations. Define $\mathrm{dev}(i)$ and $\mathrm{corr}(i, j)$ at the end of a particular epoch as below. The iterations of the epoch are indexed by $t \in [T]$ and throughout the epoch the weights $\{w_i\}$ are unchanging.

$$\mathrm{dev}(i) = \sum_{t\in[T]} (w_i X_i(t))^2,$$
$$\mathrm{corr}(i, j) = \sum_{t\in[T]} w_i w_j X_i(t) X_j(t).$$

Naturally each process p estimates these quantities using its view of the historical record; let them be $\mathrm{dev}^{(p)}(i)$ and $\mathrm{corr}^{(p)}(i, j)$.

The Gap Lemma says that if we set the deviation and correlation thresholds (α_T, β_T) properly, no good player will exceed its deviation budget, no pairs of good players will exceed their correlation budget, but some bad player or pair involving a bad player will be detected by one of these tests. One subtle point to keep in mind in this section is that random variables that depend on the coins flipped by good players can still be heavily manipulated by the scheduling power of the adversary.[7] See [38, Ch. 6] for further discussion of this issue.

[7] For example, by Doob's optional stopping theorem for martingales, it is true that $\mathbf{E}[X_i(t)] = 0$, but not true that the distribution of $X_i(t)$ is symmetric around 0, or that it is close to binomial, or that we can say anything about $X_i(t)$ after conditioning on some natural event, e.g., that it was derived from summing the values in a *full* column of $\mathsf{BB}_t(\cdot, i)$.

Lemma 5 (The Gap Lemma). *Consider any epoch in which the game does not end, and let $\{w_i\}_{i\in[n]}$ be process weights. Let G and B be the good and bad processes at the end of the epoch. With high probability,*

1. *Every good $i \in G$ has $\mathrm{dev}(i) \le w_i^2\alpha_T$, where $\alpha_T = m(T + \sqrt{T(c\ln n)^3})$.*
2. *Every pair $i, j \in G$ has $\mathrm{corr}(i,j) \le w_i w_j \beta_T$, where $\beta_T = m\sqrt{T(c\ln n)^3}$.*
3. *If the weights satisfy Invariant 1 and no processes were added to B in this epoch, then*

$$\sum_{i\in B} \max\{0, \mathrm{dev}(i) - w_i^2\alpha_T\} + \sum_{(i\neq j)\in B^2} \max\{0, \mathrm{corr}(i,j) - w_i w_j \beta_T\} \ge \frac{\epsilon}{16} f\alpha_T.$$

Proof of The Gap Lemma, Parts 1 and 2.

Part 1. Fix a good process $i \in G$ and $t \in [T]$. For $r \in [m]$, let $\delta_r \in \{-1, 0, 1\}$ be the outcome of its rth coin-flip, being 0 if the adversary never lets it flip r coins in iteration t. Then for any $r < s$, $\mathbf{E}[\delta_r\delta_s] = 0$. This clearly holds when $\delta_s = 0$, and if the adversary lets the sth flip occur, $\mathbf{E}[\delta_r\delta_s \mid \delta_s \neq 0, \delta_r] = 0$ since $\delta_s \in \{-1, 1\}$ is uniform and independent of δ_r. Therefore, $\mathbf{E}[(X_i(t))^2] = \mathbf{E}[(\sum_{r=1}^m \delta_r)^2] = \sum_{r=1}^m \mathbf{E}[\delta_r^2] + \sum_{r\neq s} \mathbf{E}[\delta_r\delta_s] = \sum_{r=1}^m \mathbf{E}[\delta_r^2] \le m$.

Now consider the sequence of random variables $(S_t)_{t\in[0,T]}$ where $S_0 = 0$ and $S_t = S_{t-1} + (X_i(t))^2 - m$. Since $\mathbf{E}[S_t \mid S_{t-1}, \ldots, S_0] \le S_{t-1}$, (S_t) is a supermartingale. For all $t \in [T]$ we guarantee $|X_i(t)| \le X_{\max}$, so $|S_t - S_{t-1}| = |(X_i(t))^2 - m| \le X_{\max}^2$. Hence, by Azuma's inequality, $S_T \le X_{\max}^2\sqrt{T(c\ln n)}$ with probability $1 - \exp\{-(X_{\max}^2\sqrt{T(c\ln n)})^2/2TX_{\max}^2\} = 1 - n^{-\Omega(c)}$. Therefore, with high probability, for all $i \in [n]$,

$$\begin{aligned}\mathrm{dev}(i) &= \sum_{t=1}^T (w_i X_i(t))^2 = w_i^2(S_T + Tm)\\ &\le w_i^2(Tm + X_{\max}^2\sqrt{T(c\ln n)})\\ &= w_i^2 m\left(T + \sqrt{T(c\ln n)^3}\right) = w_i^2\cdot\alpha_T.\end{aligned}$$

Part 2. Fix a $t \in [T]$ and let $\delta_{i,r} \in \{-1, 0, 1\}$ be the outcome of the rth coin-flip of i in iteration t. By the same argument as above, $\mathbf{E}[X_i(t)X_j(t)] = \mathbf{E}[(\sum_r \delta_{i,r})(\sum_s \delta_{j,s})] = \sum_{r,s} \mathbf{E}[\delta_{i,r}\delta_{j,s}] = 0$. Now consider the sequence $(S_t)_{t\in[0,T]}$ where $S_0 = 0$ and $S_t = S_{t-1} + X_i(t)X_j(t)$. It follows that $\mathbf{E}[S_t \mid S_{t-1}, \ldots, S_0] = S_{t-1}$, so (S_t) is a martingale. By assumption, for all t, both $|X_i(t)|, |X_j(t)| \le X_{\max}$. So, $|S_t - S_{t-1}| = |X_i(t)||X_j(t)| \le X_{\max}^2$. By Azuma's inequality, $S_T \le X_{\max}^2\sqrt{Tc\ln n}$ with probability $1 - n^{-\Omega(c)}$. Therefore, with high probability, for all i, j,

$$\begin{aligned}\mathrm{corr}(i,j) &= \sum_{t=1}^T w_i w_j X_i(t)X_j(t) \le w_i w_j X_{\max}^2\sqrt{Tc\ln n}\\ &= w_i w_j\cdot m\sqrt{T(c\ln n)^3} = w_i w_j\cdot\beta_T.\end{aligned}$$ □

Part 3 of the Gap Lemma is proved in Lemmas 6–11. By Invariant 1, the total weight loss of the good players is at most the weight loss of the bad players plus $\epsilon^2 f/8$. Define ρ to be the relative weight loss of the bad players:

$$\rho \ge 0 \text{ is such that } \sum_{i\in B} w_i = (1-\rho)f.$$

Thus, at this moment $\sum_{i\in G}(1 - w_i) \le \rho f + \epsilon^2 f/8$. Remember that the scheduling adversary can allow the protocol to progress while neglecting to schedule up to f good players. Thus, in Lemma 6 we consider an arbitrary set $\hat{G} \subset G$ of $n - 2f$ good players.

Lemma 6. *If Invariant 1 holds then*

1. *For any $\hat{G} \subseteq G$ with $|\hat{G}| = n - 2f$,*
$$\sum_{i\in\hat{G}} w_i^2 \ge (1 - \max\{\rho/2, \epsilon/8\})^2(n - 2f).$$
2. $\sum_{(i\neq j)\in B^2} w_i w_j \le (1-\rho)^2 f^2$ *and* $(1-\rho)^2 f \le \sum_{i\in B} w_i^2 \le (1-\rho)f$.

Proof. We first claim that, for any real numbers $\hat{w}_1, \ldots, \hat{w}_k \in [0, 1]$, if $\sum_{i=1}^k \hat{w}_i = (1-\hat{\rho})k$ for some $\hat{\rho} \in [0, 1]$, then $(1-\hat{\rho})^2 k \le \sum_{i=1}^k \hat{w}_i^2 \le (1-\hat{\rho})k$. The lower bound follows from Jensen's inequality and is achieved when all weights are equal. The upper bound follows from the fact that $\hat{w}_i^2 \le \hat{w}_i$.

Part 1. Note that

$$\begin{aligned}\sum_{i\in\hat{G}} w_i &= n - 2f - \sum_{i\in\hat{G}}(1 - w_i)\\ &\ge n - 2f - (\rho + \epsilon^2/8)f \qquad \text{(Invariant 1)}\\ &= (1 - \tfrac{\rho+\epsilon^2/8}{2+\epsilon})(n - 2f)\\ &\ge (1 - \max\{\rho/2, \epsilon/8\})(n - 2f)\end{aligned}$$

Thus the relative weight loss from $\hat{G}$'s point of view is less than $\hat{\rho} = \max\{\rho/2, \epsilon/8\}$, and from the first claim of the proof, $\sum_{i\in\hat{G}} w_i^2 \ge (1 - \max\{\rho/2, \epsilon/8\})^2(n - 2f)$.

Part 2. From the first claim of the proof with $\hat{\rho} = \rho$, we have $(1-\rho)^2 f \le \sum_{i\in B} w_i^2 \le (1-\rho)f$. For the other claim,

$$\begin{aligned}\sum_{(i\neq j)\in B^2} w_i w_j &= \Big(\sum_{i\in B} w_i\Big)^2 - \sum_{i\in B} w_i^2\\ &\le (1-\rho)^2 f^2 - (1-\rho)^2 f\\ &\le (1-\rho)^2 f^2.\end{aligned}$$ □

Let us recall a few key facts about the game. Before BB_t is constructed the adversary commits to its desired direction $\sigma(t)$. The $m \times n$ matrix BB_t is complete when it has $n - f$ full columns, therefore the adversary *must* allow at least $m(n - 2f)$ coins to be flipped by good players. We define $S_G(t)$ to be the weighted sum of all the coin flips flipped by good players. I.e., if the set G is stable throughout iteration t then

$$S_G(t) = \sum_{i\in G} w_i X_i(t).$$

If a process i were corrupted in the middle of iteration t then only a prefix of its coin flips would contribute to $S_G(t)$. If $\mathrm{sgn}(S_G(t)) = \sigma(t)$ then the adversary is happy. For example, it can just let the sum of the coin flips controlled by corrupted players sum up to zero, which does not look particularly suspicious. However, if $\mathrm{sgn}(S_G(t)) \neq \sigma(t)$ then the adversary must counteract the good coin flips. Due to disagreements in the state of the blackboard (see

Lemma 10), players can disagree about the sum of blackboard entries by up to f, so the adversary may only need to counteract the good players by $-\sigma(t)S_G(t)-f$. Lemma 7 lower bounds the second moment of this objective.

Lemma 7. *For all $t \in [T]$,*

$$\begin{aligned}&\mathbf{E}[(\max\{0,-\sigma(t)S_G(t)-f\})^2]\\ &\geq m\left((1-\max\{\rho/2,\epsilon/8\})^2(n/2-f)-\epsilon f/16\right).\end{aligned}$$

Proof. Let S_r, $r \geq 0$, be the weighted sum of the first $m(n-2f)+r$ coin flips generated by good players, and $Z_r = (\max\{0,-\sigma(t)S_r - f\})^2$ be the objective function for S_r. The adversary can choose to stop letting the good players flip coins at any time after $m(n-2f)$, thus $\mathbf{E}[(\max\{0,-\sigma(t)S_G(t)-f\})^2] = \mathbf{E}[Z_{2fm}]$, which we argue is at least $\mathbf{E}[Z_0]$. Note that if $Z_{r-1}=0$ then the adversary has achieved the minimum objective and has no interest in further flips, so $\mathbf{E}[Z_r \mid Z_{r-1}=0] \geq Z_{r-1}$. If $Z_{r-1}>0$, then were the adversary to allow some $i \in G$ to flip another coin, we would have $S_r = S_{r-1}+w_i\delta_r$, $\delta_r \in \{-1,1\}$, and

$$Z_r = \begin{cases}(-\sigma(t)S_{r-1}-f+w_i)^2 = Z_{r-1}+2w_i(-\sigma(t)S_{r-1}-f)+w_i^2 \\ \qquad\qquad \text{with probability } \frac{1}{2},\\ (-\sigma(t)S_{r-1}-f-w_i)^2 = Z_{r-1}-2w_i(-\sigma(t)S_{r-1}-f)+w_i^2 \\ \qquad\qquad \text{with probability } \frac{1}{2}.\end{cases}$$

Thus, $\mathbf{E}[Z_r \mid Z_{r-1}>0, |\delta_r|>0] = Z_{r-1}+w_i^2 \geq Z_{r-1}$, i.e., if the adversary is trying to *minimize* the objective function

$$(\max\{0,-\sigma(t)S_G(t)-f\})^2,$$

it will not allow any good coin flips beyond the bare minimum.

To lower bound $\mathbf{E}[Z_0]$, the analysis above shows that any adversary minimizing this objective will let the player i with the smallest weight flip the next coin (thereby minimizing w_i^2), conditioned on any prior history. Thus, in the worst case the $n-2f$ good players with the smallest weights each flip m coins.

We compute $\mathbf{E}[Z_0]$ under this strategy. Since S_0 is

$$S_0 = \sum_{i=1}^{n-2f}\sum_{r=1}^{m} w_i\delta_{i,r},$$

where $\delta_{i,r} \in \{-1,1\}$ are fair coin flips, $\Pr(-\sigma(t)S_0 \geq 0) \geq \frac{1}{2}$ by a simple bijection argument ($\delta_{i,r} \mapsto -\delta_{i,r}$). Hence, $\mathbf{E}[Z_0] \geq \frac{1}{2}\mathbf{E}[Z_0 \mid -\sigma(t)S_0 \geq 0]$. Continuing,

$$\begin{aligned}\mathbf{E}[Z_0 \mid -\sigma(t)S_0 \geq 0] &= \mathbf{E}[(-\sigma(t)S_0-f)^2 \mid -\sigma(t)S_0 \geq 0]\\ &\quad + \mathbf{E}[Z_0-(-\sigma(t)S_0-f)^2 \mid -\sigma(t)S_0 \geq 0]\\ &\geq \mathbf{E}[(-\sigma(t)S_0-f)^2 \mid -\sigma(t)S_0 \geq 0] - f^2\\ &\geq \mathbf{E}[(S_0)^2 \mid -\sigma(t)S_0 \geq 0]\\ &\quad - 2f\mathbf{E}[-\sigma(t)S_0 \mid -\sigma(t)S_0 \geq 0]\\ &= \mathbf{E}[(S_0)^2] - 2f\mathbf{E}[|S_0|].\end{aligned}$$

The first inequality comes from the fact that $Z_0 \neq (-\sigma(t)S_0-f)^2$ only when $-\sigma(t)S_0 \in [0,f)$ (given the conditioning) and in this range is $-(-\sigma(t)S_0-f)^2 \geq -f^2$. The second inequality comes from expanding $(-\sigma(t)S_0-f)^2$, linearity of expectation, and the fact that $\sigma(t)^2=1$. Since $\mathbf{E}[\delta_{i,r}\delta_{i',r'}]=0$ for $(i,r)\neq(i',r')$,

$$\begin{aligned}\mathbf{E}[(S_0)^2] &= \mathbf{E}\left[\left(\sum_{i,r} w_i\delta_{i,r}\right)^2\right] = \sum_{i,r,i',r'} w_iw_{i'}\mathbf{E}[\delta_{i,r}\delta_{i',r'}]\\ &= m\sum_{i=1}^{n-2f} w_i^2. \qquad (1)\end{aligned}$$

We bound the expected value of $|S_0|$ as follows

$$\begin{aligned}\mathbf{E}[|S_0|] &\leq \sqrt{\mathbf{E}[(S_0)^2]} \quad (\mathrm{Var}[|S_0|] = \mathbf{E}[(S_0)^2] - (\mathbf{E}[|S_0|])^2 \geq 0)\\ &= \sqrt{m\sum_{i=1}^{n-2f} w_i^2} \leq \sqrt{mn} \qquad \text{(Equation 1)}\\ \mathbf{E}[(S_0)^2] &\geq m(1-\max\{\rho/2,\epsilon/8\})^2(n-2f). \qquad \text{(Lemma 6)}\end{aligned}$$

and putting it all together we have

$$\begin{aligned}\mathbf{E}[Z_0] &\geq \frac{1}{2}\mathbf{E}[Z_0 \mid -\sigma(t)S_0 \geq 0]\\ &\geq \frac{1}{2}\left(\mathbf{E}[(S_0)^2] - 2f\cdot\mathbf{E}[|S_0|]\right)\\ &\geq \frac{1}{2}\left(m(1-\max\{\rho/2,\epsilon/8\})^2(n-2f) - 2f\sqrt{mn}\right)\\ &\geq m\left((1-\max\{\rho/2,\epsilon/8\})^2(n/2-f) - \epsilon f/16\right)\end{aligned}$$

The last line follows since $m=\Theta(n/\epsilon^2)$. □

Lemma 8. *With high probability, for every $t \in [T]$,*

$$\max\{0,-\sigma(t)S_G(t)-f\}^2 \leq cmn\ln n.$$

Proof. The total number of good coin flips is at most mn. By a Chernoff-Hoeffding bound, $S_G(t) \leq \sqrt{cmn\ln n}$ with high probability and the lemma follows. □

Lemma 9. *With high probability,*

$$\begin{aligned}&\sum_{t=1}^{T}\max\{0,-\sigma(t)S_G(t)-f\}^2\\ &\geq m\left[\left(\left(1-\max\left\{\frac{\rho}{2},\frac{\epsilon}{8}\right\}\right)^2\left(\frac{n}{2}-f\right)-\frac{\epsilon f}{16}\right)T - n\sqrt{T(c\ln n)^3}\right].\end{aligned}$$

Proof. Let $\gamma = ((1-\max\{\rho/2,\epsilon/8\})^2(n/2-f)-\epsilon f/16)$. Consider the sequence of random variables $A_0, A_1, \ldots, A_T$, where $A_0 = 0$ and $A_t = A_{t-1} + \max\{0,-\sigma(t)S_G(t)-f\}^2 - m\gamma$. By Lemma 7, $\mathbf{E}[A_t \mid A_{t-1},\ldots,A_0] \geq 0$. So, (A_t) is a submartingale. By Lemma 8, with high probability, for all $t\in[T]$, $\max\{0,-\sigma(t)S_G(t)-f\}^2 \leq m\gamma'$, where $\gamma' = cn\ln n$. Assuming this holds, $|A_t - A_{t-1}| \leq m\gamma'$ and, by Azuma's inequality, $A_T \leq -m\gamma'\sqrt{Tc\ln n}$ with probability $1-n^{-\Omega(c)}$. Therefore, with high probability,

$$\sum_{t=1}^{T}\max\{0,-\sigma(t)S_G(t)-f\}^2 = m\gamma T + A_T \geq m(\gamma T - \gamma'\sqrt{Tc\ln n}).$$ □

Lemma 10. *For every epoch in which no players are corrupted,*

$$\sum_{i\in B}\mathrm{dev}(i) + \sum_{(i\neq j)\in B^2}\mathrm{corr}(i,j) \geq \sum_{t=1}^{T}\max\{0,-\sigma(t)S_G(t)-f\}^2.$$

Proof. Define $S_B(t)$ to be the sum of coin flips declared by corrupted players. I.e., if B were stable throughout iteration t then $S_B(t) = \sum_{i\in B} w_i X_i(t)$. Then

$$\begin{aligned}&\sum_{t\in[T]} (S_B(t))^2\\ &= \sum_{t\in[T]}\left(\sum_{i\in B}(w_iX_i(t))^2 + \sum_{(i\neq j)\in B^2} w_iw_jX_i(t)X_j(t)\right)\\ &= \sum_{i\in B}\mathrm{dev}(i) + \sum_{(i\neq j)\in B^2}\mathrm{corr}(i,j).\end{aligned}$$

In iteration $t \in [T]$, the adversary must convince at least one good process p that $\mathrm{sgn}\left(\sum_i w_i X_i^{(p)}(t)\right) = \sigma(t)$. By Theorem 3, $\sum_i |X_i^{(p)}(t) - X_i(t)| \le f$ and hence the total disagreement between p's weighted sum and the true weighted sum is

$$\sum_i \left|w_iX_i^{(p)}(t) - w_iX_i(t)\right| = \sum_i w_i\left|X_i^{(p)}(t) - X_i(t)\right| \le f.$$

Thus, if $-\sigma(t)S_G(t) \ge f$ (the good players sum is in the non-adversarial direction by at least f) the bad players must correct it by setting $\sigma(t)S_B(t) \ge -\sigma(t)S_B(t) - f$. Therefore, for any $S_G(t)$ we must have $(S_B(t))^2 \ge \max\{0, -\sigma(t)S_G(t) - f\}^2$ and the lemma follows. □

Recall from Parts 1 and 2 of The Gap Lemma (Lemma 5) that every good player $i \in G$ has $\mathrm{dev}(i) \le w_i^2\alpha_T$ and every good pair $(i,j) \in G^2$ has $\mathrm{corr}(i,j) \le w_iw_j\beta_T$. Lemma 11 lower bounds the *excess* of the dev/corr-values involving bad players, beyond these allowable thresholds.

Lemma 11. *In any epoch in which no processes are corrupted, With high probability,*

$$\sum_{i\in B}\max\{0, \mathrm{dev}(i) - w_i^2\alpha_T\} + \sum_{(i\neq j)\in B^2}\max\{0, \mathrm{corr}(i,j) - w_iw_j\beta_T\} \ge \frac{\epsilon}{16}f\alpha_T.$$

Proof. By Lemma 9 and Lemma 10, with high probability,

$$\begin{aligned}&\sum_{i\in B}\mathrm{dev}(i) + \sum_{(i\neq j)\in B^2}\mathrm{corr}(i,j)\\ &\ge m\left(\left(\left(1-\max\left\{\frac{\rho}{2},\frac{\epsilon}{8}\right\}\right)^2\left(\frac{n}{2}-f\right) - \frac{\epsilon f}{16}\right)T - n\sqrt{T(c\ln n)^3}\right).\end{aligned} \tag{1}$$

Recall that $\alpha_T = m(T + \sqrt{T(c\ln n)^3})$, $\beta_T = m\sqrt{T(c\ln n)^3}$, and, by Lemma 6, that $\sum_{i\in B} w_i^2 \le (1-\rho)f$ and $\sum_{(i\neq j)\in B^2} w_iw_j \le (1-\rho)^2f^2$. Putting these together we have

$$\begin{aligned}&\alpha_T\sum_{i\in B}w_i^2 + \beta_T\sum_{(i\neq j)\in B^2}w_iw_j\\ &\le m\left(T+\sqrt{T(c\ln n)^3}\right)\cdot(1-\rho)f + m\sqrt{T(c\ln n)^3}\cdot(1-\rho)^2f^2.\end{aligned} \tag{2}$$

The expression we wish to bound is at least (1) minus (2), namely:

$$\begin{aligned}&m\Big[\left((1-\max\{\rho/2,\epsilon/8\})^2(n/2-f) - \epsilon f/16 - (1-\rho)f\right)T\\ &\quad - \left((1-\rho)f + (1-\rho)^2f^2\right)\sqrt{T(c\ln n)^3}\Big]\end{aligned} \tag{3}$$

Now depending on the larger value of $\rho/2$ and $\epsilon/8$, there are two cases expanding Equation 3.

Case 1: $\rho/2 \le \epsilon/8$. In this case, we simplify Equation 3 by setting $\rho = 0$.

$$\begin{aligned}&\text{(Equation 3)}\\ &\ge mT\left[\left(1-\frac{\epsilon}{8}\right)^2\left(1+\frac{\epsilon}{2}\right)f - \frac{\epsilon f}{16} - f\right] - f(f+1)m\sqrt{T(c\ln n)^3}\\ &\ge mT\left[\left(1+\frac{\epsilon}{4}-\frac{\epsilon^2}{8}\right)f - \frac{\epsilon f}{16} - f\right] - n^2m\sqrt{T(c\ln n)^3}\\ &\ge \frac{\epsilon}{8}fmT - n^2m\sqrt{T(c\ln n)^3} \qquad (\epsilon \le 1/2)\\ &\ge \frac{\epsilon}{16}f\alpha_T.\end{aligned}$$

Case 2: $\rho/2 > \epsilon/8$. In this case, we expand the $(1-\rho/2)^2$ term and simplify Equation 3 using the identity $n = (4+\epsilon)f$.

$$\begin{aligned}&\text{(Equation 3)}\\ &\ge mT\left[\left(\frac{n}{2}-2f\right)(1-\rho) + \left(\frac{n}{2}-f\right)\frac{\rho^2}{4} - \frac{\epsilon f}{16}\right] - n^2m\sqrt{T(c\ln n)^3}\\ &= mTf\left[(\epsilon/2)(1-\rho) + (1+\epsilon/2)\rho^2/4 - \epsilon/16\right] - n^2m\sqrt{T(c\ln n)^3},\end{aligned}$$

which is minimized when $\rho = \epsilon/(1+\epsilon/2)$, hence

$$\begin{aligned}&\ge mTf\left[\frac{\epsilon}{2}\left(1-\frac{\epsilon}{1+\epsilon/2}\right) + (1+\epsilon/2)\left(\frac{\epsilon}{1+\epsilon/2}\right)^2/4 - \epsilon/16\right]\\ &\qquad - n^2m\sqrt{T(c\ln n)^3}\\ &= mTf\left[\frac{7\epsilon}{16} - \frac{\epsilon^2}{4(1+\epsilon/2)}\right] - n^2m\sqrt{T(c\ln n)^3},\end{aligned}$$

and since $T = \Theta(n^2\ln^3 n/\epsilon^2)$ and $\alpha_T = m(T + \sqrt{T(c\ln n)^3})$, with $\epsilon < 1/2$ this is lower bounded by

$$\ge \frac{\epsilon}{4}f\alpha_T.$$

Remark 12. We are able to upper bound correlation scores between two good players, and lower bound the average correlation score between two bad players. However, the correlations between good and bad players cannot be usefully limited. This is why Lemma 10 and Lemma 11 only apply to epochs in which no processes are corrupted, since any $\mathrm{corr}(i,j)$ score is difficult to analyze when i is corrupted halfway through the epoch.

3.3 Weight Updates

When the T iterations of an epoch k are complete, we reduce the weight vector (w_i) in preparation for epoch $k+1$. According to The Gap Lemma, if an individual deviation score $\mathrm{dev}(i)$ is too large, i is bad w.h.p., and if a correlation score $\mathrm{corr}(i,j)$ is too large, $B \cap \{i,j\} \neq \emptyset$ w.h.p., so reducing *both* i and j's weights by the *same* amount preserves Invariant 1. With this end in mind,

Weight-Update (Algorithm 3) constructs a complete, vertex- and edge-capacitated graph G on $[n]$, finds a fractional maximal matching μ in G, then docks the weights of i and j by $\mu(i,j)$, for each edge (i,j).

Definition 13 (Fractional Maximal Matching)**.** Let $G = (V, E, c_V, c_E)$ be a graph where $c_V : V \to \mathbb{R}_{\geq 0}$ are vertex capacities and $c_E : E \to \mathbb{R}_{\geq 0}$ are edge capacities. A function $\mu : E \to \mathbb{R}_{\geq 0}$ is a *feasible fractional matching* if $\mu(i,j) \leq c_E(i,j)$ and $\sum_j \mu(i,j) \leq c_V(i)$. It is *maximal* if it is not strictly dominated by any feasible μ'. The *saturation level* of i is $\sum_j \mu(i,j)$; it is *saturated* if this equals $c_V(i)$. An edge (i,j) is *saturated* if $\mu(i,j) = c_E(i,j)$. (Note that contrary to convention, a self-loop (i,i) only counts *once* against the capacity of i, not twice.)

Rounding Weights Down. Recall that if p participates in a blackboard BB_t, that every other process can compute the weight vector computed from p's local view $\mathsf{BB}^{(p,t-1)}$ through blackboard $t-1$. The processes use a unified weight vector in which w_i is derived only from i's local view:

$$w_i = \begin{cases} w_i^{(i)} & \text{if } w_i^{(i)} > w_{\min} = \frac{\sqrt{n \ln n}}{T} \\ 0 & \text{otherwise.} \end{cases}$$

As we will see, the maximum pointwise disagreement $|w_i^{(p)} - w_i^{(q)}|$ between processes p, q is at most $w_{\min}$, and as a consequence, if any p thinks $w_i^{(p)} = 0$ then all processes agree that $w_i = 0$.

Excess Graph. The *excess graph* $G = (V, E, c_V, c_E)$ used in Algorithm 3 is a complete undirected graph on $V = [n]$, including self-loops, capacitated as follows:

$$c_V(i) = w_i,$$
$$c_E(i,i) = \frac{16}{\epsilon f \alpha_T} \cdot \max\{0, \mathrm{dev}(i) - w_i^2 \alpha_T\},$$
$$c_E(i,j) = \frac{16}{\epsilon f \alpha_T} \cdot 2\max\{0, \mathrm{corr}(i,j) - w_i w_j \beta_T\},$$

The reason for the coefficient of "2" in the definition of $c_E(i,j)$ is that (i,j) is a single, undirected edge, but it represents two correlation scores $\mathrm{corr}(i,j) = \mathrm{corr}(j,i)$, which were accounted for separately in Lemma 11. By parts 1 and 2 of The Gap Lemma, $c_E(i,j) = 0$ whenever both i and j are good.

The Weight-Update algorithm from the perspective of process p is presented in Algorithm 3. We want to ensure that the fractional matchings computed by good processes are numerically very close to each other, and for this reason, we use a specific maximal matching algorithm called Rising-Tide (Algorithm 4) that has a continuous Lipschitz property, i.e., small perturbations to its input yield bounded perturbations to its output. Other natural maximal matching algorithms such as *greedy* do not have this property.

3.3.1 Rising Tide Algorithm. The Rising-Tide algorithm initializes $\mu = 0$ and simply simulates the continuous process of increasing all $\mu(i,j)$-values in lockstep, so long as i, j, and (i,j) are not saturated. At the moment one becomes saturated, $\mu(i,j)$ is frozen at its current value.

Lemma 14. Rising-Tide *(Algorithm 4) correctly returns a maximal fractional matching.*

Algorithm 3 Weight-Update from the perspective of process p.

Output: Weights $(w_{i,k})_{i\in[n],k\geq 0}$ where $w_{i,k-1}$ refers to the weight w_i *after* processing epoch $k-1$, and is used throughout epoch k.

1: Set $w_{i,0} \leftarrow 1$ for all i. ▷ All weights are 1 in epoch 1.
2: **for** epoch $k = 1, 2, \ldots, K_{\max}$ **do** ▷ $K_{\max}$ = last epoch
3: Play the coin flipping game for T iterations with weights $(w_{i,k-1})$ and let $\mathrm{dev}^{(p)}$ and $\mathrm{corr}^{(p)}$ be the resulting deviation and correlation scores known to p. Construct the excess graph $G_k^{(p)}$ with capacities:

$$c_V(i) = w_{i,k-1},$$
$$c_E(i,i) = \frac{16}{\epsilon f \alpha_T} \cdot \max\left\{0, \mathrm{dev}^{(p)}(i) - w_{i,k-1}^2 \alpha_T\right\},$$
$$c_E(i,j) = \frac{16}{\epsilon f \alpha_T} \cdot 2\max\left\{0, \mathrm{corr}^{(p)}(i,j) - w_{i,k-1} w_{j,k-1} \beta_T\right\}.$$

4: $\mu_k \leftarrow$ Rising-Tide(G_k) ▷ A maximal fractional matching
5: For each i set

$$w_{i,k}^{(p)} \leftarrow w_{i,k-1} - \sum_j \mu_k(i,j).$$

6: Once $w_{i,k}^{(i)}$ is known for $i \in [n]$, set

$$w_{i,k} = \begin{cases} w_{i,k}^{(i)} & \text{if } w_{i,k}^{(i)} > w_{\min} \stackrel{\text{def}}{=} \frac{\sqrt{n \ln n}}{T} \\ 0 & \text{otherwise.} \end{cases}$$

Algorithm 4 Rising-Tide$(G = (V, E, c_V, c_E))$

1: $E' \leftarrow \{(i,j) \in E \mid c_E(i,j) > 0\}$.
2: $\mu(i,j) \leftarrow 0$ for all $i, j \in V$.
3: **while** $E' \neq \emptyset$ **do**
4: Let $\mu_{E'}(i,j) = \begin{cases} 1 & \text{if } (i,j) \in E' \\ 0 & \text{otherwise.} \end{cases}$.
5: Choose maximum $\epsilon > 0$ such that $\mu' = \mu + \epsilon \mu_{E'}$ is a feasible fractional matching.
6: Set $\mu \leftarrow \mu'$.
7: $E' \leftarrow E' - \{(i,j) \mid i \text{ or } j \text{ or } (i,j) \text{ is saturated}\}$ ▷ $\mu(i,j)$ cannot increase
8: **return** μ.

Proof. Obvious. □

Recall that $c_V(i)$ is initialized to be the (old) weight w_i and the new weight is set to be $c_V(i) - \sum_j \mu(i,j)$. We are mainly interested in differences in the new weight vector computed by processes that start from slightly different graphs G, H. Lemma 15 bounds these output differences in terms of their input differences.

Lemma 15 (Rising Tide Output)**.** *Let $G = (V, E, c_V^G, c_E^G)$ and $H = (V, E, c_V^H, c_E^H)$ be two capacitated graphs, which differ by*

$$\eta_E = \sum_{i,j} \left| c_E^G(i,j) - c_E^H(i,j) \right|$$

in their edge capacities and

$$\eta_V = \sum_i \left| c_V^G(i) - c_V^H(i) \right|$$

in their vertex capacities. Let μ_G and μ_H be the fractional matching computed by Rising-Tide *(Algorithm 4) on G and H respectively. Then:*

$$\sum_i \left| \left(c_V^G(i) - \sum_j \mu_G(i,j) \right) - \left(c_V^H(i) - \sum_j \mu_H(i,j) \right) \right| \le \eta_V + 2\eta_E.$$

See [27, Appendix] for a proof of Lemma 15.

3.4 Error Accumulation and Reaching Agreement

The maximum number of epochs is $K_{\max} = 2.5f$. Let $k \in [1, K_{\max}]$ be the index of the current epoch, and let $w_{i,k-1}$ be the weights that were used in the execution of Coin-Flip() during epoch k. Upon completing epoch k, each process p applies Algorithm 3 to update the consensus weight vector $(w_{i,k-1})_{i\in[n]}$ to produce a local weight vector $(w_{i,k}^{(p)})_{i\in[n]}$, and then the consensus weight vector $(w_{i,k})_{i\in[n]}$ used throughout epoch $k+1$.

Lemma 16 (Maintaining Invariant 1)**.** *Suppose for some $\epsilon > 0$ that $n = (4+\epsilon)f$, $m = \Theta(n/\epsilon^2)$, and $T = \Theta(n^2 \log^3 n/\epsilon^2)$. At any point in epoch $k \in [1, K_{\max}]$, with high probability,*

$$\sum_{i\in G}(1 - w_{i,k-1}) \le \sum_{i\in B}(1 - w_{i,k-1}) + \frac{\epsilon^2}{\sqrt{n}} \cdot (k-1).$$

Proof. We prove by induction on k. For the base case $k = 1$ all the weights are 1 so Lemma 16 clearly holds. We will now prove that if the claim holds for k, it holds for $k+1$ as well. Fix any good process p. The vector $(w_{i,k}^{(p)})$ is derived from $(w_{i,k-1})$ by deducting at least as much weight from bad processes as from good processes, with high probability, and $(w_{i,k})$ is derived from $(w_{i,k}^{(q)})_{q\in[n],i\in[n]}$ by setting $w_{i,k} = w_{i,k}^{(i)}$ and rounding down to 0 if it is at most $w_{\min}$. By the inductive hypothesis,

$$\sum_{i\in G}(1 - w_{i,k}^{(p)}) \le \sum_{i\in B}(1 - w_{i,k}^{(p)}) + \frac{\epsilon^2}{\sqrt{n}} \cdot (k-1).$$

Therefore,

$$\sum_{i\in G}(1 - w_{i,k}) \le \sum_{i\in B}(1 - w_{i,k}) + \frac{\epsilon^2}{\sqrt{n}} \cdot (k-1) + \sum_{i\in[n]} |w_{i,k}^{(p)} - w_{i,k}^{(i)}| + w_{\min}\, n_0,$$

where n_0 is the number of processes whose weight is rounded down to 0 after epoch k.

Hence, it suffices to show that $\sum_{i\in[n]} |w_{i,k+1}^{(p)} - w_{i,k+1}^{(i)}| + w_{\min}\, n_0 \le \epsilon^2/\sqrt{n}$. By Lemma 15, the computed weight difference between process p and any process q can be bounded by twice the sum of all edge capacity differences. According to Algorithm 3, the edge capacities differ due to underlying disagreement on the $\mathrm{dev}(i)$ and $\mathrm{corr}(i,j)$ values. Thus,

$$|w_{q,k}^{(p)} - w_{q,k}^{(q)}| \le 2 \cdot \frac{16}{\epsilon f \alpha_T} \left(\sum_i \left| \mathrm{dev}^{(p)}(i) - \mathrm{dev}^{(q)}(i) \right| + \sum_{i\ne j} \left| \mathrm{corr}^{(p)}(i,j) - \mathrm{corr}^{(q)}(i,j) \right| \right)$$

By Theorem 3, two processes may only disagree in up to f cells of the blackboards $(\mathrm{BB}_1, \ldots, \mathrm{BB}_t)$. Since the sum of each column in each blackboard is bounded by $X_{\max}$, we have $|\mathrm{dev}^{(p)}(i) - \mathrm{dev}^{(q)}(i)| < 2X_{\max}$ for at most f values of i, and $|\mathrm{corr}^{(p)}(i,j) - \mathrm{corr}^{(q)}(i,j)| < 2X_{\max}$ for at most nf pairs $i \in B, j \in (G \cup B)$. Continuing,

$$\begin{aligned} &\le 2 \cdot \frac{16}{\epsilon f \alpha_T} \left(f \cdot 2X_{\max} + nf \cdot 2X_{\max} \right) \\ &\le \frac{64(n+1)X_{\max}}{\epsilon m T} && (\alpha_T \ge mT) \\ &\le \frac{\sqrt{n \ln n}}{T} && (\text{using } m = \Omega(n/\epsilon^2)) \\ &= w_{\min} \end{aligned}$$

Now the inductive step for k holds by noticing that

$$\begin{aligned} &\sum_{i\in[n]} |w_{i,k}^{(p)} - w_{i,k}^{(i)}| + w_{\min}\, n_0 \\ &\le 2 w_{\min}\, n \\ &\le \frac{\epsilon^2}{n^{1.5} \log^{2.5} n} \cdot n && (\text{using } T = \Omega(n^2 \log^3 n/\epsilon^2)) \\ &< \frac{\epsilon^2}{\sqrt{n}}. \end{aligned}$$

Therefore, with $K_{\max} = 2.5f$ we obtain Invariant 1. That is, for any weight vector (w_i) that are used on a blackboard,

$$\begin{aligned} \sum_{i\in G}(1 - w_i) &\le \sum_{i\in B}(1 - w_i) + \frac{\epsilon^2}{\sqrt{n}} \cdot 3f \\ &\le \sum_{i\in B}(1 - w_i) + \frac{1}{8}\epsilon^2 f. && (\text{whenever } n \ge 576) \end{aligned}$$

Note that Invariant 1 is also preserved whenever a process is corrupted, transferring it from G to B. □

The next observation and Lemma 18 shows that the weight of every bad process becomes 0 after running $K_{\max}$ epochs of Weight-Updates without reaching agreement.

Observation 17. For any i and k, if there exists process p such that $w_{i,k}^{(p)} = 0$, then $w_{i,k} = 0$.

Proof. In the proof of Lemma 16 it was shown that $|w_{i,k}^{(p)} - w_{i,k}^{(i)}| \le \sqrt{n \ln n}/T = w_{\min}$, hence if $w_{i,k}^{(p)} = 0$, $w_{i,k}$ is rounded down to 0. See Algorithm 3. □

Lemma 18. *If agreement has not been reached after $K_{\max} = 2.5f$ epochs, all bad processes have weight 0, with high probability.*

Proof. There are at most f epochs in which the adversary corrupts at least one process. We argue below that after all other epochs, in the call to Weight-Update, the total edge capacity of the graph induced by B is at least 1. This implies that in each iteration of Weight-Update, either some $i \in B$ with $c_V(i) = w_i > w_{\min}$ becomes saturated (and thereafter $w_i = 0$ by Observation 17), or the total weight of all processes in B drops by at least 2. The first case can occur at most f times and the second at most $f/2$, hence after $K_{\max} = 2.5f$ epochs, all bad players' weights are zero, with high probability.

We now prove that the total edge capacity is at least 1. Recall that each edge (i, j), $i \neq j$, represents the two correlation scores $\mathrm{corr}(i, j)$ and $\mathrm{corr}(j, i)$. Hence, by Lemma 11, the sum of edge capacities on B is:

$$\begin{aligned}\sum_{\{i,j\}\subset B} c_E(i,j) &= \frac{16}{\epsilon f \alpha_T}\left(\sum_{i\in B}\max\{0, \mathrm{dev}(i) - w_{i,k}^2\alpha_T\}\right.\\ &\qquad\left. + \sum_{(i\neq j)\in B^2}\max\{0, \mathrm{corr}(i,j) - w_{i,k}w_{j,k}\beta_T\}\right)\\ &\geq \frac{16}{\epsilon f \alpha_T}\left(\frac{\epsilon}{16} f\alpha_T\right) \qquad \text{(by Lemma 5)}\\ &\geq 1.\end{aligned}$$

□

Lemma 19. *Suppose Invariant 1 holds. In any iteration in which the bad processes have zero weights, the good processes agree on the outcome of the coin flip, with constant probability.*

Proof. Let $S = \sum_i w_i X_i(t)$ be the weighted sum of the players. Through its scheduling power, the adversary may still be able to create disagreements between good players on the outcome of the coin-flip if $S \in [-f, f]$. Moreover, good processes still possess $\Omega(n)$ total weight by Invariant 1. With constant probability, $|S|$ is larger than its standard deviation, namely $\Theta(\sqrt{mn})$, which is much larger than f as $m = \Omega(n/\epsilon^2)$. Thus, with constant probability all good players agree on the outcome. □

Theorem 20. *Suppose $n = (4+\epsilon)f$ where $\epsilon > 0$, $m = \Theta(n/\epsilon^2)$, and $T = \Theta(n^2 \log^3 n/\epsilon^2)$. Using the implementation of Coin-Flip() from Section 3, Bracha-Agreement solves Byzantine agreement with probability 1 in the full information, asynchronous model against an adaptive adversary. In expectation the total communication time is $\tilde{O}((n/\epsilon)^4)$. The local computation at each process is polynomial in n.*

Proof. By Lemma 18, after $K_{\max} = 2.5f$ epochs, all bad processes' weights become zero, with high probability. From then on, by Lemma 19, each iteration of Bracha-Agreement achieves agreement with constant probability. Thus, after one more epoch, all processes reach agreement with high probability. The total communication time (longest chain of dependent messages) is $O((K_{\max}+1)mT) = \tilde{O}((n/\epsilon)^4)$. If, by chance, the processes fail to reach agreement after this much time, they restart the algorithm with all weights $w_i = 1$ and try again. Thus, the algorithm terminates with probability 1. □

REFERENCES

[1] Rosa Abrantes-Metz, Sofia B. Villas-Boas, and George G. Judge. 2013. *Tracking the Libor Rate.* Technical Report. Department of Agricultural & Resource Economics, UC Berkeley, Working Paper Series. https://ideas.repec.org/p/cdl/agrebk/qt2p33x7dk.html

[2] Miklós Ajtai and Nathan Linial. 1993. The influence of large coalitions. *Comb.* 13, 2 (1993), 129–145. https://doi.org/10.1007/BF01303199

[3] Noga Alon and Moni Naor. 1993. Coin-Flipping Games Immune Against Linear-Sized Coalitions. *SIAM J. Comput.* 22, 2 (1993), 403–417. https://doi.org/10.1137/0222030

[4] James Aspnes. 1998. Lower Bounds for Distributed Coin-Flipping and Randomized Consensus. *J. ACM* 45, 3 (1998), 415–450. https://doi.org/10.1145/278298.278304

[5] James Aspnes. 2003. Randomized protocols for asynchronous consensus. *Distributed Computing* 16, 2 (2003), 165–175.

[6] Hagit Attiya and Keren Censor. 2008. Tight bounds for asynchronous randomized consensus. *J. ACM* 55, 5 (2008), 1–26.

[7] Ziv Bar-Joseph and Michael Ben-Or. 1998. A Tight Lower Bound for Randomized Synchronous Consensus. In *Proceedings 17th Annual ACM Symposium on Principles of Distributed Computing (PODC)*. 193–199. https://doi.org/10.1145/277697.277733

[8] Amos Beimel, Eran Omri, and Ilan Orlov. 2015. Protocols for Multiparty Coin Toss with a Dishonest Majority. *J. Cryptol.* 28, 3 (2015), 551–600. https://doi.org/10.1007/s00145-013-9168-3

[9] Michael Ben-Or. 1983. Another Advantage of Free Choice: Completely Asynchronous Agreement Protocols (Extended Abstract). In *Proceedings 2nd Annual ACM Symposium on Principles of Distributed Computing (PODC)*. 27–30. https://doi.org/10.1145/800221.806707

[10] Michael Ben-Or and Nathan Linial. 1985. Collective Coin Flipping, Robust Voting Schemes and Minima of Banzhaf Values. In *Proceedings 26th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 408–416. https://doi.org/10.1109/SFCS.1985.15

[11] Michael Ben-Or, Elan Pavlov, and Vinod Vaikuntanathan. 2006. Byzantine agreement in the full-information model in $O(logn)$ rounds. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing (STOC)*. 179–186.

[12] Manuel Blum. 1981. Coin Flipping by Telephone. In *IEEE Workshop on Communications Security (CRYPTO)*. 11–15.

[13] Nicolò Bonettini, Paolo Bestagini, Simone Milani, and Stefano Tubaro. 2020. On the use of Benford's law to detect GAN-generated images. In *Proceedings 25th International Conference on Pattern Recognition (ICPR)*. 5495–5502. https://doi.org/10.1109/ICPR48806.2021.9412944

[14] Gabriel Bracha. 1987. Asynchronous Byzantine Agreement Protocols. *Inf. Comput.* 75, 2 (1987), 130–143. https://doi.org/10.1016/0890-5401(87)90054-X

[15] Gabriel Bracha and Sam Toueg. 1985. Asynchronous Consensus and Broadcast Protocols. *J. ACM* 32, 4 (1985), 824–840. https://doi.org/10.1145/4221.214134

[16] Niv Buchbinder, Iftach Haitner, Nissan Levi, and Eliad Tsfadia. 2017. Fair Coin Flipping: Tighter Analysis and the Many-Party Case. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2580–2600. https://doi.org/10.1137/1.9781611974782.170

[17] Richard Cleve. 1986. Limits on the Security of Coin Flips when Half the Processors Are Faulty (Extended Abstract). In *Proceedings of the 18th Annual ACM Symposium on Theory of Computing (STOC)*. 364–369. https://doi.org/10.1145/12130.12168

[18] Miguel Correia, Giuliana Santos Veronese, Nuno Ferreira Neves, and Paulo Verissimo. 2011. Byzantine consensus in asynchronous message-passing systems: a survey. *International Journal of Critical Computer-Based Systems* 2, 2 (2011), 141–161.

[19] Dana Dachman-Soled, Mohammad Mahmoody, and Tal Malkin. 2014. Can Optimally-Fair Coin Tossing Be Based on One-Way Functions?. In *Proceedings 11th Conference on Theory of Cryptography (TCC) (Lecture Notes in Computer Science, Vol. 8349)*. 217–239. https://doi.org/10.1007/978-3-642-54242-8_10

[20] Mark Duggan and Steven D. Levitt. 2002. Winning Isn't Everything: Corruption in Sumo Wrestling. *American Economic Review* 92, 5 (2002), 1594–1605. https://doi.org/10.1257/000282802762024665

[21] Michael J. Fischer, Nancy A. Lynch, and Michael Merritt. 1986. Easy Impossibility Proofs for Distributed Consensus Problems. *Distributed Comput.* 1, 1 (1986), 26–39. https://doi.org/10.1007/BF01843568

[22] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. 1985. Impossibility of Distributed Consensus with One Faulty Process. *J. ACM* 32, 2 (1985), 374–382. https://doi.org/10.1145/3149.214121

[23] Daniel Gamermann and Felipe Leite Antunes. 2017. Evidence of Fraud in Brazil's Electoral Campaigns Via the Benford's Law. *CoRR* abs/1707.08826 (2017). arXiv:1707.08826 http://arxiv.org/abs/1707.08826

[24] Iftach Haitner and Yonatan Karidi-Heller. 2020. A Tight Lower Bound on Adaptively Secure Full-Information Coin Flip. In *Proceedings 61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 1268–1276. https://doi.org/10.1109/FOCS46700.2020.00120

[25] Iftach Haitner, Nikolaos Makriyannis, and Eran Omri. 2018. On the Complexity of Fair Coin Flipping. In *Proceedings 16th International Conference on Theory of Cryptography (TCC) (Lecture Notes in Computer Science, Vol. 11239)*. 539–562. https://doi.org/10.1007/978-3-030-03807-6_20

[26] Iftach Haitner and Eliad Tsfadia. 2017. An Almost-Optimally Fair Three-Party Coin-Flipping Protocol. *SIAM J. Comput.* 46, 2 (2017), 479–542. https://doi.org/10.1137/15M1009147

[27] Shang-En Huang, Seth Pettie, and Leqi Zhu. 2022. Byzantine Agreement in Polynomial Time with Near-Optimal Resilience. *CoRR* abs/2202.13452 (2022). arXiv:2202.13452

[28] Jeff Kahn, Gil Kalai, and Nathan Linial. 1988. The Influence of Variables on Boolean Functions (Extended Abstract). In *Proceedings of the 29th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 68–80. https://doi.org/10.1109/SFCS.1988.21923

[29] Bruce M. Kapron, David Kempe, Valerie King, Jared Saia, and Vishal Sanwalani. 2010. Fast asynchronous Byzantine agreement and leader election with full information. *ACM Trans. Algorithms* 6, 4 (2010), 68:1–68:28. https://doi.org/10.1145/1824777.1824788

[30] Karlo Kauko. 2019. *Benford's law and Chinese banks' non-performing loans.* BOFIT Discussion Papers 25/2019. Bank of Finland, Institute for Economies in Transition (BOFIT). http://hdl.handle.net/10419/212933

[31] A Kilani and G P Georgiou. 2021. Countries with potential data misreport based on Benford's law. *Journal of Public Health* 43, 2 (2021), e295–e296. https://doi.org/10.1093/pubmed/fdab001

[32] Ben Kimmett. 2020. *Improvement and partial simulation of King & Saia's expected-polynomial-time Byzantine agreement algorithm.* Master's thesis. University of Victoria, Canada.

[33] Valerie King and Jared Saia. 2011. Breaking the $O(n^2)$ bit barrier: scalable Byzantine agreement with an adaptive adversary. *J. ACM* 58, 4 (2011), 1–24.

[34] Valerie King and Jared Saia. 2016. Byzantine Agreement in Expected Polynomial Time. *J. ACM* 63, 2 (2016), 13:1–13:21. https://doi.org/10.1145/2837019

[35] Valerie King and Jared Saia. 2018. Correction to Byzantine Agreement in Expected Polynomial Time, JACM 2016. *CoRR* abs/1812.10169 (2018). arXiv:1812.10169 http://arxiv.org/abs/1812.10169

[36] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. 1982. The Byzantine Generals Problem. *ACM Trans. Program. Lang. Syst.* 4, 3 (1982), 382–401. https://doi.org/10.1145/357172.357176

[37] Allison B. Lewko. 2011. The Contest between Simplicity and Efficiency in Asynchronous Byzantine Agreement. In *Proceedings 25th International Symposium on Distributed Computing (DISC) (Lecture Notes in Computer Science, Vol. 6950)*. 348–362. https://doi.org/10.1007/978-3-642-24100-0_35

[38] Darya Melnyk. 2020. *Byzantine Agreement on Representative Input Values Over Public Channels.* Ph. D. Dissertation. ETH Zurich.

[39] Tal Moran, Moni Naor, and Gil Segev. 2016. An Optimally Fair Coin Toss. *J. Cryptol.* 29, 3 (2016), 491–513. https://doi.org/10.1007/s00145-015-9199-z

[40] Michael O. Rabin. 1983. Randomized Byzantine Generals. In *Proceedings 24th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. 403–409. https://doi.org/10.1109/SFCS.1983.48

[41] Boudewijn F. Roukema. 2014. A first-digit anomaly in the 2009 Iranian presidential election. *J. Applied Statistics* 41, 1 (2014), 164–199. https://doi.org/10.1080/02664763.2013.838664

[42] Alexander Russell, Michael E. Saks, and David Zuckerman. 2002. Lower Bounds for Leader Election and Collective Coin-Flipping in the Perfect Information Model. *SIAM J. Comput.* 31, 6 (2002), 1645–1662. https://doi.org/10.1137/S0097539700376007

[43] Michael E. Saks. 1989. A Robust Noncryptographic Protocol for Collective Coin Flipping. *SIAM J. Discret. Math.* 2, 2 (1989), 240–244. https://doi.org/10.1137/0402020

[44] Uri Simonsohn. 2013. Just Post It: The Lesson From Two Cases of Fabricated Data Detected by Statistics Alone. *Psychological Science* 24, 10 (2013), 1875–1888. https://doi.org/10.1177/0956797613480366

[45] Uri Simonsohn, Joseph P. Simmons, and Leif D. Nelson. 2015. Better P-curves: Making P-curve analysis more robust to errors, fraud, and ambitious P-hacking, a Reply to Ulrich and Miller (2015). *Journal of Experimental Psychology* 144, 6 (2015), 1146–1152. https://doi.org/10.1037/xge0000104

[46] Cristi Tilden and Troy Janes. 2012. Empirical evidence of financial statement manipulation during economic recessions. *J. Finance and Accountancy* 10 (2012).

[47] Sam Toueg. 1984. Randomized Byzantine Agreements. In *Proceedings 3rd Annual ACM Symposium on Principles of Distributed Computing (PODC)*. 163–178. https://doi.org/10.1145/800222.806744