RANDOMIZED ALGORITHMS FOR RELIABLE BROADCAST

by

Vinod Vaikuntanathan

Submitted to the Department of Electrical Engineering and Computer Science in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2009

Author .....

Department of Electrical Engineering and Computer Science

January 30, 2009

Certified by ...

Shafi Goldwasser

RSA Professor of Electrical Engineering and Computer Science

Thesis Supervisor

Accepted by...

Terry P. Orlando

Chair, Department Committee on Graduate Students

No two on earth in all things can agree. All have some daring singularity.

Winston Churchill

Contents

List of Tables

Acknowledgments

FIRST AND FOREMOST, I would like to thank my advisor, Shafi Goldwasser. She took me under her wing during my very first months of graduate school, a time when I knew little of anything. She has been an endless source of fascinating problems, and amazingly creative ideas over the years. If there is one thing I hope I inherit from her, it would be her exquisite taste in research directions.

Getting through the frustrating first years of grad school would have been impossible without the mentorship of Alon Rosen. He taught me the cryptographic way of thinking, and always had time to listen to my ideas. Rafael Pass is a dear friend and collaborator. He has been an amazing person to work with – creative, energetic and brilliant.

I am grateful to Ran Canetti, Silvio Micali, Madhu Sudan and Ron Rivest for always being available to answer my questions and for their inspiring courses. Special thanks to Silvio for expressing keen interest in my work, and for his generous help and guidance. Ran hosted me at IBM Research during the summers of 2006 and 2007. I am very grateful for the warm and active atmosphere at IBM, and the people responsible for it.

The summer of 2007, which I spent at SRI International as an intern, was one of the most fun and productive times I have ever had. I am grateful to Chris Peikert and Brent Waters for hosting me at SRI and for the terrific collaborative work during that summer.

The results in this thesis are based on joint work with Michael Ben-Or, Shafi Goldwasser and Elan Pavlov. Michael sent me a draft of his result which started off the work in this thesis – I am very grateful to him for that. I would like to thank my super-star co-authors on all the papers I wrote during graduate school, none of which appear in this thesis, but all of which I consider formative. Special thanks to Adi Akavia, Zvika Brakerski, Susan Hohenberger, Yael Kalai, Guy Rothblum, abhi shelat, Eran Tromer and Mayank Varia.

I would like to thank the graduate students at MIT and the Weizmann Institute who contributed to my wonderful grad school experience. Thanks to Be Blackburn and Joanne Hanley for their timely and meticulous help with all the administrative details. Special thanks to Jaykumar, Sreeja and Vivek for their help and friendship. Sravana has been a wonderful companion and a source of immense inspiration – her love and support are deeply appreciated.

I thank my brother Visakh for his encouragement and all the "little-brother-ly stuff". Finally, and quite obviously, none of this would have been possible without

my physical existence and upbringing, for which I thank my parents, SUDHA and VAIKUNTANATHAN. This thesis is dedicated to them.

Please forgive me any omissions.

Abstract

In this thesis, we design randomized algorithms for classical problems in fault-tolerant distributed computing in the full-information model. The full-information model is a strong adversarial model which imposes no restrictions on the computational power of the faulty players nor on the information available to them. Namely, the faulty players are infinitely powerful and are privy to all the communications in the network.

Our main result is the construction of two efficient randomized protocols for Byzantine agreement, a classical problem in distributed computing. Byzantine agreement is the problem of simulating the reliable broadcast functionality in a network where all communication is person-to-person. We design two randomized Byzantine agreement protocols in a synchronous network with an expected round-complexity of $O(\log n)$ rounds. One of the protocols is resilient against an all-powerful, full-information adversary that corrupts less than *a third* of the number of players (whereas the other protocol is resilient against a fourth fraction of corruptions). Our protocols have the following additional features.

- The fault-tolerance of our protocols can be increased to less a half fraction of faults, if there is a public-key infrastructure setup available that allows the players to compute (public-key) digital signatures.
- Our protocols work even if the source of randomness is a "somewhat random" source (also called a Santha-Vazirani source). The price we pay is a decreased fault-tolerance.

Our second result is the design of a compiler that transforms a randomized distributed protocol that tolerates benign, fail-stop faults into a protocol that tolerates malicious, Byzantine faults. Fail-stop faults follow the protocol specification, but may stop in the middle of the execution. On the other hand, Byzantine faults are arbitrarily malicious. The resulting protocol has almost the same fault-tolerance and efficiency as the original protocol. Our compiler suggests a modular way to design distributed protocols: first, design a protocol that tolerates fail-stop faults, and use our compiler to "boost" the fault-tolerance to Byzantine faults. The design of the compiler is based on a new protocol technique that we develop, called "auditing" of distributed protocols.

Thesis Supervisor: Shafi Goldwasser
Title: RSA Professor of Electrical Engineering and Computer Science

1

Introduction

A reliable broadcast channel is a communication channel that guarantees the recipient of a message that everyone else received the same message. The existence of built-in reliable broadcast channels is a convenient abstraction, widely assumed by protocol designers in distributed computing and cryptography [GMW87, BGW88, CCD88, GGL98].

Despite its utility, reliable broadcast channels do not exist in nature. Consider the two most natural kinds of networks that we encounter, namely wired networks (a quintessential example being the internet itself) and wireless and radio networks. In the internet, where all communication is person-to-person (at best!) and some of the people involved are untrustworthy, the reliable broadcast guarantee no longer exists. Though an untrustworthy sender may claim to send the same message to everyone, he could actually send different messages to different people. In the case of a wireless or a radio network, a malicious wireless transmitter might tactfully interfere with a transmission so that a message is heard by one person but not by another. Again, in this case, the guarantee of reliable broadcast does not exist. Faced with this situation, we would like to

> *efficiently simulate a reliable broadcast channel on top of traditional networks.*

This is called the reliable broadcast problem, which is the focus of this thesis.

RELIABLE BROADCAST AND BYZANTINE AGREEMENT. In 1980, Pease, Shostak and Lamport [PSL80] insightfully defined the problem of Byzantine agreement, which is equivalent to the problem of simulating a reliable broadcast channel over point-to-point networks. Since then, Byantine Agreement has arguably become the central problem in distributed computation tolerating faulty behavior. Informally, the problem is to maintain a common view of the world in the presence of faulty processes that strive to prevent the good processes from

reaching agreement. The faults can range from simple "crashing of processes" to processes with malicious intent co-ordinating to mislead the good ones into disagreement.

Definition 1. A protocol among n players, in which each player starts with an input is a Byzantine Agreement protocol, if the following conditions hold:

- *Completeness (Validity):* If the input values of all the honest players are the same (say, a value v), then the output of all the honest players is v itself.
- *Soundness (Agreement):* No two honest players have different outputs, regardless of the inputs of the players.

Given a Byzantine agreement protocol, achieving reliable broadcast is easy: the sender in the reliable broadcast scenario sends his message to all the players, and the players then run a Byzantine agreement protocol to agree on a message. If the sender is honest, the completeness property of Byzantine agreement guarantees that all the honest players receive the sender's message. On the other hand, even a malicious sender cannot force the honest players into disagreement.

From this point on, we will focus our attention on the problem of Byzantine agreement.

OUR MODEL. The players can communicate to each other via point-to-point links. We assume that the network is synchronous: that is, there is a global (but "implicit") clock governing the time required to deliver messages. The protocol execution, thus, happens in "rounds" (for simplicity, think of each round as a tick of the global clock), and the messages sent at the end of a round are delivered at the beginning of the next round.

We address a very strong fault-model. An all-powerful adversary corrupts a subset of t players in the beginning of the protocol. The adversary is *Byzantine*, meaning that the strategy of the corrupted players is computationally unbounded and the worst possible. The adversary is a *full-information adversary*, meaning that it makes its decisions based on the information about the state of all the players (faulty as well as honest) including their coin tosses up to and including the current round, and the entire history of communications between them.[1] Thus, the adversary is privy to *all the activity in the internals of all processors as well as in the network.* No meaningful privacy guarantees can be made in the presence of such an adversary. In addition, the delivery of messages is in the *rushing* model. Namely, a dishonest player may possibly see all messages sent in round i by other players, before he sends his own round i messages.

[1]There have been at least two models of full-information adversaries in the literature. Goldreich, Goldwasser and Linial [GGL98] defines a full-information adversary as one that has access to all the communications in the network, but not the internal coin-tosses of the players. The stronger variant, which we use in this thesis, gives the adversary access to the (past) coin-tosses of all the players as well. This stronger adversary was called an intrusive adversary in the work of Chor and Dwork [CD89].

PREVIOUS WORK. Since its introduction in the work of Pease et al.[PSL80], the problem of Byzantine Agreement has been a source of enormous attention. Pease et al. proved (in [PSL80] itself) that no deterministic algorithm can achieve Byzantine Agreement among n players in the presence of t faults if $n \leq 3t$ (This bound was later extended to the case of randomized algorithms by Karlin and Yao [KY86]). They also constructed a (deterministic) algorithm that solves BA for any $n > 3t$, in a synchronous full-information network. Once the feasibility of BA was shown, further attempts concentrated on reducing the complexity of achieving agreement. The standard complexity measures of interest are the number of rounds, and the total communication and computational complexity of the protocol, the former being the most interesting of them. The protocol of [PSL80] had a round complexity of $t + 1$ rounds, which was shown to be optimal for *deterministic* protocols by Fischer and Lynch [FL82]. However, the communication complexity of the protocol was exponential in n. Following a series of works [BG89a, BGP92], Garay and Moses [GM98] constructed a BA protocol that runs for $t + 1$ rounds, with a polynomial communication.

Faced with the lower bound on the round complexity for deterministic protocols, the natural direction of research was to find ways to overcome this limitation, the first choice being to resort to randomization. This direction was pursued early on, starting with the work of Ben-Or and Rabin [Ben83, Rab83] who showed how to reach Byzantine agreement quickly given a source of "common coins".[2] Thus, the bulk of the attention was concentrated on constructing protocols that generate a common-coin in a network. Numerous subsequent works trod this path, showing how to design better common-coin protocols (see [Bra87, DSS90, CMS89, CC85] and the references therein).

This line of research culminated in the work of Feldman and Micali [FM97], who designed a protocol to generate a common coin in $O(1)$ rounds and with polynomial communication, under the assumption that the point-to-point channels connecting pairs of processors are *private*, or that the processors are computationally bounded and cryptography exists. This, in turn, gave Byzantine Agreement protocols that run in $O(1)$ rounds. On the other hand, without the assumption of private channels or a computationally bounded adversary (i.e, in the full-information model), the best known protocol that achieved Byzantine Agreement was due to Chor and Coan [CC85] and had a round complexity of $\Theta(\frac{t}{\log n})$ rounds. This raises the following natural question:

> *How efficient can Byzantine Agreement be, relying only on randomization, without using cryptographic techniques or the private channels assumption?*

Our first and main contribution in this thesis is to construct efficient Byzantine agreement protocols in the full-information model, namely without relying on cryptographic techniques or the private channels assumption.

[2]The work of Ben-Or is in the asynchronous model, where deterministic Byzantine agreement is impossible [FLP83]. Rabin's result holds in both the synchronous and asynchronous models.

Achieving results in the full information model is important, as these results hold unconditionally. Currently, all results in the computational model hold only under intractability assumptions such as the existence of one-way functions. The results in the private channels model are conditioned on the availability of private physical communication channels between pair of players. This elegant abstraction is implemented by resorting to secure encryption, the existence of which is again based on intractability assumptions.

1.1 Byzantine Agreement in the Full Information Model

The main result in this thesis is the construction of two protocols for Byzantine agreement in the full-information model with a logarithmic round-complexity. The first of these protocols achieves a fault-tolerance of $t \leq (1/4 - \epsilon)n$, whereas the second achieves a fault-tolerance of $t \leq (1/3 - \epsilon)n$. Both protocols have an expected round-complexity of $O(\log n/\epsilon^2)$ rounds. The second of these protocols achieves asymptotically optimal fault-tolerance, as Pease, Shostak and Lamport [PSL80] and Karlin and Yao [KY86] show that BA is not possible if $n \leq 3t$.

These protocols constitute an exponential improvement in the round-complexity over the best previously known Byzantine agreement protocol in the full-information model, due to Chor and Coan [CC85]. The Chor-Coan protocol had an expected round-complexity of $O(n/\log n)$ rounds.

The Byzantine agreement protocols are derived via new and collective coin-flipping protocols in the full-information model without built-in reliable broadcast channels. There is an extensive body of work [GGL98, GVZ06, Fei99, RZ01, Blu81] on efficient random selection protocols in the full information model, which we could potentially take advantage of. However, the challenge in using these protocols, is that in *all of these works, an additional assumption is made: reliable broadcast channels exist for free.* Since our goal is to design protocols that simulate reliable broadcast, this is an assumption we simply cannot make. Indeed, both the correctness and efficiency (including the round complexity) of [GGL98, GVZ06, Fei99, RZ01, BL85] hold only under the additional assumption that broadcast is an atomic unit-cost operation. Nevertheless, we use the ideas from the coin-flipping protocols of [Fei99, RZ01] in an essential way in our Byzantine Agreement protocol.

CONCURRENT WORK. In a concurrent work, King, Saia, Sanwalani and Vee [KSSV06a, KSSV06b] constructed a Byzantine agreement protocol in the synchronous full-information model with a polylog(n) round-complexity tolerating $t < (\frac{1}{3} - \epsilon)n$ dishonest players. In addition, their protocols achieve a slightly weaker notion of agreement – called "almost everywhere agreement" – with the optimal communication complexity of $\tilde{O}(n)$.

1.2 Trading off Fault-Tolerance with Other Parameters

We show various tradeoffs between the fault-tolerance of Byzantine agreement and other parameters of the system. In particular, we show how to tradeoff the fault-tolerance against the setup assumptions available, the round-complexity and the quality of randomness available.

SETUP ASSUMPTIONS. The first extension deals with increasing the fault-tolerance. The result of Pease, Shostak and Lamport [PSL80] shows that no Byzantine agreement protocol can achieve a fault-tolerance better than *a third*, unless there is some additional setup. The most common setup assumption is that of a Public-key Infrastructure. Assuming that a PKI is available, we show how to construct a logarithmic-round BA protocol even when upto *half* the players are corrupted.

One drawback of this setting is that one has to assume a computationally bounded adversary (to allow for cryptographic signtures). We note that the advantage in this protocol is that we use signature schemes, which are typically much easier to construct (and also complexity-theoretically simpler) than public-key encryption schemes (which are necessary to implement private channels).

ROUND-COMPLEXITY. The second result shows how to achieve a better round-complexity at the expense of the fault-tolerance of the BA protocol. In contrast to our main result which shows a logarithmic round BA protocol tolerating a constant fraction of faults, we show how to achieve an expected round-complexity of $O(1)$: the price we pay is that the fault-tolerance of this protocol is only $n/\log^{O(1)} n$, sublinear in the number of players.

QUALITY OF RANDOMNESS. In the context of synchronous Byzantine agreement, randomness is a critical resource that enables achieving solutions with far higher efficiency than the deterministic solutions. However, typically, the randomized algorithms for BA are designed under the assumption that the players have access to a source of unbiased, independent coins. Physical sources of randomness, on the other hand, are rarely unbiased and independent although they do seem to exhibit somewhat imperfect randomness. Thus, it is a natural question whether "imperfect randomness" is good enough for to achieve efficient Byzantine agreement in the full-information model.

This question was first posed by Goldwasser, Sudan and Vaikuntanathan [GSV05] who answered this question in the affirmative in the case where all the players have access to independent block-wise random sources with sufficient min-entropy. Recently, Kalai, Li, Rao and Zuckerman [KLRZ08] improved this result to the case where each player has an independent general weak random source. However, both these results assume that the random sources of any two players are independent of each other.

We show a Byzantine agreement protocol where each honest player has access to a Santha-Vazirani (SV) source, and where the random sources of the players are not necessarily independent. A γ-SV source is one that outputs a sequence

of bits where each bit is somewhat random (that is, has a bias of at most γ) even given all the other bits in the output. In particular, we consider the case where the concatenation of the random sources of all the players forms a γ-SV source for some $\gamma = \frac{c}{\log n}$. Given such a random source, we show how to achieve Byzantine agreement tolerating $t < 1 - \frac{5}{6}e^{2c}$ faulty players. In particular, for $\gamma = \frac{1}{12 \log n}$, this gives us a fault-tolerance of $(1 - \frac{5}{6}e^{1/6})n = 0.003n$ faulty players. As c becomes smaller and smaller, this number gets better and better (tending to $1/3$ in the limit).

1.3 Auditing Distributed Protocols and Applications

We introduce a new protocol transformation called Audit. The goal of Audit is to transform any distributed protocol Π that possibly assumes broadcast channels, into another protocol that simulates Π "as well as possible" when no reliable broadcast channels are given.

Of course, one could simulate any protocol assuming broadcast channels, by replacing each broadcast instruction of the protocol with the execution of a sub-protocol for implementing reliable broadcast. This naive approach, however, runs into trouble, as it may increase the round-complexity of the simulated protocol prohibitively. For example, consider the leader election protocols of [Fei99] and [RZ01] which run in $O(\log^* n)$ rounds. Replacing, every broadcast instruction made in [Fei99, RZ01], by the best Byzantine agreement protocol currently known for the full information model (that is, from the work in this thesis), would yield a $O(\log n \log^* n)$ leader election protocol. Moreover, even if one were to design a Byzantine agreement protocol with expected round-complexity $k = o(\log n)$, it would not merely imply an $O(k)$ factor slow-down in the number of rounds. As already observed by Chor and Rabin [CR87], the probability that all executions of a probabilistic protocol halt within the expected number of rounds proved for a single execution can be exponentially small in the number of executions.

Audit allows us to take any protocol Π designed assuming reliable automatic broadcast and the identity of a special player P (called the "auditor") and produce a protocol $\mathsf{Audit}(P, \Pi)$ which *does not assume automatic broadcast.* When the *auditor* P is a good player, the output distribution of non-faulty players in $\mathsf{Audit}(P, \Pi)$ will be as in Π. Even when P is not an honest player, the output of the non-faulty players in $\mathsf{Audit}(P, \Pi)$ will be as in Π *except* that some non-faulty players may output $\perp$. A significant feature of Audit is that the round-complexity of $\mathsf{Audit}(P, \Pi)$ is a (small) constant times that of Π.

Informally, the role of the auditor P is to "audit" the execution of $\mathsf{Audit}(P, \Pi)$ and ensure that in *each round* all honest players get the same messages, thus simulating the reliable broadcast functionality. If the auditor is honest, this will be ensured. Otherwise the worst that may happen is that some honest player will receive a $\perp$ message instead of what was sent by honest players.

In independent work, Katz and Koo [KK06] introduced the notion of moderated Verifiable Secret Sharing (VSS). The idea of a moderator is very reminiscent

of the idea of an auditing committee which consists of a single auditor. In contrast with our work, [KK06] obtain their results in the private channels model. We remark, that although our primary interest in this paper is the full information model, the Audit transformation introduced here will apply to the private channels model as well.

1.3.1 An Application of Auditing: "Boosting" Fault-Tolerance of Protocols

Designing distributed algorithms that tolerate Byzantine failures is a complex task. The task of the protocol-designer would be simpler, were there a *compiler* that takes as input a distributed protocol Π that tolerates *benign failures*, and outputs a robust distributed protocol Π' that tolerates the most *severe* failures.

The problem of designing a compiler that automatically converts any *deterministic* protocol that tolerates fail-stop faults to one that tolerates Byzantine faults was considered by Hadzilacos[Had83] and Neiger and Toueg [NT90]. They provide a procedure that converts any deterministic protocol Π_{in} that tolerate fail-stop faults into Π_{out} whose output in the presence of Byzantine faults, is identical to the output of Π_{in} in the presence of fail-stop faults. We remark that their transformation explicitly assumes that in Π_{in} the inputs of honest players is sent to all other players in the first round, however when the adversary is intrusive this restriction is unnecessary.

Bracha [Bra84] explicitly raised the question, which we partially address here, whether such a transformation is possible for randomized protocols as well. The additional challenge over the deterministic case is that a fail-stop fault in a randomized protocol will flip coins fairly as prescribed by the protocol, but a Byzantine failure could potentially use any biased coins it likes (or none at all, in particular).

We show a compiler that takes any randomized protocol Π_{in} designed to tolerate a Fail Stop t-adversary, where the *source of randomness of all players in* Π_{in} *is an SV-source* [SV84],[3] into a protocol Π_{out} that tolerates a Byzantine $\mathsf{min}(t, \frac{n}{3})$-adversary. If the round-complexity of Π_{in} is r, that of Π_{out} is $O(r \log^* n)$. Previously, Hadzilacos, Neiger-Toueg, and Bracha [Had83, NT90, Bra84] constructed such a compiler for deterministic protocols, and [Bra84] raised as an open question, whether such a compiler exists for randomized protocols.

Our main result of this section is:

Theorem 1. There is a compiler that converts any r_Π-round protocol Π in which the randomness source is a γ-SV-source and which tolerates a fail-stop t-adversary, to a $r_{\Pi'}$-round protocol Π' that uses a uniformly random source and tolerates a Byzantine t'-adversary where $t' = \mathsf{min}(t, \frac{n}{3})$, and $r_{\Pi'} = O(r_\Pi \log^* n)$.

[3] Namely, given all coins tossed by all players thus far, the probability that the next coin is "heads" is bounded between $\frac{1}{2} - \gamma$ and $\frac{1}{2} + \gamma$ for some $\gamma > 0$.

This result partially answers the open question of Bracha [Bra84]. The compiler we construct assumes that the input protocol works even if the coins of the players are drawn from an SV-source; thus, it is not the most general statement we would like to make. It is an interesting question whether the condition in the above theorem on the coins of the players in the input protocol can be removed.

1.4 Overview of the Thesis

The main contribution of this thesis is the construction of efficient Byzantine agreement (or equivalently, reliable broadcast) protocols in the full-information model. In this journey, we additionally construct tools that enable us to remove the assumption of built-in reliable broadcast channels from distributed protocols.

Chapter 2 - Preliminaries. We introduce basic notation and recall basic notions that will be used throughout the thesis.

Chapter 3 - Classical Work on Byzantine Agreement. We recall classical definitions of Byzantine agreement, reliable broadcast, graded broadcast and various random selection problems. We also prove and refine the classical connection between randomized Byzantine agreement and random selection problems.

Chapter 4 - New Byzantine Agreement Protocols. This chapter contains the main contribution of our work, namely the construction of two Byzantine Agreement protocols in the full-information model, with an expected round-complexity of $O(\log n)$.

These results proceed by constructing committee-election protocols that do not assume broadcast channels, using technical ideas from the works of Feige [Fei99] and Russell and Zuckerman [RZ01]. In particular, these results give us the most efficient leader-election, collective coin-flipping and committee-election known so far in the full-information model without built-in broadcast.

Chapter 5 - Auditing a Distributed Protocol and an Application. The main motivation behind the construction of reliable broadcast protocols is to convert distributed protocols that assume built-in broadcast channels into ones that do not make such an assumption. This transformation, however, comes at a cost: each invocation of the reliable broadcast channel in the original protocol is replaced with a reliable broadcast protocol, and this increases the round-complexity of the original protocol by a *multiplicative* factor of $O(\log n)$ (using the protocols in Chapter 4, which are the best known so far in the full-information model).

This raises the following question: given a protocol that assumes built-in reliable broadcast channels, is it possible to run (a modified version of) the

protocol that (a) does not take much more time to run than the original protocol, and (b) still provides a meaningful guarantee? We answer this question in the affirmative.

We show how to apply this to the problem of boosting the fault-tolerance of distributed protocols. Namely, we design a mechanism that transforms randomized distributed protocols tolerating fail-stop faults (satisfying certain technical conditions) into an equivalent protocol that tolerate Byzantine faults.

Chapter 6 - Extensions. In this chapter, we show various tradeoffs between the fault-tolerance of Byzantine agreement and other parameters of the system. In particular, we show how to tradeoff the fault-tolerance against the setup assumptions available, the round-complexity and the quality of randomness available.

ACKNOWLEDGMENTS. This thesis is based on the material in joint work with Michael Ben-Or, Shafi Goldwasser and Elan Pavlov, and has been supported by NSF Grants CCF-0514167 and CCF-0635297, an Israel Science Foundation Grant 700/08 and an MIT Akamai Presidential Fellowship. Portions of the thesis originally appeared as extended abstracts in the *38'th Symposium on the Theory of Computing* [BPV06] and the *47'th Foundations of Computer Science* [GPV06].

2

Preliminaries

2.1 Modeling a Synchronous Distributed System

We first present a computational model for algorithms in a synchronous distributed system. The reader might find it convenient to refer back to this section, since much of the later chapters will use the model presented here. Our modeling follows in a large part the material in Lynch [Lyn96, Chapter 2] and Feldman and Micali [FM88], while adapting them to our setting.

The modeling consists of defining the network characteristics, the computation of a protocol and the adversarial model.

THE NETWORK. We consider a distributed system made up of n players, $P_1, \ldots, P_n$, and denote by $\mathcal{P} = \{P_1, \ldots, P_n\}$ the set of players. The communication network is made up of a bidirectional communication link between every pair of players, which is the only means by which the players can communicate. In particular, we stress that there is no built-in broadcast channel in the system. One can imagine generalizations of this model where not all pairs of players are connected by direct communication links; we will deal with some of these cases in Section 2.1.1.

THE COMPUTATION. A distributed protocol Π is specified by an n-tuple of interacting (possibly randomized) Turing machines $(\Pi_1, \Pi_2, \ldots, \Pi_n)$, where we think of Π_i as the program executed by player P_i.

The execution of a distributed protocol Π proceeds in "rounds". In each round, every player P_i receives the messages that were sent to P_i in the previous round, does some local computation (as specified by its program Π_i) and sends messages to all the players. In particular, the model makes the implicit assumption that there is a global clock that governs the rate of computation, and that the messages sent by a player are received within a bounded amount of time by the intended recipient.

Formally, we will let $\mathsf{state}_{i,r}$ denote the configuration of the Turing machine Π_i in the beginning of round r. The configuration consists of the contents of all the tapes of the Turing machine, as well as the finite state control. Let $m^r_{j\to i}$ denote the message sent from player P_j to P_i in the beginning of round r. Then, the program Π_i computes

$$(\mathsf{state}_{i,r+1}, m^{r+1}_{i\to 1}, \ldots, m^{r+1}_{i\to n}) \leftarrow \Pi_i(\mathsf{state}_{i,r}, m^r_{1\to i}, \ldots, m^r_{n\to i})$$

P_i sends the messages $m^{r+1}_{i\to j}$ to player P_j and updates its local state to $\mathsf{state}_{i,r+1}$.

At some point, the Turing machine Π_i enters a special halt state and writes a string on the output tape, which is the local output of P_i on this execution.

THE RANDOMNESS. We consider randomized distributed protocols. Each of the programs Π_i that define the distributed protocol Π are allowed to access a random source, namely a sequence of bits generated according to some distribution. In such a case, each step of the Turing machines Π_i is randomized. If we denote the random tape of the Turing machine Π_i by $\overline{\rho}_i = (\rho_{i,1}, \rho_{i,2}, \ldots)$, then

$$(\mathsf{state}_{i,r+1}, m^{r+1}_{i\to 1}, \ldots, m^{r+1}_{i\to n}) \leftarrow \Pi_i(\mathsf{state}_{i,r}, m^r_{1\to i}, \ldots, m^r_{n\to i}; \rho_{i,r})$$

When considering a randomized protocol, both the output and the running time of the protocol are random variables.

We define two specific distributions of randomness in this work – the uniform distribution, and the Santha-Vazirani distribution [SV84]. The uniform distribution of randomness is the one that is traditionally assumed by randomized distributed protocols. Here the concatenation of random tapes $\overline{\rho}_1 \circ \overline{\rho}_2 \circ \ldots \circ \overline{\rho}_n$ consists of a sequence of completely unbiased and independent bits.

The second one is a distribution defined by Santha and Vazirani [SV84], called the Santha-Vazirani source (or, SV-source). Informally, in an SV-source parametrized by a real number $\gamma \in [0, \frac{1}{2}]$, each bit has a bias of upto γ, and the exact bias can depend on the values of all the previous bits. Slightly more precisely, for $\gamma \in [0, \frac{1}{2}]$, a γ-SV-source is a sequence of bits $(b_1, b_2, \ldots)$ such that for any i,

$$\frac{1}{2} - \delta \leq \Pr[b_{i+1} = 0 \mid b_1, \ldots, b_i] \leq \frac{1}{2} + \delta$$

When $\gamma = 0$, this is exactly the uniform random source, and when $\gamma = \frac{1}{2}$, this is an arbitrary string. In this case, the concatenation of the random tapes $\overline{\rho}_1 \circ \overline{\rho}_2 \circ \ldots \circ \overline{\rho}_n$ is a γ-SV source. Note that in this case, there is no independence guarantee, as any two bits used by the distributed protocol can be dependent on each other.

THE ADVERSARY. We consider a strong adversarial model, namely the full-information model. A full-information t-adversary has unbounded computational power, and can "corrupt" up to t players in the beginning of the protocol execution. On corruption of a player P_i, the program of P_i, namely Π_i, is replaced by a program $\widetilde{\Pi}_i$ that the adversary chooses.

We consider three different *degrees* of corruption – namely, fail-stop, omission and Byzantine corruption – in the increasing order of severity. In the case of fail-stop corruption, the program $\widetilde{\Pi}_i$ is the same as Π_i, except that it can *halt*, in an arbitrary time during the execution of the protocol. We stress that the time when $\widetilde{\Pi}_i$ halts can (adversarially) depend on the particular execution of the protocol so far. Once $\widetilde{\Pi}_i$ halts, it does not send or receive any more messages. A slightly stronger corruption model is the omission model, which is the same as the fail-stop model except that $\widetilde{\Pi}_i$ can continue participating in the protocol even after omitting to send or receive an arbitrary subset of the messages. The most severe corruption model is the Byzantine model, where the program $\widetilde{\Pi}$ is completely arbitrary.

A corrupted program $\widetilde{\Pi}_i$ can observe all the communication in the network, namely the messages sent between *any two players*. In each round r, the corrupted program $\widetilde{\Pi}_i$ decides what messages to send to all the players in round r (resp. decides whether to omit messages in the case of an omission adversary, or halt in the case of a fail-stop adversary), *after* observing the messages sent by the honest players in all the previous rounds, inclduding round r. The ability to observe the messages sent in the same round before deciding what to send in that round is called *rushing*. This is to model the fact that although some form of synchrony is achievable in the network, perfect synchrony is almost impossible to achieve. We stress that a corrupted program can observe *all* the messages exchanged in the network, even the ones that honest players exchange amongst themselves. In other words, no communication is assumed to be secret (this models the "full-information" aspect of the adversary).

Thus, to recap, the full-information adversary is powerful in that it has (a) unbounded computing power, (b) the ability to observe all the messages sent in the network between any two players (that is, full-information), and (c) the ability to decide what to send in a given round r (resp. whether to halt or omit messages), based on the messages that the other players send during round the same round r (that is, rushing).

Formally, a full-information t-adversary $\mathcal{A}$ is a Turing machine with unbounded computational power that outputs a sequence of at most t indices $B = (b_1, \ldots, b_t)$, and t programs $\Pi_1^*, \ldots, \Pi_t^*$. Define $\widetilde{\Pi}_i = \Pi_i$ if $i \notin B$, and $\widetilde{\Pi}_i = \Pi_j^*$ where $b_j = i$. The execution of the protocol Π against the adversary $\mathcal{A}$ is define to be the execution of the protocol $\widetilde{\Pi} = (\widetilde{\Pi}_1, \ldots, \widetilde{\Pi}_n)$. To model the full-information aspect of the adversary, the execution of the corrupted and uncorrupted programs have different syntax. While an uncorrupted program executes just as before, a corrupted program $\widetilde{\Pi}_i$ acts as follows. In round r, $\widetilde{\Pi}_i$ computes the state at the beginning of round $r+1$, namely $\mathsf{state}_{i,r+1}$, and the messages to be sent in round r, namely $\{m_{i,j,r}\}_{j \in [1 \ldots n]}$, as a function of all the messages that have been sent in the network upto and including the messages in round $r+1$. Namely, it computes

$$(\mathsf{state}_{i,r+1}, m_{i \to 1}^{r+1}, \ldots, m_{i \to n}^{r+1}) \leftarrow \Pi_i(\mathsf{state}_{i,r}, \{m_{j \to k}^{r+1}\}_{j,k \in \mathsf{honest}(\mathcal{P},\mathcal{A})})$$

One of the drawbacks of the model is that the adversary has to choose which players to corrupt in the beginning of the protocol execution. This is called a static adversarial model. The stronger model of dynamic (or adaptive) adversaries, where the adversary can choose which player to corrupt adaptively during the execution of the protocol, is much harder to handle. In fact, see Section 2.1.1 for comments on this model.

COMPLEXITY MEASURES. The main measure of the complexity of a protocol Π is its round-complexity. The round-complexity of a protocol Π against an adversary $\mathcal{A}$, denoted by $\mathsf{rounds}(\Pi, \mathcal{A})$ is defined to be the total number of rounds in an execution of $\widetilde{\Pi}$ before *all* the honest players halt. The round-complexity of Π is the supremum of this quantity over all adversaries $\mathcal{A}$. That is, $\mathsf{rounds}(\Pi) = \max_{\mathcal{A}} \mathsf{rounds}(\Pi, \mathcal{A})$. In the case of a randomized protocol, $\mathsf{rounds}(\Pi)$ is a random variable and we will be concerned with the expected value of this quantity.

We are also concerned with the communication complexity of Π, denoted by $\mathsf{cc}(\Pi)$, which is the total number of bits communicated by all the *honest* players in an execution of Π against an adversary $\mathcal{A}$. Analogous to the round-complexity, in the case of a randomized protocol, we will talk about the expected communication complexity.

2.1.1 Remarks on the Model and Extensions

AUTHENTICATED CHANNELS. A physical assumption implicit in our model is the assumption of "authenticated pairwise channels". Namely, we assume that the adversary cannot modify or inject messages into the pairwise communication channel between two honest players P_i and P_j. Without this assumption, a single faulty player could impersonate as many players as he likes, and this would make achieving Byzantine agreement (and indeed, any form of reliable distributed computation) impossible. Authenticated channels thus appears to be an essential physical assumption necessary to achieve Byzantine agreement!

PARTIAL CONNECTIVITY. Our model assumes that there is a pairwise communication channel between every two players P_i and P_j which allows reliably sending a message from P_i to P_j in one time-step. The networks that arise in practice, however, are not fully connected and are sometimes quite sparse (think of the internet, or even the MIT intranet!) In such a case, we will run one of the many efficient reliable communication protocols which simulates a complete network over any network of "sufficient" connectivity. In particular, the results of [] show that we can simulate a complete network over any $t+1$-connected network of diameter δ, using a protocol that tolerates t faults and runs in $O(\delta)$ rounds.

ADAPTIVE ADVERSARIES. Our adversarial model assumes that the adversary chooses the set of players to corrupt in the beginning of the protocol execution.

The set of corrupted players is then fixed throughout the execution of the protocol. A stronger, adaptive (or dynamic) adversary is endowed with the capability to corrupt players during the execution of the protocol. Unfortunately, constructing an efficient Byzantine agreement protocol in the full-information model tolerating an adaptive adversary is impossible, as shown by Ben-Or and Bar-Joseph [BB98]. In particular, they show that any Byzantine agreement protocol that tolerates an adaptive adversary corrupting $O(n)$ players will have an expected round-complexity of $\widetilde{\Omega}(\sqrt{n})$. In contrast, the protocols we present have a round-complexity that is logarithmic (in n).

2.2 Reliable Broadcast and Byzantine Agreement

The problem of reliable broadcast is to simulate the functionality of a broadcast channel, a mechanism by which a possibly malicious player, called the sender, can transmit the same message to all the n players. A formal definition is as below.

Definition 2 (Reliable Broadcast). Let $\mathcal{M}$ be a finite message space. Let the input of a designated player $S \in \mathcal{P}$ be a message $m \in \mathcal{M}$, and the inputs of all the other players be a special symbol $\perp$. Then, a protocol Π is said to achieve reliable broadcast among the n players if:

1. **Termination:** All the honest players P_i terminate with probability 1, where the probability is over the coin-tosses of the honest players. On termination, player P_i writes a value $m_i \in \mathcal{M}$ on the output tape.

2. **Agreement:** Any two honest players have the same output. That is, for any two honest players P_i and P_j, $m_i = m_j$.

3. **Validity:** If the sender S is honest, then every honest player outputs m. That is, if S is honest, $m_i = m$ for every honest player P_i.

We want our protocols to be probabilistic in the "best possible way": that is, we require them to be always correct, and probably fast. That is, an unlucky sequence of coin-tosses may cause our protocol to to run longer, but when it halts both agreement and validity are guaranteed to hold.

Note that the above definition requires that the termination, agreement and validity conditions hold with probability 1 over the coin-tosses of the honest players. In this case, the running-time of the protocol is allowed to be a random variable, and the principal complexity measure of interest is the *expected* running time of the protocol.

The closely related problem of Byzantine Agreement [PSL80] is as defined below.

Definition 3 (Byzantine Agreement). Let $\mathcal{M}$ be a finite message space. Let the input of each player P_i be a value $m_i \in \mathcal{M}$. Then, a protocol Π is said to achieve Byzantine Agreement among the n players if:

PROTOCOL $\Pi_{\mathsf{bcast\text{-}from\text{-}BA}}(S, m)$: CODE FOR A PLAYER P_i

Input: m if P_i is the sender S, and $\perp$ otherwise.
Output: y_i.

Step 1. If $P_i = S$, then send m to all the players through the pairwise communication channels. If $P_i \neq S$, do nothing.

Step 2. Let z_i denote the value received from S in Step 1. Run the Byzantine agreement protocol BA with input z_i. The output y_i is the same as the output of the BA protocol.

Table 2.1: Reduction from Reliable Broadcast to Byzantine Agreement

1. **Termination:** All the honest players P_i terminate with probability 1, where the probability is over the coin-tosses of the honest players. On termination, player P_i writes a value $m_i \in \mathcal{M}$ on the output tape.

2. **Agreement:** Any two honest players have the same output. That is, for any two honest players P_i and P_j, $m_i = m_j$.

3. **Validity:** If all the non-faulty players have the same input, then That is, if there is an m such that for every honest player P_i, $m_i = m$, then for every honest player .

It is elementary to see that reliable broadcast is equivalent to the problem of achieving Byzantine Agreement. We will present a proof of this very simple fact, which will serve to introduce the reader to our method of presenting the protocols and structuring the proofs. Furthermore, some of the subtleties associated with the running-time of randomized protocols under parallel composition already show up in this reduction.

Proposition 1. If $t < \frac{n}{2}$, reliable broadcast and Byzantine agreement are equivalent.

Proof. We first show that if there is a Byzantine agreement protocol Π_{BA} with round-complexity r and communication complexity c, then there is a reliable broadcast protocol $\Pi_{\mathsf{bcast\text{-}from\text{-}BA}}$ with (expected) round-complexity $r + 1$ and (expected) communication complexity $c + O(n)$. The protocol $\Pi_{\mathsf{bcast\text{-}from\text{-}BA}}$ is given in Table 2.1.

Since Π_{BA} terminates within a finite amount of time, so does $\Pi_{\mathsf{bcast\text{-}from\text{-}BA}}$. To show validity of $\Pi_{\mathsf{bcast\text{-}from\text{-}BA}}$, suppose that the sender is honest and has input m. Then, all the honest players start the BA protocol Π_{BA} with input the same input m and, by the validity of Π_{BA}, all of them output m too. This proves validity

PROTOCOL $\Pi_{\text{BA-from-bcast}}$: CODE FOR A PLAYER P_i

Input: $m_i \in \mathcal{M}$.
Output: y_i.

Step 1. Run the reliable broadcast protocol $\Pi_{\text{BCast}}(P_i, m_i)$ with P_i as the sender and m_i as the input to the sender.

Step 2. P_i receives n outputs, one from each invocation of the reliable broadcast protocol $\Pi_{\text{BCast}}(P_j, m_j)$. Let $y_{i,j}$ be the output of P_i on the j^{th} invocation, namely $\Pi_{\text{BCast}}(P_j, m_j)$.

If there is a value y such that more than $n/2$ of the $y_{i,j}$'s are equal to y, then let the output $y_i = y$, else $y_i = \perp$.

Table 2.2: Reduction from Byzantine Agreement to Reliable Broadcast

for $\Pi_{\text{bcast-from-BA}}$. All the honest players agree on the output in $\Pi_{\text{bcast-from-BA}}$, simply because of the agreement property of Π_{BA}.

The claims about the round-complexity and communication-complexity of $\Pi_{\text{bcast-from-BA}}$ are easy to verify. The overhead in the complexity is caused by the initial round, where the sender sends his input to all the players, which takes 1 round a communication equivalent to $O(n)$ messages.

For the converse, we show that if there is a reliable broadcast protocol Π_{BCast} with round-complexity r and communication complexity c, then there is a BA protocol $\Pi_{\text{BA-from-bcast}}$ with round-complexity r and communication complexity $O(n \cdot c)$. The protocol $\Pi_{\text{BA-from-bcast}}$ is given in Table 2.2.

Since Π_{BCast} terminates within a finite amount of time, so does $\Pi_{\text{BA-from-bcast}}$. To show validity, suppose that all the honest players have the same input m. Consider any honest player P_i, and the n values $y_{i,j}$ that P_i receives as the output of the reliable broadcast sub-protocols, in the beginning of Step 2. If P_j is honest (and therefore has input m, by assumption), then by the validity of Π_{BCast}, $y_{i,j} = m$ too. Since there are more than $n/2$ honest players, a majority of the $y_{i,j}$'s will be m and thus, P_i will output m. Thus, all the honest players output m, which proves validity of $\Pi_{\text{BA-from-bcast}}$. We will now show agreement. For any two honest players P_i and P_j, the set of n values that P_i and P_j receive as outputs of the n reliable broadcast subprotocols is *identical*. Thus, P_i and P_j output the same value, since the output is the majority of these n values.

The overhead in the complexity is caused by executing n instances of Π_{BCast} in parallel, which costs n times the expected communication complexity of a single invocation of Π_{BCast}. The expected round-complexity of the protocol is the expected time it takes for n independent, parallel executions of Π_{BCast} to all finish. This could be more than the expected running-time of a single execution of Π_{BCast}, and in some cases, as large as $\log n$ times the expected running-time of

a single execution of Π_{BCast}. The upshot is that the expected round-complexity of $\Pi_{\mathsf{BA\text{-}from\text{-}bcast}}$ is tightly related to the expected round-complexity of n instances of Π_{BCast} running in parallel. ■

2.3 Probabilistic Lemmas

We use the following version of Chernoff Bound.

Proposition 2. Let $X_1, X_2, \ldots, X_n$ be independent random variables such that, for $1 \leq i \leq n$, $\Pr[X_i = 1] = p_i$, where $0 < p_i < 1$. Then, for $X = \sum_{i=1}^{n} X_i$, $\mu = \mathbb{E}[X] = \sum_{i=1}^{n} p_i$, and any $\delta > 0$,

$$\Pr[X > (1+\delta)\mu] < \left[\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right]^{\mu}$$

In particular, if $\delta = 2$, we get $\Pr[X > 3\mu] < e^{-\mu}$.

3

Classical Work on Reliable Broadcast

This chapter presents two classical results from the literature on reliable broadcast, and refines these results. Our protocols for reliable broadcast in Chapter 4 build on the material in this chapter.

The first is a definition of a weaker variant of reliable broadcast, called graded broadcast. Graded broadcast is a natural relaxation of reliable broadcast, and more importantly, it can be achieved by a very efficient deterministic protocol. This primitive was first defined by Feldman and Micali [FM97], and indeed they used it as a key ingredient in their reliable broadcast protocol in the private-channels model. Here, we use graded broadcast as a tool to achieve fully reliable broadcast, but in the full-information model. This material is presented in Section 3.1.

The second result is a classical connection between randomized algorithms for reliable broadcast (and also the related problem of Byzantine agreement), and protocols that achieve various forms of random selection. The connection between Byzantine agreement and random selection was first discovered by Ben-Or [Ben83] and Rabin [Rab83], and from then on, this has been the preferred route for constructing efficient randomized protocols for reliable broadcast. We follow this time-tested path, and refine it in order to achieve some additional properties of the resulting reliable broadcast protocol, such as sequential and parallel composition. This material is presented in Section 3.2.

3.1 A Weaker Variant of Reliable Broadcast

A useful intermediate step in achieving fully reliable broadcast is to define a notion of "semi-reliable" broadcast, and achieve a round-efficient implementation of such a notion. A grotesquely over-simplified specification of a semi-reliable broadcast channel is the following:

A semi-reliable broadcast sometimes loses messages, but never delivers different messages to two different players.

Feldman and Micali [FM88] defined the notion of graded broadcast which captures and strengthens the above over-simplified specification. We next present the definition of graded broadcast and a protocol achieving this definition; the material in this section is taken from the work of Feldman and Micali [FM88].

3.1.1 Graded Broadcast

Just as in the problem of reliable broadcast, there is a sender S with an input v that he wishes to broadcast to all the players. A protocol achieves graded broadcast if, at the end of the protocol, each player outputs a value v_i and a grade g_i such that:

1. Each grade g_i can take a value in $\{0, 1, 2\}$, 0 being the lowest grade and 2 the highest.
2. If the sender S is honest, all the players receive v (the sender's value) with a grade of 2.
3. Even if the sender is dishonest, the following properties hold: (a) the grades g_i and g_j of two honest players P_i and P_j cannot differ by more than one; and (b) if P_i receives a value v_i and P_j receives a value v_j with grades of at least one, then $v_i = v_j$.

The grade represents the degree of confidence that the recipient has on the fact that everyone else received the same value. If P_i outputs a value v_i with grade $g_i = 2$ (that is, the highest grade), then P_i "knows" that all the other honest players P_j received the same value, although with possibly different grades. That is, $v_j = v_i$, but g_j can be either 1 or 2. If P_i outputs a value v_i with grade $g_i = 1$, then it might be the case that another honest player P_j either recieves $v_j = v_i$ or $v_j = \perp$, but it is never the case that $v_j \notin \{v_i, \perp\}$. Finally, a grade $g_i = 0$ does not give any guarantees on the other players' values.

More formally,

Definition 4 (Graded Broadcast). Let $\mathcal{M}$ be a finite message space. Let the input of a designated player $S \in \mathcal{P}$ be a message $m \in \mathcal{M}$, and the inputs of all the other players be a special symbol $\perp$. Then, a protocol Π is said to achieve graded broadcast among the n players if each honest player P_i outputs a pair (m_i, g_i) with $m_i \in \mathcal{M} \cup \{\perp\}$ and $g_i \in \{0, 1, 2\}$, and the following conditions hold:

1. **Termination:** All the honest players P_i terminate with probability 1, where the probability is over the coin-tosses of the honest players.
2. **Graded Agreement:** For any two honest players P_i and P_j, $|g_i - g_j| \leq 1$. Furthermore, if $g_i > 0$ and $g_j > 0$, then $m_i = m_j$.
3. **Validity:** If the sender S is honest, then every honest player outputs $(m, 2)$. That is, if S is honest, $m_i = m$ and $g_i = 2$ for every honest player P_i.

The following lemma is proven in [FM97].

Lemma 1 (Feldman-Micali [FM97]). There exists a deterministic protocol Π_{gcast} among n players which achieves graded broadcast as long as $t < \frac{n}{3}$ players are corrupted by a Byzantine adversary. Π_{gcast} runs in 4 rounds, and has a communication complexity of $O(n^2)$ bits.

Proof. The protocol is in Table 3.1. It is clear from the description of the protocol that it terminates in 3 rounds, and communicates at most $O(n^2)$ messages. We will now show that it satisfies the validity and the graded agreement properties.

To show validity, suppose that the sender S is honest; we show that all honest players P_i output $(v, 2)$ where v is the input of the sender. Consider a particular honest player P_i. In Step 1, P_i receives v from the sender, and in Step 2, it sends $y_i = v$ to all the players. In turn, P_i receives the value v from all the honest players in Step 3, and thus sets $z_i = v$, and sends z_i to all the players. In Step 4, P_i again receives v from the all the $n - t$ honest players, and thus outputs $(v, 2)$.

We now show graded agreement, when the sender S is not necessarily honest. Suppose some honest player P_i outputs $(v, 2)$, for some value v. Working backwards, this means that P_i received the value v from at least $n - t$ players in Step 4. Since there are at most t dishonest players, at least $n - 2t$ honest players sent the value v in Step 3. Thus, any other honest player P_j receives the value v from at least $n - 2t$ players. Since $n - 2t \geq t + 1$, P_j will output either $(v, 1)$ or $(v, 2)$ (the latter in case he receives more than $2t$ copies of v).

We also show that if an honest player outputs $(v, 1)$, for some value v, then all the honest players will output (v', b) or $(\perp, 0)$, where $v' = v$ and $b \in \{1, 2\}$. In other words, v' is either v or $\perp$, and never a legal value different from v. To show this, suppose some honest player P_i outputs $(v, 1)$. Then, working backwards, this means that in Step 4, $\mathsf{Num}_i(v) \geq t + 1$. Since there are at most t dishonest players, this means that at least one honest player sent v in Step 3. P_i sends v in Step 3 only if he received at least $n - t$ copies of v in Step 2. Therefore, no honest player P_j could have sent a value $v' \neq v$ in Step 3. This is because every other honest player P_j receives v from at least $(n - t) - t = n - 2t$ players in Step 3. Therefore, for any other value $v' \neq v$, at most $2t < n - t$ players sent v' to P_j in Step 2. Now, this means that $\mathsf{Num}_j(v') \leq t$ in Step 4, since no honest player sends v' in Step 3 and there are at most t dishonest players. The only possibilities for P_j then are to output $(v, 2)$, $(v, 1)$ or $(\perp, 0)$. ■

3.2 Byzantine Agreement and Random Selection

Ben-Or [Ben83] and Rabin [Rab83] showed a reduction from achieving Byzantine agreement tolerating $t < n/3$ dishonest players to solving various random selection problems tolerating t dishonest players. If the random-selection protocol has a round-complexity of r, then the expected round-complexity of the resulting

PROTOCOL Π_{gcast}: CODE FOR PLAYER P_i

Input: A value v, if P_i is the sender S, and $\perp$ otherwise.
Output: A pair (v_i, g_i), where $g_i \in \{0, 1, 2\}$.

Step 1. If $P_i = S$, then send v to all the players through the pairwise communication channels. If $P_i \neq S$, do nothing.

Step 2. Let y_i denote the value received from S in Step 1. Send y_i to all the players.

Step 3. Let $z_{i,j}$ denote the value received from P_j in Step 2. Let $z_i = \mathsf{Threshold}_{n-t}(z_{i,1}, \ldots, z_{i,n})$. Send z_i to all the players.

Step 4. For every value w, let $\mathsf{Num}_i(w)$ be the number of players that sent w to P_i in Step 3.

- If there is some w such that $\mathsf{Num}_i(w) \geq 2t+1$, then output $(w, 2)$.
- Else, if there is some w such that $2t \geq \mathsf{Num}_i(w) \geq t+1$, then output $(w, 1)$.
- Else, output $(\perp, 0)$.

Table 3.1: The Graded Broadcast Protocol of Feldman and Micali

Byzantine agreement protocol is $O(r)$. Thus, to design efficient Byzantine agreement protocols, it suffices to design efficient random-selection protocols.

What is random-selection? The term is used to refer to a broad class of problems which, very informally, require the following:

> *Given a universe U, design a protocol Π_{randsel} at the end of which all players output an element $u \in U$ such that despite the faulty behavior of at most t players, u is guaranteed to be "sufficiently random".*

Two classical examples of random selection problems are collective coin-flipping and leader-election. In collective coin-flipping, the universe U contains two elements $\{0, 1\}$, and the n players are required to run a protocol Π_{coin} at the end of which all the players output a bit b that is relatively unbiased, even though a set of t players may be dishonest and might try to bias the output bit. More formally,

Definition 5 (Collective Coin-Flipping). A protocol Π_{coin} among n players is said to be an (ϵ, δ) collective coin-flipping protocol if, at the end of the protocol, each player P_i outputs a bit b_i such that the following hold:

1. **Agreement**: With probability at least ϵ, *all* the players output the same bit b. That is,
$$\Pr[\ \exists b \ \textit{such that}\ \forall i, b_i = b] \geq \epsilon$$

2. **Randomness**: Given that all the players output the same bit b, the bias of b is at most δ. That is,
$$\frac{1}{2} - \delta \leq \Pr[\ b = 0 \mid \exists b \ \textit{such that}\ \forall i, b_i = b] \leq \frac{1}{2} + \delta$$

In the problem of leader election, the requirement is that the players run a protocol and output the identity of some player (called the "leader"), with the guarantee that with high probability, the leader is an honest player. More formally,

Definition 6 (Leader Election). A protocol Π_{leader} among n players is said to be an (ϵ, δ) leader election protocol if, at the end of the protocol, each player P_i outputs a number $\ell_i \in [1 \ldots n]$ such that the following hold:

1. **Agreement**: With probability at least ϵ, *all* the players output the same number. That is,
$$\Pr[\ \exists \ell \in [1 \ldots n] \ \textit{such that}\ \forall i, \ell_i = \ell] \geq \epsilon$$

2. **Good Leader**: Given that all the players output the same number ℓ, the probability that P_ℓ is an honest player is at least δ. That is,
$$\Pr[\ \ell \in \mathsf{honest}(\mathcal{P}, \mathcal{A}) \mid \exists \ell \ \textit{such that}\ \forall i, \ell_i = \ell] \geq \delta$$

These two problems are related to each other. In particular, it is easy to see that given a protocol for (ϵ, δ) leader election, there is also a protocol for $(\epsilon\delta, \frac{1}{2}(1-\delta))$ collective coin-flipping. This is done by simply asking the elected leader to toss a random coin and send it to all the players. The better the parameter δ is, the larger the probability that an honest player will be elected as the leader, and thus the smaller is the bias of the resulting coin-flip.

It has been shown [Sak89] that both collective coin-flipping and leader election are achievable in the full-information model only if a majority of the players are honest, that is only if $t < n/2$. Under this condition, a large number of protocols with progressively better parameters and round-complexity have been designed for these problems [Sak89, ORV94]. The best protocols known are due to Feige [Fei99] and Russell and Zuckerman [RZ01], both of which have a round-complexity of $O(\log^* n)$ (where $\log^* n$ is the inverse of the Ackermann function). For a discussion of the range of parameters ϵ and δ achievable for collective coin-flipping and leader election, see [Fei99].

BEN-OR AND RABIN'S REDUCTION. Ben-Or and Rabin presented a reduction from Byzantine agreement to the problem of collective coin-flipping. In particular, they showed that if there is a protocol for (ϵ, δ) collective coin-flipping (resp.

(ϵ, δ) leader election) with some constant values of $\epsilon, \delta > 0$ that runs in r rounds tolerating $t < n/3$ dishonest players, then there is a protocol for Byzantine agreement that runs in $O(r)$ rounds tolerating t dishonest players, too. This reduction forms the basis of all the later works [FM88, CC85, DSS90, CR93] on constructing randomized Byzantine agreement protocols; all these works use the Ben-Or-Rabin reduction, and deal only with the problem of constructing collective coin-flipping (resp. leader election) protocols.

Here, we present another random selection problem, namely committee election, which is a generalization of the leader election problem. We then show how to transform any leader election generalized problem, called committee election, and also a more general reduction that preserves the complexity of parallel executions.

A NEW RANDOM SELECTION PROBLEM – COMMITTEE ELECTION. The problem of committee election was defined and used as a subroutine in the collective coin-flipping protocols of Feige [Fei99] and Russell and Zuckerman [RZ01]. The problem is to construct a protocol among n players, where all the players output a subset of players $S \subseteq \mathcal{P}$, such that the fraction of honest players in S is "roughly the same as" the fraction of honest players in the entire player-set $\mathcal{P}$.

We require an even weaker form of committee election for our purposes. Namely, we only require that the subset S contains at least *one* honest player. More formally,

Definition 7 (Committee Election). A protocol Π_{comm} among n players is said to be an (c, ϵ, δ) *committee election* protocol if, at the end of the protocol, each player P_i outputs a subset $S_i \subseteq \mathcal{P}$ such that the following hold:

1. **Agreement**: With probability at least ϵ, *all* the players output the same set S. That is,
$$\Pr[\, \exists S \subseteq \mathcal{P} \ \textit{such that} \ \forall i, S_i = S] \geq \epsilon$$
2. **Good Committee**: Given that all the players output the same set S, the probability that S contains at least one honest player is at least δ. That is,
$$\Pr[\, S \cap \mathsf{honest}(\mathcal{P}, \mathcal{A}) \neq \emptyset \mid \exists S \ \textit{such that} \ \forall i, S_i = S] \geq \delta$$
3. **Size of the Committee**: Given that all the players output the same set S, $|S| \leq c$.

It is trivial to deterministically elect a committee with $t + 1$ players that contains at least one good player. Simply output the identities of the first $t + 1$ players, for example. The non-trivial problem here is to elect a much smaller committee. In such a case, it is easy to see that a deterministic algorithm will not suffice.

A REDUCTION FROM BYZANTINE AGREEMENT TO COMMITTEE ELECTION. Given any committee-election protocol that outputs a committee S, we can

construct a randomized Byzantine agreement protocol that runs in an (expected) $O(|S|)$ rounds. If the committee-election protocol tolerates t dishonest players, then the resulting Byzantine agreement protocol tolerates $\min(t, n/3)$ dishonest players. We state this lemma formally below.

Theorem 1 (follows from [BG89b]). There is a reduction from binary Byzantine agreement to committee election. If $\Pi_{\mathsf{comm\text{-}elect}}$ is a (c, ϵ, δ) committee election protocol tolerating t faults, then $\Pi_{\mathsf{BAreduction}}$ is a Byzantine agreemnet protocol that tolerates t faults, and runs in expected $O(c/\epsilon\delta)$ rounds.

4

New Byzantine Agreement Protocols

This chapter contains the main contribution of our work, namely the construction of two Byzantine agreement protocols in the full-information model, with an expected round-complexity of $O(\log n)$.

We proceed by first constructing committee-election protocols, and then using the reduction from Byzantine agreement to committee election (Chapter 3, Theorem 1) to derive the Byzantine agreement protocols. The construction of the two committee-election protocols build heavily on the protocols of Feige [Fei99] and Russell and Zuckerman [RZ01], respectively. The protocols of [Fei99] and [RZ01] work in the full-information model with *built-in broadcast channels*, which is exactly the assumption that what we are trying to avoid. Thus, the main technical effort is to design committee-election protocols (following [Fei99] and [RZ01]) in the full-information model *without assuming built-in broadcast channels*.

ORGANIZATION OF THIS CHAPTER. We first present the protocols of [Fei99] and [RZ01], using built-in broadcast channels in Section 4.1. Section 4.2 describes the construction of our committee-election protocols without using built-in broadcast channels. As a consequence, we obtain two Byzantine agreement protocols. Section 4.3 concludes with a discussion of some natural ways to improve the round-complexity of these protocols, and roadblocks to achieving this goal.

4.1 Committee Election Protocols

In this section, we describe the committee-election protocols of Feige [Fei99] and Russell and Zuckerman [RZ01]. Although it suffices (for the purposes of constructing Byzantine agreement, using the reduction in Theorem 1) to elect a

PROTOCOL Π_{Feige}: CODE FOR A PLAYER P_i

Input: A real number $\epsilon > 0$.
Output: The identities of a set of players $S = \{P_1, \ldots, P_k\}$.

Step 1. Define the constant $a \stackrel{\Delta}{=} \frac{2}{\epsilon^2}$. Choose a random number $B_i \in [1 \ldots \frac{n}{a \log n}]$ and use the reliable broadcast channel to *broadcast* B_i to all the players. Let $\mathsf{bin}(P_i) = B_i$.

Step 2. Let $S_B = \{P_i : \mathsf{bin}(P_i) = B\}$. Output the set S_B that has the minimal size. That is, output S_B such that $|S_B|$ is the smallest.

Table 4.1: Feige's Committee Election Protocol with Built-in Broadcast Channels.

committee that has at least one honest player, these protocols do much more: the protocols elect a committee which contains roughly the same *fraction* of dishonest players as in the entire player-set $\mathcal{P}$. The protocols are described in Sections 4.1.1 and 4.1.2.

4.1.1 Feige's Committee Election Protocol

In 1999, Feige proposed an exceedingly simple and elegant protocol – called the "lightest bin" protocol – to select from among n players, a set S of $O(\log n)$ players such that the fraction of good players in S is approximately the same as the fraction of good players in the entire player-set $\mathcal{P}$. The basic idea of Feige's protocol is *selection by elimination*: assume that there are $k \ll n$ "bins" and that each player has a "ball" in his hand. Each player picks a random bin and throws the ball into the bin. When all the n players are done, they pick the *lightest bin*, namely the bin that contains the fewest balls; the elected committee consists of the players in the lightest bin. Clearly, the lightest bin has at most $\frac{n}{k}$ players, and thus if k were sufficiently large, the selected committee is small. On the other hand, since the honest players choose at random, they are almost evenly distributed among all the bins, assuming that k is not too large (informally, this is true because of an analogy with the coupon-collector problem. See the analysis in Lemma 2 for details). Thus, with high probability (over the randomness of the honest players), each bin contains a large number of honest players. Now, if the adversary wants to include too many dishonest players in a bin, he will end up making the bin too heavy which, in particular, means that the bin will not be chosen.

A formal description and claim about Feige's protocol appears in Table 4.1 and Lemma 2.

Lemma 2. Let β denote the fraction of dishonest players in the entire player-set $\mathcal{P}$. For every constant $\epsilon > 0$, at the end of the protocol Π_{Feige}, all the players output

(the same) committee $S \subseteq \mathcal{P}$ such that $|S| \leq \frac{2 \log n}{\epsilon^2}$, and with probability at least $1 - \frac{1}{n}$, the fraction of dishonest players in the elected committee is at most $\beta + \epsilon$.

Π_{Feige} runs in 1 round, and the communication- complexity is $\tilde{O}(n^2)$ bits (where each bit that is broadcast counts as $O(n)$ bits).

Proof: Since the protocol uses a built-in broadcast channel, all the players output the same set S. We will now show that with probability at least $1 - \frac{1}{n}$, the set S contains at least $(1 - \beta - \epsilon)|S|$ honest players.

First, observe that $|S| \leq a \log n$. This is because there are n players, and $\frac{n}{a \log n}$ bins, and by the pigeonhole principle, at least one of the bins contains at most $a \log n$ players. In particular, the lightest bin contains at most $a \log n$ players.

Secondly, we claim that with probability at least $1 - \frac{1}{n}$, all the bins contain at least $(1 - \beta - \epsilon) a \log n$ honest players. Define the indicator random variables $X_{i,b}$ for $1 \leq i \leq n$ and $1 \leq b \leq \frac{n}{a \log n}$ as follows.

$$X_{i,b} = \begin{cases} 1 & \text{if player } P_i \text{ chooses bin } b \\ 0 & \text{otherwise} \end{cases}$$

Since the honest players choose their bin at random, for every honest player P_i and every bin b, the expectation $\mathbb{E}[X_{i,b}] = \frac{a \log n}{n}$.

Denote the number of honest players in a particular bin b by X_b. Then, X_b is simply the sum of all random variables $X_{i,b}$ for all honest players P_i. That is,

$$X_b = \sum_{i:P_i \text{is honest}} X_{i,b}$$

By linearity of expectation, the expected number of honest players in a bin b is

$$\mathbb{E}[X_b] = \sum_{i:P_i \text{is honest}} \mathbb{E}[X_{i,b}] = \sum_{i:P_i \text{is honest}} \frac{a \log n}{n} = (1 - \beta) a \log n$$

Since the random variables $X_{i,b}$ are independent, we can bound the probability that X_b is much smaller than the expectation, using Chernoff bound,

$$\Pr[X_b < (1-\beta-\epsilon) a \log n] \leq \Pr[|X_b - \mathbb{E}[X_b]| \leq \frac{\epsilon}{1-\beta} \mathbb{E}[X_b]] \leq 2 \cdot e^{-\epsilon^2 a \log n/(1-\beta)}$$

By a union bound, the probability that some bin b contains less than $(1 - \beta - \epsilon) a \log n$ honest players is at most

$$\Pr[\exists b \text{ such that } X_b < (1 - \beta - \epsilon) a \log n] \leq \frac{n}{a \log n} \frac{2}{n^2} < \frac{1}{n}$$

These two observations together mean that the smallest bin contains at most $a \log n - (1 - \beta - \epsilon) a \log n \leq (\beta + \epsilon) a \log n$ dishonest players, with high probability. Thus, the fraction of dishonest players in the smallest bin is at most $\beta + \epsilon$.

The claims about the round-complexity and communication-complexity of Π_{Feige} are easy to verify. ■

REMARK. Feige uses this committee-election protocol for leader-election and collective coin-flipping. This is done by recursively using the committee-selection protocol to select smaller and smaller committees and finally a single player, who is then chosen as the leader. As for coin-flipping, the chosen leader will then flip a coin and send it to all the players. However, we will not use this recursive procedure for our purposes, and instead use only the committee-selection protocol as above.

4.1.2 Russell and Zuckerman's Committee Election Protocol

In 1998, building upon the work of Ostrovsky, Rajagopalan and Vazirani [ORV94] and Zuckerman [Zuc97], Russell and Zuckerman [RZ01] designed an elegant protocol for committee election. The basic idea of the Russell and Zuckerman protocol (which we call the RZ protocol) is similar to Feige's protocol, namely "selection by elimination". As in the case of Feige's protocol, the RZ protocol also uses a *reliable broadcast channel as a built-in primitive*. However, in contrast to Feige's protocol which elects an arbitrary set of $O(\log n)$ players, the RZ protocol deterministically fixes a polynomial (in n) number of committees before the execution of the protocol, and uses the protocol to *elect one of these* $\mathsf{poly}(n)$ *committees*.

The RZ protocol works as follows: first, the players deterministically compute a collection of $m = \mathsf{poly}(n)$ "prospective committees" C_i, where each of these committees has $O(\log n)$ players. Assume that the entire player-set $\mathcal{P}$ contains a β fraction of dishonest players. Then, the set of committees $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$ has the following property:

> The number of committees C_i that have "significantly more" than a β fraction of dishonest players is "very small".

Slightly more precisely, fixing an $\epsilon > 0$ and a set B of at most βn dishonest players, call a committee C bad if the fraction of dishonest players in C is more than $\beta + \epsilon$. Let

$$\mathsf{Bad}_\epsilon(B, \mathcal{C}) = \{C_i \in \mathcal{C} : \text{ the fraction of dishonest players in } C_i \text{ is more than } \beta + \epsilon\}$$

We require that for any constant $\epsilon > 0$, there is a deterministic algorithm $\mathsf{Prospective}(1^n, \epsilon)$ which outputs a collection $\mathcal{C}$ of $m = \mathsf{poly}(n)$ committees such that for any set B of at most βn dishonest players, $|\mathsf{Bad}_\epsilon(B, \mathcal{C})| \leq 3n$.

Informally, one could think of the construction of the collection of prospective committees as achieving the following goal.

> Even though the fraction of dishonest *players* in the entire player-set $\mathcal{P}$ is β, the fraction of bad *committees* is only $\frac{3n}{m}$, which is much smaller than β.

The existence of such a collection of committees (for appropriate choices of parameters) follows by a simple probabilistic argument, which is included for completeness in Appendix A. The probabilistic argument, in addition, gives us an exponential-time deterministic algorithm for constructing the collection of prospective committees. We state this result below.

Lemma 3. There is a deterministic algorithm Prospective$(1^n, \epsilon)$ which outputs a collection $\mathcal{C}$ of $m = O(n^2)$ committees C_i, where each C_i has $O(\log n/\epsilon^2)$ players such that: for every $\epsilon > 0$ and for any set B of at most βn dishonest players, $|\mathsf{Bad}_\epsilon(B, \mathcal{C})| \leq 3n$.

Moreover, Nisan [Nis96] and Zuckerman [Zuc97] showed an explicit and polynomial (in n and $\log(1/\epsilon)$) algorithm Prospective$(1^n, \epsilon)$, using explicit constructions of extractors. In the rest of our exposition, we use the above non-explicit construction for the sake of simplicity.

Given the collection of prospective committees, the protocol proceeds as follows. In the first and the only communication-round of the protocol, the players elect one of these m prospective committees. The following paragraph contain an informal (and somewhat incorrect!) description of how a committee is elected; this paragraph is only meant to give a rough intuition of the process. We refine these ideas and make them precise in the paragraph after.

Very roughly, committee-election proceeds by elimination: each player picks a $\frac{1}{2n}$ fraction of all the committees uniformly at random, and broadcasts this choice to all the players. The semantics of a player P_i picking a committee C is that P_i chooses to *eliminate C from consideration.*

1. On the one hand, since the number of bad committees is very small, all of them will be eliminated by some honest player or the other (with high probability over their coin-tosses).

2. On the other hand, the total number of committees eliminated by all the players is at most $\frac{m}{2n} \cdot n < m$, and therefore at least one committee is not eliminated by any of the players.

Thus, all the committees that are not eliminated are good committees (namely, they have roughly a β fraction of bad players), and there is at least one committee that is not eliminated. The lexicographically smallest such committee is the elected committee.

The strategy outlined above does not work as such.[1] The modification that needs to be made is in the way the players pick the committees to be eliminated. Russell and Zuckerman use the notion of hitting sets for combinatorial rectangles to design a process of elimination. An (n, a, m)-hitting set (for our purposes) is a function $h : \mathcal{C} \rightarrow \{1 \ldots a\}^n$ (for some $a > 0$) such that for every $\{R_1, R_2, \ldots, R_n\} \subseteq 1 \ldots a$ such that $|R_i| = a - 1$, there is a $C \in \mathcal{C}$ such

[1] What fails in the informal description is property (1) above. Namely, it is not true that the uniformly random selection eliminates all bad committees with high probability.

PROTOCOL Π_{RZ}: CODE FOR A PLAYER P_i

Input: A real number $\epsilon > 0$.
Output: The identities of a set of players $S \subseteq \mathcal{P}$.

Step 1. Run the algorithm Prospective$(1^n, \epsilon)$ that outputs a collection of committees $\mathcal{C}$. Run HittingSet(1^n) that outputs a function $h : \mathcal{C} \rightarrow \{1 \ldots a\}^n$.

Step 2. Choose $r_j \in [a]$, and *broadcast* r_j to all the players.

Step 3. Let the bit-vector $\vec{e} = \langle e_1, e_2, \ldots, e_m \rangle$ where

$$e_i = \begin{cases} 1 & \text{if for some } 1 \leq j \leq n,\ h(C)|_j = r_j \\ 0 & \text{otherwise} \end{cases}$$

Output the lexicographically smallest committee C_i such that $e_i = 0$.

Table 4.2: The RZ Committee Election Protocol with Built-in Broadcast Channels.

that $h(C) \in R_1 \times \ldots \times R_n$. The existence of such a function can be easily proven via a probabilistic argument, which is included for completeness in Appendix A. The lemma is stated below.

Lemma 4. There is a deterministic algorithm HittingSet(1^n) which outputs a function $h : \{1, \ldots, m\} \rightarrow \{1, \ldots, a\}^n$ such that h is an (n, a, m)-hitting set, where $m = n^2$ and $a = \frac{n}{4 \log n}$.

Moreover, Linial, Luby, Saks and Zuckerman [LLSZ97] give explicit constructions of an (n, a, m)-hitting set for any n, a and $m = \mathsf{poly}(a \log n)$. In the rest of our exposition, we use the above non-explicit construction.

Given a collection of committees $\mathcal{C}$ and a hitting set $h : \mathcal{C} \rightarrow [a]^n$, the protocol is simple: each player P_i chooses a random number $r_i \in \{1, \ldots, a\}$ and broadcasts r_i to all the players. Every committee C such that $h(C)|_i = r_i$ for some i are eliminated. In other words, the committees that are not eliminated are exactly those C for which $h(C) \in ([a] - \{r_1\}) \times \ldots \times ([a] - \{r_n\})$. By the property of hitting sets, there is at least one such committee. On the other hand, since there are only a few bad committees, all of them are eliminated. A formal description of the protocol Π_{RZ} is in Table 4.2.

Lemma 5. Let β denote the fraction of dishonest players in the entire player-set $\mathcal{P}$. For every constant $\epsilon > 0$, at the end of the protocol Π_{RZ}, all the players output (the same) committee $S \subseteq \mathcal{P}$ such that $|S| = O(\frac{\log n}{\epsilon^2})$, and with probability at least $1 - \frac{3}{n}$, the fraction of dishonest players in the elected committee is at most $\beta + \epsilon$.

Π_{RZ} runs in 1 round, and the communication- complexity is $\tilde{O}(n^2)$ bits (where each bit that is broadcast counts as $O(n)$ bits).

Proof: Say that a committee C_i is eliminated if $e_i = 1$. Note that there is a "rectangle" $R_1 \times R_2 \times \ldots \times R_n$ where each $R_i \subseteq \{1 \ldots a\}$ is of size $a - 1$ such that for every committee C that is not eliminated, $h(C) \in R_1 \times R_2 \times \ldots \times R_n$. Since h is a hitting set, there is at least one such C. This committee C is not eliminated.

Also, out of the m committees, at most $3n$ are bad. The probability that a bad committee C_i is not eliminated is at most $(1 - \frac{1}{a})^{n-t} \leq (1 - \frac{2\log n}{n})^{(1-\beta)n} \leq \frac{1}{n^2}$. Thus, the probability that there exists a bad committee that is not eliminated is at most (by a union bound), $3n \cdot \frac{1}{n^2} \leq \frac{3}{n}$. Thus, with probability at least $1 - \frac{3}{n}$, Π_{RZ} outputs a good committee. ∎

4.2 Byzantine Agreement Protocols

In this section, we present two Byzantine agreement protocols – the first builds on Feige's committee-election protocol, and the second one is based on the RZ protocol. The first protocol achieves a fault-tolerance of $t < (1/4 - \epsilon)n$ (for any constant $\epsilon > 0$), and runs in an expected $O(\frac{\log n}{\epsilon^2})$ rounds. The second protocol achieves an almost optimal fault-tolerance of $t < (1/3 - \epsilon)n$, and has the same expected round-complexity as the first protocol, namely $O(\frac{\log n}{\epsilon^2})$ rounds. Although the second protocol is superior (in terms of fault-tolerance), we present both the protocols here as we believe that they use techniques that are independently interesting.

4.2.1 Byzantine Agreement Protocol I

This section is devoted to proving the following theorem.

Theorem 2. For any constant $\epsilon > 0$, there exists an explicit protocol BA_ϵ that reaches Byzantine Agreement in a synchronous full-information network tolerating $t < (\frac{1}{4} - \epsilon)n$ non-adaptive Byzantine faults, and runs for expected $O(\frac{\log n}{\epsilon^2})$ rounds.

We first describe an outline of the protocol. The formal description of the protocol and the proof follow.

Overview of the Byzantine Agreement Protocol

In light of the reduction from Byzantine Agreement to Committee Election (Chapter 3, Theorem 1) it suffices to construct a protocol that elects a committee of size $O(\frac{\log n}{\epsilon^2})$ tolerating $t < (1/4 - \epsilon)n$ Byzantine faults, *without a built-in reliable broadcast channel.* We will call this protocol $\Pi_{\mathsf{Elect\text{-}BPV}}$ – the protocol first appeared in a joint work of the author with Michael Ben-Or and Elan Pavlov [BPV06].

Our starting point is the committee-election protocol of Feige, namely the protocol Π_{Feige} in Table 4.1. However, we cannot use Feige's protocol as such, because it works under the assumption of a *built-in broadcast channel*, which we of course cannot assume. Thus, the main technical work in our protocol is in (significantly) modifying Π_{Feige} so that it performs correctly without a broadcast channel.

Recall that Feige's committee-election protocol involves each player choosing a random bin among a set of $O(n/\log n)$ bins. They announce this choice to all the players using the built-in broadcast channel. The bin that the smallest number of players chose is the selected bin. The committee consists of all the players that chose the selected bin.

For the rest of this section, assume that all the bins in an execution of Feige's protocol contain at least $\frac{3}{4}a\log n$ honest players. If this event does not happen, then we cannot guarantee that the elected committee is a good committee, or even that all the players agree on a single committee. Fortunately, by Lemma 2, this event happens with probability $1-\frac{1}{n}$ over the coin-tosses of the honest players.

In the absence of a built-in broadcast channel, our first idea is to let the players announce their (random) choices of the bins using a graded broadcast protocol. In particular, in the first round of our protocol, each player P_i chooses a bin B_i at random and runs a graded broadcast protocol with B_i as the input. This ensures that a dishonest player cannot "convince" two different honest players that he chose different bins. In other words, for every dishonest player P^*, there is a *unique* bin B^* such that each honest player P_i "sees" that P^* is either in bin B^*, or not in any of the bins at all. If each honest player P_i now chose the smallest bin, we have the guarantee that each such bin will contain at most a $1/3$rd fraction of dishonest players. However, different honest players might choose different bins as the lightest; in other words, there is no agreement among the honest players as to which bin is the lightest. Moreover, the honest players do not even agree on the set of players that belong to a particular bin. Despite these shortcomings, we retain the idea of asking all the players to announce their choices of a bin using a graded broadcast protocol.

Each player P_i receives the output of n graded broadcasts. P_i's output on P_j's graded broadcast is a pair $(B_{i,j}, g_{i,j})$, where $B_{i,j}$ is the identity of a bin (that is, a number between 1 and $n/a\log n$) and $g_{i,j}$ is the grade that P_i assigns to P_j's graded broadcast. P_i now classifies the graded broadcasts he receives into the following two "views". He defines the set $\mathsf{View}_i(B)$, where a player P_j is in $\mathsf{View}_i(B)$ if P_i heard P_j's graded broadcast that "I chose bin B" with *a grade of* 2. More formally,

$$\forall B, \mathsf{View}_i(B) = \{P_j \mid B_{i,j} = B \text{ and } g_{i,j} = 2\}$$

P_i also defines the set $\overline{\mathsf{View}}_i(B)$ (the "closure" of $\mathsf{View}_i(B)$), where a player P_j is in $\overline{\mathsf{View}}_i(B)$ if P_i heard P_j's graded broadcast that "I chose bin B" with *a grade of at least* 1. More formally,

$$\forall B, \mathsf{View}_i(B) = \{P_j \mid B_{i,j} = B \text{ and } g_{i,j} \geq 1\}$$

It is obvious from the definition above that if P_i is an honest player and some (not necessarily honest) player $P_j \in \mathsf{View}_i(B)$, then P_j is also in $\overline{\mathsf{View}}_i(B)$. That is,

$$\forall B, \forall \text{ honest } P_i, \mathsf{View}_i(B) \subseteq \overline{\mathsf{View}}_i(B)$$

In fact, a stronger statement is true: namely, if a player $P_j \in \mathsf{View}_i(B)$ for some honest player P_i, then P_j is also in $\overline{\mathsf{View}}_{i'}(B)$ *for all honest players* $P_{i'}$. This is true because of the "graded agreement" property of the graded broadcast – if P_i receives P_j's message (namely, "I chose bin B") with a grade of 2, then $P_{i'}$ receives it with a grade of at least 1. Formally,

$$\forall B, \forall \text{ honest } P_i, P_{i'}, \mathsf{View}_i(B) \subseteq \overline{\mathsf{View}}_{i'}(B)$$

This observation can be interpreted as saying that (although the players may not agree on the content of each bin) there is some form of "graded agreement" among the players.

The second idea is to run a protocol among the players in a bin to decide if the bin has too many players, and therefore should be eliminated from consideration. That is, in some sense, the players in a bin decide to "self-destruct" if they detect that the bin contains too many players. More concretely, the idea is to mark a bin B as "disqualified" if there is some subset of at least $\frac{3}{4}a \log n$ players in B who decide that B has too many players. Consider the following two cases.

- B has at most $a \log n$ players in total: all the honest players in B will decide *not* to eliminate B. Since the number of dishonest players in B is less than $\frac{1}{4}a \log n$, they cannot reverse the decision of the honest players. Thus, B will not be disqualified.

- B has more than $a \log n$ players in total: in this case, the honest players in B will decide to disqualify B.

A fatal flaw with this idea is that the set of players in a bin is not even well-defined at this point, since different players have different views about the composition of a bin, namely, the set of players that chose the bin. Nevertheless, we retain this idea too (in spirit).

One way to fix the flaw with this idea is to ask each subset S of $\frac{3}{4}a \log n$ players [2] to decide the fate of each bin, namely whether each bin survives or is disqualified. This decision is reached by letting the players in S run a Byzantine agreement protocol to decide on a common "view of bin B"; if the bin B is deemed too large, it is disqualified and otherwise, it survives. In particular, the players in each subset S (of size $3/4a \log n$) run a Byzantine agreement protocol, at the end of which each player $P_i \in S$ computes two sets $\mathsf{SetView}_{i,S}(B)$ and

[2] There are $\binom{n}{3/4a \log n}$ such subsets.

$\overline{\mathsf{SetView}}_{i,S}(B)$. $\mathsf{SetView}_{i,S}(B)$ (resp. $\overline{\mathsf{SetView}}_{i,S}(B)$) contains a player P if P is in $\mathsf{View}_j(B)$ (resp. $\overline{\mathsf{View}}_j(B)$) for *all the players* $P_j \in S$. More formally,

$$\mathsf{SetView}_{i,S}(B) := \bigcap_{P_j \in S} \mathsf{View}_j(B) \text{ and } \overline{\mathsf{SetView}}_{i,S}(B) := \bigcap_{P_j \in S} \overline{\mathsf{View}}_j(B)$$

These sets can be computed by asking each player in P_i to broadcast $\mathsf{View}_i(B)$ and $\overline{\mathsf{View}}_i(B)$, using a reliable broadcast protocol *among the players in* S. Since the set S is of size $O(a \log n)$, we can use a deterministic reliable broadcast protocol that runs in $O(a \log n)$ rounds (for example, the protocol of [GM98]). Of course, note that if the set S contains a large fraction of dishonest players (which could happen for many sets S), then the deterministic reliable broadcast protocol will fail. We show later that this is not a problem – in fact, all we require is the existence of *some subset* S that has a large fraction of honest players (and also satisfies certain requirements, which we specify shortly).

Now, the players in S decide whether to disqualify the bin B (based on the fact that it has too many dishonest players). Player P_i's estimation of the size of B can be based on either $\mathsf{SetView}_{i,S}(B)$ or $\overline{\mathsf{SetView}}_{i,S}(B)$. It turns out that the right decision is the conservative one: that is, decide to disqualify B if the larger set, namely $\overline{\mathsf{SetView}}_{i,S}(B)$ is too large.

More precisely, each player P_i sets the bit $\mathsf{disq}_{i,S}(B) = 1$ (meaning that player P_i decides to disqualify bin B, as part of the verdict of the subset S) if $\overline{\mathsf{SetView}}_{i,S}(B)$ *has more than* $a \log n$ *players*. That is, P_i computes

$$\mathsf{disq}_{i,S}(B) = \begin{cases} 1 & \text{if } |\overline{\mathsf{SetView}}_{i,S}(B)| > a \log n \\ 0 & \text{otherwise} \end{cases}$$

P_i also defines $\mathsf{composition}_{i,S}(B) = \mathsf{SetView}_{i,S}(B)$.

Note that the definition of $\mathsf{disq}_{i,S}(B)$ refers to the set $\overline{\mathsf{SetView}}_{i,S}(B)$ and the definition of $\mathsf{composition}_{i,S}(B)$ refers to the set $\mathsf{SetView}_{i,S}(B)$. Very roughly speaking, the rationale for this asymmetry is to ensure the following: if $\mathsf{SetView}_{i,S}(B)$ is too large, then so is $\overline{\mathsf{SetView}}_{i,S}(B)$, and if that is the case, B will be disqualified. This ensures that the composition of the bin B, namely $\mathsf{SetView}_{i,S}(B)$ is never too large for a bin B that is not disqualified .

The players in each of the sets S finally send a message containing both $\mathsf{disq}_{i,S}(B)$ and $\mathsf{composition}_{i,S}(B)$ to all the players. Each player P_j receives messages from all the subsets S about each bin B. P_j decides that a bin B is disqualified if (a) *some subset* S *of players* tells P_j to disqualify B, and (b) P_i sees that all the players in S belong to $\mathsf{View}_j(B)$. Finally, the players also consolidate the opinions of all the sets and decide on the composition of the bin.

The above description, although incomplete, contains the main ideas of the protocol. For a formal and complete protocol, see Table 4.3.

PROTOCOL $\Pi_{\text{Elect-BPV}}$: PSEUDOCODE

Input of Player P_i: $\perp$
Output of Player P_i: A committee $C \subseteq \{P_1, \ldots, P_n\}$.

Step 1 **(Each player P_i)** Let $a \in \mathbb{N}$ be a parameter of the system. Choose a random bin $B_i \in [1 \ldots \frac{n}{a \log n}]$. Do a graded broadcast of B_i to all the players.

Denote what is received by player P_i sent to him The output of P_i from all the graded broadcasts is a list of n pairs $[(B_{i,1}, g_{i,1}), \ldots, (B_{i,n}, g_{i,n})]$, where $(B_{i,j}, g_{i,j})$ is what P_i received as a results of P_j's graded broadcast.

Step 2 **(Each player P_i, locally)** Construct a local view of the composition of the committees. P_i computes two sets $\mathsf{View}_i(B)$ and $\overline{\mathsf{View}}_i(B)$ for every bin B.

$$\mathsf{View}_i(B) = \{P_j \mid B_{i,j} = B \text{ and } g_{i,j} = 2\}$$

$$\overline{\mathsf{View}}_i(B) = \{P_j \mid B_{i,j} = B \text{ and } g_{i,j} \geq 1\}$$

Step 3 **(Each subset $S \subseteq [n]$ where $|S| = a \log n$)** All the players in S run a deterministic BA protocol in S to compute the following quantities for every bin B.

$$\mathsf{SetView}_{i,S}(B) := \bigcap_{P_j \in S} \mathsf{View}_j(B) \text{ and } \overline{\mathsf{SetView}}_{i,S}(B) := \bigcap_{P_j \in S} \overline{\mathsf{View}}_j(B)$$

Each player $P_i \in S$ computes a local binary variable $\mathsf{disq}_{i,S}(B)$ as follows: $\mathsf{disq}_{i,S}(B) = 1$ if $|\overline{\mathsf{SetView}}_{i,S}(B)| > a \log n$ and 0 otherwise.

P_i sends the tuple $[S, B, \mathsf{disq}_{i,S}(B), \mathsf{SetView}_{i,S}(B)]$ to all the players.

Step 4 **(Each player P_i, locally)** P_i collects the messages received from all the players (Note that P_i could have received many messages from a single player P, since P is part of many subsets S).

If P_i received messages of the form $(S, B, 1, *)$ from some set of players S' such that (a) $|S'| \geq \frac{1}{2} a \log n$, and (b) $S' \subseteq S \subseteq \mathsf{View}_i(B)$, then set $\mathsf{disq}_i(B) = 1$.

Step 5 **(Each player P_i, locally)** For every bin B and each set $S \subseteq \mathsf{View}_i(B)$, P_i does the following: If P_i receives messages of the form $(S, B, 0, D)$ from some set of players $S' \subseteq S$ of size at least $\frac{1}{2} a \log n$, then let $\mathsf{composition}_{i,S}(B) = D$, else set $\mathsf{composition}_{i,S}(B) = \perp$.

P_i defines the final composition of the bin B, denoted $\mathsf{composition}_i(B)$ as the largest $\mathsf{composition}_{i,S}(B)$ among all subsets S.

Step 6 **(Each player P_i, locally)** P_i outputs $C := \mathsf{composition}_i(B)$ as the chosen committee, where B is the lexicographically smallest bin such that $\mathsf{disq}_i(B) = 0$.

Table 4.3: Feige's Committee-Election Protocol without Broadcast.

Formal Proof of Theorem 2

The formal specification of the protocol is in Table 4.3. To prove Theorem 2, it suffices to prove the following lemma, which states that $\Pi_{\text{Elect-BPV}}$ is a committee-election protocol that runs in $O(\log n/\epsilon^2)$ rounds, elects a committee of size $O(\log n/\epsilon^2)$, and has $o(1)$ error probabilities (namely, the probability that the players do not agree on the elected committee, as well as the probability that the elected committee is bad). Then, by Theorem 1, we get a Byzantine agreement protocol that runs in expected $O(\log n/\epsilon^2)$ rounds.

Lemma 6. For any constant $\epsilon > 0$, the protocol $\Pi_{\text{Elect-BPV}}$ is an $(O(\frac{\log n}{\epsilon^2}), 1 - \frac{1}{n}, 1 - \frac{1}{n})$ committee-election protocol in a synchronous full-information network of n players tolerating $t < (1/4 - \epsilon)n$ Byzantine faults. That is, all honest players output the same committee of size $O(\frac{\log n}{\epsilon^2})$ with probability at least $1 - \frac{1}{n}$, and given that all the players output the same committee, the probability that the elected committee is good is also at least $1 - \frac{1}{n}$. $\Pi_{\text{Elect-BPV}}$ runs in $O(\frac{\log n}{\epsilon^2})$ rounds.

Proof: We show the following:

1. With probability $1 - \frac{1}{n}$, all the honest players *agree* on the list of bins that have not been disqualified. That is, for every two honest players P_i and P_j, $\{B \mid \mathsf{disq}_i(B) = 1\} = \{B \mid \mathsf{disq}_j(B) = 1\}$. Furthermore, for every bin B that is not disqualified, all the players agree on the composition of B, namely the set of players that belong to bin B. This is shown in Lemma 7.

2. By Lemma 8, at least one bin is not disqualified.

3. By Lemma 9, all the committees that are not disqualified have at most $\frac{3}{4}a \log n$ players, of which at least a $\frac{2}{3}$ fraction are honest players.

Together, these statements mean that with probability $1 - \frac{1}{n}$, all the players agree on the identity of the committee elected. Conditioned on this event, the elected committee is good with probability at least $1 - \frac{1}{n}$. The elected committee has size $\frac{3}{4}a \log n = O(\log n/\epsilon^2)$.

Let us now determine the round-complexity of this protocol. Step 1 and step 3 are the only interactive rounds. Step 1 takes $O(1)$ rounds, and step 3 consists of a number of parallel executions all of which terminate by $a \log n + 1$ rounds. Thus, $\Pi_{\text{Elect-BPV}}$ runs in $O(\frac{\log n}{\epsilon^2})$ rounds. ■

To start with, we note the following elementary fact, which states that if a player $P \in \mathsf{View}_i(B)$ for some honest player P_i, then $P \in \overline{\mathsf{View}}_j(B)$ for every other honest player P_j. This follows directly from the "graded agreement" property of graded broadcast. Proposition 3 below records this observation.

Proposition 3 (*Graded Broadcast Lemma*). For any two honest players P_i and $P_{i'}$ and any bin B, the following is true: for all players P_j, if $P_j \in \mathsf{View}_i(B)$, then $P_j \in \overline{\mathsf{View}}_{i'}(B)$.

Proof: This follows from the graded agreement property of graded broadcast. If P_i is an honest player, then the fact that $P_j \in \mathsf{View}_i(B)$ implies (by the definition of the views) that P_j's graded broadcast (with the message "I choose bin B") was received by P_i with a grade of 2. Therefore, all other honest players $P_{i'}$ will accept P_j's graded broadcast with a grade of at least 1. This, in turn, means that $P_j \in \overline{\mathsf{View}_{i'}}(B)$ for all honest $P_{i'}$. ■

The lemma below shows that the honest players agree on the set of bins that are disqualified. Namely, for every two honest players P_i and P_j and every bin B, $\mathsf{disq}_i(B) = \mathsf{disq}_j(B)$. Moreover, for every bin B that is not disqualified, all the honest players P_i agree on its composition (that is, the set $\mathsf{composition}_i(B)$ defined in Step 5 of the protocol). Namely, the lemma says that for every two honest players P_i and P_j and for every bin B such that $\mathsf{disq}_i(B) = 0$ $(= \mathsf{disq}_j(B))$, $\mathsf{composition}_i(B) = \mathsf{composition}_j(B)$.

Lemma 7 (*Agreement Lemma*). For every two honest players P_i and P_j and every bin B, the following holds:

1. $\mathsf{disq}_i(B) = \mathsf{disq}_j(B)$.
2. If $\mathsf{disq}_i(B) = \mathsf{disq}_j(B) = 0$, then $\mathsf{composition}_i(B) = \mathsf{composition}_j(B)$.

Proof: We divide the proof of the first assertion into two cases. In both cases, we will show that if an honest player P_i sets $\mathsf{disq}_i(B) = 1$ for some bin B, then all the honest players P_j set $\mathsf{disq}_j(B) = 1$ too.

Case 1: P_i set $\mathsf{disq}_i(B) = 1$ because P_i received messages of the form $(S, B, 1, *)$ from some set $S' \subseteq S$ of at least $\frac{1}{2}a \log n$ players such that *the set S consists of less than $\frac{1}{4}a \log n$ bad players.*

The total number of players in S is $\frac{3}{4}a \log n$. Thus, the fraction of players in S that are corrupt is less than $\frac{1/4a \log n}{3/4a \log n} = \frac{1}{3}$. Therefore, the Byzantine agreement protocols within S (in Step 3) will succeed. Consequently, all the honest players $P_k \in S$ have the same value for $\mathsf{disq}_{k,S}(B)$.

Furthermore, since $|S'| \geq \frac{1}{2}a \log n$ and P_i received an $(S, B, 1, *)$ message from all the players in S', there is at least one honest player $P_k \in S'$ such that $\mathsf{disq}_{k,S}(B) = 1$. Since all the honest players P_k in S have the same value for $\mathsf{disq}_{k,S}(B)$, all of them set $\mathsf{disq}_{k,S}(B) = 1$.

Now, there are more than $\frac{1}{2}a \log n$ honest players in S, and we just showed that each such player P_k sets $\mathsf{disq}_{k,S}(B) = 1$. All these players will send a message $(S, B, 1, *)$ to all the players. Thus, each player P_j will receive messages of the form $(S, B, 1, *)$ from the (at least) $\frac{1}{2}a \log n$ honest players in S. Thus, P_j will set $\mathsf{disq}_j(B) = 1$ too.

Case 2: P_i set $\mathsf{disq}_i(B) = 1$ because P_i received messages of the form $(S, B, 1, *)$ from some set $S' \subseteq S$ of at least $\frac{1}{2}a \log n$ players such that *the set S consists of at least $\frac{1}{4}a \log n$ bad players.*

Note that P_i accepts the disqualification messages $(S, B, 1, *)$ from the players in S' only if $S' \subseteq S \subseteq \mathsf{View}_i(B)$. By Proposition 3, this means that for every other honest player $P_{i'}$, $S \subseteq \overline{\mathsf{View}}_{i'}(B)$ as well.

Let us now count the number of players in $\overline{\mathsf{View}}_{i'}(B)$. As we just saw, $\overline{\mathsf{View}}_{i'}(B)$ contains all the players in S. In particular, since S contains at least $\frac{1}{4}a \log n$ bad players, $\overline{\mathsf{View}}_{i'}(B)$ contains all these players as well. In addition, since every bin has more than $\frac{3}{4}a \log n$ honest players, all of these players are in $\overline{\mathsf{View}}_{i'}(B)$ as well. Thus, for every honest player $P_{i'}$, $|\overline{\mathsf{View}}_{i'}(B)| > \frac{1}{4}a \log n + \frac{3}{4}a \log n = a \log n$.

In fact, we can make an even stronger statement. Consider the set GS (short for "good set") that consists of some $\frac{3}{4}a \log n$ honest players that chose the bin B. By assumption, there is at least one such set. The argument in the above paragraph immediately generalizes to show that for any such "good set" GS that consists entirely of honest players,

$$|\bigcap_{P_{i'} \in GS} \overline{\mathsf{View}}_{i'}(B)| > a \log n$$

Now, all the players in the good set will decide that the bin B is too large. This is because for any honest player $P_k \in GS$,

$$\mathsf{SetView}_{k,GS}(B) = \bigcap_{P_{i'} \in GS} \overline{\mathsf{View}}_{i'}(B)$$

and thus, $|\mathsf{SetView}_{k,GS}(B)| > a \log n$.

Thus, the good set GS will decide to disqualify B, and notify all the players of this decision by sending a message $(GS, B, 1, *)$. Since $GS \subseteq \mathsf{View}_j(B)$ for every honest player P_j, P_j will accept this message and set $\mathsf{disq}_j(B) = 1$ too.

In short, we proved above that in both the cases, if an honest player P_i sets $\mathsf{disq}_i(B) = 1$ for some bin B, then every honest player P_j sets $\mathsf{disq}_j(B) = 1$ as well.

We will now prove the second assertion, namely that if a bin B is not disqualified, then all the honest players agree on the composition of B. Observe that for any bin B that is *not disqualified*, the following is true: if a set of players $S \subseteq \mathsf{View}_k(B)$ for some honest player P_k, then S must have *more than two-thirds fraction* of honest players. Otherwise, the bin B would have been disqualified by a "good set" using an argument exactly like the proof of Case 2 above.

This means that the Byzantine agreement protocol in S succeeds. Thus, all the honest players P_k in S compute the same set $\mathsf{composition}_{k,S}(B)$ (and send it to all the players). Since there are at least $\frac{1}{2}a \log n$ honest players in S, this means that any two honest players P_i and P_j will set $\mathsf{composition}_i(B) = \mathsf{composition}_j(B) = \mathsf{composition}_{k,S}(B)$. This ensures agreement. ■

Lemma 8 below shows that there is some bin B that is not disqualified.

Lemma 8 (*Survivor Lemma*). There is at least one bin B such that $\mathsf{disq}_i(B) = 0$ for all honest players P_i.

Proof: We will explicitly exhibit a bin B that cannot be disqualified.

Let $\mathcal{H}$ denote the set of all honest players. We define a set $\mathbb{C}(B)$ which contains all the players that are in $\overline{\mathsf{View}}_k(B)$ for *some* honest player P_k. Formally, define

$$\mathbb{C}(B) = \bigcup_{P_k \in \mathcal{H}} \overline{\mathsf{View}}_k(B)$$

Note that the set $\mathbb{C}(B)$ is defined purely for the purposes of our analysis, and is never explicitly computed by any of the players.

We will show that there is some bin B for which the set $\mathbb{C}(B)$ is "small" (namely, of size at most $a \log n$). Subsequently, we will show that such a bin B cannot be disqualified.

Claim 1. There exists a bin B such that $|\mathbb{C}(B)| \leq a \log n$.

Proof: First, we prove that for any two bins $B \neq B'$, $\mathbb{C}(B)$ and $\mathbb{C}(B')$ are disjoint. This essentially follows from the graded agreement property of graded broadcast. More formally, if a player $P \in \overline{\mathsf{View}}_i(B)$ for some honest player P_i and bin B, then $P \notin \overline{\mathsf{View}}_{i'}(B')$ for any honest player $P_{i'}$ and bin $B' \neq B$. This immediately implies that $\mathbb{C}(B)$ and $\mathbb{C}(B')$ are disjoint.

Moreover, since there are n players in total,

$$|\bigcup_{B} \mathbb{C}(B)| \leq n$$

Since there are $n/a \log n$ bins in total, by pigeonhole principle, it follows that at least one of them contains at most $a \log n$ players. That is, there is some bin B for which $|\mathbb{C}(B)| \leq a \log n$. ■

Consider the bin B, such that $|\mathbb{C}(B)| \leq a \log n$, guaranteed by Claim 1. We show that B cannot be disqualified. Suppose not, for the sake of contradiction. That is, suppose that there is some honest player P_j who sets $\mathsf{disq}_j(B) = 1$. P_j sets $\mathsf{disq}_j(B) = 1$ because it receives messages of the form $(S, B, 1, *)$ from some set $S \subseteq \mathsf{View}_j(B)$. Consider two cases.

1. S has at least $\frac{1}{4} a \log n$ dishonest players: Since the honest player P_j accepts messages of the form $(S, B, 1, *)$, it must be the case that $S \subseteq \mathsf{View}_j(B)$. In particular, $\mathsf{View}_j(B)$ contains all the dishonest players in S – there are at least $\frac{1}{4} a \log n$ of them, by assumption. In addition, $\mathsf{View}_j(B)$ contains all the honest players in B – there are more than $\frac{3}{4} a \log n$ such players, by assumption. Thus, $|\mathsf{View}_j(B)| > a \log n$. By Proposition 3,

$$\mathbb{C}(B) = \bigcup_{P_k \in \mathcal{H}} \overline{\mathsf{View}}_k(B) \supseteq \mathsf{View}_j(B)$$

This means that $|\mathbb{C}(B)| > a \log n$, contrary to assumption. In other words, this case cannot happen.

2. S has less than $\frac{1}{4}a \log n$ dishonest players: Since S has more than $\frac{2}{3}$ fraction of good players, Byzantine Agreement in S succeeds.

   The good players in S compute $\bigcap_{P_i \in S} \overline{\mathsf{View}_i}(B)$ to decide if the bin is too big. But, note that

$$\bigcap_{P_i \in S} \overline{\mathsf{View}_i}(B) \subseteq \bigcap_{\text{honest } P_i \in S} \overline{\mathsf{View}_i}(B) \subseteq \bigcup_{\text{honest } P_i} \overline{\mathsf{View}_i}(B) \stackrel{def}{=} \mathbb{C}(B)$$

   By the choice of B, $|\mathbb{C}(B)| \leq a \log n$, and therefore the good players in S do not disqualify B, contrary to assumption.

In short, we have shown that there is a bin B that cannot be disqualified. ■

Finally, we show that for any bin B that is not disqualified, (for any honest player P_i) $\mathsf{composition}_i(B)$ contains more than a $\frac{2}{3}$ fraction of honest players. In particular, we show that $\mathsf{composition}_i(B)$ contains at most $\frac{3}{4}a \log n$ players, of which more than $\frac{1}{2}a \log n$ are honest.

Note that the entire player-set contains at least $\frac{3}{4}$ fraction of honest players, but the committee contains only a $\frac{2}{3}$ fraction of honest players. This degradation is an artifact of our protocol.

Lemma 9 (*"The Chosen One is Good" Lemma*). For any honest player P_i, if $\mathsf{disq}_i(B) = 0$, then $\frac{3}{4}a \log n < |\mathsf{composition}_i(B)| \leq a \log n$. Furthermore, $\mathsf{composition}_i(B)$ contains at most $\frac{1}{4}a \log n$ dishonest players. In particular, the fraction of dishonest players in $\mathsf{composition}_i(B)$ is less than $\frac{1}{3}$.

Proof: Analogous to Lemma 8, define the set $\mathbb{D}(B)$ to be the set of all players that are in $\mathsf{View}_k(B)$ for some honest player P_k. The only difference between $\mathbb{C}(B)$ (defined in Lemma 8) and $\mathbb{D}(B)$ is that the former refers to $\overline{\mathsf{View}_k}(B)$ whereas the latter refers to $\mathsf{View}_k(B)$. Clearly, $\mathbb{D}(B) \subseteq \mathbb{C}(B)$.

Formally, let $\mathcal{H}$ denote the set of all honest players and define

$$\mathbb{D}(B) \stackrel{def}{=} \bigcup_{k \in \mathcal{H}} \mathsf{View}_k(B)$$

We now show that for any bin B that is not disqualified, $\mathbb{D}(B)$ is small. (Note that this is somewhat of a partial converse to the claim in Lemma 8 which states that if $\mathbb{C}(B)$ is small, then B is not disqualified).

Claim 2. If a bin B is not disqualified, then $\mathbb{D}(B)$ is small. In particular, if $\mathsf{disq}_i(B) = 0$ for some honest player P_i, then $|\mathbb{D}(B)| \leq a \log n$.

Proof: Consider an arbitrary set GS (the "good set") that consists of $\frac{3}{4}a \log n$ honest players in B; there must exist at least one such set, since each bin contains more than $\frac{3}{4}a \log n$ honest players. Since $\mathsf{disq}_i(B) = 0$, the good set GS did not disqualify B.

Suppose, for contradiction, that $|\mathbb{D}(B)| > a \log n$. For every honest player P_k, $\overline{\mathsf{View}_k}(B) \supseteq \mathbb{D}(B)$ by Proposition 3. Thus,

$$\bigcap_{P_k \in GS} \overline{\mathsf{View}_k}(B) \supseteq \mathbb{D}(B)$$

Thus,

$$|\bigcap_{P_k \in GS} \overline{\mathsf{View}}_k(B)| > a \log n$$

and therefore the good set GS will disqualify B, contrary to assumption. Thus, $|\mathbb{D}(B)| \leq a \log n$. ■

Now, since B is not disqualified, exactly as in the proof of Lemma 8, any set S whose decision matters has less than $\frac{1}{4}a \log n$ bad players.

The composition for B advertised by the good set contains all the good players in B – there are more than $\frac{3}{4}a \log n$ of them. This means that $|\mathsf{composition}_i(B)| > \frac{3}{4}a \log n$. Suppose this is not the final composition chosen. This can happen only if there exists another set $\widetilde{S}$ that advertises a larger composition for B. i.e, a composition of size more than $\frac{3}{4}a \log n$.

Now, we upper bound the number of bad players in any such advertised (and accepted) composition. Recall that the players in $\widetilde{S}$ agree on $\bigcap_{P_k \in \widetilde{S}} \mathsf{View}_k(B)$ as the composition. For any honest player $P_k \in \widetilde{S}$, $\mathsf{View}_k(B)$ has less than $\frac{1}{4}a \log n$ dishonest players. Thus, $\bigcup_{P_k \in \mathcal{H}} \mathsf{View}_k(B)$ has at most $\frac{1}{4}a \log n$ dishonest players.

We now show an upper-bound on the size of $\mathsf{composition}_i(B)$. This follows from the fact that

$$\bigcap_{P_k \in \widetilde{S}} \mathsf{View}_k(B) \subseteq \bigcap_{\text{honest } P_k \in \widetilde{S}} \mathsf{View}_k(B) \subseteq \bigcup_{P_k \in \mathcal{H}} \mathsf{View}_k(B) \stackrel{def}{=} \mathbb{D}(B)$$

Thus, since $|\mathbb{D}(B)| \leq a \log n$, $\mathsf{composition}_i(B) = \bigcap_{P_k \in \widetilde{S}} \mathsf{View}_k(B)$ is of size at most $a \log n$ too.

We just showed that $\frac{3}{4}a \log n < |\mathsf{composition}_i(B)| \leq a \log n$. Furthermore, we showed that $\mathsf{composition}_i(B)$ has at most $\frac{1}{4}a \log n$ dishonest players; in other words, at most a $\frac{1}{3}$ fraction of dishonest players. ■

4.2.2 Byzantine Agreement Protocol II

This section is devoted to proving the following theorem.

Theorem 3. For any constant $\epsilon > 0$, there exists a protocol BA_ϵ that achieves Byzantine Agreement in a synchronous full-information network of n players tolerating $t < (\frac{1}{3} - \epsilon)n$ faults. The round complexity of BA_ϵ is $O(\frac{\log n}{\epsilon^2})$, and the communication complexity is $\widetilde{O}(n^2)$ bits.

First, we describe an outline of our protocol. The formal description of the protocol and the proof will follows.

Overview of the Byzantine Agreement Protocol

As in our first construction of a Byzantine agreement protocol, we construct a committee election protocol that elects a committee of size $O(\frac{\log n}{\epsilon^2})$ tolerating $t < (1/3 - \epsilon)n$ Byzantine faults, with the protocol Π_{RZ} as the starting point. The reduction from Byzantine Agreement to Committee Election (Chapter 3,

Theorem 1) then immediately gives us a Byzantine agreement protocol with the same parameters.

The rest of this section describes in a very high level the working of our committee-election protocol $\Pi_{\text{Elect-GPV}}$. Let us first condition on the event that the random choices of the honest players eliminate all the bad committees – by Lemma 5, this event happens with probability $1 - \frac{1}{n}$. If this event does not happen, then we cannot guarantee that the committee that is elected will be good, or even that all the players agree on the elected committee. For the rest of the discussion, we assume that this event happens.

The first idea is to let the players announce their (random) choices, using a graded broadcast protocol. This ensures that a dishonest player cannot "convince" two different honest players that he chose two different sets of committees to eliminate. In other words, for every dishonest player P^*, there is a *unique* set of committees S^* (which could possibly be $\perp$) such that each honest player P_i sees that P^* chose to eliminate S^*, or chose nothing at all! If each honest player P_i now chooses the lexicographically first among the surviving committees, we have the guarantee that all the honest players will choose a good committee. However, two different honest players might choose two different committees; in other words, there is no agreement among the honest players as to which committee is chosen.

To remedy this problem, we will ask each committee (more precisely, the players in each committee) to run an agreement protocol among themselves to decide if the committee should be eliminated. A committee decides to eliminate itself (that is, "self-destruct") if it detects a possible disagreement among the players about its fate. This is, in some sense, a safety mechanism. We will show that this process ensures the following properties. First of all, all the bad committees are already eliminated by the good players' choices. The protocol within the committee C can decide to eliminate C, and not "un-eliminate" C. This one-sided decision helps ensure that a bad committee can never be chosen. Secondly, we will show that the self-destruct mechanism makes sure that all the players agree on the list of committees that have been eliminated. Thirdly, there will be at least one (necessarily good) committee that is neither eliminated nor self-destructs. These three properties will together show that the committee-election protocol succeeds.

Formal Proof of Theorem 3

The formal description of the protocol is in Table 4.4. To prove Theorem 2, it suffices to prove the following lemma, which states that $\Pi_{\text{Elect-GPV}}$ is a committee-election protocol that runs in $O(\log n/\epsilon^2)$ rounds, elects a committee of size $O(\log n/\epsilon^2)$, and has $o(1)$ error probabilities (namely, the probability that the players do not agree on the elected committee, as well as the probability that the elected committee is bad). Then, by Theorem 1, we get a Byzantine agreement protocol that runs in expected $O(\log n/\epsilon^2)$ rounds.

PROTOCOL $\Pi_{\text{Elect-GPV}}$: PSEUDOCODE

Pre-Computation: Run deterministic algorithms that on input 1^n and ϵ output a collection of committees $\mathcal{C} = \{C_1, C_2, \ldots, C_m\}$, and a hitting set $h : \mathcal{C} \rightarrow \{1 \ldots a\}^n$.

Output of Player P_i: A committee $C_i \in \mathcal{C}$.

Step 1 **(Each Player** P_i**)** Choose r_i randomly from $\{1 \ldots a\}$ and *graded broadcasts* r_i to all the players. The output of P_i from the graded broadcasts is a list of n values $\left[(r_{i,1}, g_{i,1}), (r_{i,2}, g_{i,2}), \ldots, (r_{i,n}, g_{i,n})\right]$.

Construct an m-bit list $(e_{i,1}, \ldots, e_{i,m})$ as follows: set $e_j = 1$ if there is a $k \in [1 \ldots n]$ such that $h(C_j)|_k = r_{i,k}$ and $g_{i,k} = 2$.

Remark: $e_{i,j} = 1$ if player P_i "thinks" that committee C_j has been eliminated.

Step 2 **(Each Player** P_i **in Committee** C_j**)** Set the variable

$$\mathsf{SelfDestruct}_{i,j} = \begin{cases} 1 & \text{if } \exists k \text{ where } h(C_j)|_k = r_{i,k} \text{ and } g_{i,k} \geq 1 \\ 0 & \text{otherwise} \end{cases}$$

All players in C_j run a Byzantine agreement protocol, where the input of the i^{th} player is $\mathsf{SelfDestruct}_{i,j}$. This is done by using a deterministic reliable broadcast protocol among the players in C_j. Each player sets $\mathsf{SelfDestruct}_{i,j}$ to be the output of the BA protocol.

Remark: All the players in committee C_j run an agreement protocol among themselves, to determine if some player P_i (possibly outside the committee) "thinks" that C_j is eliminated. The guarantee provided by this step is the following: If C_j is a good committee, and some honest player P_i has set $e_{i,j} = 1$ in the previous step, then all honest players P_k in C_j will set $\mathsf{SelfDestruct}_{k,j} = 1$.

Step 3 **(Each Player** P_i **in Committee** C_j**)** Send, via pairwise channels $\mathsf{SelfDestruct}_{i,j}$ to all the n players.

Local Step $3'$ **(Each Player** P_i**)** Receive for every player P_k in committee C_j, a bit $s_{i,j,k}$. Modify the list $(e_{i,1}, \ldots, e_{i,n})$ as follows: Set $e_{i,j} = 1$ if there are at least $2|C_j|/3$ k's such that $s_{i,j,k} = 1$. Otherwise, leave $e_{i,j}$ unchanged.

Remark: Consolidate the opinions of all the players in C_j about whether to self-destruct C_j, by taking a threshold vote among the players in C_j. This is the key step that ensures agreement.

Output the lexicographically smallest C_j such that $e_{i,j} = 0$.

Table 4.4: Our Committee-Election Protocol without Broadcast.

Lemma 10 (Committee-selection without Broadcast Channels). For any constant $\epsilon > 0$, the protocol $\Pi_{\text{comm-elect}}$ is an $(O(\frac{\log n}{\epsilon^2}), 1 - \frac{1}{n}, 1 - \frac{1}{n})$ committee-election protocol in a synchronous full-information network of n players tolerating $t < (1/3 - \epsilon)n$ Byzantine faults. That is, all honest players output the same committee of size $O(\frac{\log n}{\epsilon^2})$ with probability at least $1 - \frac{1}{n}$, and given that all the players output the same committee, the probability that the elected committee is good is also at least $1 - \frac{1}{n}$. $\Pi_{\text{comm-elect}}$ runs in $O(\frac{\log n}{\epsilon^2})$ rounds, and the communication complexity is $\widetilde{O}(n^2)$ bits.

Proof: We show the following.

1. First of all, Lemma 11 shows that at the end of $\Pi_{\text{Elect-GPV}}$, all the bad committees are eliminated.

2. With probability $1 - \frac{1}{n}$, all the honest players *agree* on the list of committees that have not been eliminated. This is shown in Lemma 11.

3. Finally, at the end of the protocol $\Pi_{\text{comm-elect}}$, no honest player outputs "fail". That is, with probability 1, there is at least one committee that is not eliminated. This is shown in Lemma 12.

Together, these statements mean that with probability $1 - \frac{1}{n}$, all the players agree on the identity of the committee elected, and conditioned on this event, the elected committee is good with probability $1 - \frac{1}{n}$. Clearly, the elected committee has size $O(\log n/\epsilon^2)$.

The round-complexity is the sum of the round-complexities of Steps 1, 2 and 3. Step 1 is a graded broadcast, which takes $O(1)$ rounds. Step 2 consists of a reliable broadcast protocol run within each committee in parallel. Using the deterministic reliable broadcast protocol of [GM98], this takes $O(|C|) = O(\log n/\epsilon^2)$ rounds. Step 3 is a single round, where each player communicates to every other player via the pairwise channels.

The bottleneck in the communication complexity is Steps 2 and 3. Step 2 consists of $\widetilde{O}(n^2)$ Byzantine agreement among $O(\log n)$ players, where the input in each protocol is single bit. Step 3 consists of each player in each committee sending a bit to all the players. This takes $\widetilde{O}(n^2)$ bits too. The total communication is thus $\widetilde{O}(n^2)$ bits. ■

The lemma below shows that with high probability, at the end of $\Pi_{\text{Elect-GPV}}$, all the bad committees are eliminated.

Lemma 11. With probability $1 - \frac{1}{n}$ over the coin-tosses of the honest players, all bad committees are eliminated at the end of $\Pi_{\text{Elect-GPV}}$. Formally, for every honest player P_i and for every bad committee C_j, $e_{i,j} = 1$.

Proof: First of all, by Lemma 5, for every bad committee C_j, there is an honest player P_k such that P_k eliminates C_j in the protocol Π_{RZ}. In other words, for every bad committee C_j, there is an honest player P_k such that $h(C_j)|_k = r_k$. Since P_k is honest, his graded broadcast is received by every other honest player

P_i with the maximum grade of 2. This means that every honest player sets $e_{i,j} = 1$ at the end of Step 1. ■

The next lemma shows that all the honest players agree on the list of committees that have been eliminated, at the end of the protocol. In other words, for every two honest players P_i and $P_{i'}$, the lists $\mathbf{e}_i = (e_{i,1}, \ldots, e_{i,m})$ and $\mathbf{e}_{i'} = (e_{i',1}, \ldots, e_{i',m})$ are identical, at the end of Local Step 3′.

Lemma 12. For every two honest players P_i and $P_{i'}$, $\mathbf{e}_i = \mathbf{e}_{i'}$ at the end of the protocol $\Pi_{\text{comm-elect}}$.

Proof: It is sufficient to show that for every j such that $\mathbf{e}_{i,j} = 1$, $\mathbf{e}_{i',j} = 1$ also. First of all, for every bad committee C_j, $\mathbf{e}_{i,j} = \mathbf{e}_{i',j} = 1$. This is because $\mathbf{e}_{i,j} = \mathbf{e}_{i',j} = 1$ at the end of Step 1 for every bad committee C_j (be Lemma 11), and once $e_{i,j}$ (resp. $e_{i',j}$) is set to 1, it is never reset back to 0.

In the rest of the proof, we let C_j be a good committee. If $e_{i,j}$ is 1, then it is because of one of the following reasons.

1. P_i received a bit $s_{i,j,k} = 1$ from at least $2|C|/3$ players P_k in committee C_j. Consequently, P_i sets the bit $e_{i,j} = 1$. Since C_j is a good committee and the players compute the bits $s_{i,j}$ by running a BA protocol, all other honest players $P_{i'}$ will receive at least $2|C|/3$ bits $s_{i',j,k} = 1$ too. Thus, P' will set the bit $e_{i',j} = 1$.

2. P_i set the bit $e_{i,j} = 1$ in Step 1, because it received from some player P_k a value r_j with grade 2 such that $h(C_j)|_k = r_k$. This means that every other honest player $P_{i''}$ received r_k with grade at least 1, and thus will set $\mathsf{SelfDestruct}_{i'',j} = 1$. In particular, this is true for all the honest players in the committee C_j, which means that the BA protocol will result in all of them receiving an output of 1. In turn, all the honest players $P_{i''}$ in the committee C_j sending a bit $s_{i'',j} = 1$ in Step 3. When an honest player $P_{i'}$ receives this message from all the honest players in C_j, it will set the bit $e_{i',j} = 1$ too.

■

Finally, we show that there is at least one committee is not eliminated. That is, for every honest player P_i, there is a committee C_j such that $e_{i,j} = 0$.

Lemma 13. There exists a committee C_j such that $e_{i,j} = 0$ for every honest player P_i.

Proof: Fix the collection of messages r_j sent by all the players (including the faulty ones). By Lemma 5, there is at least one committee C_j such that $h(C_j)|_k \neq r_k$ for any k. This means that $e_{i,j} = 0$ for every honest player P_i, and the committee C_j survives. ■

4.3 Can the Round-Complexity be Improved?

The main contribution to the round-complexity of the above two protocols is the Byzantine agreement protocol that we run among the players in each committee. In a sense, these two reliable broadcast protocols can be viewed as a reduction from reliable broadcast among n players to many instances of reliable broadcast among a much smaller number of players (which, in our case is $O(\log n)$).

Two possible venues to improving the round-complexity of these protocols come to mind. The first is to design a committee-election protocol that outputs a smaller committee. Since the complexity of the Byzantine agreement protocol within the committee is proportional to the size of the committee, this will directly result in a smaller round-complexity of the resulting reliable broadcast protocol. We remark that given *any* one-round committee-election protocol that outputs a committee of size c, possibly using a broadcast channel, we know how to convert it into an $O(c)$-round committee-election protocol without using broadcast. There are no lower-bounds on the number of rounds required for committee-election or even leader-election. Thus, in principle, it is possible that there is a one-round protocol for leader-election, which we can immediately convert to an $O(1)$-round reliable broadcast protocol. In fact, we show how to do exactly this in Chapter 6 for a fault-tolerance of $O(n/\log^{O(1)} n)$.

The second possibility is to use a protocol with a smaller than linear round-complexity in order to run Byzantine agreement among the players in each committee. For example, one might think of using the $O(\log n)$ round reliable broadcast protocol we just designed to bootstrap the Byzantine agreement protocol within each committee, by recursion. We remark that this strategy does not work because of a curious property of randomized protocols. Given any randomized protocol with an expected running-time of t, executing $\mathsf{poly}(n)$ *independent copies* of the protocol blows up the running-time to $t \cdot O(\log n)$. Thus, even if we have a Byzantine agreement protocol with expected running-time $O(1)$, executing it on all the $m = n^2$ committees independently in parallel will increase the running-time to $O(\log n)$, which means that we really gained nothing at all!

A third possibility is to somehow *correlate* the m Byzantine agreement protocols taking place in parallel so that their total expected running-time is much smaller than $O(\log n)$ times the individual round-complexity. Indeed, we showed how to do this for any polynomial number of Byzantine agreement protocols running *among the same set of players* (see Chapter 3). In that case, we could make sure that the total expected running-time was the same as the expected running-time of each individual protocol. The situation is different here, as we are running the different Byzantine agreement protocols on different sets of players.

5

Auditing a Distributed Protocol and an Application

The main motivation behind the construction of reliable broadcast protocols is to convert distributed protocols that assume built-in broadcast channels into ones that do not make such an assumption. This transformation, however, comes at a cost: each invocation of the reliable broadcast channel in the original protocol is replaced with a reliable broadcast protocol (as in Chapter 4), and this increases the round-complexity of the original protocol by a *multiplicative* factor of $O(\log n)$ (using the protocols in Chapter 4, which are the best known so far in the full-information model).

This raises the following question: given a general protocol that assumes built-in reliable broadcast channels, is it possible to run (a modified version of) the protocol that (a) does not take much more time to run than the original protocol, and (b) still provides a meaningful guarantee? In fact, graded broadcast can be thought of as exactly such a mechanism: it provides the functionality of a "semi-reliable broadcast channel" which is quite meaningful, although weaker than the reliable broadcast functionality, and furthermore, can be implemented very efficiently.

In this chapter, we present a general transformation of protocols, called auditing. Given any distributed protocol Π that possibly assumes broadcast channels, auditing Π results in a protocol Π' that has the effect of running Π "as well as possible" even when no reliable broadcast channels are given. Furthermore, the round-complexity of Π' is tightly related to that of Π. This transformation is inspired by the techniques of Chapter 4.

AN APPLICATION OF AUDITING: BOOSTING FAULT-TOLERANCE OF PROTOCOLS. We show how to apply the audit transformation to the problem of boosting the fault-tolerance of distributed protocols. We explain this informally below. When designing distributed protocols, it is often convenient to first design a

protocol for a weaker model of faults, and then use a general transformation to convert the protocol to also work with a stronger fault-model. The weakest model of faults that is still reasonable to consider is the fail-stop fault model, whereas the Byzantine fault-model is by far the strongest. Bracha [Bra84] and later, Neiger and Toueg [NT90] designed a mechanism that transforms a *deterministic* distributed protocol tolerating fail-stop faults into an equivalent protocol that tolerate Byzantine faults. The question of whether such a transformation can be applied to randomized protocols as well was left open by their work. We use the auditing transformation to answer this question in the affirmative.

ORGANIZATION OF THIS CHAPTER. In Section 5.1, we formally define the guarantees provided by the audited protocol. In Section 5.1.1, we construct the auditing transformation itself, and in section 5.2, we show how to apply the auditing transformation to boosting the fault-tolerance of distributed protocols.

5.1 The Guarantees of an Audited Protocol

Consider any distributed protocol Π where all the players receive the same output.[1] For any such protocol Π and for any player P (the "auditor"), we define an "audited" protocol $\Pi' = \mathsf{Audit}(P, \Pi)$. We will refer to Π' as a P-audited version of Π. Informally, the execution of Π' with the auditor P provides the following guarantees.

- If the auditor P is honest, Π' works exactly like Π. In particular, the output of the protocol is distributed exactly as in Π, and furthermore, all the honest players receive the output in Π'.
- Even if the auditor is dishonest, he can do only minimal damage – the worst he can do is set the outputs of some of the players to $\perp$. In other words, the output of the protocol is still distributed exactly as in Π except that an arbitrary subset of the honest players do not get the output in Π'.

To define this precisely, we need a notion of what it means to simulate a protocol by another protocol.

SIMULATION OF A PROTOCOL BY ANOTHER PROTOCOL. We let $\mathcal{D}(\Pi, \vec{x}, A; \vec{\mathcal{R}})$ denote the output distribution of the protocol Π against the adversary A, when the input vector of the players is $\vec{x} = (x_1, \ldots, x_n)$. Here, $\vec{\mathcal{R}} = (\mathcal{R}_1, \ldots, \mathcal{R}_n)$ where $\mathcal{R}_i$ is the random source for player P_i. Note that the sources $\mathcal{R}_i$ could, in general, be dependent on each other.

The definition below of what it means for a protocol Π' to simulate a protocol Π is cryptographic in flavor. Essentially, it says that for any adversary A' that can

[1] The condition that all the players receive the same output is made so that the definition is simple. We remark that the results in this section can be generalized to an arbitrary distributed protocol.

force a particular output distribution in Π', there is a corresponding adversary A that can force the same distribution in Π. In other words, Π' is "as good as" Π. Note that since we are in working in the full-information model, both A and A' are unbounded algorithms, can corrupt t players each statically, and can listen to all communication in the network.

Definition 8 (**Perfect Simulation of a Protocol Π by a Protocol Π'**). Protocol Π' with randomness source $\vec{\mathcal{R}}'$ is said to *perfectly simulate* Π with randomness source $\vec{\mathcal{R}}$ if,

(1) for every adversary A', there exists an adversary A such that for every input vector $\vec{x}$, the output distribution of Π' under the influence of A' is identical to the output distribution of Π under A. That is,

$$\forall A', \exists A \text{ such that } \forall \vec{x},\ \mathcal{D}(\Pi', \vec{x}, A'; \mathcal{R}') \equiv \mathcal{D}(\Pi, \vec{x}, A; \mathcal{R})$$

(2) all the players in Π' receive the output $\mathcal{D}(\Pi', \vec{x}, A'; \mathcal{R}')$.

In such a case, we write $\Pi' \sim \Pi$.

In our setting, we also need the notion of best-effort simulation of a protocol Π by a protocol Π'. Essentially, Π' "best-effort simulates" Π if the output distribution of the players in Π' is identical to that of Π, except that some players in Π' do not receive any output. We note that the set of players that do not receive any output in Π can be chosen by the adversary depending on the particular execution of the protocol.

Definition 9 (**Best-Effort Simulation of a Protocol Π by a Protocol Π'**). Protocol Π' with randomness source $\vec{\mathcal{R}}'$ is said to *best-effort simulate* Π with randomness source $\vec{\mathcal{R}}$ if the condition (1) in Definition 8 holds and

(2′) there exists a set of players S (possibly empty) that receive the output in Π', namely $\mathcal{D}(\Pi', \vec{x}, A'; \mathcal{R}')$.

In such a case, we write $\Pi' \sim_{\perp} \Pi$.

AUDITING A DISTRIBUTED PROTOCOL. Given any distributed protocol Π that possibly assumes broadcast channels, we would like execute Π "as well as possible" when no reliable broadcast channels are given. Of course, one way to do this would be to simulate every invocation of broadcast in Π with a reliable broadcast (equivalently, Byzantine Agreement) protocol. However, since the best known Byzantine Agreement protocol in the full-information model costs $\Theta(\log n)$ rounds of communication (as in Chapter 4), this transformation is not very efficient. What we would like to do is to get as good a guarantee as possible on the output of the protocol, while *preserving the efficiency of the original protocol.* This leads us to the notion of an audited protocol.

Definition 10 (Audit(P, Π)). There exists a protocol transformation Audit that has the following properties. Audit takes as input a protocol Π among n players that possibly assumes the existence of a built-in broadcast channel, as well the identity of a special player P, called the auditor. The output of Audit is a protocol Π' that does not assume reliable broadcast channels. The protocol $\Pi' = \mathsf{Audit}(P, \Pi)$ has the following guarantees.

- When P is honest, the protocol Π' perfectly simulates Π.
- Even when P is malicious, the protocol Π' best-effort simulates Π.
- The round-complexity of Π' is the same as the round-complexity of Π, upto constant factors. That is, $\mathsf{rounds}(\Pi) = O(\mathsf{rounds}(\Pi'))$.

Remark. We can generalize the definition of Audit so that the auditor is not a single player, but a set of players C (called a "committee"). In this case, we would require that Π' perfectly simulate Π whenever the committee C has a large fraction of honest players, and it best-effort simulates Π otherwise. Note that auditing by a single player is a special case of auditing by a committee. One the one hand, this generalization gives us more flexibility in the choice of the auditor; however, the complexity of the audited protocol Π' will grow linearly with the size of the set C. We do not pursue this generalization further in this work.

5.1.1 The Audit Transformation

In this section, we construct the Audit transformation that converts any protocol Π that possibly assumes reliable broadcast to an audited protocol Π' that does not assume reliable broadcast.

Informally, the motivation for the construction is the following: First, note that if Π is a protocol that implements the *reliable broadcast* functionality for a dealer D, then the guarantees provided by $\mathsf{Audit}(D, \Pi)$ are identical to the guarantees of a *semi-reliable broadcast* protocol, where the dealer D acts as the auditor in $\mathsf{Audit}(D, \Pi)$. This observation suggests a way of constructing the Audit transformation – replace each broadcast instruction in the protocol Π by the execution of a semi-reliable broadcast protocol. Such a simple transformation indeed works if Π is a *one-round* protocol. In the case of multi-round protocols, it turns out that we make this idea work using the stronger primitive of graded broadcast; *the different grades* guaranteed by graded broadcast can be used in a more sophisticated way to ensure that the dishonest players cannot skew the output distribution (beyond forcing some of the outputs to $\perp$).

Theorem 4. There exists a transformation Audit that satisfies Definition 10. Let $\Pi' = \mathsf{Audit}(P, \Pi)$. Then,

- If the fault-tolerance of Π is t, that of Π' is $\min(t, \lfloor \frac{n-1}{3} \rfloor)$.

THE TRANSFORMATION Audit(P, Π): PSEUDOCODE

Input: Protocol Π.
Output: Protocol $\Pi' = \mathsf{Audit}(P, \Pi)$

Π' works by simulating each round of Protocol Π as follows.

1. (**Each Player** P_i) If Π instructs P_i to send a message to P_j, Π' instructs the same.
2. (**Each Player** P_i) If Π instructs P_i to broadcast a message m to all the players, the following subroutine is invoked.
   (a) (**Player** P_i) Run a graded broadcast subroutine with P_i as the sender, that sends m to all the players.
   (b) (**The Auditor** P) Let m' denote the message that P received as a result of P_i's gradecast, in Step 2(a). P runs a graded broadcast subroutine to send m' to all the players.
   (c) (**Every player** P_j) Let (m_i, g_i) and (m_P, g_P) denote the outputs of P_j from the gradecast in step (a) and the gradecast of the auditor P in step (b), respectively. P_j will take m_P as the message broadcast by P_i in the underlying execution of Π.
   (d) (**Every player** P_j) Set a bit $\mathsf{fail}_j = 1$ if either
   - $m_P \neq m_i$ and $g_i = 2$, or
   - $g_P \neq 2$.
3. Finally, each player P_j gets an output O_i from the underlying execution of Π. P_j outputs O_j if $\mathsf{fail}_j = 0$, and $\perp$ otherwise.

Table 5.1: The transformation Audit

- If the round-complexity of Π is r_Π, the round-complexity of Π' is at most $3r_\Pi$.

Proof. We will show that the protocol $\Pi' = \mathsf{Audit}(C, \beta, \Pi)$ (given in Table 1) satisfies Definition 10.

If all the honest players P_j set $\mathsf{fail}_j = 1$ at the end of Π', then all of them output $\perp$, and there is nothing to prove. Our goal will be to show that if some honest player P_j sets $\mathsf{fail}_j = 0$ (that is, his output is not $\perp$) then for every time a broadcast instruction in the protocol is simulated by Step (2) of Audit, all the honest players in fact receive the same message (and thus, the functionality of broadcast is achieved).

Suppose some honest player P_j sets $\mathsf{fail}_j = 0$. This means that, in all steps of the simulation (that is, every time some possibly dishonest player P_i broadcasts a

message m), in the view of P_j, both the following hold:

(i) $g_P = 2$ for the auditor P, and

(ii) $m_P = m_i$ or $g_i \neq 2$.

This follows from Step $2(d)$ of Audit.

Since $g_P = 2$, every other honest player receives the same message from the auditor committee P, with confidence at least 1. Thus, it follows that all the players receive the same value for the broadcast of player P_i. Moreover, if P_i is good, then $g_i = 2$. Then, by condition (ii) above, $m_P = m_i$. This means that, if P_i is good, all players accept the message that P_i "broadcast" (regardless of whether the auditor P is good or not). Thus, we showed that every player gets the same message as a result of P_i's broadcast *and* moreover if P_i is good, they accept the message P_i sent. This simulates the functionality of reliable broadcast perfectly, in each round.

Since graded broadcast can be implemented with a fault-tolerance of $\lfloor n - 1/3 \rfloor$ and a round-complexity of 3, the claims about the fault-tolerance and round-complexity of Audit(P, Π) follow. ■

5.2 Boosting the Fault-Tolerance of Distributed Protocols

In this section, we show how to compile any randomized protocol that tolerates fail-stop faults into a protocol that tolerates Byzantine faults. Recall that when the corruption is fail-stop, all the players (including the corrupted ones) execute the code of the prescribed protocol, using the prescribed random source, except that the corrupted players can halt at any time during the protocol. On the other hand, the Byzantine adversary is arbitrarily malicious; in particular, a player corrupted by a Byzantine adversary does not necessarily follow the prescribed protocol nor does it use the prescribed random source.

In particular, our compiler expects as input a fail-stop-tolerant protocol where all the players have access to a γ-SV source as the source of randomness. A γ-SV source produces a sequence of possibly biased and possibly dependent bits such that the bias of each bit b_i is bounded by γ, *even when conditioned on the values of all the other bits*. In our context, giving all the players access to a γ-SV source has the following meaning: the concatenation of the random sources of all the players constitutes a γ-SV source. The compiler transforms such a protocol into a protocol that tolerates Byzantine adversaries, where the protocol gives each of the players access to a uniformly random source.[2]

In particular, our compiler has the following input-output behavior.

[2]Note that in this case, the dishonest players may choose not to use the prescribed source of randomness.

Input: Any full-information protocol $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ that tolerates a fail-stop t-adversary in which all the players have access to a γ-SV-source as the source of randomness, for some $\gamma = O(t/n)$.

Output: A protocol Π_{byz} in the full-information model that "realizes the same functionality" as $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$, tolerates a Byzantine $\min(t, \frac{n}{3})$-adversary and uses a uniformly random source. Furthermore the round-complexity of Π_{byz} is a constant times the round-complexity of $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$.

Formally, we show that for every adversary A in the output protocol Π_{byz}, there is an adversary A' and a γ-SV source $\mathcal{R}_\gamma$ in Π such that the execution of $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ using the random source $\mathcal{R}_\gamma$ under the control of the adversary A' is identical to the execution of Π_{byz} using a uniformly random source under the control of A.

REMARK. Our compiler works only in the full-information model. Namely, the compiler does not guarantee anything if the input protocol works in, say, the private channels model.

REMARK. Throughout this thesis, our definition of full-information model gives the adversary access to the messages sent through the communication channels, *as well as the internal state and the prior coin-tosses of all the honest players* (but not the future coin-tosses). This is stronger than the traditional full-information model (for example, as in [GGL98]) where the adversary has access to the messages in the communication channel, but not the internal state and the coin-tosses of the honest players. This more constrained model has been called the intrusive full-information model in the literature [CD89]. Working in the intrusive full-information model makes the results if Chapter 4 stronger.

However, working in the intrusive full-information model does not make the result in this section stronger (than a hypothetical equivalent result in the standard full-information model of [GGL98]). The reason is that even though our compiler produces an output protocol that works in the intrusive full-information model, it also expects as input a (fail-stop-tolerant) protocol that works in the same model as well!

We also remark that we do not know how to construct a compiler of this form for the standard full-information model – we leave this as a (very interesting) open question.

5.2.1 Informal Description of the Compiler

To understand our compiler, we first focus on the difference between a fail-stop adversary and a Byzantine one. Informally, a Byzantine fault is more malicious than a fail-stop fault in two aspects:

1. a Byzantine player can use arbitrary coins as the source of randomness, whereas a fail-stop player uses the prescribed source of randomness.

2. a Byzantine player can send arbitrary messages to players in every round, whereas a fail-stop player follows the prescribed protocol, and can only halt (possibly in the middle of a round).

Thus, to transform a protocol tolerating fail-stop faults to one tolerating Byzantine faults, we have to:

1. Force the Byzantine player to use the prescribed random source.

2. Restrict the Byzantine player to follow the prescribed protocol.

The second of these two issues was already addressed in the work of Neiger and Toueg [NT90], where they construct a compiler that transform a *deterministic* protocol that works against fail-stop faults into one that works against Byzantine faults. Informally, the idea there is to ask each player to prove that they executed the correct protocol with respect to the incoming messages and the randomness both of which can be assumed to be public without loss of generality (since we work in the full-information model). We state their result in Lemma 15.

The main novelty of our result is in solving the first issue, namely forcing an adversary to use the prescribed random source. We proceed to informally describe how this is done. One way to do this is to run a collective coin-flipping protocol and force each player to use the coin generated by the coin-flipping protocol.

We assume that the input protocol is public-coin – that is, the players send all the coins they use in the computation, along with the messages. We also assume that the players send their inputs in the first round; since the adversary is intrusive, we can assume this without loss of generality. Furthermore, all the random-selection protocols [Fei99, RZ01, ORV94], as well as all the Byzantine Agreement protocols we know have this property.

Our goal is to force the corrupted players in Π_{omit} to use coins drawn from a γ-SV source (as opposed to using coins with an arbitrary distribution). To do this, every time the underlying protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ asks a player P to sample a coin from the γ-SV random source, we run a $(1,\gamma)$ collective coin-flipping protocol Π_{coin} among all the players. All the players receive the output of the coin-flipping protocol with probability 1, and furthermore, the bias of the coin is at most γ. The player P is then supposed to use b as his coin.

In other words, we force all the players in Π_{omit} to use the outcome of the common-coin subroutine as the source of randomness. Now, each coin produced by the $(1,\gamma)$ collective coin-flipping Π_{coin} has a bias of at most γ and thus, the set of coins produced by the many executions of Π_{coin} form a γ-SV source.[3] Since all the players see the outcome of the coin-flipping protocol, they can then verify that P indeed used b as his coin in the protocol Π_{omit}.

However, it might be the case that the common-coin protocol Π_{coin} assumes reliable broadcast channels. If this is the case, we will instead run an audited version of Π_{coin}, where the recipient of the coin P acts as the auditor. The effect

[3]Note that the coins are not independent and the bias of each coin can adversarially depend on the previous coins and all the messages sent in the network.

of doing this is to limit the worst possible behavior of P to preventing some of the players from the seeing the outcome of the coin. But this is exactly the behavior of an omission fault, which the original protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ tolerates.

Putting together the above intuition together, our compiler proceeds in three steps.

Step 1. Convert the input protocol $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ to a protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ that tolerates *omission* faults, where in both $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ and $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$, *all* the players (including the corrupted ones) use a γ-SV source as the source of randomness.

Omission faults were introduced as an intermediate fault-model between fail-stop faults and Byzantine faults in the work of Neiger and Toueg. The difference between omission faults and fail-stop faults is that omission faults can continue omitting to send/receive messages in every round, whereas a fail-stop fault stops sending any message after it omits to send a message in some round.

Step 2. Convert $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ to a protocol Π_{omit}, where Π_{omit} tolerates omission faults where the faulty players are not required to use the prescribed random source. The honest players in Π_{omit}, however, are provided with a truly random source. In contrast, the input protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ worked only against adversaries that used the prescribed source of randomness (which is an γ-SV source).

Step 3. Convert Π_{omit} to Π_{byz} that tolerates Byzantine faults.

5.2.2 Construction of the Compiler

We prove the following theorem.

Theorem 5. Let $\gamma = O(t/n)$. Let $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ be any protocol that works against a fail-stop t-adversary and where all the players (including the dishonest ones) use an arbitrary γ-SV source as the source of randomness. There is a compiler that converts $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ into a protocol Π_{byz} that works against Byzantine adversary and where all the honest players use a uniformly random source.

If the round-complexity of $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ is r and the fault-tolerance is t, then the round-complexity of Π_{Byz} is $O(r \log^* n)$ and the fault-tolerance is $\min(t, n/3)$.

Proof. First instantiate the compiler in Lemma 14 as follows: Let the coin-flipping protocol Π_{coin} be the protocol of Feige (given in Table 4.1). This protocol runs in $O(\log^* n)$ rounds, has a fault-tolerance of $n/3$ and outputs a coin with bias $\gamma = O(t/n)$. Using this protocol as Π_{coin}, and putting together Lemma 14 and Lemma 15 gives us the statement of the theorem. ∎

Lemma 14. Assume that Π_{coin} is an r_{coin}-round $(1, \gamma)$ collective coin-flipping protocol tolerating an omission adversary that corrupts at most t_{coin} players. Then, there is a compiler that converts any r-round protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ that works

against an omission t-adversary to a protocol Π_{omit} that works against an omission $\min(t, t_{\mathsf{coin}})$-adversary. In Π_{omit}, the adversary can choose which coins to use in the protocol.

The round-complexity of Π_{omit} is $O(r \cdot r_{\mathsf{coin}})$.

Proof. Since the adversary is intrusive, we can without loss of generality assume that in every round, each player sends its coin-tosses to every other player.

Π_{omit} works exactly like $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$, except for the following:

> Whenever a player P_i in $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ is instructed to toss a coin, all the players together execute an audited protocol $\mathsf{Audit}(P_i, \Pi_{\mathsf{coin}})$, where Π_{coin} is an $(1, \gamma)$ collective coin-flipping protocol and P_i is the auditor in the protocol. Let the output (the coin) be b_i. P_i uses b_i as the coin in the underlying protocol.

This modification has the following effect.

1. When an honest player P_i is instructed to toss a coin in $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$, all the players run the collective coin-flipping protocol which results in a bit b_i. Thus, the coin b_i has a bias of at most γ, even conditioned on all the previous coin-tosses (by the randomness property of Π_{coin}). Furthermore, all the honest players see the outcome b_i, since the auditor P_i is honest.

   This simulates the effect of P_i sampling a coin b_i from a γ-SV source and sending the coin to all the players, which is exactly what happens in the execution of $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$.

2. When a faulty player P_i is instructed to toss a coin, in $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$, all the players run the collective coin-flipping protocol which results in a bit b_i, a coin with bias at most γ. However, since the auditor P_i is corrupted, only a subset S of the players see the outcome of the coin-toss.

   This simulates the effect of P_i sampling a coin b_i from a γ-SV source and sending the coin to the players in S, which is exactly what happens in the execution of $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$.

   Note that since the adversary in both cases is just an omission adversary, the corrupted player follows the instructions of the protocol, and in particular, will use b_i as the coin in the remainder of the protocol whenever it is instructed to do so.

The round-complexity of Π_{omit} is at most $O(r \cdot r_{\mathsf{coin}})$, since every round where a player in $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ is asked to sample a coin, the players execute an audited collective coin-flipping protocol which runs in $O(r_{\mathsf{coin}})$ rounds. The fault-tolerance of $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ is the minimum of the fault-tolerance of $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ and the fault-tolerance of the audited Π_{coin} protocol, which is $\min(t, n/3)$.

■

Lemma 15 ([NT90]). There are compilers that convert:

1. Any r-round protocol $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ that works against a fail-stop t-adversary to a protocol $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ that works against an omission $\min(t, \frac{n}{2})$-adversary. In both $\Pi_{\mathsf{fs}}^{\gamma-\mathsf{SV}}$ and $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$, all the players (including the dishonest ones) use a γ-SV source.

   The round-complexity of $\Pi_{\mathsf{omit}}^{\gamma-\mathsf{SV}}$ is $O(r)$.

2. Any r-round protocol Π_{omit} that works against an omission t-adversary to a protocol Π_{byz} that works against a Byzantine $\min(t, \frac{n}{3})$-adversary and runs in $O(r)$ rounds.

6

Extensions

In this chapter, we present various extensions to the main result in Chapter 4.

The first result in this vein deals with the trade-off between the fault-tolerance of the protocol and setup assumptions. The result of Pease, Shostak and Lamport [PSL80] shows that no Byzantine agreement protocol can achieve a fault-tolerance better than a third, with no prior setup. The most common setup assumption is that of a Public-key Infrastructure. In the setup of a PKI, each player P_i publishes a public-key PK_i, for which it holds the corresponding secret-key SK_i. In Section 6.1, we show how to construct a logarithmic-round BA protocol with a fault-tolerance of $t < \frac{n}{2}$ when there is a PKI setup is available. One of the drawbacks of this setting is that one has to assume a computationally bounded adversary in order to allow for cryptographic digital signatures. However, digital signatures are typically easier to construct and also complexity-theoretically simpler than public-key encryption schemes (which are necessary to implement private channels).

The second result shows a tradeoff between fault-tolerance and the round-complexity of the BA protocol. In contrast to the result in Chapter 4 which shows a logarithmic round BA protocol, we show how to achieve an expected round-complexity of $O(1)$: the price we pay is that the fault-tolerance of this protocol is only $n/\log^{O(1)} n$.

The third and final result shows a tradeoff between the fault-tolerance and the quality of the random source used in the BA protocol. In particular, we show how to run BA protocol of Chapter 4 with a γ-SV source for some $\gamma = \frac{c}{\log n}$ for some constant $c > 0$, given that the number of faults is a small constant fraction of the total number of players (where the constant depends on c). For details on the exact tradeoff, see Section 6.3.

6.1 A Trade-off between Fault-Tolerance and Trusted Setup

There are three factors that contribute to the fault-tolerance of the BA protocol of Chapter 4.

1. First, the fault-tolerance of the BA protocol can be no more than the fault-tolerance of the graded broadcast protocol used within.

2. Secondly, the fault-tolerance of the protocol is upper-bounded by $(\frac{1}{2} - \epsilon)n$ because of the following. Each honest player computes the verdict of a good committee as the verdict of the majority of the players within the committee. If the fraction of faulty players within the committee is more than a half, this could lead to different honest players computing different verdicts, even for a "good committee". Since the committee-selection process itself loses an ϵ factor in the fault-tolerance, this restricts the fault-tolerance of the BA protocol to at most $(\frac{1}{2} - \epsilon)n$, for some constant $\epsilon > 0$.

3. Thirdly, the fault-tolerance of the deterministic Byzantine agreement protocol that we run within the committees also upper-bounds the fault-tolerance of the entire protocol.

We now show how to modify each of the components in the protocol so as to achieve a fault-tolerance of $(\frac{1}{2} - \epsilon)n$. First, we replace the graded broadcast protocol invocation by an authenticated graded broadcast protocol, such as the one of Katz and Koo [KK06]. Authenticated graded broadcast achieves the same guarantees as graded broadcast, but tolerates $t < n/2$ dishonest players assuming a PKI setup among the players in the network. That is, each player P_i publishes a public-key PK_i for which he and only he knows the secret-key SK_i. In addition, here, we assume that the adversary is computationally unbounded so that it cannot forge digital signatures.

Secondly, we replace the reliable broadcast protocol within the committees by the protocol of Dolev and Strong [DS83] which assumes a PKI setup. This protocol achieves a fault-tolerance of $t < n/2$ and runs for $t + 1$ in a network of n players with t faults.

Modifying the BA protocols of Chapter 4 by replacing each invocation of graded broadcast by the authenticated graded broadcast protocol, and each invocation of Byzantine agreement by the Dolev-Strong protocol, we get the following theorem.

Theorem 6. For any constant $\epsilon > 0$, there exists a protocol BA_ϵ that achieves Byzantine Agreement in a synchronous full-information network of n players with a PKI setup, tolerating $t < (\frac{1}{2} - \epsilon)n$ faults. The round complexity of BA_ϵ is $O(\frac{\log n}{\epsilon^2})$, and the communication complexity is $\widetilde{O}(n^2)$ bits.

6.2 A Trade-off between Fault-Tolerance and Efficiency

We show two BA protocols that run in expected $O(1)$ rounds. First, we present a simple protocol with fault-tolerance $t = O(n/\log^2 n)$. Then, we show how to achieve a fault-tolerance of $t = O(n/\log^c n)$ for a constant $c < 1.58$ using a more sophisticated protocol.

A SIMPLE EXPECTED $O(1)$-ROUND PROTOCOL. The BA protocol is based on the following result of Ajtai and Linial [AL93], who show the existence of a 1-round collective coin-flipping protocol (in other words, a Boolean function) whose output cannot be influenced by any coalition of less than $\frac{n}{\log^2 n}$ players. For a string $\mathbf{x} \in \{0,1\}^n$ and $B \subseteq [n]$, let $\mathbf{x}_B$ denote the $|B|$-bit string formed by projecting $\mathbf{x}$ onto indices in B. For any $B \subseteq [n]$, we can thus write $\mathbf{x}$ as a pair $(\mathbf{x}_B, \mathbf{x}_{[n]\setminus B})$.

Theorem 7 (Ajtai-Linial [AL93]). There are constants $c, \epsilon > 0$ and a family of Boolean functions $\{f_n\}_{n=1}^{\infty}$ where $f_n : \{0,1\}^n \to \{0,1\}$ such that, for any set of variables $B \subseteq [n]$ of size at most $\frac{cn}{\log^2 n}$,

$$\epsilon \leq \Pr_{\mathbf{x}_{[n]\setminus B} \in_U \{0,1\}^{n-|B|}} [\exists \mathbf{x}_B \in \{0,1\}^{|B|} \text{ such that } f(\mathbf{x}_B, \mathbf{x}_{[n]\setminus B}) = 0] \leq 1 - \epsilon$$

This theorem directly gives a way to get a common coin with bounded bias, when $t = O(\frac{n}{\log^2 n})$: each player samples a random bit b_i, and sends it to all the players. The players compute $b = f(b_1, \ldots, b_n)$ as the common coin. Now, with high regardless of what the dishonest players do, the output of f is guaranteed to have a constant bias. Using the reduction of Ben-Or and Rabin, this gives us an expected $O(1)$-rounds BA.

AN EXPECTED $O(1)$-ROUND PROTOCOL WITH A BETTER FAULT-TOLERANCE. We construct an expected $O(1)$-*round* BA protocol that tolerates a Byzantine t-adversary for any $t = O(\frac{n}{\log^{1.58} n})$.

Theorem 8. There exists a BA protocol in a synchronous full-information network of n players tolerating $t = O(\frac{n}{\log^{1.58} n})$ Byzantine faults. The protocol runs in expected $O(1)$ rounds.

Proof. To construct the BA protocol, we will invoke Lemma 14 in Chapter 5. This lemma shows how to construct a BA protocol that runs in expected $O(1)$ rounds tolerating t faults given the following two ingredients.

1. a BA protocol that tolerates a fail-stop t-adversary where all the players (including the dishonest ones) use a γ-SV source as the source of randomness.

2. a collective coin-flipping protocol that tolerates t faults and generates a coin of bias at most γ. The protocol could possibly assume built-in reliable broadcast channels.

We now show how to construct these two ingredients.

BYZANTINE AGREEMENT PROTOCOL USING AN SV-SOURCE AGAINST FAIL-STOP FAULTS. Chor, Merritt and Shmoys [CMS89] constructed a simple *one-round* BA protocol against fail-stop faults, which uses a uniformly random source. Below, we show that the same protocol achieves BA even if the source of randomness is a $\frac{1}{2\log n}$-SV-source.

Lemma 16. There exists a BA protocol in a synchronous full-information network of n players tolerating a fail-stop t-adversary for $t < (1-\epsilon)n$ (for any $\epsilon > 0$). The protocol runs in expected $O(1)$ rounds, even if the randomness for the protocol is drawn from a γ-SV-source with $\gamma = O(\frac{1}{\log n})$.

Proof. As usual, we focus on constructing a *common-coin* protocol. The protocol proceeds as follows.

1. (Every player P_i) Sample $\log n$ random bits, and sends it to every other player.

2. (Every player P_i) If there is a unique P_j from which P_i received message 0, elect P_j as leader, else output $\perp$.

3. The leader flips a coin and sends it to everybody.

Let $\gamma = \frac{1}{2\log n}$. Let samp_i denote the $\log n$-bit string sampled by P_i from a γ-SV-source. Then, $\frac{1}{en} \leq (\frac{1}{2} - \frac{1}{2\log n})^{\log n} \leq \Pr[\mathsf{samp}_i = 0] \leq (\frac{1}{2} + \frac{1}{2\log n})^{\log n} \leq \frac{e}{n}$.

If *exactly* one of the *good* players P_i obtains $\mathsf{samp}_i = 0$ and *all* the bad players P_j obtain $\mathsf{samp}_j \neq 0$, then it is easy to see that all the honest players will choose an honest player as the leader. The probability that this happens is at least $(1-\frac{e}{n})^t \cdot \binom{n-t}{1}\frac{1}{en}(1-\frac{e}{n})^{n-t} \geq \kappa$ for some constant $\kappa > 0$. Thus, the protocol generates a (κ, γ)-common-coin in one round. ■

A COIN-FLIPPING PROTOCOL WITH BIAS $O(\frac{1}{\log n})$. Now, it suffices to design a coin-flipping protocol that generates a coin with bias at most $\frac{1}{2\log n}$. The first thought is to directly use the collective coin-flipping protocols of Feige [Fei99] or that of Russell-Zuckerman [RZ01]. However, these protocols do not guarantee that the bias of the coin generated is as small as $O(\frac{1}{\log n})$ (*even when* the number of faults is small). Thus, we construct a new collective coin-flipping protocol (assuming broadcast channels) that generates a coin with bias $\frac{1}{2\log n}$, when the number of faults $t = O(\frac{n}{\log^{1.58} n})$. More precisely,

Lemma 17. There is an $O(1)$-round $(1, \frac{1}{2\log n})$ collective coin-flipping protocol that tolerates $t < n/\log^{1.58} n$ faulty players.

Proof. First suppose that there exists a one-round collective coin-flipping protocol Π_{cc} such that for any Byzantine t-adversary, Π_{cc} generates a coin with bias $\frac{t}{n^\alpha}$. Then, we will show a $(1, \frac{1}{2\log n})$ collective coin-flipping protocol Π_{coin} against $t < \frac{n}{\log^\beta n}$ faults, for any $\beta > \frac{1}{\alpha}$. The protocol Π_{coin} assumes reliable broadcast channels, and it runs in $O(1)$ rounds.

Such a coin-flipping protocol Π_{cc} is guaranteed by the following result of Ben-Or and Linial.

Fact 1 ([BL85]). There exists a one-round $(1, \frac{t}{n^{0.63}})$ collective coin-flipping protocol among n players tolerating a Byzantine t-adversary.

Π_{coin} elects a committee (much like the Russel-Zuckerman committee selection), but it does so in three rounds. In the *first* round, elect a committee C_1 of size $\log^{1+\beta} n$ by running the Russell-Zuckerman committee-selection Π_{RZ} among n players. The probability that a bad committee is elected is at most $\frac{1}{n}$. In the second round, elect a committee C_2 of size $\log^\beta n \log\log n$ by running the Russell-Zuckerman committee-selection Π_{RZ} among the players in C_1. The probability that a bad committee is elected in this round is at most $\frac{1}{\log^{2+\beta} n}$. In the third round, run the one-round protocol Π_{coin} among the players in C_2.

The probability that C_2 is a bad committee is at most $\frac{1}{n} + \frac{1}{\log^{2+\beta} n} = o(\frac{1}{\log n})$. If C_2 is good, then running Π_{coin} generates a coin with bias at most $\frac{\log\log n}{(\log^\beta n \log\log n)^\alpha} \leq \frac{1}{3\log n}$. Thus, the total bias of the coin is at most $\frac{1}{3\log n} + o(\frac{1}{\log n}) < \frac{1}{2\log n}$. ∎

Putting together these two components gives us the BA protocol. ∎

6.3 A Trade-off between Fault-Tolerance and the Quality of Randomness

In this section, we show that the protocols of Chapter 4 work even if the random sources of the players are γ-SV sources with parameter $\gamma = O(\frac{1}{\log n})$. The only place where randomness is used in the protocols in Chapter 4 is in the committee-election protocols. Thus, it suffices to show that there is a protocol for committee election even when the players only have access to a γ-SV source. We show below that the protocol of Feige (see Table 4.1) does the trick.

Lemma 18. Let $\gamma = \frac{c}{\log n}$, and let the fraction of faults $\beta < 1 - \frac{5}{6}e^{2c}$ for some constant $c > 1$. There exists a committee-election protocol $\Pi^{\gamma-\mathsf{SV}}_{\mathsf{comm}}$ that outputs a committee S of size at most $2\log n$, such that with probability at least $1 - \frac{1}{n}$, the fraction of dishonest players in the elected committee is at most $1/3$.

$\Pi^{\gamma-\mathsf{SV}}_{\mathsf{comm}}$ uses a γ-SV source, tolerates t faulty players and runs in 1 round.

Proof. The protocol is exactly the committee-election protocol of Feige. The analysis proceeds in a very similar way to the proof of Lemma 2. The additional difficulties are caused due to the fact that the random variables used in the

analysis are not independent, which prevents us from using tail bounds such as the Chernoff bound. Fortunately for us, variants of Chernoff bound (such as the one of Schmidt, Siegel and Srinivasan [SSS95]) apply even in the case where the random variables are drawn from a distribution with bounded bias. We proceed formally below.

The main claim is that with probability at least $1 - \frac{1}{n}$, all the bins contain at least $2/3 \log n$ honest players. Given this claim, we are done, exactly as in the proof of Lemma 2.

Define the indicator random variables $X_{i,b}$ for $1 \leq i \leq n$ and $1 \leq b \leq \frac{n}{\log n}$ as follows.

$$X_{i,b} = \begin{cases} 1 & \text{if player } P_i \text{ chooses bin } b \\ 0 & \text{otherwise} \end{cases}$$

Since the honest players choose their bin using $\log n - \log\log n$ bits sampled from the γ-SV source, for every honest player P_i and every bin b,

$$(\frac{1}{2} - \gamma)^{\log n - \log\log n} \leq \mathbb{E}[X_{i,b}] \leq (\frac{1}{2} + \gamma)^{\log n - \log\log n}$$

That is, substituting the value of $\gamma = \frac{c}{\log n}$ and simplifying, we get

$$\frac{\log n}{n} e^{-2c} \leq \mathbb{E}[X_{i,b}] \leq \frac{\log n}{n} e^{2.1c}$$

Denote the number of honest players in a particular bin b by X_b. Then, X_b is simply the sum of all random variables $X_{i,b}$ for all honest players P_i. By linearity of expectation, the expected number of honest players in a bin b is bounded as follows.

$$(1 - \beta)e^{-2c} \log n \leq \mathbb{E}[X_b] \leq (1 - \beta)e^{2.1c} \log n$$

Even though the random variables $X_{i,b}$ are not independent, each $X_{i,b}$ has bounded bias given all the other $X_{i',b}$. Thus, we can use the result of [SSS95] to show that the probability that X_b is much smaller than the expectation is exponentially small in $\log n$. Setting the value of $\beta = 1 - 5/6e^{2c}$,

$$\Pr[X_b \leq 2/3 \log n] \leq 2 \cdot e^{-2 \log n}$$

By a union bound, the probability that some bin b contains less than $2/3 \log n$ honest players is at most $\frac{1}{n}$. ■

A

Combinatorial Tools

A.1 Construction of Committees

In this section, we give a probabilistic proof of the existence of a set of prospective committees, as required for the protocol of Russell and Zuckerman (see Chapter 4).

Lemma 19. There is a deterministic algorithm $\mathsf{Prospective}(1^n, \epsilon)$ which outputs a collection $\mathcal{C}$ of $m = O(n^2)$ committees C_i, where each C_i has $O(\log n/\epsilon^2)$ players such that: for every $\epsilon > 0$ and for any set B of at most βn dishonest players, $|\mathsf{Bad}_\epsilon(B, \mathcal{C})| \leq 3n$.

Proof. The proof is by probabilistic method. That is, we show that there exists such a collection of committees without giving an explicit way to find them.

Let n denote the number of players and βn be the number of bad players. Then, there exists a good collection of $m = n^2$ committees $\mathcal{C}$, where each committee is of size $n' = O(\frac{\log n}{\epsilon^2})$.

The collection of m committees is fixed by choosing m sets $C_1, C_2, \ldots, C_m$, each of size n', at random. Fix a "bad set" B of size t. For any committee C_i, the probability that the committee has more than $3n'\frac{t}{n}$ players from B is small. First of all, note that the expected size of $|C_i \cap B| = n'\frac{t}{n}$. By a simple Chernoff bound,

$$\Pr_{choice of C_i}[|C_i \cap B| > 3n'\frac{t}{n}] < e^{-\frac{n't}{n}}$$

By the choice of $n' = \frac{n \log n}{t}$,

$$\Pr_{choice of C_i}[|C_i \cap B| > 3n'\frac{t}{n}] < \frac{1}{n}$$

The expected number of bad committees is therefore at most $\frac{m}{n}$, which by the choice of m, is n. The probability that the number of bad committees is larger

than $3n$ is, again by a Chernoff bound, at most e^{-n}. This analysis was done for a fixed B. That is, we just showed that,

$$\forall B, \Pr[\text{ the number of bad committees w.r.t } B \text{ is more than } 3n] < e^{-n}$$

Now, reverse quantifiers. Use a union bound over the set of all B's. Since there exist $\binom{n}{t} \leq 2^n$ such B's,

$$\Pr[\exists B \text{ the number of bad committees w.r.t } B \text{ is more than } 3n] < e^{-n}2^n < 1$$

Therefore, there exists a choice of n^2 committees, each of size $\frac{n \log n}{t}$, such that no more than $3n$ of them are bad. ■

We obtain the following corollaries of this lemma.

Corollary 1. Let the number of players be n and the number of faults t be $\frac{n}{\log^{\beta} n}$. Then, there exists a set of n^2 committees, each of size $\log^{1+\beta} n$ such that at most $3n$ of these have more than $3 \log n$ bad players.

Corollary 2. Let the number of players be $\log^{1+\beta} n$ and the number of faults t be $3 \log n$. Then, there exists a set of $\log^{2+2\beta} n$ committees, each of size $O(\log^{\beta} n \log \log n)$ such that at most $3 \log^{1+\beta} n$ of the committees have more than $O(\log \log n)$ bad players.

HITTING SETS. To define the notion of a hitting set for combinatorial rectangles, we first need a few definitions.

Fix an $a > 0$. A *combinatorial rectangle* $R \subseteq [a]^n$ is *defined to be* $R = R_1 \times R_2 \times \ldots \times R_n$ where each $R_i \subseteq [a]$, and $|R_i| = a - 1$. The volume of R in $[a]^n$ is $\mathsf{vol}(R) = \frac{1}{a^n} \prod_{i=1}^{n} |R_i| = (1 - \frac{1}{a})^n$. We also let $\mathsf{vol}(R)$ be compactly denoted as δ.

A set $D \subseteq [a]^n$ is called an (a, n, m)-hitting set if (1) $|D| = m$, and (2) for every combinatorial rectangle $R \subseteq [a]^n$, $|D \cap R| > 0$. A pertinent question is for what values of the parameters a and n does there exist an (a, n, m)-discrepancy set. The following lemma answers this question. The proof is by a simple application of probabilistic method.

Lemma 20. There exists an (a, n, m)-hitting set D for every n, $m = n^2$ and $a = \frac{2n}{\log n}$.

Proof. The proof is by probabilistic method. $D = \{d_1, d_2, \ldots, d_m\}$ is chosen to be m random points in $[a]^n$. Fix a combinatorial rectangle R. The probability that $d_1 \in R$ is

$$\Pr_{\mathsf{choiceofD}}[d_1 \in R] = \delta$$

Thus, the expected number of points d_i that are in R is δm. The probability that the number of points d_i in R is less than $(1 - \epsilon)\delta m$ is

$$\Pr[|R \cap D| \leq (1 - \epsilon)\delta m] < e^{-\epsilon^2 \delta m}$$

This analysis was done for a fixed R. Now, reverse quantifiers.

$$\Pr[\exists R \text{ such that } |R \cap D| \leq (1-\epsilon)\delta m] \leq \text{ (number of rectangles) } e^{-\epsilon^2 \delta m}$$

The total number of rectangles is a^n. Thus

$$\Pr[\exists R \text{ such that } |R \cap D| \leq (1-\epsilon)\delta m] < a^n e^{-\epsilon^2 \delta m} < 1$$

by the choice of a and m. Thus, in particular, there exists a D such that for every rectangle R, $|D \cap R| \geq (1-\epsilon)|D| > 2n+1$. ■

Bibliography

[AL93] Miklós Ajtai and Nathan Linial. The influence of large coalitions. *Combinatorica*, 13(2):129–145, 1993.

[BB98] Ziv BarJoseph and Michael Ben-Or. A tight lower bound for randomized synchronous consensus. In *PODC*, pages 193–199, 1998.

[Ben83] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols (extended abstract). In *PODC*, pages 27–30, 1983.

[BG89a] Piotr Berman and Juan A. Garay. Asymptotically optimal distributed consensus (extended abstract). In *ICALP*, pages 80–94, 1989.

[BG89b] Piotr Berman and Juan A. Garay. Asymptotically optimal distributed consensus (extended abstract). In *ICALP*, pages 80–94, 1989.

[BGP92] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. Optimal early stopping in distributed consensus (extended abstract). In *WDAG*, pages 221–237, 1992.

[BGW88] Michael Ben-Or, Shafi Goldwasser, and Avi Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract). In *STOC*, pages 1–10, 1988.

[BL85] Michael Ben-Or and Nathan Linial. Collective coin flipping, robust voting schemes and minima of banzhaf values. In *FOCS*, pages 408–416, 1985.

[Blu81] Manuel Blum. Coin flipping by telephone. In *CRYPTO*, 1981.

[BPV06] Michael Ben-Or, Elan Pavlov, and Vinod Vaikuntanathan. Byzantine agreement in the full-information model in o(log n) rounds. In *STOC*, pages 179–186, 2006.

[Bra84] Gabriel Bracha. An asynchronous $\lceil (n-1)/3 \rceil$-resilient consensus protocol. In *PODC*, pages 154–162, 1984.

[Bra87] Gabriel Bracha. An $O(\log n)$ expected rounds randomized byzantine generals protocol. *J. ACM*, 34(4):910–920, 1987.

[CC85] Benny Chor and Brian A. Coan. A simple and efficient randomized byzantine agreement algorithm. *IEEE Trans. Software Eng.*, 11(6):531–539, 1985.

[CCD88] David Chaum, Claude Crépeau, and Ivan Damgård. Multiparty unconditionally secure protocols (extended abstract). In *STOC*, pages 11–19, 1988.

[CD89] Benny Chor and Cynthia Dwork. Randomization in byzantine agreement. 5:443–497, 1989.

[CMS89] Benny Chor, Michael Merritt, and David B. Shmoys. Simple constant-time consensus protocols in realistic failure models. *J. ACM*, 36(3):591–614, 1989.

[CR87] Benny Chor and Michael O. Rabin. Achieving independence in logarithmic number of rounds. In *PODC*, pages 260–268, 1987.

[CR93] Ran Canetti and Tal Rabin. Fast asynchronous byzantine agreement with optimal resilience. In *STOC*, pages 42–51, 1993.

[DS83] Danny Dolev and H. Raymond Strong. Authenticated algorithms for byzantine agreement. *SIAM J. Comput.*, 12(4):656–666, 1983.

[DSS90] Cynthia Dwork, David B. Shmoys, and Larry J. Stockmeyer. Flipping persuasively in constant time. *SIAM J. Comput.*, 19(3):472–499, 1990.

[Fei99] Uriel Feige. Noncryptographic selection protocols. In *FOCS*, pages 142–153, 1999.

[FL82] Michael J. Fischer and Nancy A. Lynch. A lower bound for the time to assure interactive consistency. *Inf. Process. Lett.*, 14(4):183–186, 1982.

[FLP83] Michael J. Fischer, Nancy A. Lynch, and Mike Paterson. Impossibility of distributed consensus with one faulty process. In *PODS*, pages 1–7, 1983.

[FM88] Paul Feldman and Silvio Micali. Optimal algorithms for byzantine agreement. *Symposium on the Theory of Computing*, pages 148–161, 1988.

[FM97] Pesech Feldman and Silvio Micali. An optimal probabilistic protocol for synchronous byzantine agreement. *SIAM J. Comput.*, 26(4):873–933, 1997.

[GGL98] Oded Goldreich, Shafi Goldwasser, and Nathan Linial. Fault-tolerant computation in the full information model. *SIAM J. Comput.*, 27(2):506–544, 1998.

[GM98] Juan A. Garay and Yoram Moses. Fully polynomial byzantine agreement for n $>$ 3t processors in $t + 1$ rounds. *SIAM J. Comput.*, 27(1):247–290, 1998.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC*, pages 218–229, 1987.

[GPV06] Shafi Goldwasser, Elan Pavlov, and Vinod Vaikuntanathan. Fault-tolerant distributed computing in full-information networks. In *FOCS*, pages 15–26, 2006.

[GSV05] Shafi Goldwasser, Madhu Sudan, and Vinod Vaikuntanathan. Distributed computing with imperfect randomness. In *DISC*, pages 288–302, 2005.

[GVZ06] Ronen Gradwohl, Salil Vadhan, and David Zuckerman. Random selection with an adversarial majority. *ECCC Report TR06-026*, 2006.

[Had83] Vassos Hadzilacos. Byzantine agreement under restricted types of failures. *Technical Report 18-83, Department of Computer Science, Harvard University*, 1983.

[KK06] Jonathan Katz and Chiu-Yuen Koo. On expected constant-round protocols for byzantine agreement. *ECCC Report TR06-028*, 2006.

[KLRZ08] Yael Tauman Kalai, Xin Li, Anup Rao, and David Zuckerman. Network extractor protocols. In *FOCS*, pages 654–663, 2008.

[KSSV06a] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Scalable leader election. In *SODA*, pages 990–999, 2006.

[KSSV06b] Valerie King, Jared Saia, Vishal Sanwalani, and Erik Vee. Towards secure and scalable computation in peer-to-peer networks. In *FOCS*, pages 87–98, 2006.

[KY86] Anna Karlin and Andrew Chi-Chih Yao. Probabilistic lower bounds for byzantine agreement. *Manuscript*, 1986.

[LLSZ97] Nathan Linial, Michael Luby, Michael E. Saks, and David Zuckerman. Efficient construction of a small hitting set for combinatorial rectangles in high dimension. *Combinatorica*, 17(2):215–234, 1997.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1996.

[Nis96] Noam Nisan. Extracting randomness: How and why a survey. In *IEEE Conference on Computational Complexity*, pages 44–58, 1996.

[NT90] Gil Neiger and Sam Toueg. Automatically increasing the fault-tolerance of distributed algorithms. *J. Algorithms*, 11(3):374–419, 1990.

[ORV94] Rafail Ostrovsky, Sridhar Rajagopalan, and Umesh V. Vazirani. Simple and efficient leader election in the full information model. In *STOC*, pages 234–242, 1994.

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM.*, 27:228–234, 1980.

[Rab83] Michael O. Rabin. Randomized byzantine generals. *FOCS*, pages 403–409, 1983.

[RZ01] Alexander Russell and David Zuckerman. Perfect information leader election in $\log^* n + O(1)$ rounds. *JCSS*, 63(4):612–626, 2001.

[Sak89] Michael E. Saks. A robust noncryptographic protocol for collective coin flipping. *SIAM J. Discrete Math.*, 2(2):240–244, 1989.

[SSS95] Jeanette P. Schmidt, Alan Siegel, and Aravind Srinivasan. Chernoff-hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math*, 8:331–340, 1995.

[SV84] M. Santha and U. V. Vazirani. Generating quasi-random sequences from slightly-random sources. In *FOCS*, pages 434–440, Singer Island, 1984.

[Zuc97] David Zuckerman. Randomness-optimal oblivious sampling. *Random Struct. Algorithms*, 11(4):345–367, 1997.