Communication-Efficient Signature-Free Asynchronous Byzantine Agreement

Fan Li and Jinyuan Chen
Department of Electrical Engineering
Louisiana Tech University
Email: {fli005, jinyuan}@latech.edu

***Abstract*—In this work, we focus on the problem of byzantine agreement (BA), in which n distributed processors seek to reach an agreement on an ℓ-bit value, but up to t processors might be corrupted by a Byzantine adversary and act as dishonest nodes. In particular, we consider the communication-efficient BA in an asynchronous setting, where the network communication might have arbitrarily time delay. The primary challenge of designing the BA protocol in this setting is that we need to handle both the message delay from honest nodes and the Byzantine behavior from dishonest nodes simultaneously. In this work we propose a new signature-free asynchronous byzantine agreement (ABA) protocol, which achieves the optimal communication complexity of $O(n\ell)$ when $\ell \geq t\log t$, given $n \geq 5t+1$. A protocol is said to be signature-free if the protocol design does not depend on the cryptographic machinery such as hashing and signature. To the best of our knowledge, this is the first *signature-free* ABA protocol that achieves the optimal communication complexity of $O(n\ell)$ when ℓ is almost linearly scaled with t.**

I. Introduction

Byzantine agreement (BA), as a 40-year-old distributed consensus problem (cf. [1], [2]), is one of the fundamental building blocks for distributed computing and cryptography (e.g. [1]–[12]). In the BA problem, n distributed processors seek to reach an agreement on some value, but up to t processors might be corrupted by a Byzantine adversary and act as dishonest nodes. The BA problem has been broadly studied in varying settings, such as authenticated BA (cf. [5]–[12]), unauthenticated BA (cf. [13]–[16]), BA with synchronous communication (cf. [17]–[20]), BA with asynchronous communication (cf. [21]–[33]), binary BA (cf. [13]–[15], [30], [34]), and multi-valued BA (cf. [35]–[45]). In this work, we focus on the *communication-efficient* multi-valued asynchronous BA (ABA) in an unauthenticated (signature-free) setting. A protocol is said to be signature-free if the protocol design does not depend on the cryptographic machinery such as hashing and signature.

In the research line of signature-free multi-valued ABA, the work of [39] proposed an ABA protocol with communication complexity $O(n\ell+n^5\log n)$, given $n \geq 3t+1$. This result was recently improved in [45] to the communication complexity of $O(n\ell + n^4\log n)$. The communication complexity of [39] and [45] is optimal when $\ell \geq n^4\log n$ and $\ell \geq n^3\log n$, respectively. A recent work of [16] has shown that the optimal communication complexity $O(n\ell)$ can be achieved when $\ell \geq t\log t$ in the synchronous setting. This inspires us to raise the following question.

TABLE I
Comparison of different ABA protocols

Model	Resilience	Communication complexity	Round complexity	Signature -free
[39]	$t < \frac{n}{3}$	$O(n\ell + n^5\log n)$	$O(1)$	yes
[45]	$t < \frac{n}{3}$	$O(n\ell + n^4\log n)$	$O(1)$	yes
Proposed	$t < \frac{n}{5}$	$O(\max\{n\ell, nt\log t\})$	$O(1)$	yes

Is it possible to achieve the optimal communication complexity $O(n\ell)$ when $\ell \geq t\log t$ in the signature-free asynchronous setting?

In this work, we seek to investigate the above question. Specifically, we propose a new signature-free ABA protocol that achieves the communication complexity of $O(\max\{n\ell, nt\log t\})$, given $n \geq 5t+1$. It implies that the optimal communication complexity of $O(n\ell)$ can be achieved when $\ell \geq t\log t$. To the best of our knowledge, this is the first signature-free ABA protocol that achieves the optimal communication complexity of $O(n\ell)$ when ℓ is almost linearly scaled with t.

The proposed protocol (called as A-COOL) is extended from the COOL protocol introduced in the synchronous setting [16]. Note that, in the asynchronous setting the message transmission might be delayed with arbitrary time. Therefore, we need to handle both the message delay from honest nodes and the Byzantine behavior from dishonest nodes simultaneously. This is the primary challenge of designing the BA protocol in this asynchronous setting.

We provide some comparison between different ABA protocols in Table I and Fig. 1. Note that Fig. 1 focuses on the comparison in the communication complexity exponent, which is defined in (1) (see Section II), capturing the exponent of communication complexity performance. As shown in Fig. 1, compared to the protocols in [39] and [45], A-COOL protocol provides additive gains up to 3 and 2, respectively, in terms of communication complexity exponent performance.

This paper is organized as follows. The system model is provided in Section II, while the main results are presented in Section III. The proposed A-COOL is detailed in Section IV. The analysis of A-COOL is provided in Section V. The conclusion is drawn in Section VI.

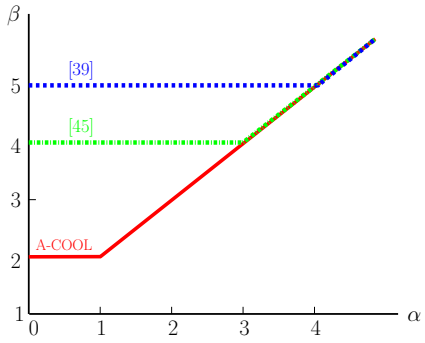

Fig. 1. Communication complexity exponent β vs. message size exponent α of the proposed A-COOL, the protocols in [39] and [45], focusing on the case with $\delta = 1$.

II. System models

We consider the ABA problem, in which a set of n distributed processors $\{P_1, P_2, \cdots, P_n\}$ seek to reach an agreement on an ℓ-bit value, but up to t processors might be corrupted by a Byzantine adversary and act as dishonest nodes. The processors are communicated over an asynchronous network. Each pair of processors is connected via a reliable and private communication channel. The index of each processor is available to all other processors. We consider a static adversary, denoted by $\mathcal{A}_t$, which selects the set of corrupted processors before starting the protocol. We assume that the adversary has complete knowledge of the state of the other processors. The processors that are not corrupted by the adversary are honest processors. We provide the formal definition of ABA as below.

Definition 1 (ABA)**.** *A protocol for a set of processors $\{P_1, P_2, \cdots, P_n\}$, where P_i holds an initial ℓ-bit input value $\boldsymbol{w}_i$, $i \in [1:n]$, is said to be an ABA protocol tolerating an adversary $\mathcal{A}_t$, if the following properties hold*

- *Termination: Every honest processor eventually almost-surely*[1] *terminates.*
- *Consistency: All of the honest processors eventually decide on the same value.*
- *Validity: If every honest processor holds an initial value $\boldsymbol{w}$, then all of the honest processors eventually decide on this initial value $\boldsymbol{w}$.*

In this work we use the following three parameters to measure the quality of a BA protocol:

- Resilience: the maximum number of dishonest nodes allowed in the protocol, denoted by t.
- Communication complexity: the total number of communication bits during the protocol execution, denoted by b.
- Round complexity: the expected number of rounds taken by the protocol to terminate, denoted by r.

[1]The probability that an honest processor is undecided after r rounds approaches 0 as r grows to infinity [46].

We define the notion of *communication complexity exponent* of the ABA protocol as

$$\beta(\alpha, \delta) \triangleq \lim_{n\to\infty} \frac{\log b(n, \delta, \alpha)}{\log n} \tag{1}$$

where α is the message size exponent, defined as $\alpha \triangleq \lim_{n\to\infty} \frac{\log \ell}{\log n}$, and δ is the faulty size exponent, defined as $\delta \triangleq \lim_{n\to\infty} \frac{\log t}{\log n}$. β, α and δ capture the exponents of communication complexity b, the message size ℓ and faulty size t respectively with n as the base, when n is large.

Recall that a protocol is said to be signature-free if the protocol design does not depend on the cryptographic machinery such as hashing and signature. A protocol is said to be information-theoretic secure if the protocol can tolerate the computationally unbounded adversary.

III. Main results

In this section we will provide the main results of this work.

Theorem 1 (ABA)**.** *For the ABA problem, the proposed A-COOL is a signature-free multi-valued ABA protocol that reaches an agreement on an ℓ-bit value with communication complexity $O(\max\{n\ell, nt\log t\})$, given $n \geq 5t+1$.*

Proof. The proposed A-COOL protocol is provided in Section IV. □

Theorem 1 implies that the proposed A-COOL protocol achieves the optimal communication complexity $O(n\ell)$ when $\ell \geq t\log t$. The following result is directly derived from Theorem 1.

Corollary 1 (Communication complexity exponent)**.** *For the proposed A-COOL protocol, when $n \geq 5t+1$, the communication complexity exponent is*

$$\beta(\alpha, \delta) = \max\{1+\alpha, 1+\delta\}. \tag{2}$$

The proposed A-COOL protocol does not assume the cryptographic machinery such as hashing and signature (signature free). Also, the proposed protocol can tolerate the computationally unbounded adversary (information-theoretic secure).

IV. The proposed A-COOL protocol

In this section, we will provide the proposed A-COOL protocol. This proposed A-COOL protocol seeks to reach an agreement on an ℓ-bit value $\boldsymbol{w}$ over n distributed processors $\{P_i\}_{i=1}^n$ under asynchronous communication. In the design of the A-COOL protocol, coding scheme is used to reduce the communication complexity during the information exchange steps. Specifically, the $(n,\ k)$ Reed-Solomon error correction code will be used in the protocol design. We will show that the proposed A-COOL is a signature-free multi-valued ABA protocol that reaches an agreement on an ℓ-bit value with communication complexity $O(\max\{n\ell, nt\log t\})$, when $n \geq 5t+1$. This result serves as the proof of Theorem 1.

The proposed A-COOL protocol is composed of at most four phases, as shown in Algorithm 1. The proposed protocol is guaranteed to satisfy the termination, consistency, and

validity conditions. In what follows, we will describe the four phases in detail. In the description of the proposed A-COOL, we will first consider the case of $t \leq \frac{n-1}{5}$ and $t = \Omega(n)$. Later on we will also discuss the case of relatively small t, compared to n.

We first define some notations that will be used in the protocol design. Let $\boldsymbol{w}^{(i)}$ be the updated value at P_i, $i \in [1:n]$. Let $\mathrm{u}_i^{[p]}(j)$ be the binary link indicator for P_i and P_j, for $p \in [1:3]$ and $i, j \in [1:n]$. Let $\mathrm{s}_i^{[p]}$ be the binary success indicator at P_i, for $p \in [1:3]$ and $i \in [1:n]$. In our design the initial value of $\boldsymbol{w}^{(i)}$ is set as $\boldsymbol{w}^{(i)} = \boldsymbol{w}_i$, $i \in [1:n]$. The binary link indicator is initially set as $\mathrm{u}_i^{[1]}(j) = 0$, $i, j \in [1:n]$.

A. Phase 1: exchange symbols and update information

The main idea of Phase 1 is to exchange coded symbols and update information.

1) Encode and exchange symbols: By applying Reed-Solomon error correction code, $P_i, i \in [1:n]$, first encodes its ℓ-bit initial value $\boldsymbol{w}_i$ into ℓ/k-bit symbols as

$$y_j^{(i)} \triangleq \boldsymbol{h}_j^{\mathsf{T}} \boldsymbol{w}_i, \quad j \in [1:n] \tag{3}$$

where $\boldsymbol{h}_j$ is defined as $\boldsymbol{h}_j \triangleq [h_{j,1}, h_{j,2}, \cdots, h_{j,k}]^{\mathsf{T}}$, for

$$h_{j,m} \triangleq \prod_{\substack{p=1 \\ p \neq m}}^{k} \frac{j-p}{m-p}, \quad j \in [1:n], \quad m \in [1:k]. \tag{4}$$

In our protocol design, the parameter k is designed as $k \triangleq \lfloor \frac{t}{5} \rfloor + 1$. Note that for the (n, k) Reed-Solomon error correction code constructed in the Galois Field $GF(2^c)$, the condition of $n \leq 2^c - 1$ needs to be satisfied (cf. [47]). In this protocol, the parameter c is designed as

$$c \triangleq \Big\lceil \frac{\max\{\ell,\ (t/5+1) \cdot \log(n+1)\}}{k} \Big\rceil. \tag{5}$$

It can be verified that the condition of $n \leq 2^c - 1$ is satisfied with the designed parameters k and c. In this protocol, each coded symbol $y_j^{(i)}$ has c bits. After this encoding process, P_i, $i \in [1:n]$, sends a pair of coded symbols $(y_j^{(i)}, y_i^{(i)})$ to P_j for $j \in [1:n] \setminus i$.

2) Update information: Upon receiving $n - t$ pairs of symbols $\{(y_i^{(j)}, y_j^{(j)})\}_j$, P_i compares the received $(y_i^{(j)}, y_j^{(j)})$ with its symbol pair $(y_i^{(i)}, y_j^{(i)})$ and sets a link indicator $\mathrm{u}_i^{[1]}(j)$ as

$$\mathrm{u}_i^{[1]}(j) = \begin{cases} 1 & \text{if } (y_i^{(j)}, y_j^{(j)}) = (y_i^{(i)}, y_j^{(i)}) \\ 0 & \text{else} \end{cases} \tag{6}$$

for $i \in [1:n]$. The value of $\mathrm{u}_i^{[1]}(j) = 0$ implies that P_i and P_j might have mismatched value, i.e., $\boldsymbol{w}^{(i)} \neq \boldsymbol{w}^{(j)}$. If $\mathrm{u}_i^{[1]}(j) = 0$, we consider the pair of $(y_i^{(j)}, y_j^{(j)})$ from P_j as a mismatched symbol pair at P_i. By counting the number of mismatched symbol pairs, P_i sets a success indicator $\mathrm{s}_i^{[1]}$ as

$$\mathrm{s}_i^{[1]} = \begin{cases} 1 & \text{if } \sum_{j=1}^{n} \mathrm{u}_i^{[1]}(j) \geq n - 2t \\ 0 & \text{else} \end{cases} \tag{7}$$

for $i \in [1:n]$. The event of $\mathrm{s}_i^{[1]} = 0$ means that the number of mismatched symbol pairs is more than $2t$, which implies that the initial value of P_i does not match the majority of other processors' initial values. If $\mathrm{s}_i^{[1]} = 0$, P_i updates $\boldsymbol{w}^{(i)}$ as $\boldsymbol{w}^{(i)} = \phi$ (a default value), else keeps the original value of $\boldsymbol{w}^{(i)}$.

3) Exchange success indicators: P_i, $i \in [1:n]$, sends the value of success indicator $\mathrm{s}_i^{[1]}$ to $P_j, j \in [1:n] \setminus i$. Upon receiving $n - t$ success indicators $\{\mathrm{s}_j^{[1]}\}_j$, P_i creates the sets:

$$\mathcal{S}_1 \triangleq \{i : \mathrm{s}_j^{[1]} = 1, j \in [1:n]\}, \quad \mathcal{S}_0 \triangleq [1:n] \setminus \mathcal{S}_1 \tag{8}$$

based on $\{\mathrm{s}_j^{[1]}\}_j$. In this step, different processors might have different views on $\mathcal{S}_0$ and $\mathcal{S}_1$, due to the inconsistent value possibly sent from dishonest processors.

Remark 1. In Phase 1, since $y_j^{(i)}$ defined in (3) has c bits, the communication complexity of exchanging coded data (see Line 3 in Algorithm 1), denoted by b_1, is $b_1 = 2cn(n-1)$ bits. Since the success indicator $\mathrm{s}_i^{[1]}$ has only 1 bit, the communication complexity of exchanging success indicators (see Line 14), denoted by b_2, is $b_2 = n(n-1)$ bits.

Remark 2. Note that dishonest processors could send arbitrary (inconsistent) value to different honest processors, which could make honest processors output their updated values differently in this phase. Phase 2 will further handle the issue of inconsistent updated values among honest processors.

B. Phase 2: mask errors, and update success indicator

In this phase, the main goal is to mask errors from honest processors, based on the result of Phase 1.

1) Mask errors: If $\mathrm{s}_i^{[1]} = 1$, P_i sets $\mathrm{u}_i^{[2]}(j) = \mathrm{u}_i^{[1]}(j), \forall j \in \mathcal{S}_1$ and $\mathrm{u}_i^{[2]}(j) = 0,\ \forall j \in \mathcal{S}_0$.

2) Exchange success indicators: If $\mathrm{s}_i^{[1]} = 1$, P_i updates $\mathrm{s}_i^{[2]}$ as in (7) using updated values of $\{\mathrm{u}_i^{[2]}(j)\}_{j=1}^n$, else sets $\mathrm{s}_i^{[2]} = 0$. Then P_i sends $\mathrm{s}_i^{[2]}$ to $P_j, j \in [1:n] \setminus i$. If $\mathrm{s}_i^{[2]} = 0$, P_i updates $\boldsymbol{w}^{(i)}$ as $\boldsymbol{w}^{(i)} = \phi$, else keeps the original value of $\boldsymbol{w}^{(i)}$.

3) Update $\mathcal{S}_0$ and $\mathcal{S}_1$: Upon receiving $n - t$ success indicators $\{\mathrm{s}_j^{[2]}\}_j$, P_i updates the sets of $\mathcal{S}_0$ and $\mathcal{S}_1$ as in (8) for $i \in [1:n]$.

Remark 3. Since the success indicator $\mathrm{s}_i^{[2]}$ has 1 bit, the communication complexity of exchanging success indicators in Phase 2 (see Line 26 in Algorithm 1), denoted by b_3, is $b_3 = n(n-1)$ bits.

C. Phase 3: mask errors, update information, and vote

In Phase 3, the main goal is to mask the rest of the errors from the honest processors, and then vote for stopping in this phase or going to Phase 4.

1) Mask errors: If $\mathrm{s}_i^{[2]} = 1$, P_i sets $\mathrm{u}_i^{[3]}(j) = \mathrm{u}_i^{[2]}(j), \forall j \in \mathcal{S}_1$ and $\mathrm{u}_i^{[3]}(j) = 0,\ \forall j \in \mathcal{S}_0$.

2) Exchange success indicator: If $\mathrm{s}_i^{[2]} = 1$, P_i updates $\mathrm{s}_i^{[3]}$ as in (7) using updated values of $\{\mathrm{u}_i^{[3]}(j)\}_{j=1}^n$, else sets $\mathrm{s}_i^{[3]} = 0$. Then P_i sends $(\mathrm{s}_i^{[3]}, y_j^{(i)})$ to $P_j, j \in [1:n] \setminus i$. If $\mathrm{s}_i^{[3]} = 0$,

Algorithm 1 : A-COOL protocol, code for P_i, $i \in [1:n]$

1: Set $\boldsymbol{w}^{(i)} = \boldsymbol{w}_i$; $\mathrm{u}_i^{[1]}(j) = 0$, $j \in [1:n]$.

Phase 1

2: P_i encodes $\boldsymbol{w}_i$ into n symbols as $y_j^{(i)} \triangleq \boldsymbol{h}_j^{\intercal}\boldsymbol{w}_i, j \in [1:n]$.
3: P_i sends $(y_j^{(i)}, y_i^{(i)})$ to P_j, $\forall j \in [1:n] \setminus i$.
4: **upon** receiving $n-t$ pairs of symbols $\{(y_i^{(j)}, y_j^{(j)})\}_j$
5: **for** each received $(y_i^{(j)}, y_j^{(j)})$ **do**
6: **if** $\big((y_i^{(j)}, y_j^{(j)}) == (y_i^{(i)}, y_j^{(i)})\big)$ **then**
7: P_i sets $\mathrm{u}_i^{[1]}(j) = 1$.
8: **else**
9: P_i sets $\mathrm{u}_i^{[1]}(j) = 0$.
10: **if** $\big(\sum_{j=1}^{n} \mathrm{u}_i^{[1]}(j) >= n-2t\big)$ **then**
11: P_i sets $\mathrm{s}_i^{[1]} = 1$.
12: **else**
13: P_i sets $\mathrm{s}_i^{[1]} = 0$ and $\boldsymbol{w}^{(i)} = \phi$.
14: P_i sends $\mathrm{s}_i^{[1]}$ to $P_j, j \in [1:n] \setminus i$.
15: **upon** receiving $n-t$ success indicators $\{\mathrm{s}_j^{[1]}\}_j$
16: P_i sets $\mathcal{S}_1 = \{i : \mathrm{s}_j^{[1]} = 1, j \in [1:n]\}$, $\mathcal{S}_0 = [1:n] \setminus \mathcal{S}_1$.

Phase 2

17: **if** $(\mathrm{s}_i^{[1]} == 1)$ **then**
18: P_i sets $\mathrm{u}_i^{[2]}(j) = \mathrm{u}_i^{[1]}(j), \forall j \in \mathcal{S}_1$.
19: P_i sets $\mathrm{u}_i^{[2]}(j) = 0, \forall j \in \mathcal{S}_0$.
20: **if** $\big(\sum_{j=1}^{n} \mathrm{u}_i^{[2]}(j) >= n-2t\big)$ **then**
21: P_i sets $\mathrm{s}_i^{[2]} = 1$.
22: **else**
23: P_i sets $\mathrm{s}_i^{[2]} = 0$ and $\boldsymbol{w}^{(i)} = \phi$.
24: **else**
25: P_i sets $\mathrm{s}_i^{[2]} = 0$.
26: P_i sends $\mathrm{s}_i^{[2]}$ to $P_j, j \in [1:n] \setminus i$.
27: **upon** receiving $n-t$ success indicators $\{\mathrm{s}_j^{[2]}\}_j$
28: P_i updates $\mathcal{S}_0$ and $\mathcal{S}_1$, based on $\{\mathrm{s}_j^{[2]}\}_j$.

Phase 3

29: **if** $(\mathrm{s}_i^{[2]} == 1)$ **then**
30: P_i sets $\mathrm{u}_i^{[3]}(j) = \mathrm{u}_i^{[2]}(j), \forall j \in \mathcal{S}_1$.
31: P_i sets $\mathrm{u}_i^{[3]}(j) = 0, \forall j \in \mathcal{S}_0$.
32: **if** $\big(\sum_{j=1}^{n} \mathrm{u}_i^{[3]}(j) >= n-2t\big)$ **then**
33: P_i sets $\mathrm{s}_i^{[3]} = 1$.
34: **else**
35: P_i sets $\mathrm{s}_i^{[3]} = 0$ and $\boldsymbol{w}^{(i)} = \phi$.
36: **else**
37: P_i sets $\mathrm{s}_i^{[3]} = 0$.
38: P_i sends $(\mathrm{s}_i^{[3]}, y_j^{(i)})$ to $P_j, j \in [1:n] \setminus i$.
39: **upon** receiving $n-t$ success indicators $\{\mathrm{s}_j^{[3]}\}_j$
40: P_i updates $\mathcal{S}_0$ and $\mathcal{S}_1$, based on $\{\mathrm{s}_j^{[3]}\}_j$.
41: **if** $(\sum_j \mathrm{s}_j^{[3]} >= n-2t)$ **then**
42: P_i sets $\mathrm{v}_i = 1$.
43: **else**
44: P_i sets $\mathrm{v}_i = 0$.
45: P_i runs the one-bit consensus with all other processors on the n votes $\{\mathrm{v}_i\}_{i=1}^n$, by using the one-bit consensus protocol from [30].
46: **if** (the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ is 1) **then**
47: P_i goes to next phase.
48: **else**
49: P_i outputs $\boldsymbol{w}^{(i)} = \phi$ as a final consensus and stops.

Phase 4

50: **if** $(\mathrm{s}_i^{[3]} == 0)$ **then**
51: $y_i^{(i)} \leftarrow \text{Majority}(\{y_i^{(j)} : j \in \mathcal{S}_1\})$.
52: P_i sends $y_i^{(i)}$ to P_j, $\forall j \in [1:n] \setminus i$.
53: **if** $(\mathrm{s}_i^{[3]} == 0)$ **then**
54: **upon** receiving $n-t$ symbols $\{y_j^{(j)}\}_j$
55: P_i decodes message and updates it into $\boldsymbol{w}^{(i)}$ based on the received symbols.
56: P_i outputs $\boldsymbol{w}^{(i)}$ as the final consensus and stops.

P_i updates $\boldsymbol{w}^{(i)}$ as $\boldsymbol{w}^{(i)} = \phi$, else keeps the original value of $\boldsymbol{w}^{(i)}$.

3) Update $\mathcal{S}_0$ and $\mathcal{S}_1$: Upon receiving $n-t$ success indicators $\{\mathrm{s}_j^{[3]}\}_j$, P_i updates the sets of $\mathcal{S}_0$ and $\mathcal{S}_1$ as in (8) for $i \in [1:n]$.

4) Vote: P_i, $i \in [1:n]$, sets a binary vote as

$$\mathrm{v}_i = \begin{cases} 1 & \text{if } \sum_j \mathrm{s}_j^{[3]} \geq n-2t \\ 0 & \text{else} \end{cases} \tag{9}$$

based on the $n-t$ received success indicators $\{\mathrm{s}_j^{[3]}\}_j$. v_i denotes a vote from P_i for stopping in this phase or going to Phase 4.

5) One-bit consensus on the n votes: $P_i, i \in [1,n]$, runs the one-bit consensus with all other processors on the n votes $\{\mathrm{v}_i\}_{i=1}^n$, by using the one-bit consensus protocol from [30]. The consensus can be reached with $O(n^2)$ bits of communication complexity, and $O(1)$ rounds of round complexity, for $t < n/5$. If the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ is 0, then every honest processor P_i outputs $\boldsymbol{w}^{(i)} = \phi$ as a final consensus and stops, else every honest processor goes to Phase 4.

Remark 4. In Phase 3, since $\mathrm{s}_i^{[3]}$ and $y_j^{(i)}$ has 1 bit and c bits respectively, the communication complexity of exchanging pairs of $(\mathrm{s}_i^{[3]}, y_j^{(i)})$, $i \in [1:n], j \in [1:n] \setminus i$ (see Line 38 in Algorithm 1), denoted by b_4, is $b_4 = (1+c)n(n-1)$ bits.

Remark 5. The work of [30] provides the most efficient unauthenticated binary ABA protocol that achieves the communication complexity $O(n^2)$ and round complexity $O(1)$,

assuming a common coin oracle. Since we run the one-bit consensus from [30], the communication complexity of invoking one-bit consensus (see Line 45 in Algorithm 1), denoted by b_5, is $b_5 = O(n^2)$ bits. Since the round complexity of this step is constant (i.e., $O(1)$) and the round complexity of other steps is also constant, it gives that the round complexity of the proposed protocol is constant, i.e., $O(1)$.

Remark 6. It can be proved that at the end of this phase, there exists at most 1 group of honest processors, where the honest processors within this group have the same non-empty updated value ($\boldsymbol{w}^{(i)} \neq \phi$), and the honest processors outside this group have the same empty updated value ($\boldsymbol{w}^{(i)} = \phi$) (see Lemma 6 in Section V-B).

D. Phase 4: exchange coded symbols and make consensus

The idea of this phase is to calibrate and update the final value of the honest processors in the set of $\mathcal{S}_0$ such that every honest processor outputs the same value as the final consensus.

1) Update symbols with majority rule: P_i, $i \in \mathcal{S}_0$, updates the value of $y_i^{(i)}$ as $y_i^{(i)} \leftarrow \text{Majority}(\{y_i^{(j)} : j \in \mathcal{S}_1\})$, where the coded symbols $\{y_i^{(j)}\}_{j\in\mathcal{S}_1}$ were received in Phase 3. Majority($\bullet$) is a function defined to return the most frequent value in the list, based on the majority rule. Note that P_i, $i \in \mathcal{S}_1$ keeps its previous value of $y_i^{(i)}$.

2) Broadcast updated symbol: $P_i, i \in [1:n]$, sends the value of $y_i^{(i)}$ to P_j, $\forall j \in [1:n] \setminus j$.

3) Decode: Upon receiving $n-t$ coded symbols $\{y_j^{(j)}\}_j$, P_i decodes message and updates it into $\boldsymbol{w}^{(i)}$ using the $n-t$ received symbols, for $i \in \mathcal{S}_0$. For P_i, $i \in \mathcal{S}_1$, it skips this step.

4) Stop: $P_i, i \in [1:n]$, outputs $\boldsymbol{w}^{(i)}$ as the final consensus and stops.

Remark 7. In Phase 4, since $y_i^{(i)}$ has c bits, the communication complexity of exchanging the coded symbols (see Line 52 in Algorithm 1), denoted by b_6, is $b_6 = cn(n-1)$ bits.

Remark 8. Note that up to $\lfloor \frac{n-k}{2} \rfloor$ Byzantine errors can be corrected in the design of an (n,k) error correction code, using some efficient decoding algorithms [47]–[49]. It can be verified that in Step 3 of this phase, every honest processor decodes and outputs the same message, with the parameter design of k and c in this protocol.

E. Performance analysis of A-COOL

This section provides the performance analysis of the proposed A-COOL protocol, as stated in Lemma 1 and Lemma 2. Lemmas 3-5 will be used for the proof of Lemma 2.

Lemma 1. *A-COOL reaches an agreement on an ℓ-bit value with the communication complexity of $O(\max\{n\ell, nt\log t\})$ bits and the round complexity of $O(1)$ rounds, given $n \geq 5t+1$.*

Proof. For the proposed A-COOL, it reaches an agreement on an ℓ-bit value given $n \geq 5t+1$ (see Lemma 2).

By summing the communication complexity $b_1, b_2, \cdots, b_6$ expressed in Remarks 1, 3, 4, 5 and 7, the total communication complexity of the proposed A-COOL, denoted by b, is computed as

$$\begin{aligned} b &= \sum_{i=1}^{6} b_i \\ &= O(cn(n-1) + n^2) \\ &= O(\max\{\ell/t, \log n\} \cdot n(n-1) + n^2) \\ &= O(\max\{\ell n^2/t, n^2 \log n\}) \quad \text{bits} \end{aligned} \tag{10}$$

where c is designed as $c = \lceil \frac{\max\{\ell,\ (t/5+1)\cdot\log(n+1)\}}{k} \rceil$ for $k = \lfloor \frac{t}{5} \rfloor + 1$. In the description of the proposed A-COOL, we just considered the case of $t \leq \frac{n-1}{5}$ and $t = \Omega(n)$. For this case, the total communication complexity shown in (10) can be rewritten as $b = O(\max\{n\ell, nt\log t\})$ bits.

For the case with relatively small t compared to n, the same communication complexity performance can be achieved. In this case, $n' = 5t+1$ processors are first selected to run the A-COOL protocol described in Section IV, by replacing n with n', and replacing c with $c' \triangleq \lceil \frac{\max\{\ell,\ (t/5+1)\cdot\log(n'+1)\}}{k} \rceil$. After the consensus, the selected n' processors will send the coded symbol of the agreed value to the rest of the processors, that is, each of the selected n' processors sends a coded symbol with c' bits. The communication complexity of this step is $n'(n-n')c'$ bits. By combining the above steps for this case, it can be computed that the total communication complexity is $b = O(\max\{\ell n, nt\log t\})$ bits.

By combining the above two cases, we conclude that the total communication complexity of A-COOL is $b = O(\max\{n\ell, nt\log t\})$ bits.

Given that the round complexity of the one-bit consensus in Phase 3 is constant, it implies that the round complexity of the proposed protocol is constant, i.e., $O(1)$. □

Lemma 2. *The proposed A-COOL is a signature-free ABA protocol, which satisfies the termination, consistency and validity conditions, given $n \geq 5t+1$.*

Proof. The proof of Lemma 2 will use tools from coding theory and graph theory. Lemmas 3-5 will be used for the proof of Lemma 2.

Lemma 3. *Given $n \geq 5t+1$, every honest processor eventually almost-surely terminates in A-COOL.*

Proof. In the proposed A-COOL protocol, given $n \geq 5t+1$, it is guaranteed that every honest processor eventually almost-surely terminates at the end of Phase 3 and outputs ϕ as a final consensus, or almost-surely terminates at the end of Phase 4 and outputs updated value $\boldsymbol{w}^{(i)}$ as a final consensus. □

Lemma 4. *Given $n \geq 5t+1$, all of the honest processors decide on the same value in A-COOL.*

Proof. See Section V-B. □

Lemma 5. *Given $n \geq 5t+1$, if every honest processor holds an initial value $\boldsymbol{w}$, then at the end of A-COOL all of the honest processors eventually decide on this initial value $\boldsymbol{w}$.*

Proof. See Section V-C. □

From Lemma 3, Lemma 4, and Lemma 5, we can conclude that given $n \geq 5t+1$, the termination, consistency and validity conditions are satisfied in the proposed A-COOL protocol. At this point, we complete the proof of Lemma 2. □

V. Analysis on consistency and validity properties of A-COOL

This section will provide an analysis of the consistency and validity properties of the A-COOL protocol. Specifically, we will prove Lemma 4 and Lemma 5 in Section V-B and Section V-C, respectively. Let us first define network groups in Section V-A.

A. Groups

We first define some groups over n distributed processors, based on the initial values of processors and the values of success indicators in each phase. We define the indices of all of the dishonest processors as Group $\mathcal{F}$, with $|\mathcal{F}| = t$. We also define some groups of honest processors as

$$\mathcal{A}_l \triangleq \{i : \boldsymbol{w}_i = \bar{\boldsymbol{w}}_l, i \notin \mathcal{F}, i \in [1:n]\},\ l \in [1:\eta] \tag{11}$$

$$\mathcal{A}_l^{[p]} \triangleq \{i : \mathrm{s}_i^{[p]} = 1, \boldsymbol{w}_i = \bar{\boldsymbol{w}}_l, i \notin \mathcal{F}, i \in [1:n]\}, l \in [1:\eta^{[p]}] \tag{12}$$

$$\mathcal{B}^{[p]} \triangleq \{i : \mathrm{s}_i^{[p]} = 0, i \notin \mathcal{F}, i \in [1:n]\} \tag{13}$$

for $p \in \{1,2,3\}$, where $\bar{\boldsymbol{w}}_1, \bar{\boldsymbol{w}}_2, \cdots, \bar{\boldsymbol{w}}_\eta$ are different non-empty ℓ-bit values and $\eta, \eta^{[1]}, \eta^{[2]}, \eta^{[3]}$ are non-negative integers with $\eta^{[3]} \leq \eta^{[2]} \leq \eta^{[1]} \leq \eta$. Group $\mathcal{A}_l$ defined in (11) denotes a subset of honest processors who have the same initial value for $l \in [1:\eta]$. Group $\mathcal{A}_l^{[p]}$ defined in (12) denotes a set of honest processors who have the same non-empty updated value at the end of Phase p for $l \in [1:\eta^{[p]}]$ and $p \in \{1,2,3\}$. Group $\mathcal{B}^{[p]}$ defined in (13) denotes a set of honest processors whose success indicators are 0 at the end of Phase p for $p \in \{1,2,3\}$.

Based on the definitions of (11)-(13), it holds true that

$$\sum_{l=1}^{\eta} |\mathcal{A}_l| + |\mathcal{F}| = n \tag{14}$$

$$\sum_{l=1}^{\eta^{[p]}} |\mathcal{A}_l^{[p]}| + |\mathcal{B}^{[p]}| + |\mathcal{F}| = n, \quad p \in \{1,2,3\}. \tag{15}$$

For some $i \in \mathcal{A}_l$, if $(\bar{\boldsymbol{w}}_l - \bar{\boldsymbol{w}}_j)$ is in the null space of $\boldsymbol{h}_i$, then $\boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_l = \boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_j$ will be satisfied for some different l and j. Based on this scenario, we further divide Group $\mathcal{A}_l$ and Group $\mathcal{A}_l^{[p]}, p \in \{1,2,3\}$ into some sub-groups defined by

$$\mathcal{A}_{l,j} \triangleq \{i : i \in \mathcal{A}_l, \boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_l = \boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_j\}, j \neq l, j, l \in [1:\eta] \tag{16}$$

$$\mathcal{A}_{l,l} \triangleq \mathcal{A}_l \setminus \{\cup_{j=1, j\neq l}^{\eta} \mathcal{A}_{l,j}\},\ l \in [1:\eta] \tag{17}$$

$$\mathcal{A}_{l,j}^{[p]} \triangleq \{i : i \in \mathcal{A}_l^{[p]}, \boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_l = \boldsymbol{h}_i^\mathsf{T}\bar{\boldsymbol{w}}_j\}, j \neq l, j, l \in [1:\eta^{[p]}] \tag{18}$$

$$\mathcal{A}_{l,l}^{[p]} \triangleq \mathcal{A}_l^{[p]} \setminus \{\cup_{j=1, j\neq l}^{\eta^{[p]}} \mathcal{A}_{l,j}^{[p]}\},\ l \in [1:\eta^{[p]}]. \tag{19}$$

B. Proof of Lemma 4

In this sub-section, we will provide the proof of Lemma 4. Specifically, given $n \geq 5t+1$, we will prove that all of the honest processors decide on the same value in A-COOL. We first provide Lemma 6, which will be used in the proof of Lemma 4.

Lemma 6. *In the A-COOL protocol with $n \geq 5t+1$, at the end of Phase 3 there exists at most 1 group of honest processors, where the honest processors within this group have the same non-empty updated value, and the honest processors outside this group have the same empty updated value.*

Proof. Note that proving Lemma 6 is equivalent to proving $\eta^{[3]} \leq 1$ (see the definition in (12)). We first provide Lemmas 7 and 8, which will be used to prove the inequality of $\eta^{[3]} \leq 1$.

Lemma 7. *It is true that $\eta^{[2]} \leq 2$ in A-COOL with $n \geq 5t+1$.*

Proof. See Appendix A. □

Lemma 8. *It is true that $\eta^{[3]} \leq 1$ when $\eta^{[2]} = 2$, given $n \geq 5t+1$ in A-COOL.*

Proof. See Appendix B. □

The result of Lemma 7 reveals that $\eta^{[2]} \leq 2$ must be satisfied. It is apparently true that $\eta^{[3]} \leq 1$ when $\eta^{[2]} < 2$, due to the fact that $\eta^{[3]} \leq \eta^{[2]}$ (see (11)-(12)). Moreover, the result of Lemma 8 reveals that $\eta^{[3]} \leq 1$ also holds true when $\eta^{[2]} = 2$. At this point, it can be concluded that $\eta^{[3]} \leq 1$ in A-COOL with $n \geq 5t+1$, which completes the proof of Lemma 6. □

In the Phase 3 of A-COOL, all processors run the one-bit consensus on the n votes $\{\mathrm{v}_i\}_{i=1}^n$, by using the one-bit consensus protocol from [30]. Based on the consensus of the votes, two cases will be considered for proving Lemma 4.

In the first case where the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ is 0, each honest processor P_i outputs $\boldsymbol{w}^{(i)} = \phi$ as a final consensus and stops in Phase 3, for $i \in [1:n]$. It implies that all of the honest processors decide on the same value.

In the second case where the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ is 1, each honest processor P_i goes to Phase 4, for $i \in [1:n]$. In this case, at least one honest processor P_i votes $\mathrm{v}_i = 1$, for $i \notin \mathcal{F}$. Otherwise, if all of the honest processors vote 0, the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ should be 0, which contradicts the condition of this case. For the honest processor P_i who votes $\mathrm{v}_i = 1$, the received success indicators satisfy

$$\sum_j \mathrm{s}_j^{[3]} \geq n - 2t. \tag{20}$$

It implies that at least $n - 3t$ honest processors send success indicators as ones in Phase 3, i.e.,

$$\sum_{j \in \cup_{l=1}^{\eta^{[3]}} \mathcal{A}_l^{[3]}} \mathrm{s}_j^{[3]} \geq n - 3t. \tag{21}$$

The results of (21) and Lemma 6 imply that at the end of Phase 3 there exists exactly 1 group of honest processors with

size at least $n-3t$, where the honest processors within this group have the same non-empty updated value, and the honest processors outside this group have the same empty updated value. In other words, in this case, it holds true that

$$\eta^{[3]} = 1 \tag{22}$$

$$|\mathcal{A}_1^{[3]}| \geq n - 3t \tag{23}$$

$$\boldsymbol{w}_i = \bar{\boldsymbol{w}}_1, \quad \forall i \in \mathcal{A}_1^{[3]} \tag{24}$$

where (22) and (23) follow from (21) and Lemma 6; (24) is from the definition of $\mathcal{A}_l^{[3]}$ in (12).

Based on the symbols of $y_i^{(j)}$ received in Phase 3, P_i, $i \in \mathcal{S}_0$, updates the value of $y_i^{(i)}$ in the first step of Phase 4 as $y_i^{(i)} \leftarrow \text{Majority}(\{y_i^{(j)} : j \in \mathcal{S}_1\})$ using the majority rule. In this step, it is true that $\mathcal{A}_1^{[3]} \subseteq \mathcal{S}_1$ and $|\mathcal{A}_1^{[3]}| > |\mathcal{F}|$, which guarantees that P_i could update the value of $y_i^{(i)}$ as

$$y_i^{(i)} \leftarrow \text{Majority}(\{y_i^{(j)} : j \in \mathcal{S}_1\}) = \boldsymbol{h}_i^{\intercal}\bar{\boldsymbol{w}}_1,$$

for $i \in \mathcal{S}_0 \setminus \mathcal{F}$. At the end of this step, the value $y_i^{(i)}$ of P_i is updated to $y_i^{(i)} = \boldsymbol{h}_i^{\intercal}\bar{\boldsymbol{w}}_1$, for $i \notin \mathcal{F}$. In the second step of Phase 4, P_i, $i \in [1:n]$, sends the updated value of $y_i^{(i)}$ to P_j, $\forall j \in [1:n] \setminus i$. In the third step of Phase 4, P_i, $i \in \mathcal{S}_0$ decodes its message, with the received $n-t$ symbols $\{y_j^{(j)}\}_j$. It is guaranteed that $n-2t$ out of $n-t$ symbols are expressed as $y_j^{(j)} = \boldsymbol{h}_j^{\intercal}\bar{\boldsymbol{w}}_1, \forall j \notin \mathcal{F}$. Since the number of symbols that are not encoded with the message $\bar{\boldsymbol{w}}_1$ is no more than t, P_i could decode its message and update it as $\boldsymbol{w}^{(i)} = \bar{\boldsymbol{w}}_1$ in this step, for $i \in \mathcal{S}_0$. Meanwhile, P_i keeps its original value and outputs $\boldsymbol{w}^{(i)} = \bar{\boldsymbol{w}}_1$, for $i \in \mathcal{S}_1$. Therefore, at the end of Phase 4, all of the honest processors decide on the same value, i.e., $\boldsymbol{w}^{(i)} = \bar{\boldsymbol{w}}_1$, $\forall i \notin \mathcal{F}$. At this point, we complete the proof of Lemma 4.

C. Proof of Lemma 5

In this subsection, we will provide the proof of Lemma 5. Specifically, we will prove that given $n \geq 5t+1$, if every honest processor holds an initial value $\boldsymbol{w}$, then at the end of A-COOL all of the honest processors eventually decide on this initial value $\boldsymbol{w}$.

In the scenario that every honest processor holds an initial value $\boldsymbol{w}$, from Phase 1 to Phase 3, all honest processors will set their success indicators as ones and keep their updated value exactly the same as the initial value $\boldsymbol{w}$. In Phase 3, the consensus of the votes $\{\mathrm{v}_i\}_{i=1}^n$ should be 1 and all honest processors will go to Phase 4. Given the same updated value $\boldsymbol{w}$ and the same nonzero success indicator as inputs at all of the honest processors, in Phase 4 all honest processors will output the values that are exactly the same as the initial value $\boldsymbol{w}$. Therefore, for this scenario, all of the honest processors eventually decide on the initial value $\boldsymbol{w}$. At this point we complete the proof of Lemma 5.

VI. Conclusion

In this work, we studied the ABA problem and proposed a new signature-free protocol, i.e., A-COOL, which is communication efficient. Specifically, the proposed A-COOL protocol achieves the optimal communication complexity $O(n\ell)$ when $\ell \geq t\log t$, given $n \geq 5t+1$. The proposed A-COOL protocol can also be extended to the asynchronous Byzantine broadcast (ABB) problem, by adding one step of broadcasting ℓ-bit message from the leader to the distributed processors. The ℓ-bit message sent from the leader will serve as the initial value at every distributed node in the ABA problem. Then, the proposed A-COOL can be applied here to achieve the same performance as in the ABA setting, in terms of resilience, communication complexity and round complexity.

Appendix

A. Proof of Lemma 7

In this sub-section, we will provide the proof of Lemma 7. Specifically, we will prove that $\eta^{[2]} \leq 2$ holds true in A-COOL with $n \geq 5t+1$. Before proving Lemma 7, we first provide Lemma 9 that will be used in the proof.

Lemma 9. *When $\eta^{[2]} \geq 1$, it is true that $|\mathcal{A}_l| \geq n - 29t/9$, for any $l \in [1:\eta^{[2]}]$.*

Proof. See Appendix C. □

We will use proof by contradiction technique in the proof of Lemma 7. Let us first assume

$$\eta^{[2]} \geq 3. \tag{25}$$

Given this assumption, the result of Lemma 9 reveals that

$$|\mathcal{A}_l| \geq n - 29t/9 \quad \forall l \in [1:\eta^{[2]}]. \tag{26}$$

By summing the bounds $|\mathcal{A}_l|, \forall l \in [1:\eta^{[2]}]$ in (26), it gives

$$\begin{aligned}\sum_{l=1}^{\eta^{[2]}} |\mathcal{A}_l| &\geq \eta^{[2]}(n - 29t/9) \\ &\geq 3(n - 29t/9) \qquad (27)\\ &= (n-t) + (2n - 26t/3) \\ &> (n-t) \qquad (28)\end{aligned}$$

where (27) follows from the assumption of $\eta^{[2]} \geq 3$; and (28) uses the condition of $n \geq 5t+1$, implying that $2n - 26t/3 > 0$.

The result of (28) reveals that $\sum_{l=1}^{\eta^{[2]}} |\mathcal{A}_l| > n-t$, which contradicts with the identify of $\sum_{l=1}^{\eta^{[2]}} |\mathcal{A}_l| \leq \sum_{l=1}^{\eta} |\mathcal{A}_l| = n-t$ (see (14)), i.e., in total there are $n-t$ honest distributed processors. Thus, we conclude that the assumption in (25) leads to a contradiction, which implies that $\eta^{[2]}$ is bounded by

$$\eta^{[2]} \leq 2.$$

At this point, we complete the proof of Lemma 7.

B. Proof of Lemma 8

This sub-section provides the proof of Lemma 8. Specifically, we will prove that $\eta^{[3]} \leq 1$ holds true when $\eta^{[2]} = 2$, given $n \geq 5t+1$ in A-COOL. This proof will use the results of Lemmas 10 and 12 provided as below.

Lemma 10. *It is true that $\eta^{[1]} = 2$ when $\eta^{[2]} = 2$, given $n \geq 5t+1$ in A-COOL.*

Proof. The proof uses the results from Lemma 9 and [16, Lemma 7] (see its statement in Lemma 11). Proof by contradiction technique is also used in this proof.

Lemma 11. *[16, Lemma 7] For $\mathcal{A}_{l,j}$ and $\mathcal{A}_{l,j}^{[1]}$ defined in* (16) *and* (18)*, and for $\eta \geq \eta^{[1]} \geq 2$, the following inequalities hold true*

$$|\mathcal{A}_{l,j}| + |\mathcal{A}_{j,l}| < k, \quad \forall j \neq l,\ j,l \in [1:\eta] \tag{29}$$

$$|\mathcal{A}_{l,j}^{[1]}| + |\mathcal{A}_{j,l}^{[1]}| < k, \quad \forall j \neq l,\ j,l \in [1:\eta^{[1]}] \tag{30}$$

where k is designed as $k \triangleq \lfloor \frac{t}{5} \rfloor + 1$.

When $\eta^{[2]} = 2$, we have $\eta^{[1]} \geq 2$, due to the fact that $\eta^{[1]} \geq \eta^{[2]}$. To prove $\eta^{[1]} = 2$ given $\eta^{[2]} = 2$, we will assume $\eta^{[1]} > 2$ and prove that this assumption will lead to a contradiction. When $\eta^{[2]} = 2$, it holds true that

$$\sum_{l=3}^{\eta} |\mathcal{A}_l| = n - |\mathcal{F}| - \sum_{l=1}^{2} |\mathcal{A}_l| \tag{31}$$

$$\leq n - t - 2(n - 29t/9) \tag{32}$$

$$\leq 4t/9 - 1 \tag{33}$$

where (31) is from (14); (32) follows from Lemma 9; and (33) uses the condition of $n \geq 5t + 1$.

Given the assumption of $\eta^{[1]} > 2$, there exists an $i^\star \in \mathcal{A}_{l^\star}^{[1]} \subseteq \mathcal{A}_{l^\star}$ for $l^\star \geq 3$, such that $\mathrm{s}_{i^\star}^{[1]} = 1$ (see (12)). It implies that

$$\sum_{j \in \cup_{l=1}^{\eta} \mathcal{A}_l} \mathrm{u}_{i^\star}^{[1]}(j) \geq n - 2t - t$$

(see (7)), which can be rewritten as

$$\sum_{j \in \mathcal{A}_{l^\star} \cup \{\cup_{l=1, l\neq l^\star}^{\eta} \mathcal{A}_{l,l^\star}\}} \mathrm{u}_{i^\star}^{[1]}(j) \geq n - 2t - t, \tag{34}$$

due to the fact that $\mathrm{u}_{i^\star}^{[1]}(j) = 0$, $\forall j \in \{\cup_{l=1}^{\eta} \mathcal{A}_l\} \setminus \{\mathcal{A}_{l^\star} \cup \{\cup_{l=1, l\neq l^\star}^{\eta} \mathcal{A}_{l,l^\star}\}\}$. For $i^\star \in \mathcal{A}_{l^\star}^{[1]} \subseteq \mathcal{A}_{l^\star}$ and $l^\star \geq 3$, the size of $\mathcal{A}_{l^\star} \cup \{\cup_{l=1, l\neq l^\star}^{\eta} \mathcal{A}_{l,l^\star}\}$ can be bounded by

$$|\mathcal{A}_{l^\star} \cup \{\cup_{l=1, l\neq l^\star}^{\eta} \mathcal{A}_{l,l^\star}\}| \leq \sum_{l=1}^{2} |\mathcal{A}_{l,l^\star}| + \sum_{l=3}^{\eta} |\mathcal{A}_l| \tag{35}$$

$$\leq 2(k-1) + 4t/9 - 1 \tag{36}$$

$$= 2\lfloor t/5 \rfloor + 4t/9 - 1 \tag{37}$$

$$< n - 3t \tag{38}$$

where (36) is from the result in Lemma 11 (see (29)) and the result in (33); (37) is from $k = \lfloor \frac{t}{5} \rfloor + 1$. Note that the result of (38) contradicts the conclusion in (34). Thus, we conclude that the assumption of $\eta^{[1]} > 2$ leads to a contradiction. It implies that $\eta^{[1]}$ is bounded by $\eta^{[1]} = 2$ when $\eta^{[2]} = 2$, which completes the proof of Lemma 10.

□

Lemma 12. *It is true that $\eta^{[3]} \leq 1$ if $\eta^{[1]} = 2$, given $n \geq 5t+1$ in A-COOL.*

Proof. See Appendix D. □

Based on the results of Lemmas 10 and 12, now we can prove Lemma 8. Given $\eta^{[2]} = 2$, Lemma 10 reveals that $\eta^{[1]} = 2$. Moreover, given $\eta^{[1]} = 2$, Lemma 12 reveals that $\eta^{[3]} \leq 1$. Therefore, it can be concluded that $\eta^{[3]} \leq 1$ when $\eta^{[2]} = 2$. At this point we complete the proof of Lemma 8.

C. Proof of Lemma 9

In this sub-section we will provide the proof of Lemma 9. Specifically, when $\eta^{[2]} \geq 1$, we will prove that $|\mathcal{A}_l| \geq n - 29t/9$ holds true, for any $l \in [1 : \eta^{[2]}]$. Before proving Lemma 9, we will first prove Lemma 13 (see below), whose result will be used in the proof of Lemma 9. The proofs of Lemmas 9 and 13 follow closely from the proofs of Lemmas 9 and 8 in the work of [16] (see Section VI-D and VI-C in [16]), respectively. The proofs borrow tools from graph theory.

We will consider a graph $G = (\mathcal{P}, \mathcal{E})$, where $\mathcal{P} = [1 : n-t]$ is a set of $n - t$ vertices and $\mathcal{E}$ is a set of edges. In this graph, there exists a set $\mathcal{C} \subseteq \mathcal{P} \setminus \{i^\star\}$ such that the following conditions are satisfied:

$$E_{i^\star,i} = 1 \quad \forall i \in \mathcal{C} \tag{39}$$

$$\sum_{j \in \mathcal{P}} E_{i,j} \geq n - 3t \quad \forall i \in \mathcal{C} \tag{40}$$

$$|\mathcal{C}| \geq n - 3t - 1 \tag{41}$$

for a given vertex $i^\star \in \mathcal{P}$, where $E_{i,j}$ is defined as $E_{i,j} = 1$ if there is an edge between vertex i and vertex j, else $E_{i,j} = 0$. We define $\mathcal{D} \subseteq \mathcal{P}$ as the set of vertices, in which each vertex in $\mathcal{D}$ is connected with at least k vertices in $\mathcal{C}$, i.e.,

$$\mathcal{D} \triangleq \Big\{ i :\ \sum_{j \in \mathcal{C}} E_{j,i} \geq k,\ i \in \mathcal{P} \setminus \{i^\star\} \Big\} \tag{42}$$

for $k = \lfloor \frac{t}{5} \rfloor + 1$. The following lemma provides a bound on the size of $\mathcal{D}$.

Lemma 13. *For any graph $G = (\mathcal{P}, \mathcal{E})$ specified by* (39)-(41) *and for the set $\mathcal{D} \subseteq \mathcal{P}$ defined by* (42)*, it is true that $|\mathcal{D}| \geq n - 29t/9 - 1$, given $n \geq 5t + 1$.*

Proof. For a graph $G = (\mathcal{P}, \mathcal{E})$ specified by (39)-(41), the number of edges connected between $\mathcal{C}$ and $\mathcal{P} \setminus \{i^\star\}$ is bounded by

$$m_e \geq |\mathcal{C}| \cdot (n - 3t - 1). \tag{43}$$

The idea of bounding the size of $\mathcal{D}$ defined as in (42) follows from the work of [16] (see VI-C in [16]). We consider the same extreme scenario, as described in [16], that has the minimum size of $\mathcal{D}$. Let $\mathcal{D}^*$ be the set defined as in (42) of the considered extreme scenario, with size $|\mathcal{D}^*|$. By considering the extreme scenario, it holds true that

$$(k-1)(n - t - 1 - |\mathcal{D}^*|) + \tau(|\mathcal{D}^*| - 1) + \tau_0 = m_e \tag{44}$$

for $\tau \triangleq |\mathcal{C}|$ and for some τ_0 such that $k \leq \tau_0 \leq \tau$. Therefore, from (44) we can bound $|\mathcal{D}^*|$ as

$$|\mathcal{D}^*| = \frac{m_e + \tau - \tau_0 - (k-1)(n-t-1)}{\tau - (k-1)} \tag{45}$$

$$\geq \frac{m_e - (k-1)(n-t-1)}{\tau - (k-1)} \tag{46}$$

$$\geq \frac{\tau(n-3t-1) - 2(k-1)(n-t-1)}{\tau - (k-1)} \tag{47}$$

$$= (n-3t-1) - \frac{2(k-1)t}{\tau - (k-1)}$$

$$\geq n - 3t - 1 - \frac{2(k-1)t}{n-3t-1-(k-1)}$$

$$\geq n - 29t/9 - 1 \tag{48}$$

for $\tau = |\mathcal{C}|$ and $k = \lfloor \frac{t}{5} \rfloor + 1$, where (46) stems from $\tau_0 \leq \tau$; (47) is from (43). Since $|\mathcal{D}^*|$ is the size of $\mathcal{D}^*$ of the extreme scenario, we can bound $|\mathcal{D}|$ as $|\mathcal{D}| \geq |\mathcal{D}^*| \geq n - 29t/9 - 1$, for any other scenario of the graph $G = (\mathcal{P}, \mathcal{E})$ that satisfy the conditions (39)-(41). At this point we complete the proof of Lemma 13.

□

Now let us focus on the proof of Lemma 9. This proof follows from the proof of Lemma 9 in the work of [16] (see Section VI-D in [16]). The proof includes four steps, which are provided as below.

Step (a): In this step, we transform the distributed network into a graph that is within the family of graphs $G = (\mathcal{P}, \mathcal{E})$ specified by (39)-(41). Specifically, when $\eta^{[2]} \geq 1$, considering a fixed $i^\star$ for $i^\star \in \mathcal{A}_{l^\star}^{[2]}$ and $l^\star \in [1 : \eta^{[2]}]$, we define a subset of $\{\cup_{p=1}^{\eta^{[1]}} \mathcal{A}_p^{[1]}\} \setminus \{i^\star\}$ of honest processors as

$$\mathcal{C}' \triangleq \{j : \mathrm{u}_{i^\star}^{[1]}(j) = 1, j \in \{\cup_{p=1}^{\eta^{[1]}} \mathcal{A}_p^{[1]}\} \setminus \{i^\star\}\}. \tag{49}$$

By following the proof idea of Lemma 9 in [16], the following conclusions hold true:

$$\mathrm{u}_{i^\star}^{[1]}(j) = 1, \quad \forall j \in \mathcal{C}' \tag{50}$$

$$\sum_{i \in \cup_{l=1}^{\eta} \mathcal{A}_l} \mathrm{u}_j^{[1]}(i) \geq n - 3t, \quad \forall j \in \mathcal{C}' \tag{51}$$

$$|\mathcal{C}'| \geq n - 3t - 1. \tag{52}$$

Then we map the distributed network into a graph $G = (\mathcal{P}', \mathcal{E}')$, where $\mathcal{P}'$ is a set of $n-t$ honest processors (vertices), and $\mathcal{E}'$ is a set of edges. In this graph, we have $\mathcal{P}' \triangleq \cup_{l=1}^{\eta} \mathcal{A}_l$ and $E_{i,j} = \mathrm{u}_i^{[1]}(j), \forall i, j \in \mathcal{P}'$. In the graph $G = (\mathcal{P}', \mathcal{E}')$, for a given $i^\star \in \mathcal{A}_{l^\star}^{[2]} \subseteq \mathcal{P}'$, there exists a set $\mathcal{C}' \subseteq \mathcal{P}' \setminus \{i^\star\}$ such that the conditions in (50)-(52) (Similar to the conditions in (39)-(41)) are satisfied. This graph $G = (\mathcal{P}', \mathcal{E}')$ falls into a family of graphs satisfying (39)-(41).

Step (b): In this step, we will bound the size of $\mathcal{D}'$, which is defined as

$$\mathcal{D}' \triangleq \Big\{ i : \sum_{j \in \mathcal{C}'} \mathrm{u}_j^{[1]}(i) \geq k, \ i \in \{\cup_{l=1}^{\eta} \mathcal{A}_l\} \setminus \{i^\star\} \Big\} \tag{53}$$

(similar to the definition of $\mathcal{D}$ in (42)), where k is defined as $k \triangleq \lfloor \frac{t}{5} \rfloor + 1$. Since the graph $G = (\mathcal{P}', \mathcal{E}')$ falls into a family of graphs satisfying (39)-(41), the result of Lemma 13 reveals that the size of $\mathcal{D}'$ is bounded by

$$|\mathcal{D}'| \geq n - 29t/9 - 1. \tag{54}$$

Step (c): In this step, by following the similar argument of Step (c) in Section VI-D of [16]. It holds true that

$$\boldsymbol{w}_i = \boldsymbol{w}_{i^\star} \quad \forall i \in \mathcal{D}'. \tag{55}$$

That is, every processor in $\mathcal{D}'$ has the same initial value as $P_{i^\star}$, for $i^\star \in \mathcal{A}_{l^\star}^{[2]}$ and $l^\star \in [1 : \eta^{[2]}]$.

Step (d): In this step, we will bound the size of $\mathcal{A}_{l^\star}$, i.e., $|\mathcal{A}_{l^\star}|$, for $l^\star \in [1 : \eta^{[2]}]$. The result of (55) reveals that $\mathcal{D}' \cup \{i^\star\} \subseteq \mathcal{A}_{l^\star}$, for $i^\star \in \mathcal{A}_{l^\star}^{[2]}$ and $l^\star \in [1 : \eta^{[2]}]$. Therefore, $|\mathcal{A}_{l^\star}|$ can be bounded by

$$|\mathcal{A}_{l^\star}| \geq |\mathcal{D}'| + 1 \geq n - 29t/9$$

for $l^\star \in [1 : \eta^{[2]}]$. At this point we complete the proof of Lemma 9.

D. Proof of Lemma 12

In this sub-section we provide the proof of Lemma 12. Specifically, we will prove that $\eta^{[3]} \leq 1$ holds true if $\eta^{[1]} = 2$, given $n \geq 5t + 1$. The proof follows from the proof of Lemma 10 in the work of [16] (see Section VI-E in [16]).

Given $\eta^{[1]} = 2$, let us first focus on the case with two conditions: $|\mathcal{A}_1^{[1]}| + |\mathcal{B}^{[1]}| \geq 2t+1$ and $|\mathcal{A}_2^{[1]}| + |\mathcal{B}^{[1]}| < 2t+1$. By following the proof steps of Lemma 10 in [16], in the first step of Phase 2, it holds true that

$$\mathrm{u}_i^{[2]}(j) = 0, \quad \forall j \in \mathcal{A}_1 \cup \mathcal{B}^{[1]}, \ i \in \mathcal{A}_{2,2}^{[1]}. \tag{56}$$

Since $|\mathcal{A}_1^{[1]}| + |\mathcal{B}^{[1]}| \geq 2t + 1$, it implies that P_i sets

$$\mathrm{s}_i^{[2]} = 0, \quad \forall i \in \mathcal{A}_{2,2}^{[1]} \tag{57}$$

in the second step of Phase 2 (see (7)). Then, after exchanging the success indicators and updating the sets of $\mathcal{S}_0$ and $\mathcal{S}_1$, it is true that

$$\mathcal{B}^{[1]} \cup \mathcal{A}_{2,2}^{[1]} \subseteq \mathcal{S}_0. \tag{58}$$

At the end of Phase 2, a subset of $\mathcal{A}_{2,1}^{[1]}$ might be in the list of $\mathcal{S}_0$ (i.e., $\mathcal{A}_{2,1}^{[1]} \cap \{i : \mathrm{s}_i^{[2]} = 0\} \subseteq \mathcal{S}_0$), while the rest of $\mathcal{A}_{2,1}^{[1]}$ is still in list of $\mathcal{S}_1$ (i.e., $\mathcal{A}_{2,1}^{[1]} \cap \{i : \mathrm{s}_i^{[2]} = 1\} \subseteq \mathcal{S}_1$). We will show that the complete set $\mathcal{A}_{2,1}^{[1]} \subseteq \mathcal{S}_0$ holds true at the end of Phase 3. By following the similar argument with proof steps of Lemma 10 in [16], in the first step of Phase 3, it is true that

$$\mathrm{u}_i^{[3]}(j) = 0, \forall j \in \mathcal{A}_{1,1}^{[1]} \cup \mathcal{A}_{2,2}^{[1]} \cup \mathcal{B}^{[1]}, i \in \mathcal{A}_{2,1}^{[1]} \cap \{i : \mathrm{s}_i^{[2]} = 1\}. \tag{59}$$

The size of $\mathcal{A}_{1,1}^{[1]} \cup \mathcal{A}_{2,2}^{[1]} \cup \mathcal{B}^{[1]}$ can be bounded by

$$|\mathcal{A}_{1,1}^{[1]} \cup \mathcal{A}_{2,2}^{[1]} \cup \mathcal{B}^{[1]}| = |\mathcal{A}_{1,1}^{[1]}| + |\mathcal{A}_{2,2}^{[1]}| + |\mathcal{B}^{[1]}| \tag{60}$$

$$= n - |\mathcal{F}| - |\mathcal{A}_{1,2}^{[1]}| - |\mathcal{A}_{2,1}^{[1]}| \tag{61}$$

$$\geq n - |\mathcal{F}| - (k-1) \tag{62}$$

$$\geq 4t + 1 - (k-1) \tag{63}$$

$$\geq 3t + 1 \tag{64}$$

where (60) and (61) use the disjoint property of sets; (62) is from Lemma 11, which implies that $|\mathcal{A}_{1,2}^{[1]}| + |\mathcal{A}_{2,1}^{[1]}| \leq k-1$; (64) is from the fact that $t \geq k-1$ based on $k = \lfloor \frac{t}{5} \rfloor + 1$. The results of (59) and (64) imply that

$$\mathrm{s}_i^{[3]} = 0, \quad \forall i \in \mathcal{A}_{2,1}^{[1]} \cap \{i : \mathrm{s}_i^{[2]} = 1\}, \tag{65}$$

that is $\mathcal{A}_{2,1}^{[1]} \cap \{i : \mathrm{s}_i^{[2]} = 1\} \subseteq \mathcal{S}_0$. Therefore, at this point we have $\mathcal{A}_{2,1}^{[1]} \subseteq \mathcal{S}_0$. Since we have $\mathcal{A}_{2,1}^{[1]} \subseteq \mathcal{S}_0$ and $\mathcal{A}_{2,2}^{[1]} \subseteq \mathcal{S}_0$ (see (58)), as well as $\mathcal{A}_2^{[1]} = \mathcal{A}_{2,1}^{[1]} \cup \mathcal{A}_{2,2}^{[1]}$, then it holds true that

$$\mathcal{A}_2^{[1]} \subseteq \mathcal{S}_0 \tag{66}$$

at the end of Phase 3, which implies that $\eta^{[3]} \leq 1$.

By following the similar proof steps of the above case, one can prove that $\eta^{[3]} \leq 1$ if $\eta^{[1]} = 2$ for the case with $(|\mathcal{A}_1^{[1]}| + |\mathcal{B}^{[1]}| < 2t+1,\ |\mathcal{A}_2^{[1]}| + |\mathcal{B}^{[1]}| \geq 2t+1)$, and the case with $(|\mathcal{A}_1^{[1]}| + |\mathcal{B}^{[1]}| \geq 2t+1,\ |\mathcal{A}_2^{[1]}| + |\mathcal{B}^{[1]}| \geq 2t+1)$. One can also prove that the case with $(|\mathcal{A}_1^{[1]}| + |\mathcal{B}^{[1]}| < 2t+1,\ |\mathcal{A}_2^{[1]}| + |\mathcal{B}^{[1]}| < 2t+1)$ does not exit. At this point we complete the proof of Lemma 12.

REFERENCES

[1] M. Pease, R. Shostak, and L. Lamport, "Reaching agreement in the presence of faults," *Journal of the ACM*, vol. 27, no. 2, pp. 228–234, Apr. 1980.

[2] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," *ACM Transactions on Programming Languages and Systems (TOPLAS)*, vol. 4, no. 3, pp. 382–401, Jul. 1982.

[3] M. Castro and B. Liskov, "Practical Byzantine fault tolerance," in *Proceedings of the third symposium on Operating systems design and implementation*, Feb. 1999, pp. 173–186.

[4] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling Byzantine agreements for cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*, Oct. 2017, pp. 51–68.

[5] D. Dolev and H. Strong, "Authenticated algorithms for Byzantine agreement," *SIAM Journal on Computing*, vol. 12, no. 4, pp. 656–666, Nov. 1983.

[6] M. Rabin, "Randomized byzantine generals," in *24th Annual Symposium on Foundations of Computer Science*, Nov. 1983.

[7] P. Feldman and S. Micali, "Byzantine agreement in constant expected time," in *26th Annual Symposium on Foundations of Computer Science*, Oct. 1985.

[8] D. Dolev and R. Reischuk, "Bounds on information exchange for Byzantine agreement," *Journal of the ACM*, vol. 32, no. 1, pp. 191–204, Jan. 1985.

[9] P. Feldman and S. Micali, "Optimal algorithms for Byzantine agreement," in *20th annual ACM symposium on Theory of computing*, Jan. 1988.

[10] C. Cachin and S. Tessaro, "Asynchronous verifiable information dispersal," in *IEEE Symposium on Reliable Distributed Systems (SRDS)*, Oct. 2005.

[11] J. Katz and C. Koo, "On expected constant-round protocols for Byzantine agreement," *Journal of Computer and System Sciences*, vol. 75, no. 2, pp. 91–112, Feb. 2009.

[12] Y. Lu, Z. Lu, Q. Tang, and G. Wang, "Dumbo-MVBA: Optimal multi-valued validated asynchronous Byzantine agreement, revisited," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Jul. 2020, pp. 129–138.

[13] B. Coan and J. Welch, "Modular construction of a Byzantine agreement protocol with optimal message bit complexity," *Information and Computation*, vol. 97, no. 1, pp. 61–85, Mar. 1992.

[14] P. Berman, J. Garay, and K. Perry, "Bit optimal distributed consensus," *Computer Science*, pp. 313–321, 1992.

[15] J. Garay and Y. Moses, "Fully polynomial Byzantine agreement for $n > 3t$ processors in $t+1$ rounds," *SIAM Journal on Computing*, vol. 27, no. 1, pp. 247–290, 1998.

[16] J. Chen, "Fundamental limits of Byzantine agreement," 2020, available on: arXiv:2009.10965v2.

[17] B. Chor and B. Coan, "A simple and efficient randomized Byzantine agreement algorithm," *IEEE Transactions on Software Engineering*, vol. 11, no. 6, pp. 531–539, Jun. 1985.

[18] B. Pfitzmann and M. Waidner, "Information-theoretic pseudosignatures and Byzantine agreement for $t \geq n/3$," IBM Research Report, 1996.

[19] P. Feldman and S. Micali, "An optimal probabilistic protocol for synchronous Byzantine agreement," *SIAM Journal on Computing*, vol. 26, no. 4, pp. 873–933, Aug. 1997.

[20] I. Abraham, S. Devadas, D. Dolev, K. Nayak, and L. Ren, "Efficient synchronous Byzantine consensus," 2017, available on ArXiv: https://arxiv.org/abs/1704.02397.

[21] M. Ben-Or, "Another advantage of free choice: Completely asynchronous agreement protocols," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Aug. 1983, pp. 27–30.

[22] G. Bracha, "An asynchronous $[(n-1)/3]$-resilient consensus protocol," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Aug. 1984.

[23] C. Attiya, D. Dolev, and J. Gil, "Asynchronous Byzantine consensus," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Aug. 1984, pp. 119–133.

[24] M. Fischer, N. Lynch, and M. Paterson, "Impossibility of distributed consensus with one faulty process," *Journal of the ACM*, vol. 32, no. 2, pp. 374–382, Apr. 1985.

[25] G. Bracha and S. Toueg, "Asynchronous consensus and broadcast protocols," *Journal of the ACM*, vol. 32, no. 4, pp. 824–840, Oct. 1985.

[26] D. Dolev, C. Dwork, and L. Stockmeyer, "On the minimal synchronism needed for distributed consensus," *Journal of the ACM*, vol. 34, no. 1, pp. 77 – 97, Jan. 1987.

[27] R. Canetti and T. Rabin, "Fast asynchronous Byzantine agreement with optimal resilience," in *25th Annual ACM Symposium on the Theory of Computing*, Jun. 1993, pp. 42–51.

[28] J. Aspnes and O. Waarts, "Randomized consensus in expected $O(N \log^2 N)$ operations per processor," *SIAM Journal on Computing*, vol. 25, no. 5, pp. 1024–1044, Oct. 1996.

[29] I. Abraham, D. Dolev, and J. Halpern, "An almost-surely terminating polynomial protocol for asynchronous byzantine agreement with optimal resilience," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Aug. 2008, pp. 405–414.

[30] A. Mostéfaoui, H. Moumen, and M. Raynal, "Signature-free asynchronous binary Byzantine consensus with $t < n/3$, $O(n^2)$ messages, and $O(1)$ expected time," *Journal of the ACM*, vol. 62, no. 4, Sep. 2015.

[31] A. Miller, Y. Xia, K. Croman, E. Shi, and D. Song, "The Honey Badger of BFT protocols," in *2016 ACM SIGSAC Conference on Computer and Communications Security*, Oct. 2016.

[32] S. Duan, M. Reiter, and H. Zhang, "BEAT: Asynchronous BFT made practical," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, Jan. 2018, pp. 2028–2041.

[33] I. Abraham, D. Malkhi, and A. Spiegelman, "Asymptotically optimal validated asynchronous Byzantine agreement," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Jul. 2019, pp. 337–346.

[34] T. Crain, "A simple and efficient asynchronous randomized binary Byzantine consensus algorithm," 2020, available on: arXiv:2002.04393.

[35] R. Turpin and B. Coan, "Extending binary Byzantine agreement to multivalued Byzantine agreement," *Information Processing Letters*, vol. 18, no. 2, pp. 73–76, Feb. 1984.

[36] M. Fitzi and M. Hirt, "Optimally efficient multi-valued byzantine agreement," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Jul. 2006, pp. 163–168.
[37] A. Patra and P. Rangan, "Communication optimal multi-valued asynchronous Byzantine agreement with optimal resilience," in *International Conference on Information Theoretic Security*, 2011, pp. 206–226.
[38] G. Liang and N. Vaidya, "Error-free multi-valued consensus with byzantine failures," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Jun. 2011, pp. 11–20.
[39] A. Patra, "Error-free multi-valued broadcast and Byzantine agreement with optimal communication complexity," in *Proceedings of the 15th international conference on Principles of Distributed Systems, OPODIS*, 2011, pp. 34–49.
[40] M. Hirt and P. Raykov, "Multi-valued Byzantine broadcast: The $t < n$ case," *Advances in Cryptology – ASIACRYPT 2014, Lecture Notes in Computer Science*, vol. 8874, pp. 448–465, Nov. 2014.
[41] C. Ganesh and A. Patra, "Broadcast extensions with optimal communication and round complexity," in *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, Jul. 2016, pp. 371–380.
[42] W. Chongchitmate and R. Ostrovsky, "Information-theoretic broadcast with dishonest majority for long messages," in *Theory of Cryptography. TCC 2018. Lecture Notes in Computer Science*, vol. 11239, Nov. 2018, pp. 370–388.
[43] C. Ganesh and A. Patra, "Optimal extension protocols for byzantine broadcast and agreement," in *Distributed Computing*, Jul. 2020.
[44] A. Loveless, R. Dreslinski, and B. Kasikci, "Optimal and error-free multi-valued Byzantine consensus through parallel execution," 2020, available on : https://eprint.iacr.org/2020/322.
[45] E. Nayak, L. Ren, E. Shi, N. Vaidya, and Z. Xiang, "Improved extension protocols for Byzantine broadcast and agreement," 2020, available on ArXiv: https://arxiv.org/abs/2002.11321.
[46] G. Bracha, "Asynchronous Byzantine agreement protocols," *Information and Computation*, vol. 75, no. 2, pp. 130–143, Nov. 1987.
[47] I. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, Jun. 1960.
[48] R. Roth, *Introduction to coding theory*. Cambridge University Press, 2006.
[49] E. Berlekamp, "Nonbinary BCH decoding (abstr.)," *IEEE Trans. Inf. Theory*, vol. 14, no. 2, pp. 242–242, Mar. 1968.