# Scalable Network-Coded PBFT Consensus Algorithm

Beongjun Choi, Jy-yong Sohn, Dong-Jun Han, and Jaekyun Moon
School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST)
Email: {bbzang10, jysohn1108, djhan93}@kaist.ac.kr, jmoon@kaist.edu

***Abstract*—We suggest a general framework for *network-coded* Practical Byzantine Fault Tolerant (PBFT) consensus for enabling agreement among distributed nodes under Byzantine attacks. The suggested protocol generalizes existing *replication* and *sharding* schemes which are frequently used for consensus in current blockchain systems. Using the proposed algorithm, it is possible to reach a consensus when the available bandwidth is considerably smaller on individual links compared to that required for conventional schemes. It is shown that there exists an upper bound on the number of nodes that can participate in the protocol, given a maximum bandwidth constraint across all pairwise links. Furthermore, the protocol that achieves the upper bound is provided by using a set of *constant weight codes*.**

## I. Introduction

The Practical Byzantine Fault Tolerance (PBFT) algorithm proposed in [1] is an efficient algorithm for agreement under Byzantine attacks in asynchronous distributed networks. Malicious attacks by Byzantine nodes can be arbitrary, such as transmitting a faulty data or declining to respond to the requests. The PBFT algorithm offers both the *safety* (agreed data must be the same) and *liveness* (agreement is eventually made unless the data is faulty) properties given that at most $\lfloor \frac{m-1}{3} \rfloor$ out of $m$ nodes are Byzantine. Many cryptocurrency platforms including Ripple, Tendermint and Hyperledger fabric use consensus protocols based on [1].

In the PBFT algorithm, each node exchanges data with all other nodes in reaching a consensus. In this process, high communication demand arises among nodes, which prevents the algorithm from scaling with the number of nodes. One major approach to scale a consensus protocol is to use *sharding* [2], [3], which processes data among small multiple groups of nodes called *shards*. The data is divided into several chunks, and each shard is responsible for each data chunk. The shards operate in parallel, allowing each node to operate with less bandwidth and storage compared to the full replication.

The present work proposes a novel framework which generalizes the existing sharding protocol, by applying the concept of network coding [4]. The proposed protocol considers all possible scenarios of dividing the entire data into multiple blocks and distributing them across the nodes. The sharding protocol, on the other hand, considers only some limited scenarios under the following constraints: 1) nodes in the same shard process exactly the same data block, 2) nodes in different shards process disjoint data. Under the scenario where link capacity is limited between any two arbitrary nodes, the suggested scheme can significantly reduce the maximum required bandwidth across the nodes compared to the sharding protocol by properly distributing the data blocks. The suggested scheme allows balancing the communication burden among different links, which improves the *scalability* of the consensus protocol with the number of participants.

*Contributions:* We propose a novel network-coded consensus protocol which opens new possibilities for improving scalability, by reducing the maximum required bandwidth between nodes. Under the communication bandwidth constraint, it turns out that there exists an upper bound on the number of nodes that can participate in the consensus protocol. Furthermore, we show that the network-coded protocol achieving the above upper bound can be designed by utilizing a set of *constant weight codewords*.

*Related Works:* The ideas on reducing the communication burden of PBFT-based consensus algorithms have been studied in [2], [3]. The total communication burden of the PBFT protocol is given by $\mathcal{O}(m^2)$ where $m$ is the number of nodes, since each node should communicate with all the remaining nodes to reach a consensus[1]. Sharding protocols [2], [3] can reduce the complexity to $\mathcal{O}(m^2/s^2)$ where $s$ is the number of shards. The suggested network-coded protocol can relax the maximum communication burden between different nodes, while maintaining the total required bandwidth compared to sharding. Meanwhile, there has been a recent work [6] which employs a classical coding scheme to enhance security by encoding the data blocks. However, the verification of raw data requires decoding, which is available only when a large number of nodes transmit the encoded blocks they have. This incurs a high communication burden, which limits the application of the method of [6] in practice. On the other hand, in the proposed scheme, each node can check the validity of data blocks without any communication across the nodes.

*Notations:* For a positive integer $n$, $[n]$ represents the index set $\{1, 2, \cdots, n\}$. For a matrix $G$, $G_{i,j}$ and $G_{i,:}$ indicate the element at position $(i, j)$ and the $i$th row vector of $G$, respectively. The Hamming distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ is denoted as $d_H(\mathbf{x}, \mathbf{y})$. Finally, for a given binary code $C$, $d_{min}(C)$ represents the minimum Hamming distance among codewords.

[1]Recently, it has been shown that LinBFT [5] can further reduce the communication complexity to $\mathcal{O}(m)$ by introducing techniques such as verifiable random functions. The suggested scheme in this paper is based on the PBFT algorithm but it can be also applied to LinBFT, which may further reduce the communication burden.

## II. System Model

### A. Consensus Protocol

The suggested network-coded consensus protocol is based on the PBFT algorithm [1]. Simillar to [1], we consider systems with up to $f$ Byzantine nodes. Consider a system where a client requests $m$ nodes to reach a consensus on a given data. The consensus protocol consists of 5 stages (request, pre-prepare, prepare, commit, reply) as illustrated in Fig. 1(b). First, in the *request* stage, the client sends the entire data to a single node called *primary node*. The remaining $m-1$ nodes are called *backup nodes*. In the *pre-prepare* stage, the primary node distributes the data to the backup nodes; it does not deliver the entire data, but rather only a subset of data based on some predefined rule. The example of the predefined rule is illustrated in Fig. 1(a) where the individual nodes are responsible for different sets of data blocks. Here, although the primary node can access the entire data blocks, it is only responsible for a subset of the entire data in the following *prepare* and *commit* stages. For example, node 1 in Fig. 1(a) is responsible for data blocks 1 and 2 only. Then, in the *prepare* stage, each backup node determines whether the received data blocks are trustworthy (*e.g.* via signature test), and transmits the valid data blocks to other nodes. In the *commit* stage, each node transmits the data blocks approved by at least $2f$ nodes at the *prepare* stage. Here, in the *prepare* and *commit* stages, any two nodes communicate with each other only on the data blocks shared by them. After collecting the data blocks at the *commit* stage, each node decides that the system has reached a consensus on a data block if more than $2f$ nodes commit the corresponding block. Note that the consensus process of each block proceeds in parallel. When all data blocks are agreed, the entire data is considered to have reached a consensus.

In the consensus protocol, we assume that the size of the data is small enough. One such application is the consensus in the sensor networks [7]. Thus, we consider scenarios of transmitting plaintext data at any stage, instead of sending the message digest produced by a hash function.

### B. System Variables

The data is split into $n$ data blocks with equal size and stored in $m$ nodes, where each node can have multiple data blocks. Let $G$ be an assignment matrix of size $m \times n$ where each element $G_{i,j}$ represents

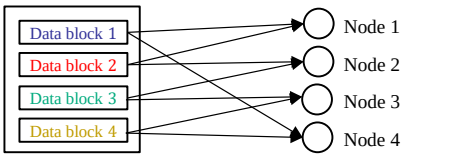
$$G_{i,j} = \begin{cases} 1, & \text{if node } i \text{ stores data block } j \\ 0, & \text{otherwise.} \end{cases}$$

Using the definition, the assignment matrix of protocol illustrated in Fig. 1(a) is expressed as

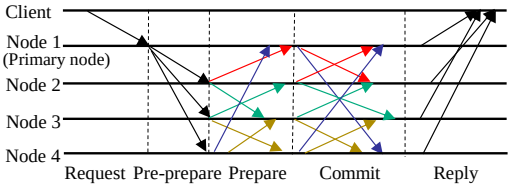
$$G = [1\ 1\ 0\ 0;\ 0\ 1\ 1\ 0;\ 0\ 0\ 1\ 1;\ 1\ 0\ 0\ 1].$$

Now we define three performance metrics for a given $G$. Here, the size of entire data is assumed to be normalized to one. The storage size or space of node $i$ is defined as

$$\rho_i(G) = \sum_{j\in[n]} G_{i,j}/n, \tag{1}$$




(a) A predefined data block assignment rule




(b) Network-Coded PBFT algorithm

Fig. 1: Network-coded PBFT consensus protocol

which indicates the number of data blocks stored in node $i$. Moreover, the distribution degree of $j$th data block is defined as

$$\eta_j(G) = \sum_{i\in[m]} G_{i,j}/m, \tag{2}$$

which represents the fraction of nodes having $j$th data block. Note that nodes $a$ and $b$ share $\sum_k G_{a,k}G_{b,k}$ data blocks where the size of each block is $1/n$. The required communication bandwidth from node $a$ to $b$ is defined as

$$\gamma_{a,b}(G) = \sum_{k\in[n]} G_{a,k}G_{b,k}/n. \tag{3}$$

For notational simplicity, we omit $G$ in (1), (2), (3) unless necessary. In the presented paper, we assume a homogeneous model where each node has identical storage space, i.e.,

$$\rho_i = \rho, \quad \forall i \in [m].$$

The set of assignment matrices $G \in \{0,1\}^{m\times n}$ with storage space $\rho$ is denoted as $\mathcal{G}(m,n,\rho)$. Here, we provide the relationship between $\rho$ and $\eta_j$ in the following proposition.

**Proposition 1.** *For an assignment matrix $G \in \mathcal{G}(m,n,\rho)$, the following holds:*

$$\rho = \frac{1}{n}\sum_{j\in[n]} \eta_j.$$

*Proof.* The number of ones in assignment matrix $G$ can be counted by two distinct ways: summing up the number of ones in $G$ row-by-row gives us the total value of $mn\rho$, while the column-wise summation gives us $\sum_{j\in[n]} m\eta_j$. Thus, $mn\rho = \sum_{j\in[n]} m\eta_j$, which completes the proof. □

Given an assignment matrix $G$, the following proposition provides the maximum number of Byzantine nodes that the network-coded protocol can tolerate.

**Proposition 2.** *The network-coded consensus protocol with assignment matrix $G \in \{0,1\}^{m\times n}$ tolerates $f$ Byzantine nodes if and only if*

$$\min_{j\in[n]} \eta_j \geq \frac{3f+1}{m}. \tag{4}$$

*Proof.* Recall that the PBFT algorithm can tolerate $f$ Byzantine nodes if the number of nodes containing the data is greater than or eqaul to $3f+1$. Note that in the coded PBFT, $j$th data block is stored in $m\eta_j$ nodes. Thus, for each block $j \in [n]$, $3f+1 \leq m\eta_j$ is required, which completes the proof. □

In the PBFT consensus algorithm, communication burden is dominated by the *prepare* and *commit* stages since all nodes sharing the same data block should communicate with each other. The total required bandwidth for exchanging data in the *commit* stage, denoted by

$$\gamma_{total} = \sum_{a\in[m]} \sum_{b\in[m],b\neq a} \gamma_{a,b}, \tag{5}$$

is derived in Proposition 3. The required bandwidth of the *prepare* stage is very similar to that of the *commit* stage; the only difference is that the primary node does not transmit data.

**Proposition 3.** *Assume that every node follows the consensus protocol described in Section II-A. Then, the total required bandwidth for exchanging data in the commit stage is*

$$\gamma_{total} = \frac{1}{n}\sum_{j=1}^{n} \binom{m\eta_j}{2}. \tag{6}$$

*Proof.* Consider an assignment matrix $G \in \mathcal{G}(m,n,\rho)$. The total required bandwidth for exchanging $j$th data block is $\frac{1}{n}\binom{m\eta_j}{2}$ since $m\eta_j$ nodes containing $j$th data block communicate with each other, and $1/n$ is the normalized data block size. Thus, the total required bandwidth is expressed as (6) by summing up all data blocks. □

**Remark 1.** *For a given assignment matrix $G \in \mathcal{G}(m,n,\rho)$, $\gamma_{total}$ is minimized when $\eta_j = \eta = \rho$ holds for all $j \in [n]$. In other words, in order to minimize $\gamma_{total}$, the number of nodes that store each data block should be identical.*

Moreover, here we define the *maximum link bandwidth*

$$\gamma_{max} = \max_{a.b\in[m],a\neq b} \gamma_{a,b}, \tag{7}$$

which represents the maximum required bandwidth among different links. In this paper, we focus on reducing $\gamma_{max}$ for increasing scalability.

## III. Motivating Example

Let $A$ and $B$ be $8\times 8$ assignment matrices defined as

$$A = \begin{bmatrix} 1&1&1&1&0&0&0&0\\ 1&1&0&0&1&1&0&0\\ 1&0&1&0&1&0&1&0\\ 1&0&0&1&0&1&1&0\\ 0&1&1&0&1&0&0&1\\ 0&1&0&1&0&1&0&1\\ 0&0&1&1&0&0&1&1\\ 0&0&0&0&1&1&1&1 \end{bmatrix}, \; B = \begin{bmatrix} \mathbf{1}_{4\times4} & \mathbf{0}_{4\times4}\\ \mathbf{0}_{4\times4} & \mathbf{1}_{4\times4} \end{bmatrix},$$

where $\mathbf{1}_{4\times4}$ and $\mathbf{0}_{4\times4}$ are all-one and all-zero matrices of size $4\times4$, respectively. Here, $B$ is the assignment matrix corresponding to the classical sharding protocol with two shards. Notice that both assignment matrices have storage space $\rho = 0.5$, and can tolerate $f = 1$ Byzantine node among $m = 8$ nodes, according to Proposition 2. When we use assignment matrix $A$, the maximum link bandwidth is $\gamma_{max}(A) = 0.25$, since any two nodes share at most 2 data blocks in common. Similarly, we have $\gamma_{max}(B) = 0.5$. Note that both $A$ and $B$ have $\eta_j = 0.5$ for all $j \in [n]$, i.e., each data block is stored in exactly half of the nodes. Thus, according to Proposition 3, both schemes have the same total bandwidth $\gamma_{total}$. From the above example, it is seen that appropriate design of an assignment matrix can reduce $\gamma_{max}$, while maintaining the same total required bandwidth $\gamma_{total}$.

## IV. Main Results

As all nodes sharing the same data block need to communicate with one another for consensus, the main obstacle to scaling the PBFT algorithm is communication burden. We address this scalability problem by load balancing. With the existing sharding protocol, the maximum link bandwidth is equal to the size of the data held by each node (when the two nodes are in the same shard). By applying the network-coded PBFT, it turns out that the maximum link bandwidth can be reduced, while the total required bandwidth remains almost the same.

### A. Problem Statement with Main Results

Consider a system where the storage space of individual nodes is given as $\rho$. We provide answers to the following two questions: First, given the bandwidth constraint $\gamma_t$ such that

$$\gamma_{max} \leq \gamma_t, \tag{8}$$

how many nodes can participate in the network-coded protocol? In other words, how can we construct an upper bound on $m$ as a function of $\gamma_t$? Second, how to design an assignment matrix $G$ to achieve the upper bound on $m$ under the bandwidth constraint (8)? We first state the following lemma, which shows the relationship between 1) the Hamming distance between two rows of $G$ and 2) the required bandwidth for the users corresponding to these rows.

**Lemma 1.** *For a given assignment matrix $G \in \mathcal{G}(m,n,\rho)$, the following holds:*

$$d_H(G_{a,:}, G_{b,:}) = 2n(\rho - \gamma_{a,b}(G)). \tag{9}$$

*Proof.* From the definition of $\gamma_{a,b}$,

$$n\gamma_{a,b}(G) = |\{k \in [n] : G_{a,k} = G_{b,k} = 1\}|.$$

Also, note that $G_{a,:}$ and $G_{b,:}$ are vectors with length $n$ and weight $n\rho$. Therefore,

$$\begin{aligned} n\rho - n\gamma_{a,b}(G) =& |\{k \in [n] : G_{a,k} = 1, G_{b,k} = 0\}| \\ =& |\{k \in [n] : G_{a,k} = 0, G_{b,k} = 1\}|, \end{aligned}$$

which completes the proof. □

From Lemma 1, the bandwidth constraint (8) reduces to $\min_{a,b\in[m],a\neq b} d_H(G_{a,:}, G_{b,:}) \geq 2n(\rho - \gamma_t)$. The problem

TABLE I: $A(8,d,4)$ where $d = 2\lceil n(\rho-\gamma_t)\rceil$

| $\gamma_t$ | 0 | 0.125 | 0.25 | 0.375 | 0.5 |
|---|---|---|---|---|---|
| $d$ | 8 | 6 | 4 | 2 | 0 |
| $A(8,d,4)$ | 2 | 2 | 14 | 70 | $\infty$ |

of designing $G$ under the above constraint is related to constructing a *binary constant weight code*, which has been studied in information theory community [8]. By definition, binary constant weight codes $C \subset \mathbb{F}_2^n$ are codes where every codeword has the equal Hamming weight $w$. Although it remains as an open problem for general parameters, there have been several studies to find the maximum size $A(n,d,w)$ of binary constant weight code $C$ with a minimum distance at least $d$. We define $C(n,d,w)$ as a set of binary constant weight codewords with maximal size as

$$C(n,d,w) = \{\underline{c}_1, \cdots, \underline{c}_{A(n,d,w)}\}, \tag{10}$$

where $\underline{c}_i \in C(n,d,w)$ is a binary codeword of size $1 \times n$.

**Theorem 1.** *Consider $G \in \mathcal{G}(m,n,\rho)$. If $\gamma_{max}(G) \leq \gamma_t$ holds, then the number of rows is upper bounded as $m \leq A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$. Moreover, the assignment matrix $G^*$, which achieves the upper bound on $m$, can be constructed from constant weight code $C^* = C(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$, i.e.,*

$$G^* = \begin{bmatrix} \underline{c}_1 \\ \underline{c}_2 \\ \vdots \\ \underline{c}_{A(n,2\lceil n(\rho-\gamma)\rceil,n\rho)} \end{bmatrix} \tag{11}$$

*satisfies $\gamma_{max}(G^*) \leq \gamma_t$.*

*Proof.* We first prove the upper bound on $m$ by contrapositive. Consider $G \in \mathcal{G}(m,n,\rho)$ that satisfies

$$m > A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho), \tag{12}$$

and code $C = \{G_{1,:}, G_{2,:}, \cdots, G_{m,:}\}$ corresponding to $G$. Then, $C \subseteq \mathbb{F}_2^n$ is a constant weight code with weight $n\rho$. We want to find a bound on the minimum distance of code $C$. Since $A(n,d,w)$ is a monotonic decreasing function of $d$, we obtain $d_{min}(C) < 2\lceil n(\rho-\gamma_t)\rceil$ from (12). Note that the Hamming distance between two arbitrary binary vectors with the same weight is an even number. Thus,

$$\min_{a,b\in[m],a\neq b} d_H(G_{a,:}, G_{b,:}) = d_{min}(C) < 2n(\rho-\gamma_t).$$

holds. Combining with Lemma 1, we have $\gamma_{max}(G) > \gamma_t$, which completes the first part of the proof. Now we prove that $\gamma_{max}(G^*) \leq \gamma_t$. First, denote $m_0 \triangleq A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$. Since $C^* = C(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho) = \{\underline{c}_1, \cdots, \underline{c}_{m_0}\}$ is a constant weight code with minimum distance at least $2\lceil n(\rho-\gamma_t)\rceil$, we have

$$\min_{a,b\in[m_0],a\neq b} d_H(G^*_{a,:}, G^*_{b,:}) = d_{min}(C^*) \geq 2\lceil n(\rho-\gamma_t)\rceil \geq 2n(\rho-\gamma_t).$$

Combining Lemma 1 completes the proof. □

In Table I, we present values of $A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$ for different bandwidth constraint $\gamma_t$ when $n = 8$, $\rho = 0.5$. Also, we illustrate the number of maximum nodes as a function of $\gamma_t$ in Fig. 2. It is shown that the maximum number of nodes that can participate in the protocol increases as the available bandwidth increases.

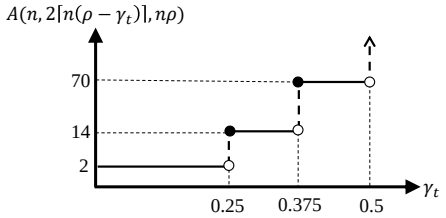



Fig. 2: $A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$ when $n = 8$ and $\rho = 0.5$

### *B. Assignment Matrix Design Considering Byzantine Nodes*

The assignment matrix designed in Section IV-A does not guarantee the Byzantine tolerance. Here we provide the necessary condition on the storage space $\rho$ against $f$ Byzantine nodes in the following proposition.

**Proposition 4.** *If the system can tolerate $f$ Byzantine nodes, the required storage space $\rho$ is lower bounded as*

$$\rho \geq \frac{3f+1}{m}. \tag{13}$$

*The equality holds if and only if $\eta_j = \frac{3f+1}{m}, \ \forall j \in [n]$.*

*Proof.* Consider $G \in \mathcal{G}(m,n,\rho)$. From Propositions 1 and 2,

$$\rho = \frac{1}{n}\sum_{j\in[n]} \eta_j \geq \frac{(3f+1)}{m} \tag{14}$$

holds, which directly leads to (13). Now we prove the equality condition. According to (14), the equality in (13) holds if and only if $\frac{1}{n}\sum_{j\in[n]} \eta_j = (3f+1)/m$. Recall that $\eta_j \geq (3f+1)/m$ for all $j \in [n]$ from Proposition 2. Thus, setting $\eta_j = \frac{3f+1}{m}$ for all $j \in [n]$ is the unique solution which satisfies the equality of (13). □

Now we provide a strategy for designing the assignment matrix $G$ which tolerates $f$ Byzantine nodes, when the number of users $m$ and the bandwidth constraint $\gamma_t$ are given. We first set $\rho = \frac{3f+1}{m}$, which is the minimum value that satisfies the constraint (13). Next, we choose an arbitrary $n$ which satisfies $m \leq A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho)$. Consider the set of constant weight codewords in (10) for $d = 2\lceil n(\rho-\gamma_t)\rceil$ and $w = n\rho$:

$$C(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho) = \{\underline{c}_1, \cdots, \underline{c}_{A(n,2\lceil n(\rho-\gamma_t)\rceil,n\rho)}\}.$$

Then, construct $G = \left[\underline{c}_{i_1}; \ \underline{c}_{i_2}; \ \cdots; \ \underline{c}_{i_m}\right]$ which satisfies $\eta_j \geq (3f+1)/m$ for all $j \in [n]$. If we cannot find such $G$, repeat this procedure for various $n$. Here we provide an example of designing $G$ for $f = 1, m = 8$ and $\gamma_t = 0.25$. Note that $\rho = (3f+1)/m = 0.5$ in this setting. First, select $n = 8$ which satisfies $m \leq A(n, 2\lceil n(\rho-\gamma_t)\rceil, n\rho) = A(8,4,4) = 14$. Next, construct an assignment matrix $G$ by stacking $m = 8$ codewords among $A(8,4,4) = 14$ codewords that satisfy $\eta_j(G) \geq (3f+1)/m$ for all $j \in [n]$. An example of such construction is matrix $A$ in Section III. Note that $m\eta_j(A) = 3f+1$ holds for all $j \in [8]$.

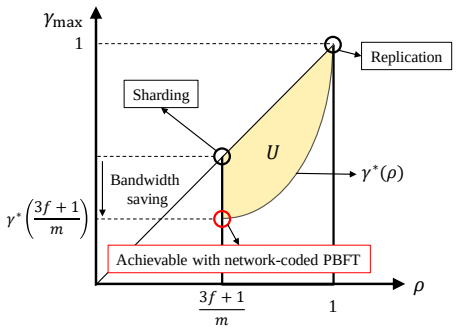



Fig. 3: The feasible region against $f$ Byzantine nodes

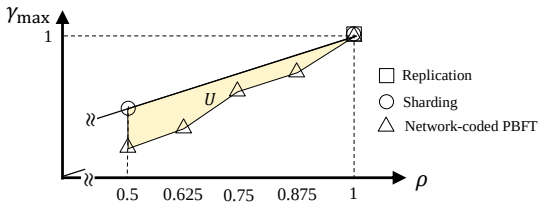



Fig. 4: The feasible points for given $m = 8, n = 8$ and $f = 1$

TABLE II: $\gamma^*(\rho)$ for given $m = 8, n = 8$ and $f = 1$

| $\rho$ | 0.5 | 0.625 | 0.75 | 0.875 | 1 |
|---|---|---|---|---|---|
| $\gamma^*(\rho)$ | 0.25 | 0.375 | 0.625 | 0.75 | 1 |

### C. Comparison with Sharding and Replication

Now we investigate the required amount of $(\rho, \gamma_{max})$ resources to tolerate $f$ Byzantine nodes out of $m$ nodes. We say a $(\rho, \gamma_{max})$ point is *feasible* if there exists a consensus protocol that can tolerate $f$ Byzantine nodes, using the storage space $\rho$ and the maximum link bandwidth $\gamma_{max}$. Our goal is to find the feasible $(\rho, \gamma_{max})$ region for three different schemes: replication, sharding and the network-coded scheme. Here we provide two necessary conditions on feasible $(\rho, \gamma_{max})$ pairs, which are common for all schemes. First, $\gamma_{max} \leq \rho$ since a node cannot transmit more data than it has. Second, $\rho \geq \frac{3f+1}{m}$ according to Proposition 4.

We illustrate the feasible $(\rho, \gamma_{max})$ pairs for various schemes in Fig. 3. In the conventional replication scheme where every node has the entire data, it is necessary for each node to exchange the entire data with other nodes. Thus, the replication scheme corresponds to $(\rho, \gamma_{max}) = (1, 1)$. For sharding, the entire data is divided into $s$ partitions, where $s$ is the number of shards. Since each node transmits the entire data it holds to other nodes in the same shard, we have $\gamma_{max} = \rho$. Thus, the feasible region for sharding is specified as

$$V = \left\{(\rho, \gamma_{max}) = (\frac{1}{s}, \frac{1}{s}) : s \in \mathbb{N}, \frac{n}{s} \in \mathbb{N}, \ \frac{3f+1}{m} \leq \frac{1}{s} \leq 1\right\}.$$

Note that $s$ divides $n$, since $n$ is the number of data blocks and each node has the equal storage size. For the network-coded scheme, there exists an additional constraint $\gamma_{max} \geq \gamma^*(\rho)$ which can be obtained from Theorem 1, where

$$\gamma^*(\rho) = \min\left\{\gamma_t \in \{0, \frac{1}{n}, \cdots, \rho\} : A(n, 2\lceil n(\rho - \gamma_t)\rceil, n\rho) \geq m\right\}.$$

Define $U$ as the set of points $(\rho, \gamma_{max})$ satisfying the three constraints provided above. Among the points in $U$, we are especially interested in the feasibility of the point $(\rho', \gamma'_{max}) = (\frac{3f+1}{m}, \gamma^*(\frac{3f+1}{m}))$ which has the optimal resource efficiency. From Propositions 1 and 2, it can be shown that the point $(\rho', \gamma'_{max})$ is feasible if there exists an assignment matrix $G \in \mathcal{G}(m, n, \rho')$ satisfying the constraint $\eta_j = \frac{3f+1}{m}$ for all $j \in [n]$.

We provide an example for the feasible $(\rho, \gamma_{max})$ pairs for various schemes in Fig 4. Table II shows the feasible $(\rho, \gamma^*(\rho))$ pairs of the proposed scheme illustrated in Fig 4. It can be seen that the suggested scheme significantly reduces $\gamma_{max}$ compared to sharding while maintaining $\rho$.

## V. Further Research Topics

*1) Generalize to the systems using message digests:* This paper focuses on transmitting plaintext data in the *prepare* and *commit* stages, assuming the data block size is small enough. However, the size of the data tends to be large in many blockchain applications, *e.g.*, 2000 bits per transaction in Bitcoin [9]. In such applications, it is more efficient to transmit message digests produced by a hash function rather than plaintexts. Applying the suggested network coding scheme in systems using hash functions in the *prepare* and *commit* stages remains as a future research topic.

*2) Data integrity:* Consider the scenarios where there are more Byzantine nodes than expected. In such cases, some data blocks can be modified by the Byzantine nodes. Compared to the existing sharding protocol, the portion of data blocks that ensures integrity can be increased when the proposed coding scheme is applied. Assume that arbitrary two nodes are Byzantine in the example of Section III. When using assignment matrix $A$, at most two data blocks can be modified. On the other hand, at most four data blocks can be modified when assignment matrix $B$ is used. This shows the potential for improving data integrity by employing the proposed scheme. Note that this advantage can be observed even when message digests are utilized in the *prepare* and *commit* stages.

## References


[1] M. Castro, B. Liskov *et al.*, "Practical byzantine fault tolerance," in *OSDI*, vol. 99, 1999, pp. 173–186.

[2] L. t. Luu, "A secure sharding protocol for open blockchains," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pp. 17–30.

[3] M. Zamani, M. Movahedi, and M. Raykova, "Rapidchain: scaling blockchain via full sharding," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, pp. 931–948.

[4] R. Ahlswede, N. Cai, S.-Y. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[5] Y. Yang, "Linbft: Linear-communication byzantine fault tolerance for public blockchains," *arXiv preprint arXiv:1807.01829*, 2018.

[6] S. Li, M. Yu, S. Avestimehr, S. Kannan, and P. Viswanath, "Polyshard: Coded sharding achieves linearly scaling efficiency and security simultaneously," *arXiv preprint arXiv:1809.10361*, 2018.

[7] S. Kar and J. M. Moura, "Distributed consensus algorithms in sensor networks: Quantized data and random link failures," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1383–1400, 2010.

[8] A. E. Brouwer, L. B. Shearer, N. Sloane *et al.*, "A new table of constant weight codes," in *IEEE Transactions on Information Theory*, 1990.

[9] "Blockchain explorer," https://www.blockchain.com/explorer, accessed: 2019-01-20.