Generalized Byzantine Agreement
with Incomplete Views

Hanjun Li

CMU-CS-19-134
December, 2019

Computer Science Department
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:
Vipul Goyal (Chair)
Bryan Parno

Submitted in partial fulfillment of the requirements for the degree of Master of Science.

Copyright © 2019 Hanjun Li
Keywords: Distributed Algorithm, Byzantine Agreement
Abstract

Byzantine agreement is the problem where a set of participants, each holding an input value, try to reach agreement on an output in the presence of corrupted parties. This problem is well studied in the standard model, where each participant has a complete view of the whole network. This thesis solves the byzantine agreement problem in a relaxed model where every participant only knows and communicates with a subset (its “view”) of other parties. We parameterize our model by $\alpha$, the maximum fraction of corruption in each honest “view”, and $\delta$, the minimum fraction of overlapping between any pair of honest “views”. We present a protocol that runs in expected polynomial round assuming $\delta > 2\alpha$. If we further assume $\alpha \leq 1/2 - \epsilon$ for any constant $\epsilon$, the protocol runs in expected constant round. We also show the tightness of our assumptions by proving impossibility results for $\alpha \geq 1/2$ and for $\delta \leq 2\alpha$. 
Acknowledgments

I would like to thank my advisor, Prof. Vipul Goyal, for his guidance and advice, and my collaborator, Justin Raizes, for his interesting ideas and perspectives during our discussions.

I would also like to thank Prof. Bryan Parno for serving as a member of my thesis committee, Prof. Dave Eckhardt for his general advice on my thesis, and Tracy Farbacher for arranging and advertising my thesis defense.
## Contents

1. **Introduction** ........................................... 1

2. **Preliminaries** ......................................... 3
   2.1 **Notations** ........................................ 3
   2.2 **Unique Digital Signature and PKI** ................. 3
   2.3 **Verifiable Random Function** ...................... 4
   2.4 **Our Model and Main Results** ..................... 6

3. **Positive Result** ....................................... 9
   3.1 **Graded Broadcast** ................................ 9
   3.2 **Leader Selection** ................................ 10
   3.3 **Main protocol** ................................... 12

4. **Negative Result** ...................................... 15
   4.1 **Broadcast from Byzantine Agreement** ............... 15
   4.2 **Negative Result for Broadcast** .................. 16

5. **Conclusions** ........................................ 21

**Bibliography** ............................................. 24
Chapter 1
Introduction

Byzantine Agreement is a very important and well studied problem in distributed computing and cryptography. At a high level, it is the problem where a set of $N$ participants, each holding an input value, try to reach agreement on an output in the presence of corrupted participants. The number of corrupted participants is usually denoted $t$. The problem is first introduced and studied by [LSP], [PSL80], in which both an upper bound and a lower bound on the size of tolerable corruption is proven: Byzantine Agreement is achievable if and only if less than $1/3$ of the participants are corrupted. In these results, the adversary may be computationally unbounded, as no cryptography tools are used. In [FL82], [GM98], both a lower bound on the round complexity, $t + 1$, and a fully polynomial protocol achieving this bound is shown. In the case of a sparse network, [Doi82], showed that Byzantine Agreement is achievable if and only if $t$ is less than $1/2$ of the connectivity of the underlying communication network.

To circumvent the upper bound on $1/3$ tolerable corruption, a popular additional assumption is a public-key infrastructure (PKI) setup, in which participants are assigned public keys of the others before entering the protocol. This setting is often referred to as authenticated. The results in [LSP], [PSL80], showed that authenticated Byzantine Agreement is achievable if and only if less than $1/2$ of the participants are corrupted. In these results, the adversary is assumed to be polynomial time algorithms, as digital signatures are only computationally secure. In [DS83], the same lower bound $t + 1$ on the round complexity, and a fully polynomial protocol achieving this bound is shown.

Finally, to circumvent the lower bound on $t + 1$ round, randomization is introduced to solve the Byzantine Agreement problem. Randomization was first considered in [BO83], [Rab83]. In the plain setting (without PKI), [FM97] first showed a protocol tolerating the optimal $1/3$ corruption that runs in expected constant round. In the authenticated setting, [FG03] showed a protocol tolerating the optimal $1/2$ corruption that runs in expected constant round with specific number theoretic assumptions. [KK06], showed a protocol tolerating $1/2$ corruption that also runs in expected constant round while only assuming the PKI setup. [Mic17], and [MV17] showed an alternative construction also tolerating $1/2$ corruption and runs in expected constant round.

**Our Contribution:** This thesis considers a generalization of the authenticated setting. It is motivated by the real world peer-to-peer networks, where not every pair of participants are con-
nected directly. In our setting, every participant only knows and communicates with a subset (its “view”) of other participants, and during the PKI setup is only assigned the public keys of those in its view. We parameterize our model by $\alpha$, the maximum fraction of corruption in each honest “view”, and $\delta$, the minimum fraction of overlapping between any pair of honest “views”. Note that when $\delta = 1$, our model becomes the standard model where the set of participants form a complete graph. We present a protocol that runs in expected polynomial round assuming $\delta > 2\alpha$. If we further assume $\alpha \leq 1/2 - \epsilon$ for any constant $\epsilon$, the protocol runs in expected constant round. We also show the tightness of our assumptions by proving impossibility results for $\alpha \geq 1/2$ and for $\delta \leq 2\alpha$.

**Related Works:** Motivated by peer-to-peer networks that exist in the real world, a line of work considers solving the Byzantine Agreement problem in networks of bounded degree. However, a trivially impossible case is when an honest participant is surrounded by corrupted ones. To circumvent this impossibility, [DPPU88] first introduced the concept of “almost everywhere” agreement, that only requires agreement among all but a linear (of the size of corruption) number of honest participants, and gave some positive results. Later works followed this paradigm and aimed to improve the expected round and communication complexity [BG89], [BG93], [KSSV06], to improve the size of tolerable corruption [Upf94], [BOR96], and to minimize the fraction of honest participants “given up” [CGO10].

Although this thesis also considers the Byzantine Agreement problem in a sparse (i.e. incomplete) network, our setting differs from the above line of work (referred to as “bounded degree” setting) in several aspects. First, in the bounded degree setting, every participant is assumed to know the whole network topology, and their general strategy is to simulate a complete network on specific families of bounded degree graph. In our setting, honest participants are not assumed such knowledge, and our results hold for any network. Second, in the bounded degree setting, the adversary is assumed to corrupt participants randomly while in our setting, the adversary corrupts adaptively. Finally, the bounded degree setting only considers “almost everywhere” agreement, while our result achieves agreement for all honest participants. Additionally, as will be shown in later sections, our setting defends certain sybil attack because only corrupted participants that are connected to some honest participant have an effect in the protocol. An adversary can spawn any number of corrupted participants while having no real effect. This is not true for the bounded degree setting.
Chapter 2

Preliminaries

2.1 Notations

We use \{0, 1\}∗ to denote the set of finite binary strings. When describing a probabilistic algorithm \(F(\cdot), F(x)\) refers to the probability space that assigns any string \(y\) the probability that \(F\) on input \(x\) outputs \(y\). We write \(y \xleftarrow{R} F(x)\) to describe assigning to \(y\) an element randomly selected according to \(F(x)\). In contrast, for a deterministic algorithm \(G(\cdot)\), we simply write \(y = G(x)\) to describe the output of \(G\) on input \(x\) being \(y\).

We call a function \(f(k)\) negligible if for every polynomial \(p(\cdot)\), we have \(f(k) < 1/p(k)\) for large enough \(k\). Usually, the function \(f(k)\) calculates some probability, and \(k\) is the security parameter. We call a function \(g(k)\) overwhelming if \(1 - g(k)\) is negligible.

When describing some algorithm \(T\), we write \(T^{F(\cdot)}\) if \(T\) is given oracle access to some functionality \(F(\cdot)\). That is, \(T\) can get the result of \(F(x)\) for any query \(x\), but \(T\) doesn’t know the code (e.g. some hard-coded information) of \(F(\cdot)\).

We will use the usual notation \(G = (V, E)\) to represent the communication graph among the participants. \(V\) is the set of nodes in the graph. Each node \(P_i\) represents the participant \(P_i\). \(E\) is the set of edges. Each edge \((P_j, P_k)\) represents the fact that \(P_k\) is in the view of \(P_j\) (see Section 2.2). Our model assumes only bi-directional edges. That is, if \(P_k\) is in the view of \(P_j\) then \(P_j\) is also in the view of \(P_k\). We write \(\Gamma_i\) as the inclusive neighbor of \(P_i\) in \(G\) (i.e. all participants in the view of \(P_i\) including itself).

2.2 Unique Digital Signature and PKI

Similar to other authenticated Byzantine Agreement protocols, we will use a Digital Signature scheme to ensure that a corrupted participant can either choose to forward or ignore a signed message, but can never forge a signed message. In addition, we also require that for any public key and any message \(m\), there is a unique valid signature. The additional uniqueness constraint is introduced in [GO92], and a construction based on the RSA assumption is provided in [MRVil]. Briefly, we summarize the notion of Unique Digital Signature below.
Notation: A Unique Digital Signature scheme is a triple of polynomial time computable algorithms (Gen, Sign, Verify) as described below.

- Gen(·) is a probabilistic algorithm that takes a unary string of length \( k \), the security parameter, as input, and outputs two binary strings, a public key \( P_k \) and a secret key \( S_k \).
  We write this as \((S_k, P_k) \xleftarrow{R} \text{Gen}(1^k)\).
- Sign(·, ·) is a deterministic algorithm that takes in the secret key \( S_k \) and a message \( x \), and produces a signature \( \text{Sig}_{P_k}(x) \).
  We write this as \( \text{Sig} = \text{Sign}(S_k, x) \).
- Verify(·, ·, ·) is a probabilistic algorithm that takes a public key \( P_k \), a message \( x \) and a signature \( \text{Sig} \) as input, and outputs a bit \( b \in \{0, 1\} \).
  We write this as \( b \xleftarrow{R} \text{Verify}(P_k, x, \text{Sig}) \).

We now briefly describe the correctness and the security of a Unique Digital Signature scheme.

**Correctness:**
- Verify accepts (i.e. outputs 1) a valid signature produced by Sign with overwhelming probability over \((P_k, S_k) \xleftarrow{R} \text{Gen}(1^k)\)
- There do not exist values \((P_k, x, \text{Sig}_1, \text{Sig}_2)\) such that \( \text{Sig}_1 \neq \text{Sig}_2 \), and \( \text{Verify}(P_k, x, \text{Sig}_1) = \text{Verify}(P_k, x, \text{Sig}_2) = 1 \).

**Security:** Let \( T \) be any polynomial time algorithm. The probability that \( T \) wins the following game must be negligible:
- \( \text{Run}(P_k, S_k) \xleftarrow{R} \text{Gen}(1^k)\):
- \( \text{Run}(x, \text{Sig}) \xleftarrow{R} T^{\text{Sign}(S_k, \cdot)}(1^k, P_k)\)
- \( T \) wins if \( 1 \xleftarrow{R} \text{Verify}(P_k, x, \text{Sig}) \)

This security definition is also called existentially unforgeable.

**Public-key Infrastructure:** Before using a Digital Signature scheme for authenticating messages, a trusted setup phase is required to first assign each participant \( P_i \) its key pair \((P_{ki}, S_{ki})\), and next distribute public keys of the participants. This setup phase is generally referred to as a Public-key infrastructure (PKI) setup. In our relaxed model, the setup phase only assigns to each participants a subset of the public keys. If the public key of \( P_j \) is assigned to \( P_i \), we say \( P_i \) trusts \( P_j \). The set of trusted participants by \( P_i \) is called the view of \( P_i \). In the later sections, we will use the player id to identity its public key. For example a signature by \( P_i \) on some message \( m_i \) will be written as \( \text{Sig}_i(m_i) \).

### 2.3 Verifiable Random Function

When describing and analyzing our protocols, we will assume that every participant has access to a public random function \( H \), mapping \( \{0, 1\}^* \) to \( \{0, 1\}^k \) for any \( k \). In this idealized model, when \( H \) receives a query string \( x \), it selects a \( k \) bit string uniformly at random as its output \( H(x) \).
All further query of \( x \) all result in the same output \( H(x) \).

Note that the random oracle \( H \) is only introduced to simplify our analysis. The usage of \( H \) can be replaced by a verifiable random function (VRF) scheme which can be constructed under the RSA assumption [MRVil]. At a high level, an VRF scheme lets a participant to calculate a pseudo-random string \( v \) based on a seed \( x \), and also a proof that this \( v \) was correctly calculated. Another participant cannot distinguish \( v \) from a truly random string in polynomial time, but can verify that \( v \) is correctly calculated based on \( x \). We now briefly summarize the notion of VRF below.

**Notation:** A Verifiable Random Function scheme is a triple of polynomial time computable algorithms \( (G, F, V) \) as described below.

- \( G(\cdot) \) is a probabilistic algorithm that takes a unary string of length \( k \), the security parameter, as input, and outputs two binary strings, a public key \( PK \) and a secret key \( SK \).
  
  We write this as \( (PK, SK) \leftarrow^R G(1^k) \)

- \( F(\cdot, \cdot) \) is a deterministic algorithm that takes two binary strings, the secret key \( SK \) and a seed \( x \), and outputs two binary strings, the value \( v \) and its corresponding proof \( proof \).
  
  We write this as \( (v, proof) = F(SK, x) \). For convenience, we sometimes write \( F = (F_1, F_2) \) where \( v = F_1(SK, x) \) and \( proof = F_2(SK, x) \).

- \( V(\cdot, \cdot, \cdot, \cdot) \) is a probabilistic algorithm that takes four binary strings, the public key \( PK \), the seed \( x \), the value \( v \), and the proof \( proof \), as input, and output a bit \( b \in \{1, 0\} \). We write this as \( b \leftarrow^R V(PK, x, v, proof) \)

At a high level, we can think of \( G \) as the function generator, \( F \) as the function evaluator, and \( V \) as the function verifier. We now describe the correctness and the security of a VRF scheme.

**Correctness:** The following must hold with overwhelming probability over \( (PK, SK) \leftarrow^R G(1^k) \):

- For all \( x \) in its domain, \( F_1(SK, x) \) produces a string in its correct range.

- For all \( x \) in its domain, if \( (v, proof) = F(SK, x) \), then \( 1 \leftarrow^R V(PK, x, v, proof) \).

- For every \( x, v_1, v_2, proof_1, proof_2 \) such that \( v_1 \neq v_2 \), either \( 0 \leftarrow^R V(PK, x, v_1, proof_1) \) or \( 0 \leftarrow^R V(PK, x, v_2, proof_2) \).

At a high level, the correctness requires that the proof produced by \( F_2 \) can be uniquely verified by \( V \).

**Security:** Let \( T = (T_E, T_j) \) be any pair of polynomial time algorithm (in the security parameter \( k \)). The advantage that \( T \) has in succeeding the following game over \( 1/2 \) (i.e. random guessing) must be negligible.

- Run \( (PK, SK) \leftarrow^R G(1^k) \)

- Run \( (x, state) \leftarrow^R T_E^{FSK}(1^k, PK) \)

- Randomly choose a bit \( r \leftarrow^R \{0, 1\} \):
if $r = 0$, run $v = F_1(SK, x)$

- if $r = 1$, sample $v$ at random from the range of $F_1$.

- Run $b \leftarrow \text{R} T_{SK}^{F(SK, \cdot)}(1^k, v, \text{state})$ where $b$ is the guess of $T$. $T$ succeeds if $x$ is in the domain of $F_1$, $x$ is not asked as a query to $F(SK, \cdot)$ by $T_E$ or $T_J$, and $b = r$.

At a high level, the security requires that the output of $F_1$ is indistinguishable from a random string.

**Using VRF in place of $H(\cdot)$:** We now describe how to use a VRF scheme to simulate the random oracle $H$ used in our protocol. In the trusted PKI setup phase, each participant $P_i$ is assigned its own key pair $(PK_i, SK_i)$, and also the set of public keys in its view. In our protocol, whenever a participant $P_i$ receives a value $v_j$ originated from $P_j$, and needs to evaluate $H(v_j)$, we instead ask $P_j$ to attach the evaluation $(r_j, \text{proof}_j) = F(SK_j, v_j)$ and its public key $PK_j$ to $v_j$. That is, $P_i$ will actually receive $v_j' = (v_j, r_j, \text{proof}_j, PK_j)$, and can simply run $\text{V}(PK_j, v_j, r_j, \text{proof}_j)$ to verify that $r_j$ is the correct result. Each message is additionally signed by $P_j$ so that no one can forge an evaluation of $H(v_j)$.

In the case where $P_j$ is not trusted by $P_i$, (hence all messages of $v_j$ is forwarded by some other participant) our protocol always guarantees that at least one forwarding is from an honest participant, who has verified its signature and the included public key $PK_i$. Whenever $P_j$ receives two contradicting $PK_i$ and $PK_i'$, $P_j$ discards all messages from $P_i$ since it must have been corrupted.

### 2.4 Our Model and Main Results

Our model is a relaxation from the standard one assumed in most authenticated Byzantine Agreement protocols (e.g. [Mic17], [KK06]). The PKI setup in our model is described in Section 2.2.

The communication and adversary model are described below.

**Communication:** All participants communicate in synchronized rounds, over authenticated and private point-to-point channels. Note that such authenticated channels only exists between two parties who trusts each other, so our communication network is sparse. Messages are sent at the start of a round, and received by the end of the round.

**The Adversary:** The adversary is a polynomial time algorithm, that can adaptively corrupt honest participants during the protocol. Corrupted participants can deviate from the protocol in arbitrary ways. At the start of any round, the adversary may corrupt additional players before receiving messages from all honest participants, and then decide what to send from all corrupted participants. The adversary knows the public keys from all participants at the beginning.

**Results:** We first state the standard definition of a Byzantine Agreement protocol. For simplicity, we only consider the binary version.
Definition 1. (Byzantine Agreement) For a set of participants $P_1, \ldots, P_N$, where each $P_i$ holds an initial input $v_i \in \{0, 1\}$, when the protocol terminates, the following conditions must hold for any adversary:

- (Validity) If all honest participants begin with the same input $v$, they also output $v$.
- (Agreement) All honest participants output the same value

Let $\alpha_i$ be the fraction of corrupted participants in the view of an honest participant $P_i$. We define $\alpha \equiv \max_i \{\alpha_i\}$. Let $\delta_{ij}$ be the fraction of overlapping between the views of two honest participants $P_i, P_j$ (i.e. $\delta_{ij} = |\Gamma_i \cap \Gamma_j|/|\Gamma_i|$). We similarly define $\delta \equiv \min_{i,j} \{\delta_{ij}\}$. Now we are ready to state our results as the following theorems:

**Theorem 1.** If $\delta > 2\alpha$, there exists an expected $O(n)$ round Byzantine Agreement protocol in our model, assuming a unique digital signature scheme and a verifiable random function scheme. Further if $\alpha = 1/2 - \epsilon$ for any constant $\epsilon$, there exists an expected constant round Byzantine Agreement protocol in our model.

**Theorem 2.** If $\alpha \geq 1/2$ or $\delta \leq 2\alpha$, there does not exist a Byzantine Agreement protocol in our model, even assuming a unique digital signature scheme and a verifiable random function scheme.
Chapter 3

Positive Result

This section first introduces several subprotocols as building blocks, and in the end uses them in our main protocol. For simplicity, in the following discussion we will assume that all honest participants have a view of the same size $n$. However, we note that all of our results hold without this restriction. Intuitively, since every pair of honest participants has at least a $\delta$ overlapping in their views, the size of their views can not differ by too much.

3.1 Graded Broadcast

A Graded Broadcast protocol is a similar but weaker notion to broadcast. At a high level, it simulates a broadcast in which some participants fail to receive the message. It, however, guarantees that all participants that successfully receive a message indeed receive the same message. In addition, if the sender is an honest participant, the broadcast always succeeds. The name “Graded Broadcast” comes from the way a participant decide whether to accept a message: a successful message is assigned a positive grade, while a failed message is assigned grade 0. This idea is first introduced in [FM97] in the standard model. We now give a formal definition in our model.

Definition 2. (Graded Broadcast) For a set of participants $S = \{P_1, \ldots, P_N\}$, and a distinguished dealer $P_d \in S$ holding an initial message $m$, when the protocol terminates, the following conditions must hold for any adversary:

- Each honest participant $P_i$ in the view of $P_d$ (i.e. $P_i \in \Gamma_d$) outputs $(m_i, g_i)$, where $g_i \in \{0, 1\}$.
- (Validity) If $P_d$ is honest, then $m_i = m$, and $g_i = 1$ for all honest $P_i \in \Gamma_d$.
- (Agreement) If two honest participants $P_i, P_j \in \Gamma_d$ outputs $(m_i, 1)$, and $(m_j, 1)$, then $m_i = m_j$.

Next we give our protocol (Algorithm 1) that achieves Graded Broadcast assuming $\delta > \alpha$. 

9
Algorithm 1 Graded Broadcast

1: The dealer $P_d$ signs message $m$, and sends $(m, \text{Sig}_d(m))$ to all participants in its view.
2: Every honest participant $P_i$ in the view of $P_d$ verifies the received signature, and forwards the received messages. If the signature is not valid, it follows through Step 3, but always outputs $(\phi, 0)$ in Step 4.
3: Every honest participant $P_i$ in the view of $P_d$ again forwards the received messages.
4: Every honest participant $P_i$ in the view of $P_d$ verifies received messages:
   - If there are only valid signatures of message $m$, then $P_i$ outputs $(m, 1)$.
   - Else: there are contradicting valid signatures of $m \neq m'$ or there are no valid signatures. $P_i$ outputs $(\phi, 0)$.

We now show the following claims about Algorithm 1.

Claim 3. (Validity) If the dealer $P_d$ is honest with message $m$ and if $\delta > \alpha$, then all honest participants $P_i$ in the view of $P_d$ output $(m, 1)$ at the end of the Graded Broadcast protocol in Algorithm 1.

Proof. Since $P_d$ is honest, only signatures of the message $m$ is ever sent out by $P_d$. By the security of a Digital Signature scheme, no contradicting signatures (on a different message $m' \neq m$) can be forged.

By our model assumption, there are at least $(\delta - \alpha)n > 1$ honest participants in the overlapping between any honest $P_i$ and $P_d$. Therefore, $P_i$ always receives at least 1 valid signature of $m$. Hence $P_i$ outputs $(m, 1)$.

Claim 4. (Agreement) If two honest participants $P_i, P_j$ in the view of $P_d$ outputs $(m_i, 1)$ and $(m_j, 1)$, and if $\delta > \alpha$, then $m_i = m_j$.

Proof. Since $P_i$ is honest, he must have seen a valid $(m_i, \text{Sig}_d(m_i))$ in Step 2. By our model assumption, there are at least $(\delta - \alpha)n > 1$ honest participants in the overlapping between any honest $P_j$ and $P_i$. Therefore, $P_i$ always receives at least 1 valid signature of $m_i$. Hence $P_j$ never outputs $(m_j, 1)$ for any $m_j \neq m_i$.

The above claims lead to the following lemma that we will use as a building block to prove Theorem 1.

Lemma 5. If $\delta > \alpha$, there exists a three round Graded Broadcast protocol.

Proof. By Algorithm 1, Claim 3, and Claim 4.

3.2 Leader Selection

A Leader Selection protocol is used for electing an honest leader that is agreed on by all honest participants. However, we only need this to happen with some probability, to which we refer as the fairness of the protocol. If the fairness is a constant, then running the protocol repeatedly will give us an honest leader in expected constant round. If the fairness is $\Omega(1/n)$, then running the protocol repeatedly will give us an honest leader in expected $O(n)$ round. Our definition is
similar to the one present in [KK06]. Our protocol construction is inspired by the ConcreteCoin protocol present in [Mic17]. We now give a formal definition in our model:

**Definition 3.** (Leader Selection) For a set of participants $P_1, ..., P_N$ and fairness $\gamma$, when the protocol terminates, the following conditions must hold with probability at least $\gamma$:

- Every honest participant $P_i$ outputs $P_l$ and $P_l$ is honest by the end of the protocol.

When such an event happens, we say that an honest leader $P_l$ is elected.

Next, we give our protocol (Algorithm 2) that achieves Leader Selection assuming $\delta > 2\alpha$. Its fairness is analyzed below.

---

**Algorithm 2** Leader Selection

**Input:** $r$

1. Let $r$ be given (representing the current iteration number in the outer protocol). Every honest participant $P_i$ sends $m_i = (i, \text{Sig}_i(r))$ to all other participants in its view.
2. Every honest participant $P_i$ forwards messages with valid signatures to all other participants in the view of $P_i$.
3. Every honest participant $P_i$ receives at most $n$ forwarded messages from each $P_j$ in its view (and ignore the messages after the first $n$). $P_i$ computes a set $S_i$ of messages that are forwarded by at least $(\delta - \alpha)n$ participants in its view, and send $S_i$ to all participants in its view.
4. Every honest participant $P_i$ receives a set $S_j$ from every participant $P_j$ in its view.
   - $P_i$ computes a set $S_i^*$ of messages that appear in at least $(1 - \alpha)n$ received sets.
   - For every $m_k \in S_i^*$, $P_i$ computes $H(m_k)$ and outputs $P_l$ where $l$ is the smallest id such that for all $m_k \in S_i^*$ $H(m_l) \leq H(m_k)$.

**Output:** $P_l$

---

We now show the following claims about Algorithm 2.

**Claim 6.** In any iteration $r$, if $P_i, P_j$ are any two honest participants, then $m_j = (j, \text{Sig}_j(r)) \in S_i^*$ in Step 4.

**Proof.** Consider any honest participant $P_k$. By our model assumption, there are at least $(1 - \alpha)n$ honest participants in the view of $P_j$, and at least $((1 - \alpha) - (1 - \delta))n = (\delta - \alpha)n$ of them are also in the view of $P_k$. Therefore, in Step 3 $P_k$ receives $m_j$ forwarded by at least $(\delta - \alpha)n$ honest participants. Hence $m_j \in S_k$.

By the argument above, every honest $P_k$ in the view of $P_i$ sends $S_k$ with $m_j \in S_k$ in Step 3. Since there are at least $(1 - \alpha)n$ of them, we have $m_j \in S_i^*$ in Step 4.

**Claim 7.** In any iteration $r$, if $P_i$ is honest, and if $\delta > 2\alpha$, then $S_i$ contains at most $2n$ messages from corrupted participants.

**Proof.** We first calculate the total number of times any message from a corrupted participants gets forwarded to $P_i$. By our model assumption, there are at most $\alpha n$ corrupted participants in the view of $P_i$ that can each forward $n$ corrupted messages. The rest $(1 - \alpha)n$ honest participants will each forward at most $\alpha n$ corrupted messages. In total, we get

$$\alpha n^2 + (1 - \alpha)\alpha n^2 = \alpha(2 - \alpha)n^2$$
For a corrupted message to be accepted into $S_i$, it must be forwarded at least $(\delta - \alpha)n$ times. Therefore, the number of accepted corrupted messages is at most
\[
\alpha(2 - \alpha)n^2 \leq \frac{\alpha}{\delta - \alpha}2n < 2n
\]

The above claims lead to the following lemma that we will use as a building block to prove Theorem 1.

**Lemma 8.** If $\delta > 2\alpha$, there exists a three round Leader Selection protocol with fairness $1/(5n)$. Further if $\alpha < 1/2 - \epsilon$ for some positive constant $\epsilon$, then there exists a three round Leader Selection protocol with constant fairness.

**Proof.** We show that Algorithm 2 is such a protocol. Let $C$ be the set of all honest participants. Consider the union of all honest $S_i^*: S = \cup_{P \in C} S_i^*$. If $P_l$ is an honest participant and $l$ is the smallest id such that for all $m_k \in S^*$ $H(m_l) \leq H(m_k)$, then by Claim 6, $P_l \in S_i^*$ for all honest $P_i$. Hence all honest $P_i$ will output $P_l$ in Step 4 (i.e. an honest leader $P_l$ is elected). Furthermore, by the definition of Random Oracle, $H(m_k)$ are uniform random and independent for all $m_k \in S^*$. Therefore, the probability that an honest leader $P_l$ is elected is exactly $|C|/|S^*|$. We now calculate the total number corrupted messages in $S^*$. In Step 4, any corrupted message ever accepted into an honest $S_i^*$ must appears in at least $(1 - \alpha)n$ sets, of which at least $(1 - 2\alpha)n$ are from an honest participant. Therefore, the number of corrupted messages in $S_i$ in Step 3 is $2nC$. Therefore, the number of corrupted messages in $S^*$ is at most
\[
\frac{2n|C|}{(1 - 2\alpha)n} = \frac{2}{1 - 2\alpha}|C|
\]
And we have $|S^*| \leq |C| + \frac{2}{1 - 2\alpha}|C|$. Since the number of corrupted participants in the view of any honest party is an integral number, we have $1/2 - \alpha \geq 1/(2n)$. The probability that an honest leader is elected is given by
\[
\frac{|C|}{|S^*|} \geq \frac{|C|}{|C| + \frac{2}{1 - 2\alpha}|C|} = \frac{1}{1 + 4n} \geq \frac{1}{5n}
\]
In the case of $\alpha < 1/2 - \epsilon$ for some constant $\epsilon$, we get
\[
\frac{|C|}{|S^*|} \geq \frac{|C|}{|C| + \frac{2}{2\epsilon}|C|} = \frac{1}{1 + 1/\epsilon}
\]
Which is a constant. $\square$

### 3.3 Main protocol

The main protocol is inspired by the one present in [Mic17]. At a high level the protocol ensures that if one honest participant decides to terminate with some output $v \in \{0, 1\}$, it is sure that no
other honest participant terminates with a different $v' \neq v$ in the same round, and that all honest participants will be able to terminate in the next round. When all honest participants hold the same value, they terminates immediately. When an honest leader is elected, they terminates with probability $1/2$. Overall, if an honest leader is elected with constant probability, then our main protocol terminates with expected constant round. Similarly, if an honest leader is elected with probability $\Omega(1/n)$, then our main protocol terminates with expected $O(n)$ rounds. The main protocol is shown in Algorithm 3.

Algorithm 3 Byzantine Agreement

**Input:** $v_i \in \{0, 1\}$: the initial value; $r \leftarrow 0$: the current iteration; $h_i \leftarrow 0$: whether to halt.

1: Every honest $P_i$ runs a Graded Broadcast protocol as the dealer with message $v_i$. In the end, $P_i$ outputs $(v_j, g_j)$ for every participant $P_j$ in its view, and accepts only values with grade 1.
   - If $h_i > 0$, do nothing.
   - If at least $(1 - \alpha)n$ 0s are accepted, then set $v_i \leftarrow 0$ and $h_i \leftarrow 1$.
   - If more than $(1 - \alpha)n$ 1s are accepted, then set $v_i \leftarrow 1$.
   - Otherwise, set $v_i \leftarrow 0$.

2: Every honest $P_i$ runs a Graded Broadcast protocol as the dealer with message $v_i$. In the end, $P_i$ outputs $(v_j, g_j)$ for every participant $P_j$ in its view, and accepts only values with grade 1.
   - If $h_i > 0$, do nothing.
   - If at least $(1 - \alpha)n$ 1s are accepted, then set $v_i \leftarrow 1$ and $h_i \leftarrow 1$.
   - If more than $(1 - \alpha)n$ 0s are accepted, then set $v_i \leftarrow 0$.
   - Otherwise, set $v_i \leftarrow 1$.

3: Every honest $P_i$ sends a random bit $b \leftarrow \{0, 1\}$ to every participants in its view. In the end, $P_i$ receives $b_j$ from every participant $P_j$ in its view.

4: Every honest $P_i$ runs a Leader Selection protocol with input $r$ and outputs $P_l$.

5: Every honest $P_i$ runs a Graded Broadcast protocol as the dealer with message $v_i$. In the end, $P_i$ outputs $(v_j, g_j)$ for every participant $P_j$ in its view, and accepts only values with grade 1.
   - If $h_i > 0$, do nothing.
   - If more than $(1 - \alpha)n$ 1s are accepted, then set $v_i \leftarrow 1$.
   - If more than $(1 - \alpha)n$ 0s are accepted, then set $v_i \leftarrow 0$.
   - If $P_l$ is in the view of $P_i$, then set $v_i \leftarrow b_l$.

6: Set $r \leftarrow r + 1$.
   - If $h_i = 2$ halts with $v^* = v_i$.
   - If $h_i = 1$, sets $h_i \leftarrow 2$.

Go back to Step 1.

**Output:** $v_i^* \in \{0, 1\}$

We now show the following claims about Algorithm 3.

**Claim 9.** (Validity) If every honest participant $P_i$ has the same initial value $v$, then they all terminate in the second iteration.

**Proof.** By our model assumption and the correctness of Graded Broadcast, every honest participant $P_i$ in Step 1 will output $(v, 1)$ from at least $(1 - \alpha)n$ honest Graded Broadcast.
If \( v = 0 \), \( P_i \) sets \( h_i \leftarrow 1 \) in Step 1. If \( v = 1 \), since \( \alpha < \frac{1}{2} \), there cannot be more than \( (1 - \alpha)n \) 1s in step 1, \( P_i \) keeps \( v_i \) unchanged and sets \( h_i \leftarrow 1 \) in Step 2 by the same argument. Once \( h_i = 1 \), \( v_i \) never changes and \( P_i \) halts in the next iteration with \( v^* = v_i \).

Claim 10. If some honest participant \( P_i \) sets \( h_i \leftarrow 1 \) in iteration \( r \) with \( v_i = v \), and if \( \delta > 2\alpha \), then every other honest participant \( P_j \) will have set \( h_j \leftarrow 1 \) with \( v_j = v \) by the end of iteration \( r + 1 \), and all of them halt by the end of iteration \( r + 2 \).

Proof. It suffice to consider the first honest \( P_i \) that sets \( h_i \leftarrow 1 \).

- Suppose this happened in Step 1, consider a different honest participant \( P_j \), who didn’t set \( h_j \leftarrow 1 \) in Step 1 (otherwise, we are done). By our model assumption, of at least \( (1 - \alpha)n \) participants who Graded Broadcasted messages 0 to \( P_i \), at least \( ((1 - \alpha) - (1 - \delta))n = (\delta - \alpha)n > \alpha n \) of them are also in the view of \( P_j \). By correctness of Graded Broadcast, \( P_j \) cannot accept more than \( (1 - \alpha)n \) values of 1 in Step 1. Hence \( P_j \) sets \( v_j \leftarrow 0 \). That is, by the end of Step 1, every honest party \( P_j \) holds \( v_j = 0 \). Repeating the argument from Claim 9, \( P_j \) keeps \( v_j \) unchanged in this iteration. If \( P_j \) didn’t set \( h_j \leftarrow 1 \) in Step 1, it will in the next iteration.

- Suppose this happened in Step 2. By assumption, \( P_i \) is the first honest party who sets \( h_i \leftarrow 1 \), and no other honest \( P_j \) have set \( h_j \leftarrow 1 \) in Step 1. The rest follows from exactly the argument in the previous case.

Note that by Claim 10, honest participants always halt with the same value. It now suffice to only consider the case when no honest participant \( P_i \) has set \( h_i \leftarrow 1 \):

Claim 11. Suppose no honest participant has set \( h_i \leftarrow 1 \) in some iteration \( r \). If an honest leader \( P_l \) is elected in iteration \( r \), and if \( \delta > 2\alpha \) then with probability \( \frac{1}{2} \) \( P_l \) will set \( P_i \) in iteration \( r + 1 \).

Proof. Suppose some honest \( P_i \) in the view of \( P_l \) in Step 5 accepted at least \( (1 - \alpha)n \) values of some value \( v \). By the argument from Claim 10, no other honest \( P_j \) have accepted more than \( (1 - \alpha)n \) values of the different value \( v^c \). That is, in Step 5 the honest participants in the view of \( P_l \) either set their values to the same \( v \), or to \( b_l \), the random bit from \( P_l \).

With probability \( \frac{1}{2} \), \( b_l = v \), and all honest participants in the view of \( P_l \) hold the same value in the next iteration. Since there are at least \( (1 - \alpha)n \) of them, \( P_l \) will set \( h_l \leftarrow 1 \) in the next iteration.

Theorem 1 now follows directly from Lemma 5, Lemma 8, Algorithm 3, Claim 9, Claim 10, and Claim 11.
Chapter 4

Negative Result

Closely related to the Byzantine Agreement problem is the Broadcast problem. We give the standard definition below:

**Definition 4.** (Broadcast) For a set of participants $S = \{P_1, ..., P_N\}$, and a distinguished dealer $P_d \in S$ holding a message $m$, when the protocol terminates, the following conditions must hold for any adversary:

- (Agreement) Every honest participant $P_i$ outputs the same $m^*$, for some $m^*$.
- (Validity) If the $P_d$ is honest, then $m^* = m$.

In the standard model, it’s clear how to use a Byzantine Agreement protocol to implement Broadcast: the dealer simply sends its message to every other participant, and then all participants ran a Byzantine Agreement protocol to decide on an output message. With some extra steps, we can also achieve the same in our relaxed model. Note that for some $P_i$ not in the view of $P_d$, it enters the protocol with the player id $P_d$, but not its public key $P_{kd}$.

In the standard model assuming a PKI setup, while Byzantine Agreement is not possible in the presence of more than $1/2$ corruption, Broadcast is possible for any number of corruption [DS83]. As will be shown in this section, Broadcast in our relaxed model is equivalent to Byzantine Agreement. We first show how to use Byzantine Agreement to implement Broadcast, and then show impossibility results for Broadcast assuming $\alpha \geq 1/2$, or $\delta \leq 2\alpha$.

### 4.1 Broadcast from Byzantine Agreement

We now show our protocol (Algorithm 4) to achieve Broadcast in our relaxed model, assuming $\delta > 2\alpha$. For simplicity, we only consider the binary version.
Algorithm 4 Broadcast

1: The dealer $P_d$ runs a Graded Broadcast protocol with message $m \in \{0, 1\}$. In the end, every honest $P_i$ in the view of $P_d$ (i.e. $P_i \in \Gamma_d$) outputs $(m_i, g_i)$.

2: For every honest $P_i \in \Gamma_d$:
   - If $g_i = 1$, then send $m_i$ to all other participants in the view of $P_i$, and sets $v_i \leftarrow m_i$.
   - Else, sets $v_i \leftarrow 0$

3: For every honest $P_i \notin \Gamma_d$:
   - If $P_i$ receives at least $(\delta - \alpha)n$ messages of a unique message $m$, then set $v_i \leftarrow m_i$
   - Otherwise, set $v_i \leftarrow 0$

4: Every honest participant $P_i$ runs a Byzantine Agreement protocol with input $v_i$, and use its output $v^*$ as the broadcast output.

We now show the following claims about Algorithm 4.

Claim 12. (Agreement) If $\delta > 2\alpha$, the output of every honest $P_i$ is the same.

Proof. This follows directly from the correctness of Byzantine Agreement (Theorem 1).

Claim 13. (Validity) If $P_d$ is honest with message $m \in \{0, 1\}$, and if $\delta > 2\alpha$, then every honest $P_i$ outputs $m$.

Proof. By the correctness of Graded Broadcast (Lemma 5), if $P_d$ is honest, then every honest $P_i \in \Gamma_d$ outputs $(m, 1)$ in Step 1, and sets $v_i \leftarrow m$ in Step 2.

For every $P_i \notin \Gamma_d$, by our model assumption, at least $(\delta - \alpha)n$ honest participants are in the overlapping of the views of $P_d$ and $P_i$. Hence $P_i$ receives $m$ at least $(\delta - \alpha)n$ times.

Note that any honest $P_j$ participant either sends out $m$ (in case of $P_j \in \Gamma_d$), or nothing in Step 2. $P_i$ can only receive some $m' \neq m$ corrupted participants in its view in Step 3. By assumption, there are at most $\alpha n < (\delta - \alpha)n$ corrupted participants in the view of $P_i$. Hence $P_i$ also sets $v_i \leftarrow m$ in Step 4.

By the correctness of Byzantine Agreement (Theorem 1), all honest participants output $m$.

The above claims lead to the following lemma that we will use as a building block to prove Theorem 2.

Lemma 14. Assuming $\delta > 2\alpha$, there exists a Broadcast protocol.


4.2 Negative Result for Broadcast

The case of $\alpha \geq 1/2$ We start with the case where $\alpha \geq 1/2$. Let $p, r$ be positive integers. Without loss of generality, assume $p > r$. Figure 4.1a shows a configuration $C_1$ where a total of $N = r + 2p$ participants are divided into 4 groups, $A, B, C$ and $F$ with sizes $|A| = r, |B| = p - r, |C| = r$, and $|F| = p$. Groups $A, B, C$ are honest participants, while $F$ are corrupted. Group $A$ sees $A, B, F$ in its view, group $C$ sees $C, B, F$ in its view, and group $B$ sees $A, C$, and only $p - r$ corrupted participants of $F$ in its view.
We now show the following claims:

**Claim 15.** \( C_1 \) represents a valid configuration with \( \alpha = 1/2 \) and any \( \delta = (2p - r)/2p \).

**Proof.** We first verify each of \( A, B, C \)'s view:

\[
|\Gamma_a| = |\Gamma_b| = |\Gamma_c| = 2p
\]

Within each view, we verify that the corruption in \( \Gamma_a, \Gamma_c \) are exactly \( 1/2 \), and the corruption in \( \Gamma_b \) is \( (p - r)/2p < 1/2 \).

We finally verify that the overlapping between \( A \) and \( C \) is exactly \( (2p - r)/(2p) \). The overlappings between \( B \) and \( A \) and between \( B \) and \( C \) are also both \( (2p - r)/(2p) \).

Let \( P_d \in C \) be some honest participant in group \( C \) with message \( m \in \{0, 1\} \). We now claim that broadcast is impossible for \( P_d \) in \( C_1 \).

**Claim 16.** Broadcast is impossible for \( P_d \) in configuration \( C_1 \).

**Proof.** Suppose there is a protocol \( \Pi \) that achieves Broadcast with dealer \( P_d \) in configuration \( C_1 \). We consider an adversary with the following strategy, as illustrated in Figure 4.1b.

The adversary locally simulate a group \( C' \) of size \( r \), with the same ids as \( C \) by with newly assigned keypairs for their digital signatures. Group \( C' \) runs the protocol \( \Pi \) honestly, with the corresponding \( P_d' \in C' \) start with message \( m' \neq m \). When \( \Pi \) requires participants in \( C' \) to send messages to \( B \), they pretend that \( B \) have ignored their messages.

The corrupted group \( F \) disables \( r \) of them, and let the rest run the protocol \( \Pi \) honestly, except they ignore all messages from \( B \) and \( C \). When \( \Pi \) requires them to send messages to \( B \) and \( C \),
they pretend that $B$ and $C$ have ignored its message.
By symmetry, it’s clear that group $A$ cannot distinguish $C$ from $C'$, hence $P_d$ from $P'_d$. Hence broadcast is impossible for $P_d$. \hfill \square

The above claims lead to the following lemma that we will use as a building block to prove Theorem 2.

**Lemma 17.** Assuming $\alpha \geq 1/2$, for any $0 < \delta < 1$, there does not exist a Broadcast protocol in our model.

**Proof.** It’s clear that for any positive integer $N \geq 5$ and valid $0 < \delta < 1$, we can choose positive $p, r$ such that $(2p - r)/(2p) = \delta$, and $2p + r = N$. By Claim 15 and Claim 16, we can construct a valid configuration $C_1$ with $\alpha = 1/2$, in which Broadcast is impossible for some honest participant. \hfill \square

**The case of $\delta \leq 2\alpha$**  Now it suffice to assume $\alpha < 1/2$. We similarly show an impossibility result for Broadcast in the case of $\alpha < 1/2$ and $\delta \leq 2\alpha$. Let $p, r$ be positive integers, and assume $p > r$. Figure 4.2a shows a configuration $C_2$, where a total of $N = 2p + 2r$ participants are divided into 4 groups, $A, B, C$ and $F$, with sizes $|A| = r, |B| = p, |C| = r, \text { and } |F| = p$. Groups $A, B, C$ are honest participants, while $F$ are corrupted. Group $A$ sees $A, B, F$ in its view; group $C$ sees $C, B, F$ in its view; and group $B$ sees $A, C$, and only $p - r$ corrupted participants of $F$ in its view.

We now show the following claims:
Claim 18. \( C_2 \) represents a valid configuration with \( \alpha = p/(2p + r) < 1/2 \) and \( \delta = 2\alpha \).

Proof. We first verify each of \( A, B, C \)'s view:

\[
|\Gamma_a| = |\Gamma_b| = |\Gamma_c| = 2p + r
\]

Within each view, we verify that the corruption in \( \Gamma_a, \Gamma_c \) are exactly \( p/(2p+r) \), and the corruption in \( \Gamma_b \) is \( (p-r)/(2p+r) < p/(2p+r) \).

We finally verify that the overlapping between \( A \) and \( C \) is exactly \( 2p/(2p + r) = 2\alpha \). The overlaps between \( B \) and \( A \) and between \( B \) and \( C \) are both \( 2p/(2p + r) = 2\alpha \).

Let \( P_d \in C \) be some honest participant in group \( C \), with message \( m \in \{0,1\} \), we similarly claim that broadcast is impossible for \( P_d \) in \( C_2 \).

Claim 19. Broadcast is impossible for \( P_d \) in configuration \( C_2 \).

Proof. The proof is analogous to the proof of Claim 16. The adversarial strategy is illustrated by Figure 4.2b, with the only difference being that the adversary no longer disables \( r \) of the corrupted participants in \( F \).

The above claims lead to the following lemma that we will use as a building block to prove Theorem 2.

Lemma 20. Assuming \( \alpha < 1/2 \) and \( \delta \leq 2\alpha \), there does not exist a Broadcast protocol in our model.

Proof. Similar to Lemma 17 it follows from Claim 18 and Claim 19.

Theorem 2 now follows directly from, Lemma 14, Lemma 17 and Lemma 20.
Chapter 5

Conclusions

This thesis presents a generalization of the standard Byzantine Agreement problem, where every honest participant knows and communicates with only a subset of all participants. This generalization is motivated by real world peer-to-peer networks, where not every pair of participants are directly connected. Our setting is parameterized by $\alpha$, the maximum fraction of corruption in each honest “view”, and $\delta$, the minimum fraction of overlapping between any pair of honest “views”.

In Chapter 3, we present a protocol that runs in expected polynomial round assuming $\delta > 2\alpha$. If we further assume $\alpha \leq 1/2 - \epsilon$ for any constant $\epsilon$, the protocol runs in expected constant round. In Chapter 4, we show that it’s impossible to achieve Byzantine Agreement assuming $\alpha \geq 1/2$ or $\delta \leq 2\alpha$. Together, they show that our assumption for the positive result is tight. However, whether there is an expected constant round protocol assuming only $\delta > 2\alpha$ is still open.

One future direction to extend this thesis is to consider single direction edges in our communication graph. That is, if some honest participant $P_i$ is in the view of another, $P_j$, it’s possible that $P_j$ is not in the view of $P_i$. Some of our idea in Chapter 3 still work with stronger parameter assumptions. However, we don’t have matching positive results and negative results for this setting yet.
Bibliography


