Distributed Computing (2022) 35:333–355
https://doi.org/10.1007/s00446-021-00416-4

Revisiting asynchronous fault tolerant computation with optimal resilience

Ittai Abraham[1] · **Danny Dolev**[2] · **Gilad Stern**[2]

Received: 8 December 2020 / Accepted: 13 December 2021 / Published online: 3 February 2022

Abstract

The celebrated result of Fischer, Lynch and Paterson is the fundamental lower bound for asynchronous fault tolerant computation: any 1-crash resilient asynchronous agreement protocol must have some (possibly measure zero) probability of not terminating. In 1994, Ben-Or, Kelmer and Rabin published a *proof-sketch* of a lesser known lower bound for asynchronous fault tolerant computation with optimal resilience in face of a Byzantine adversary: if $n \leq 4t$ then any t-resilient asynchronous verifiable secret sharing protocol must have some **non-zero** probability of not terminating. Our main contribution is to revisit this lower bound and provide a rigorous and more general proof. Our second contribution is to show how to avoid this lower bound. We provide a protocol with optimal resilience that is almost surely terminating for a *strong common coin* functionality. Using this new primitive we provide an almost surely terminating protocol with optimal resilience for asynchronous Byzantine agreement that has a new *fair validity* property. To the best of our knowledge this is the first asynchronous Byzantine agreement with fair validity in the information theoretic setting.

1 Introduction

One of the most important models of distributed computing is the *Asynchronous communication model*. Intuitively, this model captures the highest level of network un-reliability. It allows the adversary to delay each message arrival in an adaptive manner up to any finite amount. A basic question of distributed computing is:

Is there a fundamental limit to fault tolerant computation in the Asynchronous model?

The celebrated Fischer, Lynch, and Paterson (FLP) [12] impossibility result from 1985 is perhaps the most well known such fundamental limitation. It states that reaching *agreement*, even in the face of just one crash failure, is impossible for deterministic protocols. More formally, FLP [12] prove that any protocol that solves Agreement in the asynchronous model that is resilient to at least one crash failure must have a *non terminating* execution. Thus, no protocol can solve Agreement in this model in finite time, but using randomization, it is possible to define a measure on the number of rounds and obtain protocols that have a finite *expected* termination. Given the FLP [12] impossibility it is natural to ask:

Is this potentially measure zero event of non termination the only limitation for fault tolerant computation in the asynchronous model?

In 1983, Ben-Or et al. [6] initiated the study of secure multiparty computation in the asynchronous model. Their fundamental result is that the answer above is *yes* when there are $n > 4t$ servers and an adversary that can corrupt at most t parties in a Byzantine (fully malicious) manner. They show that *perfect* security with finite expected run time can be obtained for any functionality.

The BCG [6] work left open the domain of $3t < n \leq 4t$ (with $n = 3t$ it is known that Byzantine agreement is impossible, see [11]). In 1993, Canetti and Rabin [9] obtained a protocol for Asynchronous Byzantine Agreement with optimal resilience ($3t < n$). Their protocol had an "annoying property": the non-termination event has a **non-zero** probability measure. This problematic non-zero probability of non-termination came from their verifiable secret sharing protocol. In 1994, Ben-Or et al. [7] addressed this problem. They provided an optimal resilience asynchronous secure multiparty computation protocol with the same "annoying property": the non-termination event has a **non-zero** prob-

✉ Gilad Stern
gilad.stern@mail.huji.ac.il

[1] VMWare Research, Herzliya, Israel

[2] The Hebrew University in Jerusalem, Jerusalem, Israel

ability measure.[1] Moreover, BKR [7] claim that this is unavoidable. That is, if $n \leq 4t$ then any t-resilient asynchronous verifiable secret sharing protocol A must have some **non-zero** probability $q_A > 0$ of not terminating. Unfortunately, BKR [7] only provided a *proof-sketch* of this lower bound.

1.1 First contribution: lower bounds on asynchronous verifiable secret sharing with optimal resilience

25 years after the publication of the proof-stretch of BKR [7], the main contribution of this paper is a rigorous proof of the lower bound theorem. We believe that our work will help provide clarity and better understanding of the asynchronous model and its impossibility results. In addition, our lower bound proof improves over the BKR proof-sketch in two important ways:

One weakness of the BKR [7] proof-sketch is that its arguments only imply a lower bound for verifiable secret sharing schemes that have perfect hiding and binding properties. This raises a natural question: Can allowing some error probability in the Asynchronous Verifiable Secret Sharing scheme (AVSS) remove the need for a non-zero probability of non-termination? Our proof strengthens the BKR lower bound claim and proves this is not the case. We prove that even AVSS schemes with constant error must have a non-zero probability of non-termination.

A second weakness of the BKR [7] proof-sketch is that its arguments assume the share (and reconstruct) protocols terminate in a fixed (constant) number of rounds. VSS protocols whose share protocol terminates with probability 1 have been shown to be useful in other contexts [1]. Our proof strengthens the BKR [7] lower bound claim and proves that a non-zero probability of non-termination must occur even if the share and reconstruct protocols only terminate with probability 1.

1.2 Second contribution: upper bounds on strong common coin and asynchronous Byzantine agreement with fair validity

What are the implications of this lower bound? Does it imply that *all* optimal resilience secure computation must have a non-zero probability of non-termination? We know that this is not the case. In fact, Ben-Or [5] and Bracha [8] prove that Byzantine agreement has a measure zero probability of non-termination (it almost surely terminates) with optimal resilience. However the expected time of termination of these protocols is exponential.

The work of [2] shows that almost surely termination is possible even with a polynomial expected number of rounds. This is obtained using a certain type of a *weak common coin* functionality that is also almost surely terminating. This gap raises a natural question:

Are there other functionalities (stronger than a weak coin, but weaker than verifiable secret sharing) that can be implemented in the asynchronous model for $n = 3t + 1$ that are almost surely terminating?

Our first upper bound contribution is to answer this question in the affirmative. We show that a certain type of *strong common coin* is possible to implement in an almost surely terminating manner. The difference between a weak common coin and a strong common coin is that in a strong common coin protocol, all parties output the same value while in a weak common coin, with constant probability, different parties may output different values for the coin.

What is the advantage of a strong common coin over a weak common coin? With a strong common coin we know how to obtain asynchronous Byzantine agreement with a *fair validity* property. In Byzantine agreement protocols with fair validity all parties are required to output a nonfaulty party's input with probability $\frac{1}{2}$ or greater, in addition to the regular requirements from Byzantine agreement protocols. We do not know how to obtain this validity property with a weak coin.

Our second upper bound contribution is a Byzantine Agreement protocol in the Asynchronous model for $n = 3t + 1$ with *fair validity* that is almost surely terminating. To the best of our knowledge this is the first Asynchronous Byzantine Agreement protocol with *fair validity* in the information theoretic setting.

2 Lower bound

2.1 Communication model and definitions

This work deals with a fully-connected asynchronous network with a Byzantine adversary. This means that every two parties have a direct link between them. The direct link is assumed to be secure and authenticated. By the link being secure, we mean that if party P_i sends a message to party P_j, no other party can read its contents. By the link being authenticated, we mean that if party P_i receives a message from P_j, then P_j actually sent that message to P_i. In an asynchronous network any message is eventually received in finite time, but there is no bound on the amount of time the message can be delayed. The network has n parties, and the adversary can control up to t parties. A party controlled by the adversary

[1] BCG [6]: "our protocol, as well as the verifiable secret sharing protocol of [CR93], have the following annoying property: the exponentially small error probability includes an exponentially small non-zero probability of not terminating. This should be contrasted with the asynchronous Byzantine Agreement problem where the randomized protocol terminates with probability 1".

can deviate arbitrarily from the protocol, and is said to be faulty. Our work deals with information-theoretic security, and thus we assume the adversary is potentially computationally unbounded.

In the following definitions and discussions when a party has called a protocol and is running it to completion, we say that it "participates in the protocol".

Definition 1 A Byzantine Asynchronous Verifiable Secret Sharing (AVSS) protocol, comprised of a pair of protocols (S, R),[2] has a designated dealer called D, which receives a secret s from a finite field $\mathcal{F}$ as input. For $\epsilon > 0$, such a protocol is called an almost-surely terminating $(1-\epsilon)$-correct t-resilient Byzantine AVSS protocol if the three following properties hold for every adversary controlling t parties at most, and any message scheduling:

1. *Termination*
   (a) If the dealer is nonfaulty and all nonfaulty parties participate in protocol S, then each nonfaulty party will almost-surely eventually complete protocol S.
   (b) If some nonfaulty party completed protocol S, then each nonfaulty party that participates in S will almost-surely eventually complete protocol S.
   (c) If all of the nonfaulty parties finished protocol S and began protocol R, they will all almost-surely complete protocol R.
2. *Correctness* Once the first nonfaulty party has completed protocol S, there exists some value $r \in \mathcal{F}$ such that with a probability of at least $(1-\epsilon)$:
   (a) If the dealer is nonfaulty, $r = s$.
   (b) Every nonfaulty party that completes protocol R outputs the value r.
3. *Secrecy* If the dealer is nonfaulty, and no honest party has began protocol R, no adversary can gain any information about s. More precisely, denote V^s to be the adversary's view of an execution of S with a nonfaulty dealer sharing s before some nonfaulty party calls protocol R. If the dealer is nonfaulty, then for any given adversary and message scheduling, the distribution of V^s is the same for all possible secrets s.

When we say that some party almost-surely completes the protocol, we mean that it completes the protocol in finite time with probability 1. This also means that for every $\epsilon > 0$ there exists some number $N \in \mathbb{N}$ such that the probability that the party exchanges more than N messages with all parties during protocol S is less than ϵ. It is important to note that those values might need to be adjusted based on the adversary and scheduling as well. Similarly, if all parties almost-surely terminate, for every $\epsilon > 0$ there exists some $N \in \mathbb{N}$ such that the probability that there exists a nonfaulty party who exchanges more than N messages is no greater than ϵ.

[2] S is the protocol for sharing a secret and R is the protocol for reconstructing it.

2.2 Lower bound statement and proofs

The main result shown in this section is proving the following theorem:

Theorem 1 *For any $\epsilon \in (0, \frac{1}{2}]$ and $n, t \in \mathbb{N}$ such that $4t \geq n > t$ there does not exist an almost-surely terminating $\left(\frac{1}{2}+\epsilon\right)$-correct t-resilient Byzantine AVSS protocol (S, R) for n parties.*

This is done by first proving a slightly weaker version of the theorem, only showing that no such protocol exists for $n = 4, t = 1$ when sharing a binary secret $s \in \{0, 1\}$. Using standard methods, this result can be expanded to any $4t \geq n \geq 3t + 1$, and to a multivalued secret, and an explanation of how to do that is provided below. Therefore, the first goal in this section is proving the following theorem:

Theorem 2 *For any $\epsilon \in (0, \frac{1}{2}]$ there is no almost-surely terminating $\left(\frac{1}{2}+\epsilon\right)$-correct 1-resilient Byzantine AVSS protocol (S, R) for sharing a binary secret $s \in \{0, 1\}$ with 4 parties.*

Let the parties be A, B, C, D, and let D be the dealer. By way of contradiction, assume the parties run an almost-surely terminating $\left(\frac{1}{2}+\epsilon\right)$-correct t-resilient Byzantine AVSS protocol. The theorem is proven using two main lemmas. The first lemma describes possible malicious behaviour by a faulty dealer during protocol S. The second lemma describes possible malicious behaviour by another party in protocol R.

In both of the described attacks, parties A, B and D communicate freely in a synchronous manner throughout protocol S and party C does not send or receive any messages. By communicating in a synchronous manner we mean that we proceed in "rounds" consisting of waiting until parties complete their computation and send messages. Then, once parties sent all of the messages in a given round, all messages are delivered and the next round begins.[3] As will be shown below, in this setting parties A, B and D must complete protocol S. After parties A, B and D have completed protocol S, party C receives all messages from parties A and B, but does not receive messages from party D. Then, parties A, B and C run protocol R with their communication being conducted in a synchronous manner. Again, as shown below, parties A, B and C must complete protocol R in this setting

[3] In order to avoid extreme cases of unending computation, a maximal number of computation steps can be imposed on each party in each round.

as well. All messages to and from D are delayed throughout protocol R until all three parties complete the protocol. After parties A, B and C complete protocol R, all messages are delivered without delay. Note that the behaviour in protocol S is consistent with party C being crashed. Furthermore, the behaviour in protocol R is consistent with party C being delayed earlier, and party D being malicious or crashed.

Before we state the first lemma we define a distribution of views where the system is synchronous, the dealer D and parties A, B are nonfaulty and party C has crashed. In such a setting, from the Termination property of $AVSS$ all nonfaulty parties almost-surely complete protocol S. Set some $1 > \epsilon' > 0$ and let $N \in \mathbb{N}$ be a number such that if parties A, B, D participate in protocol S in the setting described above, the probability that one of them runs for more than N rounds throughout protocol S is smaller than ϵ'. From the Termination property of the protocol, and since the setting is synchronous, such a value N must exist. Define $long$ to be the event in which either A, B or D run for longer than N rounds throughout protocol S, and $\overline{long}$ to be its complement.

Definition 2 For every $s \in \{0, 1\}$, $P \in \{A, B\}$, let $\pi_{s,P}$ be the distribution of P's view when a nonfaulty D shares the value s and party C is faulty and silent, conditioned upon the event $\overline{long}$.

In the attack described in the first lemma, the dealer, D has a nonzero probability of causing parties A and B to complete protocol S, but the distribution of party A's view conditioned upon the attack's success is $\pi_{0,A}$, while the conditional distribution of party B's view is $\pi_{1,B}$.

Lemma 1 *A faulty dealer D has a strategy such that with some nonzero probability the following event holds:*

1. *Parties A and B complete protocol S.*
2. *The conditional distribution of party A's view is $\pi_{0,A}$.*
3. *The conditional distribution of party B's view is $\pi_{1,B}$.*

Intuitively, the dealer will try to make party A and party B complete protocol S while seeing contradictory worldviews. Conditioned on this nonzero probability event the following happens: on the one hand, in A's view, the execution of S looks like one in which D shared the value 0 and C was faulty and silent (corresponding to the distribution $\pi_{0,A}$). On the other hand, in B's view the execution of S looks like one in which D shared the value 1 and C was faulty and silent (corresponding to the distribution $\pi_{1,B}$). After A and B complete protocol S, C starts participating in the protocol, and completes protocol S as well. Then all three parties participate in protocol R, and eventually complete it and output some value r. This value r will be used in the next attack in the second lemma.

The crux of this attack is that parties A and B should not know whether 0 or 1 is shared during S in a run in which D is nonfaulty because of the Secrecy property. Leveraging this ambiguity, the faulty dealer tries to make them complete S with incompatible views, which they will have to resolve during protocol R.

Afterwards, in order to prove the main result, we prove the following lemma for every $1 > \epsilon' > 0$ (arbitrarily close to 0):

Lemma 2 *Without loss of generality, the adversary has a strategy controlling party B such that when a nonfaulty D shares the value 0, with probability at least $1 - \epsilon'$:*

- *A's view during protocol S is distributed according to the distribution $\pi_{0,A}$,*
- *C outputs 0 at the end of protocol R with probability $\frac{1}{2}$ or less.*

The value 0 in the lemma is used when in the previous attack party C outputs the value $r = 0$ with probability $\frac{1}{2}$ or less. The claim is "without loss of generality", in the sense that if C outputs the value $r = 1$ with probability $\frac{1}{2}$ or less then we switch A with B as well as 0 with 1.

In this attack, B acts normally throughout S, and then throughout R acts as if the attack in Lemma 1 took place. As opposed to the previous attack which can succeed with some tiny nonzero probability, B's attack succeeds (i.e. it looks as if the attack in Lemma 1 took place) with a probability arbitrarily close to 1. This means that the event that C outputs 0 with probability $\frac{1}{2}$ or less can occur with a probability arbitrarily close to 1. Note that if the dealer shares 0, then every nonfaulty party should output 0 with probability $\frac{1}{2} + \epsilon$ or greater, and thus C can only fail to output 0 with probability $\frac{1}{2} - \epsilon$. Therefore, for a small enough ϵ' we reach a contradiction. This concludes the proof for Theorem 2.

Several random variables are defined in order to prove the aforementioned lemmas. Technically, the distribution of the random variables could depend on the dealer, the adversary and on the message scheduling. Throughout all of the analysis these factors are strictly defined, and are therefore omitted from the definitions of the random variables.

Let M^s_{XY} be the distribution of messages exchanged between party X and party Y during the sharing protocol S if the network is synchronous, party D is a nonfaulty dealer sharing the value s, party C is faulty and silent, and no nonfaulty party calls protocol R. Let R^s_X be the distribution of the internal randomness of party X throughout the sharing protocol in the described setting. Let V^s_X be the distribution of party X's view of the share phase in the setting described above. For every pair of parties X, Y and value s let $r^s_X \sim R^s_X$ be a random variable describing X's randomness in a given run, $m^s_{XY} \sim M^s_{XY}$ be a random variable describing the messages between parties X and Y with a nonfaulty dealer sharing s in the described setting, and $v^s_X \sim V^s_X$ be a random variable

describing X's view in the run. Note that r_X^s is necessarily part of v_X^s in some way, as well as m_{XY}^s for every party Y.

For any distribution $\mathcal{D}$, $x \in \mathcal{D}$ means that x has a nonzero probability under $\mathcal{D}$. Party X's view v_X is *consistent* with s if $v_X \in V_X^s$. Similarly, a set of messages m_{XY} exchanged between party X and party Y is *consistent* with the secret s if $m_{XY} \in M_{XY}^s$.

For the first part of the analysis, assume D is corrupted by the adversary. The overarching goal is to prove Lemma 1 while party A sees a view consistent with D sharing the value 0 and party B sees a view consistent with D sharing the value 1. Intuitively, from the Secrecy property, neither party A nor party B should be able to tell which value was shared throughout protocol S, and thus D could send messages in that manner and neither party would notice.

The adversary's strategy is as follows: Party D samples $s_A \leftarrow R_A^0 | \overline{long}$ and $s_{AB} \leftarrow M_{AB}^0 | r_A^0 = s_A, \overline{long}$ and then $s_B \leftarrow R_B^1 | m_{AB}^1 = s_{AB}, \overline{long}$. Afterwards it samples $s_{AD} \leftarrow M_{AD}^0 | m_{AB}^0 = s_{AB}, r_A^0 = s_A, \overline{long}$ and $m_{BD} \leftarrow M_{BD}^1 | m_{AB}^1 = s_{AB}, r_B^1 = s_B, \overline{long}$. The random variables s_A, s_B are D's guesses of A and B's randomness, and the variables s_{AB}, s_{AD}, s_{BD} are the messages D predicts will be sent. Define party X's randomness throughout this run to be r_X and the messages exchanged between parties X and Y in this run to be m_{XY}. Finally, define the event G, in which $s_A = r_A, s_B = r_B$.

Before showing this behaviour can be used as part of Lemma 1, we need to show that these distributions are well-defined and samplable. Note that the setting is entirely synchronous. For simplicity, assume that the number of bits in a message and the amount of randomness needed in each round are bounded. In addition, assume that every party sends some message indicating that it completes protocol S. These assumptions are used in order to simplify the proof of the next lemma.[4]

Lemma 3 *For every values* m'_{AD}, m'_{AB}, r'_A *such that* $\Pr[m_{AD}^0 = m'_{AD}, m_{AB}^0 = m'_{AB}, r_A^0 = r'_A, \overline{long}] \neq 0$*:*

$$\Pr[m_{AD}^0 = m'_{AD}, m_{AB}^0 = m'_{AB}, r_A^0 = r'_A | \overline{long}] = \Pr[m_{AD}^1 = m'_{AD}, m_{AB}^1 = m'_{AB}, r_A^1 = r'_A | \overline{long}]$$

[4] In order to prove the general case, the dealer can simulate the entire run for parties A, B, D round-by-round twice, once sharing the value 0 and once sharing the value 1. The dealer will only accept pairs of runs in which the messages exchanged between parties A and B are the same. Proving that there must exist such a pair of runs requires proving a lemma similar to the following lemma without conditioning upon the event $\overline{long}$. Note that since the rounds almost-surely terminate, the sampling process will also terminate with probability 1. This will result in slight differences in the attacks and proofs, but with very similar techniques and ideas. The main difference is that all of the sampled probabilities will not be conditioned upon the event $\overline{long}$.

Proof First note that from the Termination property of $AVSS$, if all nonfaulty parties participate in protocol S they will all complete it. Observe a scheduling in which the communication between parties A, B and D is synchronous, party C is silent throughout all of protocol S and no nonfaulty party calls protocol R at all. In this case, since no nonfaulty party calls protocol R, the Secrecy property must hold at the time the parties complete protocol S.

Seeking a contradiction, assume the lemma doesn't hold and show a violation of the Secrecy property. In that case, observe the scenario in which the adversary controls party A and the nonfaulty dealer shares the value 0. Party A acts like a nonfaulty party would in protocol S. From D and B's point of view, party A acts as a nonfaulty party and C acts as a faulty party which doesn't send any messages. Since they cannot distinguish between the scenarios, the messages must be distributed according to the distributions M_{AD}^0, M_{AB}^0 and A's randomness must be distributed according to R_A^0. Since $\Pr[m_{AD}^0 = m'_{AD}, m_{AB}^0 = m'_{AB}, r_A^0 = r'_A, \overline{long}] \neq 0$, there is a nonzero probability that parties A, B, D complete protocol S with A having exchanged those messages in fewer than N rounds. All parties also know when other parties complete the protocol because they send a message indicating they completed protocol S. No nonfaulty party called R yet, and $\Pr[m_{AD}^0 = m'_{AD}, m_{AB}^0 = m'_{AB}, r_A^0 = r'_A | \overline{long}] \neq \Pr[m_{AD}^1 = m'_{AD}, m_{AB}^1 = m'_{AB}, r_A^1 = r'_A | \overline{long}]$. In other words, if party A acted in the exact same way, and D were sharing the value $s = 1$ instead, A would have had a different probability of seeing these values if the event $\overline{long}$ took place. Since A's random values and the messages it exchanges are a part of its view, as well as its knowledge as to whether the event $long$ happened, this means that this adversary's view in the case $s = 0$ is distributed differently than it would be in the case $s = 1$ reaching a contradiction. For completeness, all messages to and from C can be sent and received after parties A, B and D complete protocol S in order for the scheduling to be valid. □

The probabilities are equal for every event with nonzero probability in the case that the dealer is sharing the value 0. Since both must be probability spaces $\Pr[m_{AD}^0 = m'_{AD}, m_{AB}^0 = m'_{AB}, r_A^0 = r'_A | \overline{long}] = 0$ if and only if it is also true that $\Pr[m_{AD}^1 = m'_{AD}, m_{AB}^1 = m'_{AB}, r_A^1 = r'_A | \overline{long}] = 0$, and thus the distributions must be identical. In addition, the exact same arguments can be made for B instead or if party D shared the value $s = 1$.

A direct corollary is that any of the marginal and conditional probabilities are also the same. For example:

Corollary 1 *For every* $m'_{AD} \in M_{AD}^0 | \overline{long}$*:*

$$\Pr[m_{AD}^0 = m'_{AD} | \overline{long}] = \Pr[m_{AD}^1 = m'_{AD} | \overline{long}].$$

Furthermore, for every $m'_{AB} \in M^0_{AB}|\overline{long}$:

$$\Pr[m^0_{AD} = m_{AD}|m^0_{AB} = m'_{AB}, \overline{long}] = \Pr[m^1_{AD} = m_{AD}|m^1_{AB} = m'_{AB}, \overline{long}].$$

Proof Each of these equalities is shown individually. Define the distributions $X^s = M^s_{AB}|\overline{long}$ and $Y^s = R^s_A|\overline{long}$.

$$\begin{aligned}
&\Pr[m^0_{AD} = m'_{AD}|\overline{long}] \\
&= \sum_{\substack{m'_{AB}\in X^0,\\ r'_A\in Y^0}} \Pr[m^0_{AD}=m'_{AD}, m^0_{AB}=m'_{AB}, r_A=r'_A|\overline{long}] \\
&= \sum_{\substack{m'_{AB}\in X^1,\\ r'_A\in Y^1}} \Pr[m^1_{AD}=m'_{AD}, m^1_{AB}=m'_{AB}, r_A=r'_A|\overline{long}] \\
&= \Pr[m^1_{AD} = m'_{AD}|\overline{long}]
\end{aligned}$$

Summing over $m'_{AB} \in X^0 = M^0_{AB}|\overline{long}$ is the same as summing over $m'_{AB} \in X^1 = M^1_{AB}|\overline{long}$ because for every $m'_{AB} \in M^0_{AB}|\overline{long}$ there must exist m'_{AD}, r'_A such that $\Pr[m^0_{AD} = m'_{AD}, m^0_{AB} = m'_{AB}, r^0_A = r'|\overline{long}] \neq 0$. From previous observations, this means that $\Pr[m^1_{AD} = m'_{AD}, m^1_{AB} = m'_{AB}, r^1_A = r'_A|\overline{long}] \neq 0$ and thus $m'_{AB} \in M^0_{AB}|\overline{long}$ as well. The same reasoning holds about r'_A. The argument also clearly works in reverse. This argument can also be made for any other subset of the three variables.

For the second property, note that $\Pr[m^0_{AB} = m'_{AB}|\overline{long}] \neq 0$ and thus the probability is well defined. In that case:

$$\begin{aligned}
&\Pr\left[m^0_{AD} = m'_{AD}|m^0_{AB} = m'_{AB}, \overline{long}\right] \\
&= \frac{\Pr\left[m^0_{AD} = m'_{AD}, m^0_{AB} = m'_{AB}|\overline{long}\right]}{\Pr\left[m^0_{AB} = m'_{AB}|\overline{long}\right]} \\
&= \frac{\Pr\left[m^1_{AD} = m'_{AD}, m^1_{AB} = m'_{AB}|\overline{long}\right]}{\Pr\left[m^1_{AB} = m'_{AB}|\overline{long}\right]} \\
&= \Pr\left[m^1_{AD} = m'_{AD}|m^1_{AB} = m'_{AB}, \overline{long}\right]
\end{aligned}$$

It is important to notice that all of those arguments could have been made with any subset of the three random variables described in the lemma. □

Corollary 2 *The values sampled by D in the described attack are sampled from well-defined, samplable distributions.*

Proof We go through each sampled value and check if the distribution is well-defined. First, D samples $s_A \leftarrow R^0_A|\overline{long}$. From the definitions of ϵ' and the corresponding N, the probability that A's view throughout protocol S is of length greater than N is no greater than ϵ'. This means that the event $\overline{long}$ happens with probability $1 - \epsilon' > 0$ at the very least, and thus there must also exist some value $r_A \in R^0_A|\overline{long}$. D then samples $s_{AB} \leftarrow M^0_{AB}|r^0_A = s_A, \overline{long}$. Since $\Pr\left[r^0_A = r_A|\overline{long}\right] \neq 0$, there must be some set of messages $m'_{AB} \in M^0_{AB}|\overline{long}$ such that $\Pr\left[m^0_{AB} = m'_{AB}, r^0_A = s_A|\overline{long}\right] \neq 0$ and thus the distribution is well defined. The argument for s_{AD} is identical. D then samples $s_B \leftarrow R^1_B|m^1_{AB} = s_{AB}, \overline{long}$. Following similar arguments, $\Pr\left[m^0_{AB} = s_{AB}|\overline{long}\right] \neq 0$ and thus from Corollary 1, $\Pr\left[m^1_{AB} = s_{AB}|\overline{long}\right] \neq 0$. Now, following similar arguments both s_B and s_{BD} are sampled from well-defined distributions. D can easily sample from these distributions by simulating all runs with parties A, B and D that take no more than N rounds to terminate. This is possible because of the assumption that the size of messages and randomness in each round is bounded. If that is not the case, D can simulate the protocol step by step and sample values that way. □

In general the strategy from this point on is to prove that if any party's view is consistent with some secret it must complete protocol S. The next step is to show that if event G occurs, A and B's view must be consistent with some secret s (not the same one) and the event $\overline{long}$ must take place. In the runs described by the random variables party C is faulty and doesn't send any messages, and thus we prove that there are runs in which parties A and B must complete protocol S even without receiving any messages from C.

Lemma 4 *If party A's view is consistent with some secret $s \in \{0, 1\}$ then it must almost-surely complete the protocol, even without receiving any messages from party C.*

Proof Since party A's view is consistent with the secret s, A could have had this exact view with a nonfaulty dealer D sharing s if C were faulty and silent. Since in that run A and B are nonfaulty and D is a nonfaulty dealer, from the Termination property of $AVSS$, A must almost-surely complete protocol S in that run. A can't tell the difference between its view in this run and the view in which C was faulty, and thus A must complete protocol S if its view is merely consistent with s. Party C's messages won't be infinitely delayed in this run because the probability that party A has an infinitely large view is 0. After completing protocol S, all of party C's messages can be delivered. □

This exact argument can be made for party B as well. We now turn to show that if D acts according to the described strategy, there is a nonzero probability that A and B complete protocol S with the desired distribution of views, conditioned upon the event $\overline{long}$.

Lemma 5 *If the dealer is faulty, $s_A = r_A, s_B = r_B$, D exchanges the messages s_{AD} and s_{BD} with parties A and B respectively, parties A and B exchange the messages s_{AB} between them, and the scheduling is as described above, then party A's view is distributed according to $V^0_A|\overline{long}$ and party B's view is distributed according to $V^1_B|\overline{long}$.*

Proof The random variable v_A^0 is defined to be party A's view during protocol S with a nonfaulty dealer D sharing the value $s = 0$, and a faulty C which remains silent. Since no messages are received from party C, A's view consists of $m_{AB}^0, m_{AD}^0, r_A^0$. In the run described in the lemma no messages are sent or received from party C either and thus party A's view consists of s_{AB}, s_{AD}, r_A. Technically the ordering could also matter, but note that the scheduling is deterministic and looks identical in both runs, so the order in which messages are received is ignored.

Observe some m'_{AB}, m'_{AD}, r'_A such that $\Pr[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A = r'_A|\overline{long}] \neq 0$:

$$\begin{aligned}
&\Pr\left[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A^0 = r'_A|\overline{long}\right] \\
&\quad = \Pr\left[m_{AD}^0 = m'_{AD}|m_{AB}^0 = m'_{AB}, r_A^0 = r'_A, \overline{long}\right] \\
&\qquad \times \Pr\left[m_{AB}^0 = m'_{AB}|r_A^0 = r'_A, \overline{long}\right] \cdot \Pr\left[r_A^0 = r'_A|\overline{long}\right]
\end{aligned}$$

On the other hand:

$$\begin{aligned}
&\Pr\left[s_{AB} = m'_{AB}, s_{AD} = m'_{AD}, r_A = r'_A|G\right] \\
&\quad = \Pr\left[s_{AD} = m'_{AD}|s_{AB} = m'_{AB}, r_A = r'_A, G\right] \\
&\qquad \times \Pr\left[s_{AB} = m'_{AB}|r_A = r'_A, G\right] \Pr\left[r_A = r'_A|G\right] \\
&\quad = \Pr\left[s_{AD} = m'_{AD}|s_{AB} = m'_{AB}, s_A = r'_A\right] \\
&\qquad \times \Pr\left[s_{AB} = m'_{AB}|s_A = r'_A\right] \Pr\left[s_A = r'_A\right] \\
&\quad = \Pr\left[m_{AD}^0 = m'_{AD}|m_{AB}^0 = m'_{AB}, r_A^0 = r'_A, \overline{long}\right] \\
&\qquad \times \Pr\left[m_{AB}^0 = m'_{AB}|r_A^0 = r'_A, \overline{long}\right] \Pr\left[r_A^0 = r'_A|\overline{long}\right] \\
&\quad = \Pr\left[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A^0 = r'_A|\overline{long}\right]
\end{aligned}$$

Where the second to last equality stems from the definitions of the random variables s_A, s_{AB}, s_{AD}.

The analysis for B's view can be done in a similar fashion, finding that:

$$\begin{aligned}
&\Pr\left[s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, r_B = r'_B|G\right] \\
&\quad = \Pr\left[s_{BD} = m'_{BD}|s_{AB} = m'_{AB}, r_B = r'_B, G\right] \\
&\qquad \times \Pr\left[r_B = r'_B|s_{AB} = m'_{AB}, G\right] \Pr\left[s_{AB} = m'_{AB}|G\right] \\
&\quad = \Pr\left[s_{BD} = m'_{BD}|s_{AB} = m'_{AB}, s_B = r'_B\right] \\
&\qquad \times \Pr\left[s_B = r'_B|s_{AB} = m'_{AB}\right] \Pr\left[s_{AB} = m'_{AB}|G\right] \\
&\quad = \Pr\left[m_{BD}^1 = m'_{BD}|m_{AB}^1 = m'_{AB}, r_B^1 = r'_B, \overline{long}\right] \\
&\qquad \times \Pr\left[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}\right] \Pr\left[s_{AB} = m'_{AB}|G\right]
\end{aligned}$$

Where the final equality stems from the definition of the random variables s_{BD}, s_B. Now observe the messages between parties A and B:

$$\begin{aligned}
&\Pr\left[s_{AB} = m'_{AB}|G\right] \\
&\quad = \sum_{r'_A \in R_A^0|\overline{long}} \Pr\left[s_{AB} = m'_{AB}|r_A = r'_A, G\right] \\
&\Pr\left[r_A = r'_A|G\right] \\
&\quad = \sum_{r'_A \in R_A^0|\overline{long}} \Pr\left[s_{AB} = m'_{AB}|s_A = r'_A\right] \Pr\left[s_A = r'_A\right] \\
&\quad = \sum_{r'_A \in R_A^0|\overline{long}} \Pr\left[m_{AB}^0 = m'_{AB}|r_A^0 = r'_A, \overline{long}\right] \\
&\Pr\left[r_A^0 = r'_A|\overline{long}\right] \\
&\quad = \Pr\left[m_{AB}^0 = m'_{AB}|\overline{long}\right] = \Pr\left[m_{AB}^1 = m'_{AB}|\overline{long}\right]
\end{aligned}$$

Where the third equality stems from the definitions of s_A and s_{AB}, and the last equality stems from Corollary 1. Completing the original analysis:

$$\begin{aligned}
&\Pr\left[s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, r_B = r'_B|G\right] \\
&\quad = \Pr\left[m_{BD}^1 = m'_{BD}|m_{AB}^1 = m'_{AB}, r_B^1 = r'_B, \overline{long}\right] \\
&\qquad \times \Pr\left[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}\right] \Pr\left[s_{AB} = m'_{AB}|G\right] \\
&\quad = \Pr\left[m_{BD}^1 = m'_{BD}|m_{AB}^1 = m'_{AB}, r_B^1 = r'_B, \overline{long}\right] \\
&\qquad \times \Pr\left[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}\right] \\
&\qquad \times \Pr\left[m_{AB}^1 = m'_{AB}|\overline{long}\right] \\
&\quad = \Pr\left[m_{AB}^1 = m'_{AB}, m_{BD}^1 = m'_{BD}, r_B^1 = r'_B|\overline{long}\right]
\end{aligned}$$

An equality holds for every event with nonzero probability and both views must define probability spaces, and thus the distributions must be the same. □

Lemma 6 *If the dealer is faulty, $s_A = r_A, s_B = r_B$, D sends messages to parties A and B according to s_{AD} and s_{BD}, and the scheduling is as described above, then parties A and B complete protocol S having exchanged s_{AD}, s_{BD} with D respectively and s_{AB} between them.*

Proof If party D correctly guesses $s_A = r_A, s_B = r_B$, then the messages A and B exchange with each other and with D in response to each of D's messages become entirely deterministic and dictated only by D's messages. This means that since D's messages are always going to be consistent with the sampled values s_{AD}, s_{BD}, parties A and B are going to send the appropriate responses to D, as well as exchange the messages s_{AB} sampled by D between them. In that case, from Lemma 5 party A's view is distributed according to $V_A^0|\overline{long}$ and party B's view is distributed according to $V_B^1|\overline{long}$. This means that party A's view is consistent with $s = 0$ and party

B's view is consistent with $s = 1$. From Lemma 4, parties A and B almost-surely complete protocol S in finite time. □

In order for the scheduling to be valid, once parties A and B complete protocol S, all messages to and from party C are instantly delivered. Note that party D hasn't sent any messages to party C. There is a nonzero probability of $s_A = r_A, s_B = r_B$, and thus Lemma 1 is proven by combining Lemmas 5 and 6.

Now observe the following behaviour and scheduling after protocol S: party D now stays silent throughout all of protocol R, and all of the messages to and from parties A, B and C are synchronously delivered. Since all nonfaulty parties participate in protocol S, and some nonfaulty party completed protocol S, all nonfaulty parties almost-surely complete it as well. Similarly, since all nonfaulty parties completed protocol S and participate in protocol R, they all almost-surely complete it as well. Define O_C to be the random variable describing the output of party C during these runs, conditioned upon the event G. In other words, only observe the runs in which party D correctly guessed the other parties' randomness. Now, it is either the case that $\Pr[O_C = 0] \leq \frac{1}{2}$ or the case that $\Pr[O_C = 1] \leq \frac{1}{2}$.

The rest of the section proves Lemma 2 by describing attacks in which the adversary controls either party A or party B and simulates the previous adversary's behaviour conditioned upon the event G. It is possible for the adversary to simulate that event with probability $1 - \epsilon'$ even though the event has a negligible probability of occurring in the original attack, gaining a significant advantage. First assume that $\Pr[O_C = 0] \leq \frac{1}{2}$. In that case, the adversary can control party B with some specific scheduling in such a way that if a nonfaulty dealer shares the value 0 and the event $\overline{long}$ takes place, party A's view throughout the protocol must be distributed according to $V_A^0 | \overline{long}$. Party B also acts in a way similar to the way it would have acted in the previous attack. This means that all parties act in the same way they would have acted in the original attack, and thus party C outputs 0 with probability $\frac{1}{2}$ or less if the event $\overline{long}$ takes place. Since the event $\overline{long}$ takes place with at least a probability of $1-\epsilon'$, this proves the lemma.

Lemma 7 *If* $\Pr[O_C = 0] \leq \frac{1}{2}$, *there exist an adversary controlling party* B *and a scheduling such that with probability* $1 - \epsilon'$ *or greater the following things hold when a nonfaulty dealer* D *shares the value* 0*:*

- *party* A*'s view during protocol* S *is distributed according to* $V_A^0 | \overline{long}$,
- *party* C *outputs* 0 *at the end of protocol* R *with probability* $\frac{1}{2}$ *or less.*

Proof The scheduling is described only when the dealer shares the value $s = 0$ and no party runs for longer than N rounds. Any other valid scheduling can take place if those conditions don't hold. The adversary takes control of party B, and makes it act as a nonfaulty party would throughout all of protocol S. All communications between parties A, B and D are synchronous throughout protocol S. In addition, all messages to and from C are delayed until parties A, B and D complete protocol S. After completing the protocol, messages between parties D and C are further delayed until parties A, B and C complete protocol R. Since party B is acting as a nonfaulty party would, parties A and D can't tell the difference between this run and a run in which party C is faulty and silent. As discussed above, in this setting parties A, B and D must complete protocol S.

Let $\hat{m}_{XY}$ be the messages party X exchanged with party Y throughout protocol S, and let $\hat{r}_X$ be party X's randomness throughout the protocol. After completing protocol S, party B simulates all runs in which $\overline{long}$ takes place when a nonfaulty dealer shares the value 1. If there is no such run in which the messages $\hat{m}_{AB}$ are exchanged between parties A and B, party B acts as a nonfaulty party throughout protocol R. Otherwise, party B uses its simulations in order to sample some random values $\hat{s}_B \leftarrow R_B^1 | m_{AB}^1 = \hat{m}_{AB}, \overline{long}$ and some messages $\hat{s}_{BD} \leftarrow M_{BD}^1 | m_{AB}^1 = \hat{m}_{AB}, r_B^1 = \hat{s}_B, \overline{long}$. Note that clearly $\Pr[m_{AB}^0 = \hat{m}_{AB} | \overline{long}] \neq 0$, and thus also $\Pr[m_{AB}^1 = \hat{m}_{AB} | \overline{long}] \neq 0$ from Corollary 1. This means that the above distributions are well-defined. From this point on, party B acts as a nonfaulty party would act with a view consisting of $\hat{m}_{AB}, \hat{s}_{BD}, \hat{s}_B$. After parties A and B complete protocol S all messages between parties A, B and C, including the messages previously sent, are synchronously delivered. All messages to and from party D are delayed until the rest of the parties complete protocol R. It is important to note that in this scheduling, all the messages party D sent to party C throughout protocol S are also delayed until parties A, B and C complete protocol R.

Recall that m_{XY} is defined as the messages exchange by parties X and Y and r_X is defined as X's randomness throughout the attack described in Lemma 1. Now observe a snapshot of the values party A saw throughout protocol S and the values party B claims it saw throughout the protocol. For any values r'_A, r'_B, m'_{AB}, m'_{BD}, m'_{AD} such that $\Pr[m_{AB} = m'_{AB}, m_{AD} = m'_{AD}, m_{BD} = m'_{BD}, r_A = r'_A, r_B = r'_B | G] \neq 0$ the following also holds:

$$
\begin{aligned}
&Pr[m_{AB} = m'_{AB}, m_{AD} = m'_{AD}, m_{BD} = m'_{BD}, r_A = r'_A,\\
&r_B = r'_B | G] = \Pr[s_{AB} = m'_{AB}, s_{AD} = m'_{AD}, s_{BD} = m'_{BD},\\
&r_A = r'_A, r_B = r'_B | G] = \Pr[s_{AB} = m'_{AB}, s_{AD} = m'_{AD},\\
&s_A = r'_A | G] \Pr[s_B = r'_B | s_{AB} = m'_{AB}, s_{AD} = m'_{AD},\\
&s_A = r'_A] \Pr[s_{BD} = m'_{BD} | s_{AB} = m'_{AB}, s_{AD} = m'_{AD},\\
&s_A = r'_A, s_B = r'_B] = \Pr[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD},
\end{aligned}
$$

$$r_A^0 = r'_A|\overline{long}]\Pr[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}]$$
$$\times \Pr[m_{BD}^1 = m'_{BD}|m_{AB}^1 = m'_{AB}, r_B^1 = r'_B, \overline{long}]$$

Where the last equality stems from several facts. From Lemma 5 $\Pr[s_{AB} = m'_{AB}, s_{AD} = m'_{AD}, r_A = r'_A|G] = \Pr[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A^0 = r'_A|\overline{long}]$. From the way the random variable s_B is sampled, given s_{AB} the variable s_B is independent of the variables s_{AD}, s_A. Now, $\Pr[s_B = r'_B|s_{AB} = m'_{AB}] = \Pr[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}]$ from the definition of s_B. A similar argument can be made for the final expression.

On the other hand, note that since parties A, B and D are acting as nonfaulty parties throughout S, their actions are distributed identically to the setting in which C is faulty and silent. In this setting, the event $\overline{long}$ takes place with probability $1-\epsilon'$ at the very least. If this event takes place, then party B sees that the messages $\hat{m}_{AB}$ can be exchanged in some run in which the event $\overline{long}$ takes place, and thus executes the attack and samples some values. Therefore, conditioned upon the event $\overline{long}$:

$$\Pr[\hat{m}_{AB} = m'_{AB}, \hat{m}_{AD} = m'_{AD}, \hat{s}_{BD} = m'_{BD}, \hat{r}_A = r'_A,$$
$$\hat{s}_B = r'_B|\overline{long}]$$
$$= \Pr[\hat{m}_{AB} = m'_{AB}, \hat{m}_{AD} = m'_{AD}, \hat{r}_A = r'_A|\overline{long}]$$
$$\times \Pr[\hat{s}_B = r'_B|\hat{m}_{AB} = m'_{AB}, \hat{m}_{AD} = m'_{AD},$$
$$\hat{r}_A = r'_A, \overline{long}]\Pr[\hat{m}_{BD} = m'_{BD}|\hat{m}_{AB} = m'_{AB},$$
$$\hat{m}_{AD} = m'_{AD}, \hat{r}_A = r'_A, \hat{s}_B = r'_B, \overline{long}]$$
$$= \Pr[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A^0 = r'_A|\overline{long}]$$
$$\times \Pr[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}]$$
$$\times \Pr[m_{BD}^1 = m'_{BD}|m_{AB}^1 = m'_{AB}, r_B^1 = r'_B, \overline{long}]$$

Where the last equality stems from similar arguments. First of all, note that from A's point of view, party C is acting like a faulty party which is staying silent throughout protocol S and parties B, D are acting as nonfaulty parties with D sharing the value 0. Therefore, $\Pr[\hat{m}_{AB} = m'_{AB}, \hat{m}_{AD} = m'_{AD}, \hat{r}_A = r'_A|\overline{long}] = \Pr[m_{AB}^0 = m'_{AB}, m_{AD}^0 = m'_{AD}, r_A^0 = r_A|\overline{long}]$. This also means that if the event $\overline{long}$ takes place, party A's view is distributed according to $V_A^0|\overline{long}$. From the way $\hat{s}_B$ is sampled, given $\hat{m}_{AB}$, the random variable $\hat{s}_B$ is entirely independent of $\hat{m}_{AD}, \hat{r}_A$. Taking that fact into consideration, and looking at the definition of $\hat{s}_B$, $\Pr[\hat{s}_B = r'_B|\hat{m}_{AB} = m'_{AB}, \hat{m}_{AD} = m'_{AD}, \hat{r}_A = r'_A, \overline{long}] = \Pr[r_B^1 = r'_B|m_{AB}^1 = m'_{AB}, \overline{long}]$. A similar argument can be made for $\hat{s}_{BD}$.

Party B's behaviour is identical to the behaviour it would have in the attack described in the first part, and party A's view is identical to that view as well. From this point on, protocol R is run in the exact same way, and neither party A nor party C can tell the difference between the runs in which party B was faulty and the event $\overline{long}$ occurred, and the runs in which party D was faulty, given that event G occurred. In the previous attack, parties A and C output some value before receiving messages from party D during protocol R, and thus must do so in this scenario as well. In order for the scheduling to be valid, all of the messages to and from party D are received some finite time after party A and party C output a value. The distribution of A and C's views in the beginning of protocol R is identical to the distribution of their views in the previous attack. Furthermore, party B's actions are defined by the view it is simulating in the beginning of protocol R as well. Since parties A, B and C's actions are determined by their view at any point in time, the distribution of their views throughout the rest of protocol R is identical in both runs as well, and thus the distributions of their outputs must be the same as well. Therefore, if the event $\overline{long}$ occurred, the probability that party C outputs 0 is $\frac{1}{2}$ or less. The event $\overline{long}$ takes place with probability $1-\epsilon'$ or more, which completes the lemma. □

Now assume that $\Pr[O_C = 1] \leq \frac{1}{2}$. In that case:

Lemma 8 *If* $\Pr[O_C = 1] \leq \frac{1}{2}$*, there exist an adversary controlling party A and a scheduling such that with probability* $1-\epsilon'$ *or greater the following things hold when a nonfaulty dealer D shares the value* 1*:*

- *party B's view during protocol S is distributed according to* $V_B^1|\overline{long}$*,*
- *party C outputs* 1 *at the end of protocol R with probability* $\frac{1}{2}$ *or less.*

Proof The scheduling is identical to the scheduling described in the previous lemma, and party A similarly acts as a nonfaulty party throughout all of protocol S. Following the exact same arguments, parties A, B and D must complete protocol S without party C sending or receiving any messages. Similarly define $\hat{m}_{XY}$ to be the messages party X and Y exchanged throughout protocol S, and $\hat{r}_X$ to be party X's randomness throughout the protocol.

After completing protocol S, party A simulates all runs in which $\overline{long}$ takes place when a nonfaulty dealer shares the value 0. If there is no such run in which the messages $\hat{m}_{AB}$ are exchanged between parties A and B, party A acts as a nonfaulty party throughout protocol R. Otherwise, using those simulations, party A samples random values $\hat{s}_A \leftarrow R_A^0|m_{AB}^0 = \hat{m}_{AB}, \overline{long}$ and messages $\hat{s}_{AD} \leftarrow M_{AD}^0|m_{AB}^0 = \hat{m}_{AB}, r_A^0 = \hat{s}_A, \overline{long}$. Note that in this case clearly $\Pr\left[m_{AB}^1 = m_{AB}|\overline{long}\right] \neq 0$, and thus also $\Pr\left[m_{AB}^0 = m_{AB}|\overline{long}\right] \neq 0$ from Corollary 1. This means that the above distributions are well-defined. From this point on, party A acts as a nonfaulty party would act with a view consisting of $\hat{m}_{AB}, \hat{s}_{AD}, \hat{s}_A$. The scheduling from this point on is identical to the scheduling described in the previous lemma.

Recall that m_{XY} is defined as the messages exchange by parties X and Y and r_x is defined as X's randomness throughout the attack described in Lemma 1. Now observe a snapshot of the values party A saw throughout protocol S and the values party B claims it saw throughout the protocol. For any values r'_A, r'_B, m'_{AB}, m'_{BD}, m'_{AD} such that $\Pr[m_{AB} = m'_{AB}, m_{AD} = m'_{AD}, m_{BD} = m'_{BD}, r_A = r'_A, r_B = r'_B|G] \neq 0$ first analyse the variable s_A. For the analysis, define the distribution $X = R^0_A|\overline{long}$:

$$\begin{aligned}
&\Pr\left[s_A = r'_A | s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, s_B = r'_B\right] \\
&= \Pr\left[s_A = r'_A | s_{AB} = m'_{AB}\right] \\
&= \frac{\Pr\left[s_{AB} = m'_{AB} | s_A = r'_A\right] \Pr\left[s_A = r'_A\right]}{\sum_{\bar{r}_A \in X} \Pr\left[s_{AB} = m'_{AB} | s_A = \bar{r}_A\right] \Pr\left[s_A = \bar{r}_A\right]} \\
&= \frac{\Pr\left[m^0_{AB} = m'_{AB} | r^0_A = r'_A, \overline{long}\right] \Pr\left[r^0_A = r'_A | \overline{long}\right]}{\sum_{\bar{r}_A \in X} \Pr[m^0_{AB} = m'_{AB} | r^0_A = \bar{r}_A, \overline{long}] \Pr\left[r^0_A = \bar{r}_A | \overline{long}\right]} \\
&= \Pr\left[r^0_A = r'_A | m^0_{AB} = m'_{AB}, \overline{long}\right]
\end{aligned}$$

Where the first equality stems from the fact that given s_A, s_{AB} is independent of s_{BD}, s_B, from which the reverse also follows. In addition, the third equality stems from the definition of s_{AB}. Now continue the analysis in a similar fashion to the one in the previous lemma:

$$\begin{aligned}
&\Pr[m_{AB} = m'_{AB}, m_{AD} = m'_{AD}, m_{BD} = m'_{BD}, \\
&\qquad r_A = r'_A, r_B = r'_B | G] \\
&= \Pr[s_{AB} = m'_{AB}, s_{AD} = m'_{AD}, s_{BD} = m'_{BD}, r_A = r'_A, \\
&\quad r_B = r'_B | G] \\
&= \Pr[s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, r_B = r'_B | G] \\
&\quad \times \Pr[s_A = r'_A | s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, s_B = r'_B] \\
&\quad \times \Pr[s_{AD} = m'_{AD} | s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, s_A = r'_A, \\
&\qquad s_B = r'_B] \\
&= \Pr[m^1_{AB} = m'_{AB}, m^1_{BD} = m'_{BD}, r^1_B = r'_B | \overline{long}] \\
&\quad \times \Pr[r^0_A = r'_A | m^0_{AB} = m'_{AB}, \overline{long}] \\
&\quad \times \Pr[m^0_{AD} = m'_{AD} | m^0_{AB} = m'_{AB}, r_A = r_A, \overline{long}]
\end{aligned}$$

Where the last equality stems from several facts. From Lemma 5, $\Pr[s_{AB} = m'_{AB}, s_{BD} = m'_{BD}, r_B = r'_B|G] = \Pr[m^1_{AB} = m'_{AB}, m^1_{BD} = m'_{BD}, r^1_B = r'_B|\overline{long}]$. From the definition of s_A, given s_{AB}, it is independent of the variables s_{BD}, S_B, and then from the way s_A is sampled $\Pr[s_A = r'_A | s_{AB} = m'_{AB}] = \Pr[r^0_A = r'_A | m^0_{AB} = m'_A B, \overline{long}]$. Also, from the definition of the random variable s_{AD}, given s_{AB} and s_A, the variable s_{AD} is independent of the variables s_{BD}, s_B, and then the equality stems from the definition of s_{AD} and from the previous analysis.

On the other hand, note that since parties A, B and D are acting as nonfaulty parties throughout S, their actions are distributed identically to the setting in which C is faulty and silent. In this setting, the event $\overline{long}$ takes place with probability $1 - \epsilon'$ at the very least. Note that if this event takes place, then party A sees that the messages $\hat{m}_{AB}$ can be exchanged in some run in which the event $\overline{long}$ takes place, and thus executes the attack and samples some values. Therefore, conditioned upon the event $\overline{long}$:

$$\begin{aligned}
&\Pr[\hat{m}_{AB} = m'_{AB}, \hat{s}_{AD} = m'_{AD}, \hat{m}_{BD} = m'_{BD}, \hat{s}_A = r'_A, \\
&\qquad \hat{r}_B = r'_B | \overline{long}] \\
&= \Pr[\hat{m}_{AB} = m'_{AB}, \hat{m}_{BD} = m'_{BD}, \hat{r}_B = r'_B | \overline{long}] \\
&\quad \times \Pr[\hat{s}_A = r'_A | \hat{m}_{AB} = m'_{AB}, \hat{m}_{BD} = m'_{BD}, \hat{r}_B \\
&= r'_B, \overline{long}] \Pr[\hat{s}_{AD} = m'_{AD} | \hat{m}_{AB} = m'_{AB}, \hat{m}_{BD} \\
&= m'_{BD}, \hat{s}_A = r'_A, \hat{r}_B = r'_B, \overline{long}] \\
&= \Pr[m^1_{AB} = m'_{AB}, m^1_{BD} = m'_{BD}, r^1_B = r'_B | \overline{long}] \\
&\quad \times \Pr[r^0_A = r'_A | m^0_{AB} = m'_{AB}, \overline{long}] \\
&\quad \times \Pr[m^0_{AD} = m'_{AD} | m^0_{AB} = m'_{AB}, r^0_A = r'_A, \overline{long}]
\end{aligned}$$

Where the last equality stems from similar arguments. First of all note that from B's point of view, party C is acting like a faulty party which is staying silent throughout protocol S and parties A, D are acting as nonfaulty parties with D sharing the value 1. Therefore, $\Pr[\hat{m}_{AB} = m'_{AB}, \hat{m}_{BD} = m'_{BD}, \hat{r}_B = r'_B|\overline{long}] = \Pr[m^1_{AB} = m'_{AB}, m^1_{BD} = m'_{BD}, r^1_B = r'_B|\overline{long}]$. This also means that if the event $\overline{long}$ occurs, party B's view is distributed according to $V^1_B|\overline{long}$. From the way $\hat{s}_A$ is sampled, given $\hat{m}_{AB}$, the random variable $\hat{s}_A$ is entirely independent of $\hat{m}_{BD}, \hat{r}_B$. Taking that into consideration, from the definition of $\hat{s}_A$, $\Pr[\hat{s}_A = r'_A | \hat{m}_{AB} = m'_{AB}, \hat{m}_{BD} = m'_{BD}, \hat{r}_B = r'_B, \overline{long}] = \Pr[r^0_A = r'_A | m^0_{AB} = m'_{AB}, \overline{long}]$. A similar argument can be made for $\hat{s}_{AD}$.

From this point on the rest of the argument is identical to the argument in the previous lemma, finding that if event $\overline{long}$ occurs, the probability that party B outputs 0 is $\frac{1}{2}$ or less. Since the event $\overline{long}$ occurs with probability $1 - \epsilon'$ at the very least, this completes the proof. □

If $\Pr[O_C = 0] \leq \frac{1}{2}$, Lemma 7 shows that Lemma 2 holds when $s = 0$ with the adversary controlling B. On the other hand, if $\Pr[O_C = 1] \leq \frac{1}{2}$, Lemma 8 shows that Lemma 2 holds when $s = 1$ with the adversary controlling A. Since either $\Pr[O_C = 0] \leq \frac{1}{2}$ or $\Pr[O_C = 1|G] \leq \frac{1}{2}$, Lemma 2 must hold. Assume w.l.o.g that $\Pr[O_C = 0] \leq \frac{1}{2}$. Then, if a nonfaulty dealer D shares the value 0, an adversary has a strategy controlling B such that for any $1 > \epsilon' > 0$ party C outputs 0 with probability no greater than $\frac{1}{2}$ if an event occurs with probability $1 - \epsilon'$ or more. If that event doesn't occur (with probability ϵ' or less), party C might output 0 with any

probability. So in total, the probability that C outputs 0 is no greater than $\left(1-\epsilon'\right)\cdot\frac{1}{2}+1\cdot\epsilon'$ All nonfaulty parties, including C, must output 0 with probability $\frac{1}{2}+\epsilon$ or more. Therefore, pick an ϵ' such that:

$$\left(1-\epsilon'\right)\cdot\frac{1}{2}+1\cdot\epsilon' < \frac{1}{2}+\epsilon$$
$$\frac{1}{2}-\frac{1}{2}\cdot\epsilon'+\epsilon' < \frac{1}{2}+\epsilon$$
$$\frac{1}{2}+\frac{1}{2}\cdot\epsilon' < \frac{1}{2}+\epsilon$$
$$\epsilon' < 2\epsilon$$

which reaches a contradiction, completing our proof.

2.3 Extending the impossibility result

This subsection completes the proof of the main theorem:

Theorem 1 *For any $\epsilon \in (0, \frac{1}{2}]$ and $n, t \in \mathbb{N}$ such that $4t \geq n > t$ there does not exist an almost-surely terminating $\left(\frac{1}{2}+\epsilon\right)$-correct t-resilient Byzantine AVSS protocol (S, R) for n parties.*

This is done by showing how to extend the result of Theorem 2 to multivalued secrets and to any n, t such that $4t \geq n > t$. In order to extend the proof to a multivalued secret, it is enough to note that any protocol in which the dealer can share values from some set V can be used for sharing binary values. For example, this can be done by mapping the possible values to the values 0 and 1 in some deterministic fashion. Using this mapping, all parties can participate in the protocol for multivalued secrets, and then convert their output to 0 or 1, achieving the wanted result. Clearly a nonfaulty dealer could have just been trusted to share the value 0 or 1 in the first place, but this technique also forces a faulty dealer to essentially share either 0 or 1 as well.

Extending the result to any $n, t \in \mathbb{N}$ such that $4t \geq n$ requires more intricate arguments. One could hope that it is enough to show impossibility for any n, t such that $4t \geq n \geq 3t+1$. That is because for any n, t such that $3t \geq n$, an adversary can choose to corrupt only t' parties such that $4t' \geq n \geq 3t'+1$ following the same strategy, and the impossibility result should hold in this case as well. However, if an adversary attempts to do that for $n = 2$ or $n = 3$, it may not corrupt any party for the inequality to hold. Therefore, separate arguments need to be made at least for $n = 2, t = 1$, and for $n = 3, t = 1$. The proofs in those cases are very similar to the general case in which $2t \geq n > t$ and $3t \geq n > 2t$, so the general proof is provided instead. First observe the case that $2t \geq n > t$.

Lemma 9 *For any $\epsilon \in (0, \frac{1}{2}]$, and $n, t \in \mathbb{N}$ such that $2t \geq n > t$ there does not exist an almost-surely terminating $(\frac{1}{2}+\epsilon)$-correct t-resilient Byzantine AVSS protocol (S, R) for n parties.*

Proof Assume by way of contradiction such a protocol exists. Assume that the dealer is nonfaulty and that the adversary corrupts $t \geq \frac{n}{2}$ parties. Throughout protocol S, the adversary instructs all parties to act as nonfaulty parties would. Since this setting is identical to one in which all parties are nonfaulty and are participating in the protocol, all parties must complete the protocol. Now, the adversary instructs the parties it controls to proceed immediately to protocol R, and all of the nonfaulty parties are delayed until the faulty parties complete the protocol. The exact same run could have taken place if the nonfaulty parties were faulty, acted correctly throughout protocol S, and then went silent in the beginning of protocol R. In that case, from the Termination property, all of the other parties must complete protocol R even if the faulty parties are silent. From the Correctness property, once the nonfaulty parties complete protocol S, every nonfaulty party that completes protocol R outputs s with probability $(\frac{1}{2}+\epsilon)$ or greater. The adversary instructs the faulty parties to act as nonfaulty parties would, and thus they all output s with the aforementioned probability. Therefore, the adversary can perform a test to distinguish between the case in which $s = 0$ and $s = 1$ with an advantage of 2ϵ at the very least, even before any nonfaulty party calls R. In other words, the adversary's view is not distributed independently of s even before some nonfaulty party calls R, contradicting the Secrecy property. □

Next observe the case of $n, t \in \mathbb{N}$ such that $3t \geq n > 2t$. The argument in this case closely follows ideas from the DLS lower bound [10]. As stated in [4], any $AVSS$ protocol trivially yields a reliable broadcast protocol with the same probability of termination and correctness. This is achieved by the dealer first sharing the value s, and all nonfaulty parties participating. Then, once a party completes protocol S it participates in protocol R, and outputs s with the desired probability. Therefore, the proof of impossibility closely resembles proofs of the impossibility of broadcast if $3t \geq n > 2t$.

Lemma 10 *For any $\epsilon \in (0, \frac{1}{2}]$ and $n, t \in \mathbb{N}$ such that $3t \geq n > 2t$, there is no almost-surely terminating $(\frac{1}{2}+\epsilon)$-correct t-resilient Byzantine AVSS protocol (S, R) for n parties.*

Proof Assume by way of contradiction such a protocol exists. Start by partitioning the parties into 3 sets, $P_A = \{P_1, \ldots, P_t\}$, $P_B = \{P_{t+1}, \ldots P_{2t}\}$ and $P_C = \{P_{2t+1}, \ldots, P_n\}$. Now, note that there are t parties in P_A and in P_B, and $n-2t$ parties in P_C. Assume that the dealer is P_n, and that the adversary controls all parties in P_C. It is important to note that $t \geq n-2t \geq 1$, so P_n is in the set P_C, and the adversary does not control more than t parties. Now, the adversary instructs

all of the parties in P_C to communicate with all parties in P_A as nonfaulty parties would with P_n sharing the value 0. At the same time, the adversary instructs all of the parties in P_C to communicate with all parties in P_B as nonfaulty parties would with P_n sharing the value 1. It is important to note that parties in P_C do not send messages to parties parties in P_A that they should have as a result of receiving messages from parties in P_B and vice-versa. All communication between parties in P_A and parties in P_B is delayed, and all other communication is delivered instantly.

Now, note that from the point of view in all parties in P_A, this run could have taken place if the parties in P_B were faulty and silent and all of the parties in P_A and P_C were nonfaulty. From the Termination property, this means that all parties in P_A and P_C must almost-surely complete the run of protocol S. For similar reasons, all parties in P_B and P_C must almost-surely complete their run of protocol S as well. Afterwards, all parties participate in protocol R, with the parties in P_C continuing to act as if they were nonfaulty with the dealer having shared the value 0 when communicating with parties in P_A. Similarly, they continue to act as if they were nonfaulty with the dealer having shared the value 1 when communicating with parties in P_B. Again, from the point of view of all parties in P_A, this run could have taken place if all parties in P_A and P_C were nonfaulty, the parties in P_B were faulty, and the dealer shared the value 0. Therefore, from the Termination property, all of the parties in P_A must complete protocol R, and from the Correctness property, they must output the value 0 in the end of it with probability $(\frac{1}{2}+\epsilon)$ or greater. Using the exact same argument, all parties in P_B must complete protocol R and output the value 1 in the end of it with probability $(\frac{1}{2}+\epsilon)$ or greater. After all parties in P_A and P_B complete protocol R, all of the delayed communication between them is delivered. Now, since some nonfaulty parties output 0 with probability $(\frac{1}{2}+\epsilon)$ or greater and some nonfaulty parties output 1 with probability $(\frac{1}{2}+\epsilon)$, there is no value r such that all nonfaulty parties output r with probability $(\frac{1}{2}+\epsilon)$ or greater, contradicting the Correctness property of $AVSS$.

Using slight simplification of the arguments, it is possible to show that no asynchronous reliable broadcast protocol exists with $(\frac{1}{2}+\epsilon)$ probability of success if $3t \geq n > 2t$. □

Finally, observe the case that $4t \geq n > 3t$. In order to prove that no almost-surely terminating $(\frac{1}{2}+\epsilon)$-correct t-resilient AVSS protocol exists, we show that if one exists, then such a protocol exists for $n = 4, t = 1$ as well. However, from Theorem 2 no such protocol exists, which proves Theorem 1. In other words, it is enough to prove the following lemma:

Lemma 11 *If for some $n, t \in \mathbb{N}$ such that $4t \geq n > 3t$ and $\epsilon \in (0, \frac{1}{2}]$ there exists an almost-surely terminating $(\frac{1}{2}+\epsilon)$-correct t-resilient Byzantine AVSS protocol (S, R) for n parties, then such a protocol exists for 4 parties, out of which 1 is faulty.*

Proof Assume that for some $\epsilon \in (0, \frac{1}{2}]$ and $n, t \in \mathbb{N}$ such that $4t \geq n > 3t$, there exists an almost-surely terminating $(\frac{1}{2}+\epsilon)$-correct t-resilient Byzantine $AVSS$ protocol (S, R) for n parties. Assume without loss of generality that P_n is the dealer in this protocol. Now we will construct a protocol for 4 parties, A, B, C, D such that D is the dealer and one party is faulty. Partition all of the parties into 4 sets $P_A = \{P_1, \ldots, P_t\}$, $P_B = \{P_{t+1}, \ldots, P_{2t}\}$, $P_C = \{P_{2t+1}, \ldots, P_{3t}\}$, $P_D = \{P_{3t+1}, \ldots, P_n\}$. Intuitively, party X is responsible for simulating all of the parties in P_X. This simulation should be constructed in such a way that all of the runs in the original n-party protocol are mapped to equivalent runs in the 4-party protocol.

Note that in this setting, adversaries can only choose to corrupt one of the 4 P_X sets in the n-party protocol. The sets P_A, P_B, P_C are each of size t, and the set P_D is of size $n-3t < t$, meaning that in each of those cases the adversary simulates no more than t corrupt parties. Furthermore, $n - 3t > 0$, so A, B, C and D each simulate at least one party. Clearly if no such protocol can exists when dealing with an adversary limited to corrupting one of the P_X sets, no such protocol exists when the adversary has free rein to choose any subset of t parties to corrupt.

The simulation takes place as follows: the parties A, B, C and D each internally run all of the parties in the sets P_A, P_B, P_C and P_D respectively. Party D is the dealer in the protocol, so it receives some input s. D runs the dealer P_n with the input s as well. When in the simulation some $P_i \in P_X$ sends a message m to some party $P_j \in P_X$ (i.e. one party that X simulates sends a message to another party that X simulates), party X sees that event taking place and continues simulating P_j after having received the message m from P_i. When in the simulation some $P_i \in P_X$ sends a message m to some party $P_j \in P_Y$ such that $X \neq Y$ (i.e. one party that X simulates sends a message to a party that Y simulates), party X sees that event taking place, and sends P_Y the message (m, i, j). Now, if P_Y receives a message (m, i, j) from P_X such that $P_i \in P_X$ and $P_j \in P_Y$, then Y continues simulating P_j after having received the message m from P_i. Finally, when party X sees that all of the parties in P_X completed protocol S it completes protocol S. In addition, when party X sees that all of the parties in P_X completed protocol R, it sets o_i to be P_i's output for every $P_i \in P_X$,outputs majority$_{P_i \in P_X}\{o_i\}$, breaking ties arbitrarily, and terminates.

Now observe all possible runs in the simulation by parties A, B, C and D. Those runs map directly to all possible runs between parties $P_1, \ldots, P_n$ where the adversary controls one of the sets P_X of size no greater than t, and the scheduling is exactly the scheduling in which the parties were simulated to receive the messages. Since there is a direct one-to-one

mapping between runs in the original protocol and in the simulation, the probability that all of the $AVSS$ properties hold in the simulated run is exactly the same as the probability in the original run. Let parties X, Y, Z be the nonfaulty parties. The fact that each property holds in the new protocol is shown individually:

Termination If D is nonfaulty and all nonfaulty parties participate in S, then P_n, the dealer in the simulated protocol, is nonfaulty and all simulated nonfaulty parties participate in the protocol as well. In that case, all parties in P_X, P_Y, P_Z almost-surely complete the protocol in the simulation, after which parties X, Y, Z see that the parties they simulate completed protocol S and complete it as well. Similarly, assume without loss of generality that party X completed the protocol. In that case, it saw that all of the parties in P_X completed protocol S. If party Y participates in protocol S, then all nonfaulty parties in P_Y participate in the simulated run of protocol S and almost-surely complete the simulated protocol as well, at which point Y completes protocol S. The same holds for party Z. Finally, if parties X, Y and Z complete protocol S and start protocol R, then in the simulation all parties in P_X, P_Y, P_Z complete protocol S and start protocol R. Those are all of the nonfaulty parties in the simulation, so all of them complete protocol R as well. After that, parties X, Y and Z see that all of the parties they simulate completed the protocol R, perform some local computations, and complete protocol R as well.

Correctness Assume the first nonfaulty party to complete protocol S is X. Before completing protocol S, it sees that all of the parties in P_X completed protocol S in the simulated run as well. At that time, there exists some value $r \in \mathcal{F}$ such that with probability $(\frac{1}{2}+\epsilon)$ or greater, all parties in P_X, P_Y, P_Z that complete protocol R output r. Furthermore, if D is nonfaulty, then so is the simulated dealer P_n, in which case $r = s$. Now if some nonfaulty party Y completes protocol R, it first saw that all of the parties in P_Y completed the protocol and output some values. As noted above, with probability $(\frac{1}{2}+\epsilon)$ all parties in P_Y output r. If that happens, Y sees that the value output by the majority is r and outputs r as well.

Secrecy Let the faulty party be F. If $F = D$, then the secrecy property holds trivially. Otherwise, P_n, the dealer in the simulated protocol, is not in P_F. F's view consists of the view of all of the parties it simulates in P_F. In the simulated protocol, the adversary's view is distributed independently of the secret s because the adversary doesn't control the dealer. In other words, the collective view of all parties in P_F is distributed independently of s, completing the proof. □

3 Strong common coin

The main goal of this section is to construct a strong common coin primitive. This primitive is defined as follows:

Definition 3 Protocol CC is an ϵ-biased almost-surely terminating common coin protocol if the following properties hold:

1. *Termination* If all nonfaulty parties participate in the CC protocol they almost-surely complete it. Furthermore, if some nonfaulty party completes protocol CC, every nonfaulty party that begins the protocol almost-surely completes it as well.
2. *Correctness* For every value $b \in \{0, 1\}$, there is at least a $\frac{1}{2}-\epsilon$ probability that every nonfaulty party that completes the protocol outputs b. Regardless, all nonfaulty parties that complete the protocol output the same value with probability 1.

This definition has three natural desired properties of a common coin protocol: the protocol almost-surely terminates, it has an arbitrarily small bias (as a parameter of the protocol), and the output value is always agreed upon by all parties. Previous works have achieved some subset of those properties, but not all three together. For example, the protocol in [9] doesn't always terminate and the parties don't always agree on the output value. On the other hand, the protocol described in [2] always terminates, but can completely fail $O\left(n^2\right)$ times. This also means that if just one common coin instance is required, there is no guarantee that the protocol will yield the desired properties.

Throughout the following sections assume the number of nonfaulty parties is t such that $3t+1 \leq n$. The following protocols use the protocols $SVSS$ and BA, which are resilient to this number of faulty parties. The Shunning Verifiable Secret Sharing protocol ($SVSS$), as defined in [2], has a designated dealer with some input s and it consists of two sub-protocols, $SVSS - Share$ and $SVSS - Rec$.

Definition 4 A pair $(SVSS - Share, SVSS - Rec)$ is said to be an $SVSS$ protocol if it has the following properties:

1. *Validity of termination* If a nonfaulty dealer initiates $SVSS - Share$ and all nonfaulty parties participate in the protocol, then every nonfaulty party eventually completes $SVSS - Share$.
2. *Termination* If a nonfaulty party completes either protocol $SVSS - Share$ or $SVSS - Rec$, then all nonfaulty parties that participate in the protocol eventually complete it. Moreover, if all nonfaulty parties begin protocol $SVSS - Rec$, then all nonfaulty parties eventually complete protocol $SVSS - Rec$.
3. *Binding* Once the first nonfaulty party completes an invocation of $SVSS - Share$ with session id (c, d), there is a value r such that either:
   - the output of each nonfaulty party that completes protocol $SVSS - Rec$ is r; or

- there exists a nonfaulty party P_i and a faulty party P_j such that P_j is shunned by P_i starting in session (c, d).

4. *Validity* If the dealer is nonfaulty with input s, then the binding property holds with $r = s$.
5. *Hiding* If the dealer is nonfaulty and no nonfaulty party invokes protocol $SVSS - Rec$, then the faulty parties learn nothing about the dealer's value.

Party P_i shuns party P_j if it accepted messages from it in the current invocation, but won't accept any messages from it in future interactions. For our purposes it is enough to note that fewer than n^2 shunning events can take place overall.

Definition 5 A protocol is said to be an almost-surely terminating binary Asynchronous Byzantine Agreement protocol if each nonfaulty party has an input from {0, 1}, and the following properties hold:

1. *Termination* If all nonfaulty parties participate in the protocol, all nonfaulty parties almost-surely eventually complete the protocol. Furthermore, if some nonfaulty party completes the protocol, all nonfaulty parties that participate in it do so as well.
2. *Validity* If all nonfaulty parties have the same input $\sigma \in \{0, 1\}$, every nonfaulty party that completes the protocol outputs σ.
3. *Correctness* All nonfaulty parties that complete the protocol output the same value $\sigma \in \{0, 1\}$.

Let $SVSS$ be a protocol with the $SVSS$ properties, and BA be an almost-surely terminating binary Asynchronous Byzantine Agreement protocol, as described in [2]. Both of these protocols are resilient to t faulty parties such that $3t + 1 \leq n$.

In addition to these two protocols, the common coin protocol requires a protocol for agreeing on a common subset of parties for which some condition holds. In order to do that, in the protocol each party P_i employs a "dynamic predicate" Q_{ir} for each round r. Intuitively $Q_{ir}(j)$ denotes whether P_i saw that some irreversible condition holds with regard to P_j. For every value $j \in [n]$, $Q_{ir}(j) \in \{0, 1\}$ at any given point in time. Initially, $\forall j \in [n]\ Q_{ir}(j) = 0$, and for any such j, $Q_{ir}(j)$ can turn into 1, but not back to 0. We generally think of these conditions as ones that "spread" in the following manner: if some nonfaulty party P_i sees that $Q_{ir}(k) = 1$, then eventually every nonfaulty party P_j also sees that $Q_{jr}(k) = 1$. The idea of a dynamic predicate and for the protocol below are described in [7].

Definition 6 Protocol CS is a common subset protocol, with a dynamic predicate Q_i and a number $k \leq n$ as input, if it has the following properties:

1. *Termination* Assume that for every pair of nonfaulty parties P_i, P_j that participate in the CS protocol and value $k \in [n]$ if $Q_i(k) = 1$ then eventually $Q_j(k) = 1$. If all nonfaulty parties invoke the protocol, and there exists a set $I \subseteq [n]$ such that:
   - $|I| \geq k$, and
   - for every nonfaulty party P_i, eventually $\forall j \in I\ Q_i(j) = 1$,

   then all nonfaulty parties almost-surely complete the invocation of CS. Furthermore, if some nonfaulty party completes protocol CS, then every nonfaulty party that participates in the protocol almost-surely completes it as well.
2. *Correctness* All nonfaulty parties that complete an invocation of CS output the same set $S \subseteq [n]$. Furthermore, $|S| \geq k$ and for every $j \in S$ there exists a nonfaulty party P_i such that $Q_i(j) = 1$.

A construction of a common subset protocol resilient to t faulty parties such that $3t + 1 \leq n$ is shown and proven with slight changes in [7]. For completeness, another construction and proof with the aforementioned properties is provided below.

Algorithm 1 $CommonSubset_r(Q_{ir}, k)$

Code for P_i:

1. Initialize $c_{ir} = 0$.
2. For every $j \in [n]$, once $Q_{ir}(j)$ becomes 1, if $c_{ir} < k$, begin BA_{jr} with input 1.
3. If at any point BA_{jr} terminates with output 1 for any $j \in [n]$, set $c_{ir} = c_{ir} + 1$.
4. Once $c_{ir} \geq k$, begin BA_{jr} with input 0 for every $j \in [n]$ such that $Q_{ir}(j) = 0$ at this point in time.
5. Denote b_{jr} to be the output of BA_{jr}. Output $\{j | b_{jr} = 1\}$.
6. Continue participating in BA_{jr} for every $j \in [n]$ until they terminate even after completing this invocation of $CommonSubset_r$.

Note that throughout this discussion we assume $k \leq n$ and $3t + 1 \leq n$.

Lemma 12 *Assume there exists a set $I \subseteq [n]$, such that $|I| \geq k$ and that for every nonfaulty party P_i, eventually for every $j \in I$, $Q_{ir}(j) = 1$. In addition, assume that for every pair of nonfaulty parties P_i, P_j that participate in the $CommonSubset_r$ protocol and value $k \in [n]$ if $Q_i(k) = 1$ then eventually $Q_j(k) = 1$. If all nonfaulty parties invoke $CommonSubset_r$ for a given r, then at least k invocations of BA_{jr} almost-surely terminate with output 1.*

Proof Note that c_{ir} is incremented only when BA_{jr} terminates with output 1. In addition, every nonfaulty party P_i inputs 0 to any BA_{jr} invocation only after having $c_{ir} \geq k$. This means that if some nonfaulty party inputs 0 to some

invocation of BA_{jr} then it must have completed at least k prior invocations with output 1. From the Validity property of the BA_{jr} protocol, at least one nonfaulty party P_i input the value 1 to the protocol, and this only happens if it sees that $Q_{ir}(j) = 1$. By assumption, every other nonfaulty party P_l that participates in the $CommonSubset_r$ protocol eventually sees that $Q_{lr}(j) = 1$ and begins the BA_{jr} protocol with input 1 if it hasn't done so earlier with the input 0. From the Correctness property of protocol BA, every other nonfaulty party also outputs 1 for the same invocations of BA, which proves our lemma. Therefore, assume no nonfaulty party inputs the value 0 to any invocation of BA_{jr} ever for any $j \in [n]$. In that case, every nonfaulty party P_i invokes $CommonSubset_r$, and eventually for every $j \in I$ $Q_{ir}(j) = 1$. Every nonfaulty party then begins participating in BA_{jr} with input 1 for every $j \in I$. From the Validity and Termination properties of BA all nonfaulty parties almost-surely complete those invocations of BA_{jr} with output 1. Since $|I| \geq k$, this completes the proof. □

Theorem 3 *The CommonSubset protocol is a common subset protocol resilient to any number of faulty parties t such that* $3t + 1 \leq n$ *and* $k \leq n$.

Proof Each property is proven separately.

Correctness If two nonfaulty parties P_i, P_l complete $CommonSubset_r$ then they must have completed BA_{jr} for every $j \in [n]$. From the Correctness property of BA, they completed each of those invocation with the same output b_{jr} and thus both output $S_r = \{j | b_{jr} = 1\}$. Next, show that for every $j \in S_r$, $Q_{ir}(j) = 1$ for at least one nonfaulty party P_i. Assume by way of contradiction $Q_{ir}(j) = 0$ for every nonfaulty party P_i for some $j \in S_r$. If that is the case, and some nonfaulty party completed $CommonSubset_r$, every nonfaulty party that participated in BA_{jr} at that point must have input 0. From those parties' point of view, this run is identical to one in which all nonfaulty parties' inputs are 0, and some might be slow. From the Validity property of BA, all nonfaulty parties must have then output 0 in BA_{jr}. However, in that case $b_{jr} \neq 1$, and thus $j \notin S_r$ reaching a contradiction. Finally, show that $|S_r| \geq k$. Assume by way of contradiction $|S_r| < k$. In that case, all parties completed all invocations of BA_{jr}, with at most $k - 1$ terminating with output 1. Since nonfaulty parties increment c_{ir} exactly once for every BA session that outputs the value 1, this means that for every nonfaulty party P_i, $c_{ir} < k$. Since $k \leq n$, BA_{jr} terminated with output 0 for at least one $j \in [n]$. Observe BA_{jr} for that j. Nonfaulty parties participate in any BA_{jr} session only if either $Q_{ir}(j) = 1$ or $c_i \geq k$. Since $c_i < k$, $Q_{ir}(j)$ must equal 1 at the time of invoking BA_{jr} for every nonfaulty party P_i, and P_i input 1 to the BA_{jr} call. From the Validity property of BA_{jr}, all nonfaulty parties must output 1 in BA_{jr} reaching a contradiction.

Termination First assume that all nonfaulty parties participate in the protocol, and that there exists some set $I \subseteq [n]$ such that $|I| \geq k$, and that for every nonfaulty party P_i and $j \in I$ eventually $Q_{ir}(j) = 1$ almost-surely. From Lemma 12, all nonfaulty parties almost-surely eventually complete at least k invocations of BA_{jr} with output 1. At that point, $c_{ir} \geq k$ holds for every nonfaulty party P_i. Because of line 4, every nonfaulty party P_i participates in BA_{jr} for every $j \in [n]$ such that $Q_{ir}(j) = 0$ at that point in time. It is important to note that if $Q_{ir}(j) \neq 0$ then it must equal 1, which means that P_i has already invoked BA_{jr} with input 1 previously. In other words, all nonfaulty parties have invoked BA_{jr} for every $j \in [n]$, so from the Termination property of BA they almost-surely complete all of those invocations. At that point they reach line 6 of the protocol, and complete $CommonSubset_r$.

For the second part of the property observe some nonfaulty party P_l that participates in $CommonSubset_r$. If some nonfaulty party P_i completed the protocol, it must have completed the BA_{jr} invocation for every $j \in [n]$. Let S_r be P_i's output in this invocation of the $CommonSubset_r$ protocol. From the Correctness property of $CommonSubset_r$, for every $j \in S_r$, $Q_{kr}(j) = 1$ for some nonfaulty party P_k. Since for some nonfaulty party P_k $Q_{kr}(j) = 1$, by assumption eventually $Q_{lr}(j) = 1$ as well. At that point, if P_l hasn't started participating in BA_{jr} with input 0, it starts participating in it with input 1. P_i completed each of those BA invocations, so from the Termination property of BA, P_l almost-surely completes them as well. Note that after completing the $CommonSubset_r$ invocation, all nonfaulty parties continue participating in all relevant BA invocations until they terminate. From the Correctness property of BA, party P_l outputs 1 in every BA_{jr} invocation such that $j \in S_r$ because P_i must have output 1 in that invocation as well. From the Correctness Property of $CommonSubset_r$, $|S| \geq k$ and thus at that point $c_{lr} \geq k$. At that point, P_l inputs 0 to every BA invocation it hasn't started participating in yet. Following similar arguments, from the Termination property of BA P_l almost-surely completes all of those invocations and then completes the protocol. □

Using the previously discussed primitives, the rest of this section describes and proves the Correctness of a common coin protocol. Intuitively, in the protocol several weak coins are flipped using the $SVSS$ protocol. These coins should behave as fully unbiased coin in most cases, but n^2 of these coins could fail because the $SVSS$ protocol could fail n^2 times. In this context a coin failing means that it can be totally biased, or not agreed upon. This means that enough weak coins need to be flipped so that the n^2 failures are not significant. The number of weak coin flips is set to be proportional to n^4 and to a function of the acceptable bias in the final coin, and each party outputs the value it saw in a

majority of the rounds. From the properties of the binomial distribution the n^2 faulty coin flips should not significantly bias the result given that around n^4 coins are flipped.

Algorithm 2 $CoinFlip\ (\epsilon)$

Code for P_i:

- Let $k = 4\left\lceil \left(\frac{e}{\epsilon \cdot \pi}\right)^2 n^4 \right\rceil$
- For r = 1 to k:
  1. Sample $b_{ir} \leftarrow \{0, 1\}$ uniformly. Call $SVSS - Share_{ir}\ (b_{ir})$ as dealer.
  2. Participate in $SVSS - Share_{jr}$ with P_j as dealer for every $j \in [n]$.
     Note this means the party begins participating in iteration r's $SVSS - Share$ invocations only after completing iteration $r - 1$.
  3. Define the dynamic predicate Q_{ir} as follows for every $j \in [n]$:
     $$Q_{ir}(j) = \begin{cases} 1 & SVSS - Share_{jr} \text{ has been completed} \\ 0 & else \end{cases}$$
  4. Participate in $CommonSubset_r\ (Q_{ir}, n - t)$, denote its output as S_{ir}.
  5. After $CommonSubset_r$ terminates, invoke $SVSS - Rec_{jr}$ for every $j \in S_{ir}$, let the reconstructed value be b_{ijr}.
  6. For every $j \in S_{ir}$ compute $b'_{ijr} = b_{ijr} \mod 2$.
     Compute $b'_{ir} = \bigoplus_{j \in S_{ir}} b'_{ijr}$ and continue to the next iteration.
- After completing the final iteration, compute $b'_i = \text{majority}_{r \in [k]} \{b'_{ir}\}$.
- Participate in a final BA invocation with input b'_i. After completing the BA invocation, output its output. In addition, continue participating in all relevant invocations of BA, $SVSS$ and $CommonSubset$ until they terminate.

Theorem 4 *For every $\epsilon \in \left(0, \frac{1}{2}\right)$ protocol $CoinFlip\ (\epsilon)$ is an ϵ-biased almost-surely terminating common coin protocol resilient to t faulty parties such that $3t + 1 \leq n$.*

Proof Each property is proven individually. Throughout this proof let $k = 4\left\lceil \left(\frac{e}{\epsilon \cdot \pi}\right)^2 n^4 \right\rceil$ as defined in the protocol.

Termination First show that if all nonfaulty parties participate in the $CoinFlip\ (\epsilon)$ protocol they all almost-surely complete it. In order to do that, we first show that if all nonfaulty parties start the r'th iteration of the loop in protocol $CoinFlip\ (\epsilon)$, then they all almost-surely complete it. Note that all nonfaulty parties continue participating in all calls to the $SVSS$ and $CommonSubset$ protocol until they terminate, so if all nonfaulty parties started participating in them, their Termination properties continue to hold. If all nonfaulty parties start the r'th iteration of $CoinFlip\ (\epsilon)$, every nonfaulty party P_i samples a random value b_{ir}, invokes $SVSS - Share_{ir}$ as dealer, and participates in $SVSS - Share_{jr}$ with P_j as dealer for every $j \in [n]$. From the Validity of Termination property of $SVSS$, since all nonfaulty parties participate in $SVSS - Share_{jr}$ for every nonfaulty dealer P_j, all nonfaulty parties eventually complete $SVSS - Share_{jr}$. Once party P_i completes $SVSS - Share_{jr}$, $Q_{ir}(j)$ becomes 1. This means that there exists a set $I \subseteq [n]$,such that $|I| \geq n - t$ and for every nonfaulty party P_i, eventually $\forall j \in I\ Q_{ir}(j) = 1$. In addition, let P_i, P_j be a pair of nonfaulty parties, and let $Q_{ir}(k) = 1$ for some $k \in [n]$. P_i only sets $Q_{ir}(k)$ to 1 if it completed the $SVSS - Share_{kr}$ call, and from the Termination property of the protocol, P_j eventually completes $SVSS - Share_{kr}$ and updated $Q_{jr}(k)$ to 1. In other words, all conditions of the Termination property of the $CommonSubset_r$ protocol hold. All nonfaulty parties participate in $CommonSubset_r$ because they started iteration r and continue participating in it even after completing $CoinFlip$ until the invocation of $CommonSubset_r$ terminates locally. From the Termination property of $CommonSubset$, all nonfaulty parties almost-surely complete $CommonSubset_r$. From the Correctness property of $CommonSubset$, for every $j \in S_r$, $Q_{ir}(j) = 1$ for at least one nonfaulty party P_i. This means that for every $j \in S_r$ at least one nonfaulty party completed $SVSS - Share_{jr}$. Therefore, from the Termination property of $SVSS$, all other nonfaulty parties complete $SVSS - Share_{jr}$ as well. After that, all nonfaulty parties reach step 5 of the iteration, and invoke $SVSS - Rec_{jr}$ for every $j \in S_r$. Again, from the Termination property of $SVSS$, all nonfaulty parties complete $SVSS - Rec_{jr}$ for every $j \in S_r$. After that, all nonfaulty parties perform local computations in step 6, and reach the end of the iteration.

Since all parties start with the same parameter ϵ they all compute the same value k. Note that this means that all nonfaulty parties begin the first iteration, and won't stop before completing the k'th iteration. Using a simple inductive argument and the previous claim, all nonfaulty parties almost-surely complete k iterations. After completing all k iteration, every nonfaulty party then performs a local computation and participates in the last BA invocation. From the Termination property of the BA protocol, all nonfaulty parties almost-surely complete that BA invocation, and then output its value and complete the protocol.

For the second part of the property, assume some nonfaulty party P_i completed the $CoinFlip$ protocol. Before completing the protocol, it must have completed the $CommonSubset_r$ protocol for every $r \in [k]$ and output some set S_r. It also completed the $SVSS - Rec_{jr}$ protocol for every $r \in [k]$, $j \in S_r$, and the final BA protocol. Now observe some other nonfaulty party P_l that participates in the protocol. For every nonfaulty party P_k and value $j \in [n]$, if $Q_{kr}(j) = 1$ it must have first completed the invocation of $SVSS - Share_{jr}$. From the Termination property of $SVSS$, every other nonfaulty party P_m that participates in $SVSS - Share_{jr}$ completes the protocol as well and sets $Q_{mr}(j) = 1$. Therefore, if P_l participates in $CommonSubset_r$ it almost-surely completes it as well, and from the Correctness property it outputs S_r as well. P_l then calls $SVSS - Rec_{jr}$ for every $j \in S_r$ and since those are the same invocations that P_i completed, P_l completes them

as well. This means that for every $r \in [k]$, P_l almost-surely completes $CommonSubset_r$ and $SVSS - Rec_{jr}$ for every $j \in S_r$, after which it continues to the next iteration. After completing all k iterations, P_l performs some local computations and participates in the BA protocol as well. Since P_i completed the BA protocol, P_l must almost-surely complete the protocol as well, and then complete the protocol.

Correctness Every nonfaulty party that completes the protocol participates in the final call to the BA protocol and outputs its output. From the Correctness property of BA, all nonfaulty parties output the same value in the BA protocol, and thus they all output the same value in the $CoinFlip$ protocol. This proves the second part of the property

We now turn to deal with the first part of the property. Every nonfaulty party that completes $CoinFlip$ must have completed all iterations of the loop in protocol $CoinFlip$. In each iteration, from the Correctness property of *Common Subset* there exists some set S_r such that every nonfaulty party that completes the call to $CommonSubset_r$ outputs S_r. From the Correctness property of $CommonSusbet$, at the time some party completes $CommonSubset_r$, for every $j \in S_r$ there exists some nonfaulty party P_i such that $Q_{ir}(j) = 1$. P_i only sets $Q_{ir}(j) = 1$ if it has has already completed $SVSS - Share_{jr}$. In other words, at the time some nonfaulty party completes $CommonSubset_r$ there exists some nonfaulty party that completes $SVSS - Share_{jr}$ for every $j \in S_r$. From the binding property of $SVSS$, at that time some value s'_{jr} is set such that every nonfaulty party that completes $SVSS - Rec_{jr}$ either outputs s'_{jr}, or some nonfaulty party shuns some faulty party starting in that $SVSS$ session. Denote $c'_{jr} = s'_{jr} \mod 2$, and $c'_r = \bigoplus_{j \in S_r} c'_{jr}$. Note that s'_{jr} is supposed to be either 0 or 1 but in the case of sharing over a large field, this cannot be enforced for faulty dealers.

For every $j \in S_r$, no nonfaulty party calls $SVSS - Rec_{jr}$ before completing the $CommonSubset_r$ invocation, at which time S_r is set already. From the hiding property of $SVSS$, before some nonfaulty party invokes $SVSS - Rec_{jr}$ for any nonfaulty dealer P_j, the faulty (and nonfaulty) parties' view is distributed independently of the value b_{jr} shared by P_j. This also means that the values shared by any nonfaulty party P_j such that $j \in S_r$ are entirely independent of other values shared by all other parties in S_r. From the Validity property of $SVSS$, $c'_{jr} = b_{jr}$ for every nonfaulty P_j. Since $|S_r| \geq n - t$, there exists at least one nonfaulty party P_j such that $j \in S_r$. Note that $c'_r = 0$ if and only if $\bigoplus_{l \in S_r \setminus \{j\}} c'_{lr} = c'_{jr} = b_{jr}$. b_{jr} is sampled uniformly from $\{0, 1\}$ and entirely independently from the rest of the values, and thus the probability that $c'_r = 0$ for any $r \in [k]$ is exactly $\frac{1}{2}$. Using similar arguments it can also be shown that the values c'_r are independent of values computed in all other iterations.

For each $r \in [k]$ either every nonfaulty party P_i that completes the r'th iteration computes $b'_{ir} = c'_r$ or some nonfaulty party shuns some faulty party starting in iteration r. Overall, there can occur fewer than n^2 shunning events, and thus for at least $k - n^2$ different iterations every nonfaulty party P_i that completes the r'th iteration computes $b'_{ir} = c'_r$. This means that if $\left|\left\{r | c'_r = 1\right\}\right| > \frac{k}{2} + n^2$, then regardless of the faulty parties' actions, every nonfaulty party P_i that completes all k iterations outputs $b'_{ir} = c'_r = 1$ for at least $\left\lfloor \frac{k}{2} \right\rfloor + 1$ of those iterations, and thus inputs 1 to the BA invocation at the end of the protocol. From the Correctness property of BA, if every nonfaulty party that participates in a BA invocation inputs the value 1, then every nonfaulty party that completes the invocation outputs 1. In that case, all nonfaulty parties output 1 in the end of the $CoinFlip$ protocol. The exact same argument can be made stating that all nonfaulty parties output 0 if there are $\frac{k}{2} + n^2$ rounds in which $c'_r = 0$. It is left to show that $\Pr\left[\left|\left\{r | c'_r = 1\right\}\right| > \frac{k}{2} + n^2\right] \geq \frac{1}{2} - \epsilon$. If that is the case, every nonfaulty party that completes the protocol outputs 1. Since for every $r \in [k]$, $\Pr\left[c'_r = 1\right] = \frac{1}{2} = \Pr\left[c'_r = 0\right]$, the case for 0 is entirely symmetric. Define the random variable $X = \left|\left\{r | c'_r = 1\right\}\right|$. Each c'_r is an independent Bernoulli variable with probability $\frac{1}{2}$ of being 1, and thus $X \sim Bin\left(k, \frac{1}{2}\right)$. In this analysis we use the fact that:

$$n! \leq e \cdot n^{n+\frac{1}{2}} \cdot e^{-n}$$
$$n! \geq \sqrt{2\pi} \cdot n^{n+\frac{1}{2}} \cdot e^{-n}$$

Start by bounding the size of $\binom{2n}{n}$ for any n:

$$\begin{aligned}\binom{2n}{n} &= \frac{(2n)!}{(n!)^2} \\ &\leq \frac{e\,(2n)^{2n+\frac{1}{2}}\,e^{-2n}}{\left(\sqrt{2\pi}\,(n)^{n+\frac{1}{2}}\,e^{-n}\right)^2} \\ &= \frac{e}{2\pi} \cdot \frac{(2n)^{2n+\frac{1}{2}}}{(n)^{2n+1}} \\ &= \frac{e}{2\pi} \cdot 2^{2n+\frac{1}{2}} \cdot \frac{1}{\sqrt{n}}\end{aligned}$$

Denote $k = 4\left\lceil c^2 n^4 \right\rceil$ with $c = \frac{e}{\epsilon \cdot \pi}$, and $\mu = \frac{k}{2} = 2\left\lceil c^2 n^4 \right\rceil$. Now bound the probability that X is very close to μ:

$$\begin{aligned}&\Pr\left[\mu - n^2 \leq X \leq \mu + n^2\right] \\ &= \sum_{\mu - n^2 \leq l \leq \mu + n^2} \binom{2\mu}{l} \left(\frac{1}{2}\right)^{2\mu} \\ &\leq \left(2n^2 + 1\right) \binom{2\mu}{\mu} \left(\frac{1}{2}\right)^{2\mu}\end{aligned}$$

$$\leq \left(2n^2+1\right)\frac{e}{2\pi}\cdot 2^{2\mu+\frac{1}{2}}\cdot\frac{1}{\sqrt{\mu}}\left(\frac{1}{2}\right)^{2\mu}$$
$$= \left(2n^2+1\right)\cdot\frac{e}{2\pi}\cdot\frac{1}{\sqrt{\mu}}\cdot\sqrt{2}$$

Substituting back $\mu = 2\left\lceil c^2n^4\right\rceil$:

$$\begin{aligned}
&\Pr\left[\mu - n^2 \leq X \leq \mu + n^2\right] \\
&\leq \left(2n^2+1\right)\cdot\frac{e}{2\pi}\cdot\frac{1}{\sqrt{\mu}}\cdot\sqrt{2} \\
&= \left(2n^2+1\right)\cdot\frac{e}{2\pi}\cdot\frac{1}{\sqrt{2\left\lceil c^2n^4\right\rceil}}\cdot\sqrt{2} \\
&\leq \left(2n^2+1\right)\cdot\frac{e}{2\pi}\cdot\frac{1}{cn^2} \\
&= \frac{2n^2+1}{n^2}\cdot\frac{e}{2\pi}\cdot\frac{1}{c} \\
&\leq 4\cdot\frac{e}{2\pi}\cdot\frac{1}{c} = \frac{2e}{\pi}\cdot\frac{1}{c}
\end{aligned}$$

Since the cases that $X > \mu + n^2$ and $X < \mu - n^2$ are entirely symmetric:

$$\begin{aligned}
\Pr\left[X > \mu + n^2\right] &= \frac{1}{2}\left(1 - \Pr\left[\mu - n^2 \leq X \leq \mu + n^2\right]\right) \\
&\geq \frac{1}{2}\left(1 - \frac{2e}{\pi}\cdot\frac{1}{c}\right) \\
&= \frac{1}{2} - \frac{e}{\pi}\cdot\frac{1}{c}
\end{aligned}$$

Finally, substituting $c = \frac{e}{\epsilon\cdot\pi}$ and $\mu = \frac{k}{2}$:

$$\begin{aligned}
\Pr\left[X > \frac{k}{2} + n^2\right] &\geq \frac{1}{2} - \frac{e}{\pi}\cdot\frac{1}{c} \\
&= \frac{1}{2} - \frac{e}{\pi}\cdot\frac{\epsilon\cdot\pi}{e} \\
&= \frac{1}{2} - \epsilon
\end{aligned}$$

which completes the proof. □

4 Fair agreement

This section deals with constructing a Byzantine Agreement protocol with strong properties. First of all, the regular notions of Correctness (i.e. agreement) and Termination are preserved. In addition to that, a stronger notion of Validity is achieved in the case of multivalued agreement. If all nonfaulty parties have the same input σ, they all output σ; however, if that is not the case, the probability that all nonfaulty parties output some nonfaulty party's input is at least $\frac{1}{2}$. This also nicely extends to natural notions of fairness in the case of a non-Byzantine adversary.

Definition 7 A protocol is said to be a Fair Byzantine Agreement protocol if it has the following properties:

1. *Termination* If all nonfaulty parties participate in the protocol, they all almost-surely complete it. Furthermore, if some nonfaulty party completes the protocol, all other nonfaulty parties that participate in it almost-surely complete it as well.
2. *Validity* If all nonfaulty parties have the same input to the protocol, they output that value. Otherwise, with probability at least $\frac{1}{2}$, all nonfaulty parties output some nonfaulty party's input.
3. *Correctness* All nonfaulty parties that complete the protocol output the same value.

This notion of fairness is closely related to other notions of fairness, such as having a $\frac{1}{n}$ probability (or close to such a probability) of choosing any nonfaulty party's input [13]. As stated, this notion of fairness is identical to the quality property of [3]. In an asynchronous setting achieving full fairness is not possible, because a faulty party's messages could be delayed until all other parties have completed the protocol, and thus the probability of its input being chosen is 0. However, close inspection of the arguments below shows that there is nearly a uniform probability of choosing the input of some nonfaulty party that was allowed to participate in the protocol, leading to a property very close to the fairness property of [13]. The goal in this section is to design a Fair Byzantine Agreement protocol. In order to do so, a protocol for choosing one element out of m elements in an almost fair way is described.

Definition 8 A Fair Choice protocol has the following properties if all nonfaulty parties that participate in it have the same input $m \geq 3$:

1. *Termination* If all nonfaulty parties participate in the protocol they all almost-surely complete it. Furthermore, if some nonfaulty party completes the protocol, all nonfaulty parties that participate in it almost-surely complete it as well.
2. *Validity* For any set $G \subseteq \{0, \ldots, m-1\}$ such that $|G| > \frac{m}{2}$ the probability that all nonfaulty parties that complete the protocol output some $i \in G$ is at least $\frac{1}{2}$.
3. *Correctness* All nonfaulty parties that complete the protocol output the same value $i \in \{0, \ldots, m-1\}$.

Algorithm 3 $FairChoice(m)$

Code for P_i:

1. Set $N = 2^l$ for the smallest $l \in \mathbb{N}$ such that $4m^2 \geq N \geq 2m^2$ and set $\epsilon = \frac{1}{100m \log_2 m}$.
2. For every $i \in [l]$ participate in $CoinFlip_i$ (ϵ) and let the i'th output be b_i.
3. Let r be the number whose binary representation is $b_1 b_2 \ldots b_l$. Output $r \mod m$ and terminate.

Theorem 5 *FairChoice is a Fair Choice protocol resilient to any number of faulty parties t such that $3t + 1 \leq n$.*

Proof Each property is proven individually. Throughout the analysis, unless explicitly stated differently all logarithms are treated as logarithms with base 2.

Termination If all nonfaulty parties participate in the protocol and have the same input m, they all compute the same values l and ϵ. They then all participate in the $CoinFlip$ protocol l times with the same parameter ϵ and from the Termination property of the $CoinFlip$ protocol, they almost-surely complete those calls to the protocol. Afterwards every nonfaulty party performs some local computations and completes the protocol. On the other hand, if some nonfaulty party completes the $FairChoice$ protocol, it must have first completed all l invocations of the $CoinFlip$ protocol with parameter ϵ. Observe some other nonfaulty party P_i that participates in the $FairChoice$ protocol with the same input m. It must have computed the same values l and ϵ, and then participated in l invocations of the $CoinFlip$ protocol with the same parameter ϵ. Since some nonfaulty party completed all l of those invocations, from the Termination property of the $CoinFlip$ protocol, P_i almost-surely completes them as well. Finally P_i performs some local computations and completes the protocol.

Correctness Observe two nonfaulty parties that complete the protocol. Since they both have the same input m, they must have computed the same value l, and participated in l invocations of the $CoinFlip$ protocol. From the Correctness property of the $CoinFlip$ protocol, for every $i \in [l]$ they must have output the same value $b_i \in \{0, 1\}$ in the i'th invocation of the $CoinFlip$ protocol. This means that they compute the same number r, and then output output $r \mod m \in \{0, \ldots, m-1\}$.

Validity Intuitively, there are more values in G than values not in G and each value $i \in G$ has almost the same number of values $k \in \{0, \ldots, N-1\}$ such that $k \equiv i \mod m$. Furthermore, each value in $\{0, \ldots, N-1\}$ has nearly the same probability of being sampled. If every number had the exact same probability of being sampled, and each value $i \in G$ had exactly the same number of values $k \in \{0, \ldots, N-1\}$ such that $k \equiv i \mod m$ it is clear that the property holds. It is only left to show that these slight differences aren't big enough for the property not to hold.

Consider the case in which all nonfaulty parties that participate in the protocol have the same input m. Let N, l, ϵ be defined as they are in the protocol. Consider some $G \subseteq \{0, \ldots, m-1\}$ such that $|G| > \frac{m}{2}$. For every $i \in \{0, \ldots, m-1\}$ define the set $S_i = \{j \in \{0, \ldots, N-1\} \mid j \equiv i \mod m\}$. Define $S = \cup_{i \in G} S_i$. First, bound the size of S. For every $i \in \{0, \ldots, m-1\}$, $|S_i| \geq \lfloor \frac{N}{m} \rfloor \geq \frac{N}{m} - 1$. Since $|G|, m \in \mathbb{N}$:

$$
\begin{aligned}
&|G| > \frac{m}{2} \\
&2|G| > m \\
&2|G| \geq m + 1 \\
&|G| \geq \frac{m}{2} + \frac{1}{2}
\end{aligned}
$$

Note that for every $i \neq j$ $S_i \cap S_j = \emptyset$ and thus:

$$
\begin{aligned}
|S| &= \sum_{i \in G} |S_i| \\
&\geq \left(\frac{N}{m} - 1\right)|G| \\
&\geq \left(\frac{N}{m} - 1\right)\left(\frac{m}{2} + \frac{1}{2}\right) \\
&= (N - m)\left(\frac{1}{2} + \frac{1}{2m}\right)
\end{aligned}
$$

As shown in the proof of the Correctness property, all nonfaulty parties that complete the protocol first complete l invocations of the $CoinFlip$ protocol, output the same bits b_i for every $i \in [l]$, then compute the same value r and output $r \mod m$. In that case, all nonfaulty parties output some $i \in G$ if and only if $r \in S$. From the Correctness property of the $CoinFlip$ protocol, for every $j \in [l]$ and $b \in \{0, 1\}$, $\Pr\left[b_j = b\right] \geq \frac{1}{2} - \epsilon$ regardless of the adversary's actions. For every number r denote r_i to be the $i'th$ bit in its binary representation. Therefore:

$$
\begin{aligned}
&\Pr[i \in G] \\
&\quad = \Pr[r \in S] \\
&\quad = \sum_{r' \in S} \Pr\left[r = r'\right] \\
&\quad = \sum_{r' \in S} \Pr\left[\bigwedge_{j=1}^{l} r_j = r'_j\right] \\
&\quad \geq \sum_{r' \in S} \left(\frac{1}{2} - \epsilon\right)^l \\
&\quad = |S| \left(\frac{1}{2} - \epsilon\right)^l
\end{aligned}
$$

$$
\begin{aligned}
&\geq (N-m)\left(\frac{1}{2}+\frac{1}{2m}\right)\left(\frac{1}{2}-\epsilon\right)^{\log N} \\
&= (N-m)\left(\frac{1}{2}+\frac{1}{2m}\right)\left(\frac{1}{2}\right)^{\log N}(1-2\epsilon)^{\log N} \\
&= \left(1-\frac{m}{N}\right)\left(\frac{1}{2}+\frac{1}{2m}\right)\left(1-\frac{2}{100m\log m}\right)^{\log N} \\
&\geq \left(1-\frac{m}{2m^2}\right)\left(\frac{1}{2}+\frac{1}{2m}\right)\left(1-\frac{1}{50m\log m}\right)^{\log 4m^2} \\
&= \left(\frac{1}{2}+\frac{1}{2m}-\frac{1}{4m}-\frac{1}{4m^2}\right) \\
&\quad\times\left(\left(1-\frac{1}{50m\log m}\right)^{m\log m}\right)^{\frac{2\log m+2}{m\log m}}
\end{aligned}
$$

At this point recall that $m \geq 3$. Clearly $\frac{2\log m+2}{m\log m} \leq \frac{4\log m}{m\log m} = \frac{4}{m}$ for any $m \geq 2$. Secondly, note that the expression $\left(1-\frac{x}{n}\right)^n$ approaches e^{-x} from below in a monotonously increasing manner for $1 > x > 0$. Plugging in $m = 3$, $\left(1-\frac{1}{50\cdot 3\log 3}\right)^{3\log 3} \geq \frac{99}{100}e^{-\frac{1}{50}}$, and from the previous observation $\left(1-\frac{1}{50m\log m}\right)^{m\log m} \geq \frac{99}{100}e^{-\frac{1}{50}}$ for every $m \geq 3$. Combining these observations:

$$
\begin{aligned}
&\Pr[i \in G] \\
&\geq \left(\frac{1}{2}+\frac{1}{2m}-\frac{1}{4m}-\frac{1}{4m^2}\right) \\
&\quad\times\left(\left(1-\frac{1}{50m\log m}\right)^{m\log m}\right)^{\frac{2\log m+2}{m\log m}} \\
&\geq \left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}}
\end{aligned}
$$

First, note that clearly:

$$
\lim_{m\to\infty}\left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} = \frac{1}{2}\cdot 1 = \frac{1}{2}
$$

In addition, setting $m = 3$ and checking numerically:

$$
\left(\frac{1}{2}+\frac{1}{4\cdot 3}-\frac{1}{4\cdot 3^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{3}} \approx 0.534 > 0.5
$$

Next observe the derivative of the expression with respect to m and check when it is negative.

$$
\begin{aligned}
&\frac{d}{dm}\left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} \\
&= \left(-\frac{1}{4m^2}+\frac{1}{2m^3}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} \\
&\quad+\left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} \\
&\quad\times\left(\frac{-4\ln\left(\frac{99}{100}e^{-\frac{1}{50}}\right)}{m^2}\right) \\
&= \left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} \\
&\quad\times\left(\frac{2-m}{4m^3}-\frac{16m\left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\ln\left(\frac{99}{100}e^{-\frac{1}{50}}\right)}{4m^3}\right) \\
&= \frac{1}{4m^3}\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} \\
&\quad\times\left(2-m-\left(8m+4-\frac{4}{m}\right)\ln\left(\frac{99}{100}e^{-\frac{1}{50}}\right)\right)
\end{aligned}
$$

Now note that for any $m \geq 3$:

$$
\frac{1}{4m^3}\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} > 0
$$

and thus the whole expression is negative if:

$$
\begin{aligned}
0 &> 2-m-\left(8m+4-\frac{4}{m}\right)\ln\left(\frac{99}{100}e^{-\frac{1}{50}}\right) \\
&= 2-m+\left(8m+4-\frac{4}{m}\right)\ln\left(\frac{100}{99}e^{\frac{1}{50}}\right)
\end{aligned}
$$

Numerically we can find that $0.031 \geq \ln\left(\frac{100}{99}e^{\frac{1}{50}}\right) > 0$ and thus:

$$
\begin{aligned}
&2-m+\left(8m+4-\frac{4}{m}\right)\ln\left(\frac{100}{99}e^{\frac{1}{50}}\right) \\
&\leq 2-m+(8m+4)\ln\left(\frac{100}{99}e^{\frac{1}{50}}\right) \\
&\leq 2-m+(8m+4)\,0.031 \\
&= 2-m+0.248m+0.124 \\
&= 2.124-0.752m
\end{aligned}
$$

Finally check if this term is negative:

$$
\begin{aligned}
&2.124-0.752m < 0 \\
&2.124 < 0.752m \\
&\frac{2.124}{0.752} \approx 2.824 < m
\end{aligned}
$$

Since $m \geq 3$:

$$
2-m+\left(8m+4-\frac{4}{m}\right)\ln\left(\frac{100}{99}e^{\frac{1}{50}}\right) < 0
$$

and thus for every $m \geq 3$:

$$\frac{d}{dm}\left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} < 0$$

Combining the fact that at $m = 3$ the expression is greater than $\frac{1}{2}$, that the derivative is negative for any $m \geq 3$ and that the expression approaches $\frac{1}{2}$ as m approaches infinity, for any $m \geq 3$:

$$\Pr[i \in G] \geq \left(\frac{1}{2}+\frac{1}{4m}-\frac{1}{4m^2}\right)\left(\frac{99}{100}e^{-\frac{1}{50}}\right)^{\frac{4}{m}} > \frac{1}{2}$$

completing the proof. □

A Fair Byzantine Agreement protocol that uses the Fair Choice protocol is described below. In this Fair Byzantine Agreement protocol, each party P_i has some input x_i. The construction makes use of a Broadcast protocol.

Definition 9 A Broadcast protocol is a protocol with a designated sender P_i with some input v, which has the following properties:

1. *Termination* If P_i is nonfaulty and all nonfaulty parties participate in the protocol, they all complete the protocol. Furthermore, if some nonfaulty party completes the protocol, every other nonfaulty party that participates in it does so as well.
2. *Validity* If P_i is nonfaulty, every nonfaulty party that completes the protocol outputs v.
3. *Correctness* All nonfaulty parties that complete the protocol output the same value.

Let A-Cast be a Broadcast protocol, for example as described in [8].

Algorithm 4 FBA

Code for P_i with input x_i:

1. A-Cast x_i and participate in every other party's $A-Cast$. Denote the output of P_j's A-Cast to be x'_j.
2. Define the dynamic predicate Q_i as follows:
$$Q_i(j) = \begin{cases} 1 & P_j\text{'s A-Cast has been completed} \\ 0 & else \end{cases}$$
3. Participate in $CommonSubset(Q_i, n-t)$.
4. After completing the $CommonSubset$ protocol, let S be the protocol's output and let $m = |S|$. Wait to complete P_j's A-Cast for every $j \in S$.
5. If there exists some value x such that $\left|\left\{x'_j = x | j \in S\right\}\right| > \frac{m}{2}$, output x and complete the protocol. Otherwise, continue to the next step.
6. Participate in $FairChoice(m)$, and let the output be k.
7. Let j be the k'th biggest value in S, with the 0'th being understood as the biggest value, 1'st as the second biggest, etc.
8. Output x'_j and terminate.

Theorem 6 *Protocol FBA is a Fair Byzantine Agreement protocol for any number of faulty parties t such that $3t+1 \leq n$.*

Intuitively, each party A-Casts its input value, and the parties agree on a subset of parties of size $n-t$ at the very least whose values have been received using the $CommonSubset$ protocol. If all nonfaulty parties have the same input, they will see that a majority of the parties sent the same value and output that value in line 5, achieving the first part of the Validity property. Otherwise, the parties choose the value sent by one of those parties "almost fairly" using the $FairChoice$ Protocol. Since more than half of the parties in the agreed upon subset are nonfaulty, the probability that a nonfaulty party will be chosen is at least $\frac{1}{2}$. A formal proof is provided below.

Proof Again, each property is proven individually.

Termination If all nonfaulty parties participate in the FBA protocol, they all A-Cast some values in step 1 and participate in each other's A-Casts. Since all of the senders in those A-Casts are nonfaulty and all nonfaulty parties participate in all of those A-Casts, from the Termination property of A-Cast they all complete each of those calls. This means that for every pair of nonfaulty parties P_i, P_j eventually $Q_i(j) = 1$. In other words, since there are at least $n-t$ nonfaulty parties there exists a set $I \subseteq [n]$ such that for every nonfaulty party P_i, eventually $\forall j \in I\ Q_i(j) = 1$. From the Termination property of $CommonSubset$, all nonfaulty parties almost-surely eventually complete the protocol. From the Correctness property of $CommonSubset$, all nonfaulty parties output the same $S \subseteq [n]$ and for every $j \in S$ there exists some nonfaulty party P_i such that $Q_i(j) = 1$. A nonfaulty party sets $Q_i(j) = 1$ only if it completed P_j's A-Cast, and from the Termination property of A-Cast, all nonfaulty parties that participate in that A-Cast complete it as well. This means that all nonfaulty parties complete all relevant A-Cast invocations and then finish step 4 of the protocol. From the Correctness property of A-Cast, all nonfaulty parties receive the same value x'_k in P_k's A-Cast for every $k \in S$. If some nonfaulty party completes the protocol in step 5, then there exists some x such that $\left|\left\{x'_j = x | j \in S\right\}\right| > \frac{m}{2}$. Every other nonfaulty party sees that this holds as well and completes the protocol in step 5. Otherwise, all nonfaulty parties participate in $FairChoice(m)$ with the same $m = |S|$. Note that from the Correctness property of the $CommonSubset$ protocol, they all output some set S such that $|S| \geq n-t \geq 3$. Therefore, from the Termination property of the $FairChoice$ protocol, they all almost-surely complete the $FairChoice$ protocol as well. Afterwards they perform some local computations and complete the protocol.

For the second part of the property, assume some nonfaulty party P_i completed the FBA protocol. This means it

must have completed the $CommonSubset$ protocol and if it didn't complete the FBA protocol in step 5, it completed the $FairChoice$ protocol as well. Observe some other nonfaulty party P_j that participates in the protocol. First, P_j A-Casts some value and participates in every other party's A-Cast. P_j then participates in the $CommonSubset$ protocol. Every nonfaulty party that participates in the $CommonSubset$ protocol also participates in each of the A-Cast calls. For every pair of nonfaulty parties P_k, P_l that participate in $CommonSubset$ and value $m \in [n]$, if $Q_k(m) = 1$ P_k must have completed P_m's A-Cast. From the Termination property of A-Cast, P_l will eventually complete P_m's too A-Cast and set $Q_l(m) = 1$. Therefore, the conditions of the second part of the Termination property of the $CommonSubset$ protocol hold, and thus since P_i completed the $CommonSubset$ protocol P_j almost-surely completes it as well with some output S. From the Correctness property of $CommonSubset$, for every $k \in S$, $Q_l(k) = 1$ for some nonfaulty party P_l. This means that P_l completed P_k's A-Cast, which means P_j does so as well. If P_j completes the protocol in step 5 after completing all of the A-Cast invocations we are done. Otherwise, after completing all of the relevant A-Casts, P_j participates in the $FairChoice$ protocol. In that case there must not exist any x such that $\left|\left\{x'_k = x | k \in S\right\}\right| > \frac{m}{2}$. From the Correctness property of A-Cast, P_i must have output the same value in each of those A-Casts, seen that there does not exist any x such that $|\{x'_k = x | k \in S\}| > \frac{m}{2}$, and then invoked and completed the $FairChoice$ protocol. From the Termination property of the $FairChoice$ protocol, P_j almost-surely completes it as well, performs some local computations, and then finally completes the FBA protocol.

Correctness Let P_i, P_j be two nonfaulty parties that completed the protocol. They must have both first completed the $CommonSubset$ protocol and from its Correctness property output the same set $S \subseteq [n]$. They then completed P_k's A-Cast for every $k \in S$. From the Correctness property of A-Cast, they both received the same value x'_k for every $k \in S$. If there exists some x such that $\left|\left\{x'_k = x | k \in S\right\}\right| > \frac{m}{2}$ then they both must output that value and complete the protocol. Note that clearly there cannot be more than one such value. If there isn't any such value x, then they both participated in the $FairChoice$ protocol and from the Correctness property of the protocol output the same value $k \in \{0, \ldots, m-1\}$. They then both took the k'th biggest value in S and output the value corresponding to that party's A-Cast. Again, from the Correctness property of A-Cast P_i, P_j must have received the same value in that A-Cast and thus output the same value.

Validity First assume that all nonfaulty parties have the same input x. In that case, in the beginning of the protocol each nonfaulty party that participates in the protocol A-Casts x. Let P_i be some nonfaulty party that completed the protocol. Before completing the protocol it participated in all relevant A-Casts and in protocol $CommonSubset$, and completed it with some output S. The input to the $CommonSubset$ protocol is $n - t$, so from the Correctness property of $CommonSubset$, $|S| \geq n - t$. P_i then completed P_j's A-Cast for every $j \in S$. From the Validity property of A-Cast, P_i received the value $x'_j = x$ from every nonfaulty party P_j such that $j \in S$. Let G be the set of all $j \in S$ such that P_j is nonfaulty. Since there are at most t faulty parties P_k such that $k \in S$, $|G| \geq |S| - t = m - t$. Note that $m \geq n - t > 2t$ and thus $\frac{m}{2} > t \Rightarrow m - t > \frac{m}{2}$. Since $|G| \geq m - t > \frac{m}{2}$, and for every $j \in G$, P_i received the value $x'_j = x$, P_i sees that $\left|\left\{x'_j = x | j \in S\right\}\right| > \frac{m}{2}$. This means that in step 5, P_i outputs x and completes the protocol.

On the other hand, if it is not the case that all nonfaulty parties had the same input, for every nonfaulty party P_j let x_j be its input. Observe some nonfaulty party P_i that completed the protocol. Following the exact same arguments as above, P_i must have participated in all A-Casts, completed the $CommonSubset$ protocol with some output S such that $m = |S| \geq n - t$, and completed P_j's A-Cast for every $j \in S$. Note that from the Validity property of A-Cast, for every nonfaulty party P_j, P_i received the value $x_j = x'_j$ in P_j's A-Cast. If P_i output some value in step 5, it must have found some value x such that $\left|\left\{x'_j = x | j \in S\right\}\right| > \frac{m}{2}$. As previously shown $\frac{m}{2} > t$, and thus $\left|\left\{x'_j = x | j \in S\right\}\right| \geq t + 1$. There are t faulty parties at most, which means that there must be some nonfaulty party P_j such that $x_j = x'_j = x$. In other words, if P_i completed the protocol in step 5 the property holds. Otherwise, P_i must have invoked and completed protocol $FairChoice$ before completing the FBA protocol. Define G as defined above. As previously shown $|G| > \frac{m}{2}$. Let $S_G \subseteq \{0, \ldots, m-1\}$ be all of the numbers $k \in \{0, \ldots, m-1\}$ such that the k'th biggest value in S (as defined in the protocol) is in G. Note that each $k \in \{0, \ldots, m-1\}$ corresponds to a unique value $j \in S$, and thus $|S_G| = |G| > \frac{m}{2}$. From the Correctness property of $FairChoice$, P_i outputs some $k \in S_G$ with probability $\frac{1}{2}$ at the very least. P_i then finds the corresponding $j \in G$ and outputs $x'_j = x_j$. Since by definition P_j is a nonfaulty party, P_i output some nonfaulty party's input, completing the proof. □

References

1. Abraham, I., Dolev, D., Gonen, R., Halpern, J.: Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, PODC'06, Association for Computing Machinery, New York, pp. 53–62 (2006)
2. Abraham, I., Dolev, D., Halpern, J.Y.: An almost-surely terminating polynomial protocol for asynchronous byzantine agreement with optimal resilience. In: Proceedings of the Twenty-Seventh ACM

Symposium on Principles of Distributed Computing, PODC'08, Association for Computing Machinery, New York, pp. 405–414 (2008)
3. Abraham, I., Malkhi, D., Spiegelman, A.: Validated asynchronous byzantine agreement with optimal resilience and asymptotically optimal time and word communication (2018)
4. Backes, M., Datta, A., Kate, A.: Asynchronous computational VSS with reduced communication complexity. In: Cryptographers' Track at the RSA Conference, Springer, pp. 259–276 (2013)
5. Ben-Or, M.: Another advantage of free choice (extended abstract): completely asynchronous agreement protocols. In: Proceedings of the Second Annual ACM Symposium on Principles of Distributed Computing, PODC '83 (1983)
6. Ben-Or, M., Canetti, R., Goldreich, O.: Asynchronous secure computation. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, STOC'93, Association for Computing Machinery, New York, pp. 52–61 (1993)
7. Ben-Or, M., Kelmer, B., Rabin, T.: Asynchronous secure computations with optimal resilience (extended abstract). In: Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing, PODC'94, Association for Computing Machinery, New York, pp. 183–192 (1994)
8. Bracha, G.: Asynchronous byzantine agreement protocols. Inf. Comput. **75**(2), 130–143 (1987)
9. Canetti, R., Rabin, T.: Fast asynchronous byzantine agreement with optimal resilience. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, STOC '93, ACM, New York, pp. 42–51 (1993)
10. Dwork, C., Lynch, N., Stockmeyer, L.: Consensus in the presence of partial synchrony. J. ACM **35**(2), 288–323 (1988)
11. Fischer, M.J., Lynch, N.A., Merritt, M.: Easy impossibility proofs for distributed consensus problems. In: Proceedings of the Fourth Annual ACM Symposium on Principles of Distributed Computing, PODC'85, Association for Computing Machinery, New York, pp. 59–70 (1985)
12. Fischer, M.J., Lynch, N.A., Paterson, M.S.: Impossibility of distributed consensus with one faulty process. J. ACM **32**(2), 374–382 (1985)
13. Kuo, P.-C., Chung, H., Chao, T.-W., Cheng, C.-M.: Fair byzantine agreements for blockchains. IEEE Access **8**, 70746–70761 (2020)