Distributed Consensus in the Presence of Sectional Faults

S. Amitanand* I. Sanketh* K. Srinathan†* V. Vinod* C. Pandu Rangan‡*

ABSTRACT

Consider a synchronous network of n players, each with a local input. The goal of *distributed consensus* is to globally agree on one of the valid inputs even if some non-trivial subset of the players are *faulty*. By valid input, we mean the input of any non-faulty player. Extant results in Byzantine agreement literature capture the behaviour of faulty players in an "all-or-nothing" fashion. For instance, a (Byzantine) faulty player is completely unconstrained and could behave differently with different players. This leads to a gross underestimation of the achievable fault-tolerance. In this work, we propose a fault-model that considerably improves the estimation of fault-tolerance and helps capture real-life scenarios better. For instance, if two (honest) players were part of the same LAN (which is essentially a broadcast network), it is impossible for a external faulty player to behave differently with these two players (though the faulty player may behave with "equal" malice with both these players!). Among our results, we introduce the *sectional* fault-model that is more general and can capture practical scenarios not captured by any extant model. We provide a complete characterization of the tolerable faults and present efficient protocols to achieve consensus. We remark that the results of this paper strictly generalize the extant characterizations of fault-tolerance. For example, consider a network of four players P_1, P_2, P_3 and P_4, under the corrupting influence of a Byzantine adversary given by the adversary structure $\mathcal{A} = \{(P_1, P_2), (P_2, P_3), (P_4)\}$. Agreement is *impossible* in such a scenario, since the three sets from $\mathcal{A}$ cover the player set. However, it would be evident from our results that consensus in the above scenario was indeed possible if (and only if) the players P_1, P_3 and P_4 belonged to a single LAN in the network!

Keywords: distributed consensus, sectional faults

1. INTRODUCTION

The problem of *consensus* is a classic problem in distributed computing. In many practical situations, it is necessary for a group of processes in a distributed system to reach agreement on some issue, despite the presence of some faulty processes. Formally, a protocol among a group of players (some of which may be faulty), each having a value (from $\{0, 1\}$), is said to achieve consensus, if, at the end of the protocol, all honest players agree on a value and the following conditions hold:

1. *Agreement:* All honest players agree on the same value.
2. *Validity:* If all honest players start with the value v, then all honest players agree on v.
3. *Termination:* All honest players eventually agree.

There exists a rich literature in this field. The players' distrust in each other and in the underlying synchronous network is usually modeled via a centralized *adversary* that has control over some of the players. Many different adversary models have been considered. These approaches can be classified according to a number of criteria. Some prominent ones are briefly discussed below.

1. *Computational resources:* The adversary may be computationally limited (to probabilistic polynomial time) as in [5, 14] or computationally unbounded as in [14, 12].
2. *Control over corrupted players:* If the adversary can only stop the working of any of the corrupted players, then it is a *fail-stop* adversary [13](or an *omission* adversary if the crashed player may subsequently recover). A *Byzantine* or *active* adversary [12] can also take complete control of the corrupted players and alter the behaviour of the corrupted players in an arbitrary fashion. Mixed adversaries have also been considered [10, 1].
3. *Mobility:* A *static* adversary decides on the set of players that it would corrupt before the protocol begins its execution, unlike an *adaptive* adversary [4] which

†Financial support from Infosys Technologies Limited, India, is acknowledged.

‡This author would like to thank Prof. Kwangjo Kim, ICU, South Korea, for several helpful discussions. The financial support from the Ministry of Information Technology, Government of South Korea is also warmly acknowledged.

*Department of Computer Science and Engineering, Indian Institute of Technology, Madras, Chennai-600036, INDIA. e-mail:{anand,sanketh,ksrinath,vinodv}@cs.iitm.ernet.in, prangan@iitm.ac.in

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
PODC'03, July 13–16, 2003, Boston, Massachusetts, USA.
Copyright 2003 ACM 1-58113-708-7/03/0007...$5.00.

chooses the players to corrupt as the protocol execution proceeds, or a *mobile* adversary [15, 9] where different set of players are corrupted at different times during the protocol run-time.

4. *Corruption capacity:* In the *threshold* specification, like in [12], the number of corrupted players, at any given time, is limited to at most t. In case of the *non-threshold* specification, like in [7], an *adversary structure*, that is a monotone set of subsets of player set, is used wherein the adversary is permitted to corrupt the players of any one set in the adversary structure.

In this paper, we postulate an adversary that is computationally unbounded, Byzantine, static and non-threshold. In this adversarial setting, it has been proved [7] that distributed consensus among n players $\mathcal{P} = \{P_1, \ldots, P_n\}$, where every pair of players is connected by a point-to-point communication channel,[1] tolerating an adversary characterized by the adversary structure $\mathcal{A}$ is possible if and only if $\mathcal{A}$ satisfies the property $\mathcal{Q}^{(3)}$, that is, no *three* sets in $\mathcal{A}$ cover $\mathcal{P}$.[2] As a special instance, the necessary and sufficient condition for the threshold adversary case will be $t < \frac{n}{3}$ [14].

A natural question that arises is the following: *how can the fault-tolerance bound of $\mathcal{Q}^{(3)}$ be improved?* There exist many (independent) approaches to achieving the same. Below, we list three of the prominent approaches used in the literature.

1. Assume additional primitives more powerful than just point-to-point channels.

2. Use a stronger adversary model that gives some additional "knowledge" about the fault in the system.

3. Suitably relax the constraints of the *consensus* problem itself.

There exists enough evidence in the literature that all the above approaches are quite powerful and can *strictly* improve the fault-tolerance. For example, consider the threshold case bound of $t < \frac{n}{3}$. Using the first approach, [8] showed that in the presence of a broadcast channel among every three players, the bound can be improved to $t < \frac{n}{2}$. The second approach was used in [7, 11]; infact, their non-threshold adversarial model helped improve the $t < \frac{n}{3}$ bound to $\mathcal{Q}^{(3)}$.[3] As a part of this work, we use the approach to in turn improve the $\mathcal{Q}^{(3)}$ bound. The third approach was adopted in [6], thus proving that "detectable" Byzantine agreement was possible with $t < n$, again significantly improving the $t < \frac{n}{3}$ bound. Note that in each case, there is a strict improvement in the fault-tolerance.

In this paper, we use a blend of the first and second approaches. More precisely, the additional "knowledge" about the faults in the system is derived from the properties of the underlying synchronous communication network. Informally, a faulty player cannot "lie differently" to the players in each of the specified subset(s). As far as we know, this line of research has received little attention. Related (though only slightly) work is the one by Babaoglu and Drummond [2] who consider the Byzantine agreement problem in faulty broadcast networks, as part of which they develop simple two-round protocols for *partition faults*, in the threshold adversarial model. Partition faults are broadcast networks with two partitions of the player set – one in which all receive the same value and others who do not.

[1] The point-to-point channel is assumed to be authenticated.

[2] More generally, an adversary structure $\mathcal{A}$ is said to satisfy the property $\mathcal{Q}^{(k)}$ if no k sets in $\mathcal{A}$ cover $\mathcal{P}$. This is denoted by $\mathcal{Q}^{(k)}(\mathcal{P}, \mathcal{A})$.

[3] Note that there are exponential number of adversary structures that satisfy $\mathcal{Q}^{(3)}$ but are impossible to tolerate using a threshold protocol.

2. THE SECTIONAL FAULT MODEL

In this section, we define our adversarial model. We first define the *Basic Sectional Fault* model, followed by the *Sectional Fault* model and finally the *Generalized Sectional Fault* model. In each case, we begin with motivation(s) for the model and subsequently give the intuition behind the fault abstraction and formally define the adversary.

Consider the network in Figure 1 under the corrupting influence of the active adversary given by the adversary structure $\{(a_1, a_2), (b_1, b_2), (c_1, c_2)\}$ (throughout this paper we abuse "adversary structure" to mean the maximal basis of the adversary structure). Clearly, $\mathcal{Q}^{(3)}$ is not satisfied; thus no known protocol can achieve consensus. However, we note that the very character of the network presents a feature that can aid the consensus protocol. If we assume that every player on a LAN can listen to all messages that are received at the LAN (essentially making a LAN a local broadcast channel), a faulty player reporting a value to all the other players in the player set cannot quote different values to the players on this LAN (without being identified as a faulty player). In fact, we shall show (see Theorem 3.2) that consensus in this network is actually possible! Intuitively, it

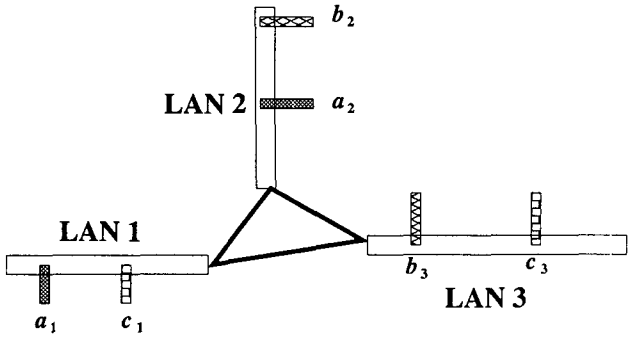

Figure 1: Example Network

is clear that all the players in the same LAN comprise an equivalence class with respect to the received value. Thus, we model the adversary as follows.

DEFINITION 2.1 (BASIC SECTIONAL ADVERSARY). *The adversary is characterized by an ordered pair $(\mathcal{A}, \Pi)$, where $\mathcal{A}$ is the adversary structure as in [11, 7] and Π (hereafter called the* partition*) is a partition of the player set. An adversary can corrupt any one set from $\mathcal{A}$. All players in one set of Π can listen to messages received by any player of that set.*[4]

Evidently, the Basic Sectional Adversary model is based on the first approach, namely, of using more powerful primitives than just point-to-point channels. However, it is also possible to view the Basic Sectional Adversary model to be

[4] This implies that the adversary cannot send different values to the players of a set in Π without being detected.

based on the second approach, that is, using a stronger adversary model that gives additional knowledge about the fault in the system; more precisely, the additional knowledge is that the adversary cannot send different values to the players of a set in Π without being detected.

In more general scenarios, the additional knowlegde could be sender dependent; i.e., the adversary cannot send different values to the players of a set in Π without being detected, where the set Π is depends on the sender's identity.[5] In order to incorporate the above discussed "locality" of partitions, we augment the previous model and extend the idea of a "global" partition to a player-wise view of the partition as follows. We associate with each player P_i an independent partition of the player set Π_i. Informally, each of the players P_i, $i = 1, \ldots, n$ behaves as if the "global" partition was Π_i, $i = 1, \ldots, n$ respectively.

DEFINITION 2.2 (SECTIONAL ADVERSARY). *The adversary is characterized by an ordered pair $(\mathcal{A}, \vec{\Pi})$, where $\mathcal{A}$ is the adversary structure as in [11, 7] and $\vec{\Pi} = \langle \Pi_1, \Pi_2, \ldots, \Pi_n \rangle$, with Π_k being a partition of the player set. An adversary can corrupt any one set from $\mathcal{A}$. All players in one set of Π_k can listen to messages sent by player k to any player of that set.*

In the final extension to the fault model, we incorporate uncertainty about the fault into the model. On the lines of [11], we model possibly incomplete knowledge of the partition by a set of partitions.

DEFINITION 2.3 (GENERALIZED SECTIONAL ADVERSARY). *The generalized adversary is characterized by an ordered pair $(\mathcal{A}, \vec{\psi})$, where $\mathcal{A}$ is the adversary structure as in [11, 7] and $\vec{\psi} = \langle \psi_1, \psi_2, \ldots, \psi_n \rangle$, with ψ_k being a collection of partitions of the player set. An adversary can corrupt any one set from $\mathcal{A}$ and choose one partition from each ψ_k, which defines the Sectional adversary description for this run.*[6]

The organization of the rest of the paper is as follows. In Section 3, we completely characterize the tolerable adversaries in the Sectional and the Generalized Sectional adversary models. To demonstrate sufficiency, we present a simple (but inefficient) protocol. Subsequently, in Sections 5 and 6, efficient (but more complex) protocols are presented. The results for the basic Sectional adversary follow as a special case of those of the Sectional adversary.

3. CHARACTERIZATION OF DISTRIBUTED CONSENSUS

THEOREM 3.1 (TOP-LEVEL CHARACTERIZATION). *In any (of the three) adversary model(s),* consensus *tolerating $(\mathcal{A}, X)$ is possible*[7] *if and only if* consensus *is possible tolerating $(\mathcal{A}', X)$ for every $\mathcal{A}'$ such that $\mathcal{A}' \subseteq \mathcal{A}$ and $|\mathcal{A}'| = 3$.*

[5] In fact, the set Π could depend on both the sender's and the receiver's identity. We consider this in the full version of the paper.

[6] Technically, the notion of a partition is essential only for corrupt players; a honest player will anyway not "lie", leave alone "lie differently". Thus, without any loss in correctness, we can have the adversary to choose a partition for all the players (rather than just the corrupted players).

[7] The variable X could vary depending on the model, but it does not matter.

Proof : In the "$\Longrightarrow$" direction, the statement is trivially true. For the "$\Longleftarrow$" direction, we prove by induction on $|\mathcal{A}|$ that a protocol for agreement tolerating $(\mathcal{A}, X)$ always exists provided *consensus* is possible tolerating $(\mathcal{A}', X)$ for every $\mathcal{A}'$ such that $\mathcal{A}' \subseteq \mathcal{A}$ and $|\mathcal{A}'| = 3$. The proof is based on the *player simulation* idea of [11]. The basis case is when $|\mathcal{A}| = 3$. In this case, $\mathcal{A}' = \mathcal{A}$ and hence a protocol exists. Now, assume that the theorem is true if $|\mathcal{A}| < k$. Consider the case where $|\mathcal{A}| = k$. Let (Y_1, Y_2, Y_3, Y_4) be a four-partition of $\mathcal{A}$, with $|Y_i| > 0$, $i = 1, 2, 3, 4$. Then, the four structures $\mathcal{A}_1$, $\mathcal{A}_2$,$\mathcal{A}_3$ and $\mathcal{A}_4$, where $\mathcal{A}_i = \mathcal{A} \setminus Y_i$, $i = 1, 2, 3, 4$, are such that each of them is strictly smaller than k. Thus, from the induction hypothesis, there exist four consensus protocols $\Gamma_1, \Gamma_2, \Gamma_3$ and Γ_4 tolerating $(\mathcal{A}_1, X)$, $(\mathcal{A}_2, X)$,$(\mathcal{A}_3, X)$ and $(\mathcal{A}_4, X)$ respectively. Next, one can construct a consensus protocol among four "virtual" players that tolerates one active fault (using standard Byzantine Agreement protocols for the threshold fault model [3]). Then, the four virtual players can be simulated by the recursively constructed protocols $\Gamma_1, \Gamma_2, \Gamma_3$ and Γ_4 respectively. Note that, by our choice of $\mathcal{A}_1$, $\mathcal{A}_2$,$\mathcal{A}_3$ and $\mathcal{A}_4$, each $Z \in \mathcal{A}$ is guaranteed to occur in three of the four $\mathcal{A}_i$'s, assuring the honesty of at least three of the four virtual players. Thus the constructed protocol achieves consensus. □

3.1 On Tolerable Sectional Adversaries

In this subsection, we present the characterization of tolerable Sectional adversaries $(\mathcal{A}, \vec{\Pi})$ where $|\mathcal{A}| = 3$. This result, along with Theorem 3.1, complete the required necessary and sufficient conditions.

First, we define the notion of *verifiability* of an adversary set w.r.t an adversary structure. We use the notation $U(A, \mathcal{A})$ to denote the set of players who belong to A and who do not belong to any other adversary set in $\mathcal{A}$. We write $U(A)$ if the adversary structure is clear from the context.

DEFINITION 3.1 (VERIFIABILITY). *A Sectional adversary set A $(\in \mathcal{A})$ is said to be* verifiable *w.r.t $(\mathcal{A}, \vec{\Pi})$ if there exists a player i in $U(A)$ such that for some set π in Π_i and $\forall A' \in \mathcal{A} \setminus \{A\}$, $\pi \cap U(A') \neq \emptyset$.*

Note that for a set to be verifiable w.r.t a Sectional adversary structure consisting of more than two sets, the adversary structure should satisfy $\mathcal{Q}^{(2)}$.

THEOREM 3.2. *Consensus tolerating the adversary structure $(\mathcal{A}, \vec{\Pi})$, where $\mathcal{A} = \{A_1, A_2, A_3\}$, is possible if and only if $A_1 \cup A_2 \cup A_3 \neq \mathcal{P}$ or each A_i is verifiable w.r.t. $(\mathcal{A}, \vec{\Pi})$.*

Proof of the necessary condition:

We now prove the necessary part of the characterization, i.e. we show that if $A_1 \cup A_2 \cup A_3 = \mathcal{P}$ and some A_i is not verifiable w.r.t. $(\mathcal{A}, \vec{\Pi})$, consensus is impossible. We assume that such a protocol exists and obtain a contradiction. Informally, the proof aims to construct three *scenarios* of protocol execution by defining the player inputs and behaviour in each case. Before proceeding with the proof, we first introduce some notation. We denote the messages sent from player x to player y in round r as $M^r_{xy}(\delta)$ where δ is the scenario under consideration. We also let $\mathcal{M}^r_x(\delta)$ denote the ordered list of all the messages sent to the player x through rounds $1, 2, \ldots, r$.

We define two scenarios X and Y to be *indistinguishable* with respect to player x after r rounds if $\mathcal{M}_x^r(X) = \mathcal{M}_x^r(Y)$ and player x's initial values in X and Y are the same.

We first deal with the case when $\mathcal{Q}^{(2)}(\mathcal{A})$ does not hold. In this case, it is clear that agreement is not possible satisfying the validity condition. In what follows, we assume that $\mathcal{Q}^{(2)}(\mathcal{A})$ holds and therefore, $U(A_1), U(A_2)$ and $U(A_3)$ are all non-empty. Without loss of generality, we suppose that A_2 is not verifiable. We now describe three scenarios, α, β and γ of protocol execution and show that the requirements of consensus in these three scenarios imply a contradiction to the existence of a protocol. For each scenario, we specify the behaviour of players belonging to each adversary set.

1. **Scenario α:** In this scenario, the adversary corrupts players belonging to the adversary set A_3. Players in $A_1 \setminus A_3$ and $A_2 \setminus A_3$ are all honest and start with input 0.

2. **Scenario β:** The adversary corrupts players belonging to the adversary set A_2. Players in $U(A_3)$ have input 1 while players in $A_1 \setminus A_2$ have input 0.

3. **Scenario γ:** The adversary corrupts players belonging to the adversary set A_1. Players in $A_3 \setminus A_1$ and $A_2 \setminus A_1$ are honest and start with the input value 1.

We now describe the adversary strategy for each of the scenarios.

1. In scenario α, the faulty players send the same messages as in scenario β. Formally, $\forall r\ M_{zx}^r(\alpha) = M_{zx}^r(\beta)$, $z \in A_3$.

2. In scenario γ, the faulty players send the same messages as in scenario β. Formally, $\forall r\ M_{zx}^r(\gamma) = M_{zx}^r(\beta)$, $z \in A_1$.

3. In scenario β, the adversary corrupts players belonging to the adversary set A_2. In their communication with players in $U(A_1)$ the faulty players belonging to $U(A_2)$ send the same messages that were sent by them in scenario α. In their communication with players in $U(A_3)$ they send the same messages as in scenario γ. Note that since A_2 is not verifiable, there does not exist a scenario, in which a player belonging to $U(A_2)$ is constrained to send the same value to a player each in $U(A_1)$ and $U(A_3)$. Formally, $\forall r\ M_{zx}^r(\beta) = M_{zx}^r(\alpha)$, $z \in A_2$ and $x \in A_1$.

4. Players belonging to more than one adversary set send the same messages as in the scenario in which they are honest. Therefore, they send the same messages in all the three scenarios.

5. In their communication with players belonging to more than one adversary set, faulty players send messages with the sole constraint that they be consistent with the possible existence of any partitions.

We complete the proof by separately proving the following statement: No protocol can achieve agreement in all three scenarios (see Lemma 3.3).

Evidently, the above statement completes the contradiction to the assumption that consensus is possible if the necessary condition does not hold. This, therefore, completes the proof of the necessary part of Theorem 3.2. □

We show the following two lemmas as a prelude to proving Lemma 3.3.

LEMMA 3.1. *The two scenarios, α and β are indistinguishable to any player belonging to $U(A_1)$.*

Proof : Players belonging to $U(A_1)$ start with the same value in both the scenarios α and β. Hence they behave similarly in the first round of both the scenarios. By the specified adversarial strategy, players belonging to A_2 and A_3 send the same messages to players in $U(A_1)$ in both the scenarios. By induction on the number of rounds, it follows that $U(A_1)$ receive the same messages in both the scenarios, i.e. $\forall r\ M_x^r(\alpha) = M_x^r(\beta),\ x \in U(A_1)$. □

LEMMA 3.2. *The two scenarios, β and γ are indistinguishable to any player belonging to $U(A_3)$.*

Proof: Similar to the proof of Lemma 3.1. □

LEMMA 3.3. *No protocol can achieve agreement in all three scenarios.*

Proof: Suppose there exists a protocol which achieves agreement in all the three scenarios. From the validity condition for agreement, it follows that the honest players in each scenario, must agree on the value 0 in scenario α, and the value 1 in scenario γ. Players in $U(A_1)$, who behave honestly in both the scenarios, α and β perceive both these scenarios to be indistinguishable (Lemmas 3.1 and 3.2), hence decide on the same value in both the scenarios, viz, 0. Similarly, the players in $U(A_3)$ decide on the same value in scenarios β and γ, namely 1. This contradicts the agreement condition for consensus in scenario β. □
This completes the proof of necessity for Theorem 3.2.

Proof of sufficiency (for Theorem 3.2):

We now give a protocol to achieve consensus whenever the necessary condition of Theorem 3.2 is satisfied and then prove the correctness of the protocol.

We first consider the case when $A_1 \cup A_2 \cup A_3$ does not cover $\mathcal{P}$. In this case, there always exists a honest player in $\mathcal{P} - (A_1 \cup A_2 \cup A_3)$ and the protocol trivially follows.

We now consider the case when $A_1 \cup A_2 \cup A_3 = \mathcal{P}$ and each of the sets A_1, A_2 and A_3 is verifiable w.r.t the adversary structure A_1, A_2, A_3. In this case, corresponding to each adversary set, there exists a triplet of players, one in each adversary set, one of which is constrained to send the same value to the other two. The sender is called the *representative*. These set of nine players are denoted by $p_{i,j}$, $1 \leq i, j \leq 3$, such that $p_{1,j}, p_{2,j}$ and $p_{3,j}$ belongs to A_j. For each i, we require that $p_{i,j} \in U(A_j)$. The representative of A_i, is the player $p_{i,i}$. All other (six) players are called *verifiers*. The representative $p_{i,i}$, is constrained to send the same value to the other two players (verifiers), viz, $p_{i,j},\ j \neq i$. We call this ordered tuple of nine players, consisting of the three representatives and the six verifiers, the *committee* of this adversary structure: i.e. the set of real players that help achieve agreement tolerating this adversary structure. The protocol, denoted by $\mathcal{C}$, is described below.

- *Round 1 :* All representatives($p_{i,i}$) send their value to the corresponding verifiers ($p_{i,j},\ j \neq i$).

- *Round 2 :* Verifiers and representatives together send the value received from the corresponding representatives to all the players. Each player then concludes three values, one for every adversary set, by taking the majority among the three values $\{ p_{i,1}, p_{i,2}, p_{i,3} \}$ corresponding to that adversary set. Each player then takes the final value of the protocol to be the majority of the three values (that were concluded for each i).

From Lemmas 3.4 and 3.5 and the fact that the protocol certainly terminates, we know that the protocol C achieves consensus. □

LEMMA 3.4 (AGREEMENT). *At the end of protocol C, all the players conclude on the same final value.*

Proof : In any execution of the protocol, only one player of the three players in the triplet corresponding to one adversary set can be corrupted by the adversary. Even if the representative is corrupted, due to the restrictions imposed by the partition $\vec{\Pi}$, the representative will be forced to send the same value to both the (honest) verifiers. Therefore, of the three versions of the representative's value that all players receive, at least two versions will be the same and will also be received the same by all the players. This means that all players will conclude on the same value for each of the representatives. Thus the agreement on each of the three values is assured. Since all the players take the majority of the three values, agreement on the final value is also assured. □

LEMMA 3.5 (VALIDITY). *At the end of protocol C, all the players agree on a honest player's value.*

Proof : The protocol agrees on the majority of the values of the three representatives. In any execution of the protocol, at least two of the three representatives are honest and therefore, the agreed value will be a honest player's value. □

3.2 Characterization of tolerable Generalized Sectional Adversaries

In the Generalized Sectional adversary model, a player is associated with more than one partition of the player set, one of which may be in use during any execution of the protocol.

THEOREM 3.3. *Consensus is achievable in the presence of Generalized Sectional adversary, ($\mathcal{A}$, $\vec{\Pi}$) where $\vec{\Pi} = \langle \Pi_1, \Pi_2, \ldots, \Pi_n \rangle$, and Π_k is a collection of partitions of the player set, if and only if consensus is achievable in each of the Sectional adversaries formed by choosing one partition each from each Π_i.*

Proof : To see that the above condition is necessary, we note that if consensus is not achievable for the Sectional adversary corresponding to a particular combination of partitions, then consensus will not be achievable in the Generalized Sectional adversary, when the adversary chooses this combination of partitions.

To show that this condition is sufficient, we give a protocol, which will achieve consensus tolerating every adversary substructure $\mathcal{A}' \subset \mathcal{A}$ of size 3. This combined with Theorem 3.1 completes the proof of the sufficiency condition.

Let the adversary structure that is being tolerated be $\mathcal{A}' = \{ A_1, A_2, A_3 \}$.

If $\mathcal{P} \neq A_1 \cup A_2 \cup A_3$ then there exists an honest player, and hence, consensus can be trivially achieved.

If $\mathcal{P} = A_1 \cup A_2 \cup A_3$ then from the necessary condition it follows that for every combination of partitions, $\mathcal{A}'$ must be verifiable. For each combination i, let $a^i_{1,2} \in U(A_2)$ and $a^i_{1,3} \in U(A_3)$ belong to the same partition for $a^i_{1,1} \in U(A_1)$. Hence, if the i^{th} combination is in operation, then the representative $a^i_{1,1}$ will be forced to send the same value to the verifiers $a^i_{1,2}$ and $a^i_{1,3}$. Let p_1, p_2 and p_3 be three players in $U(A_1)$, $U(A_2)$, and $U(A_3)$ respectively. We describe a protocol to broadcast p_1's value to all the players. Similar protocols can be used to broadcast p_2's and p_3's values to all the players and these can be put together to achieve consensus as in the protocol of Theorem 3.2.

- *Round 1:* Player p_1 broadcasts its value to all the representatives, $a^i_{1,1}$ for every combination i.
- *Round 2 :* Each virtual player representative, $a^i_{1,1}$, broadcasts the value it received from p_1 to the corresponding verifiers.
- *Round 3 :* The virtual player verifiers, $a^i_{1,2}$ and $a^i_{1,3}$ send the value they have received from the corresponding virtual player representative $a^i_{1,1}$ to all the players. The players then use the following algorithm to commonly agree upon a value corresponding to A_1.
  - If any player receives the same value, v, from all players $a^i_{1,1}$ and $a^i_{1,2}$ for every combination i, or if the player receives the same value, v, from players $a^i_{1,1}$ and $a^i_{1,3}$ for every combination i, then the player accepts v as A_1's value. This corresponds to the case when the set A_1 is not corrupted.
  - Otherwise let j be the smallest index such that the values quoted by $a^j_{1,2}$ and $a^j_{1,3}$ are the same, namely, v. Then the player accepts v as A_1's value. Note that some such j exists because some combination of partitions is in operation and there exists a committee guaranteed to work with this.

We note that if the adversary set A_1 is not corrupted, player p_1 is honest and all representatives, $\forall\, i\ a^i_{1,1}$, are honest. Moreover, if A_2 is not corrupted, the values reported by players $a^i_{1,2}$ match with those of players $a^i_{1,1}$. By the first clause of the algorithm described in Round 3 in the above protocol, all players agree on on p_1's value. In case A_1 is corrupted, the partition forces some player $a^j_{1,1}$ to send the same value to the corresponding verifiers and the second clause applies. We note that since all verifiers are honest in this case, the smallest i for which this happens is the same for all players. It may also happen that a player concludes on a value v using the first clause in this case too, but the honesty of the verifiers assures us that this value is the same as the value that other players, possibly using the second clause, agree on. This proves that the above protocol succeeds in broadcasting player p_1's value if A_1 is not corrupted or in achieving agreement on some value if A_1 is corrupted.

Similar protocols can be run for players p_2 and p_3 to complete a protocol on the lines of the protocol in the proof of Theorem 3.2. The assurance of agreement, along with the fact that at least two of the players p_1, p_2 and p_3 are honest, assures the correctness of the protocol. □

4. ACHIEVING POLYNOMIAL COMPLEXITY

In this section, we consider the issue of complexity for the case of the Sectional adversary. The bottleneck in most distributed protocols is the communication complexity rather than the computation complexity. We therefore analyze all our protocols for their communication complexity alone; nevertheless, we remark that all presented protocols have a computation complexity polynomial in the input size.

We present a protocol with communication complexity polynomial in the number of players, n, for the case of the Sectional adversary, unlike the ones in the previous section that were polynomial in the size of the maximal basis of the adversary structure.[8] In the next section, we modify this protocol to achieve a more *efficient* protocol. We then present methods to transform the final version of the protocol to a protocol tolerating the Generalized Sectional adversary.

The key to the development of a better protocol is the use of virtual players. Theorem 3.2 assures us that, corresponding to each adversary structure of three sets which together cover $\mathcal{P}$, there exists a committee of players, which can be used to perform consensus when this adversary structure is in operation. Thus, for every adversary structure $\mathcal{A}' = \{A_1, A_2, A_3\}$ of size three such that $A_1 \cup A_2 \cup A_3 = \mathcal{P}$, there exists a *virtual player* on the lines of the *player simulation* idea of [11]. We denote the set of virtual players by $\mathcal{VP}$. Let

$$f_{adv} : \mathcal{VP} \rightarrow \mathcal{A} \times \mathcal{A} \times \mathcal{A}$$

be the function such that $f_{adv}(V) = \{Y_1, Y_2, Y_3\}$ implies that the virtual player V corresponds to the adversary structure $\{Y_1, Y_2, Y_3\}$.

Let $\mathcal{A}_v \subset 2^{\mathcal{P} \cup \mathcal{VP}}$ denote the adversary structure representing the failure knowledge of the players and virtual players taken together, i.e. the adversary structure on the set $\mathcal{P} \cup \mathcal{VP}$. The structure $\mathcal{A}_v$ is constructed as follows. Let the original adversary structure $\mathcal{A} = \{B_1, B_2, \ldots, B_w\}$. Then,

$$\mathcal{A}_v = \{(B_i \cup \{Z \in \mathcal{VP} | B_i \notin f_{adv}(Z)\}) | i = 1, 2, \ldots, w\}$$

In other words, corresponding to every adversary set in the original adversary structure, any virtual player, whose corresponding adversary structure contains this set is honest. All other virtual players are assumed to be faulty (This is an over-estimation of the actual failure knowledge but it will do for the purpose of the protocol).

The following observation regarding the set of players and virtual players taken together is crucial to the development of a polynomial communication complexity protocol.

OBSERVATION 4.1. *$\mathcal{Q}^{(3)}(\mathcal{P} \cup \mathcal{VP}, \mathcal{A}_v)$ holds.*[9]

Proof : Suppose that this is not so. Then there exist three adversary sets $\mathcal{A}_v^1, \mathcal{A}_v^2, \mathcal{A}_v^3$ of $\mathcal{A}_v$ which cover $\mathcal{P} \cup \mathcal{VP}$. Each $\mathcal{A}_v^i$ consists of players and virtual players, with the player subset corresponding to an adversary set of the original adversary structure $\mathcal{A}$. By the assumption made at the beginning of the proof, the player subsets of each of $\mathcal{A}_v^i$'s together cover the player set $\mathcal{P}$. These, therefore, constitute an adversary substructure (of $\mathcal{A}$) of size 3 which violates $\mathcal{Q}^{(3)}$. By the characterization condition (Theorem 3.2), it follows that there exists a honest virtual player corresponding to this substructure. This virtual player does not appear in any of the virtual player subsets of $\mathcal{A}_v^i$'s, thereby contradicting the earlier assumption. □

This observation directly motivates an agreement protocol on the set $\mathcal{P} \cup \mathcal{VP}$. In fact, any deterministic agreement protocol where:

1. the initial values of the virtual players are generated by agreement among the representatives of the virtual player committee;

2. the operations of the protocol for virtual players are substituted by agreement protocols among the virtual player committees;

achieves consensus, in the sense that it satisfies the conditions of the consensus problem among the real players.

The protocol thus formed is still potentially exponential in communication complexity because there can be as many as $\binom{|\mathcal{A}|}{3}$ virtual players. We note, however, that two virtual players which share the same committee are effectively the same. By this we mean that firstly, they have the same input values, and secondly, they receive the same messages and hence the execution of a deterministic protocol is in no way different for both of them. This allows all virtual players with the same committees to be simulated by one committee alone, with the simulation of equivalent virtual players being left to the local computation of the participating players. This modification reduces the communication complexity to that of a protocol among $O(n^9)$ committees and n real players. A Phase King protocol [3, 1] on these $O(n^9)$ (real and virtual) players results in a protocol with polynomial communication complexity. The discussion so far leads to the following theorem.

THEOREM 4.1. *Distributed consensus tolerating the Sectional adversary $(\mathcal{A}, \vec{\Pi})$ can be achieved in polynomial communication complexity.*

5. CONSTRUCTING AN EFFICIENT PROTOCOL

A more efficient protocol relies on a close analysis of two components - one, the internal protocol among the virtual players, and two, the Phase King protocol itself. In this section, we show how modifications on these lines can bring down the communication complexity to a slim $O(n^4)$. We first describe, briefly, the Phase King protocol and then proceed with the modifications.

The Phase King protocol was originally proposed by [3]. This protocol (more correctly, a recursive modification of this protocol) could achieve bit-optimal consensus in the case of the threshold adversary. This protocol was extended to the case of a mixed non-threshold adversary in [1]. For the proof of correctness of this protocol and its analysis, we refer the reader to [3, 1]. Here we only present the protocol listing (Figure 2). Note that, in our case, it is enough to restrict the set of kings to the real player set $\mathcal{P}$ alone, since at least one honest real player is guaranteed.

We introduce the notation $q(\mathcal{A})$, or more simply q, to denote the largest integer k for which $\mathcal{Q}^{(k)}(\mathcal{A})$ holds. The condition $Q^{(3)}(\mathcal{A})$ then translates to $q \geq 3$.

[8] Note that the size of the maximal basis of the adversary structure could be exponential in the number of players.

[9] In fact, $\mathcal{Q}^{(3)}(\mathcal{VP}, \mathcal{A}_v)$ holds.

```
for k = 1 to |P| do  (* Set of kings *)
begin                (* Start of a phase *)
   send(value)       (*Universal Exchange 1; all players send.*)
   receive(V)
   V^i = {r ∈ P ∪ VP, r sent i}, i = 0, 1
   for i = 0, 1 do
      if Q^(1)(P − V^i) does not hold
            send(1)
      else
            send(0) (*Universal Exchange 2; all players send.*)
   receive(R)
   R^i = {r ∈ P ∪ VP, r sent 1 }, i = 0, 1
   if Q^(1)(R^1) holds
      value = 1
   else
      value = 0
   (for the king p_k only)
      send(value)   (* King's Broadcast *)
   receive(king's value)
   if Q^(1)(P − R^value) holds
      value = king's value
end                  (* End of the phase *)
```

Figure 2: Description of the Phase King protocol for a player in $\mathcal{P} \cup \mathcal{VP}$

5.1 Modification to the virtual player committee protocol

The aim of this modification is to move virtual player participation in the protocol from the repeated use of the internal agreement protocol, used to simulate the virtual player, (which is primarily responsible for the high communication complexity of all previous protocols) to local simulation of virtual player behaviour by the real players as much as possible. To this end, we assert the following lemma.

LEMMA 5.1. *For any deterministic protocol, a player's protocol run can be completely simulated if the following are known: (1) The player's initial value, and (2) The messages that the player receives.*

Proof : The player's initial value and the messages that the player receives completely determine the view of the protocol and so, the rest of the protocol behaviour can be simulated. □

We, therefore, modify virtual player behaviour so that the messages a virtual player receives can be made known (reliably and efficiently) to all other players. This will allow all other (real) players to simulate virtual player behaviour, reducing the overhead involved in the virtual player internal agreement protocols. To this end, we introduce a primitive called the public internal agreement protocol, which is only a slightly modified form of the protocol used for sufficiency in Theorem 3.2.

We present the details of this internal agreement protocol, doubling as broadcast, that we call public internal agreement in Figure 3. It is clear that the procedure of Figure 3 works and performs its objective.

We now modify the actions of the Phase King protocol for virtual players so that the two conditions in Lemma 5.1 are satisfied. This then allows real players to simulate the

- *Round 1:* Virtual player representatives send the inputs of the agreement protocol to the verifiers.
- *Round 2:* Verifiers and representatives together send the inputs to all other players. Recipients apply the majority function on the values received (first on the values received from each triplet corresponding to each adversary set and then on the three values thus obtained) to determine the result of the internal agreement protocol.

Figure 3: Public internal agreement (PIA)

complete virtual player behaviour locally itself, leading to a significant reduction in the communication complexity. The modifications are as follows.

1. send(input) in the very first round of the protocol for a virtual player is replaced by PIA(input) for that player.
2. receive(value) in any round is replaced by PIA (received_value).

We now analyze the complexity of the new protocol thus obtained. We note that, in the Phase King protocol, a real player acting as representative for many virtual players sends the same value for all those virtual players and hence step 1 in the PIA protocol can be performed in $O(n^2)$ bits for all virtual players. Step 2 then takes $O(n^2)$ bits per real player (and hence, $O(n^3)$ over all virtual players) because each verifier sends $O(n)$ bits for every possible representative that the verifier verifies. Thus overall, each *global* invocation of PIA (i.e. by all virtual players) in the Phase King protocol incurs an overhead of $O(n^3)$ bits. Each round involves $O(n)$ (global) receive operations and there are $O(n)$ rounds, accounting for $O(n^2)$ global invocations of PIA. Hence, the overall contribution of virtual players to the communication complexity of the Phase King protocol is $O(n^5)$ bits, which dominates the communication cost of real players (which is $O(n^3)$ bits, in fact). Thus, we have the following theorem.

THEOREM 5.1. *The Phase King protocol on $\mathcal{P} \cup \mathcal{VP}$ with the modification listed above achieves consensus and involves a communication complexity of $O(n^5)$.*

5.2 Modification to the Phase King protocol for virtual players

In this subsection, we further improve the communication complexity of the protocol by taking advantage of some specific aspects of the Phase King protocol, bringing the communication complexity down to $O(n^4)$.

We now describe, informally, the changes we propose to make and the motivation behind them. Each exchange round in the Phase King protocol is of the form $\forall i$ *receive*$(mesg_i)$, followed by a computational step, and then resulting in $\forall i$ *send*(new_mesg_i). In the current version of the protocol, what happens in a round is that virtual players reach (public) agreement on each of the messages to be received, i.e. each $mesg_i$, and all real players use this knowledge to decide on the new_mesg_i of the virtual player. We claim that, instead, it is enough for virtual players to reach agreement on new_mesg_i directly based on the new_mesg_i's of

the virtual player representatives. After this modification, virtual players need only to engage in $O(1)$ global *receives*, leading to a factor of n improvement in the communication complexity.

We now describe formally the modified Phase King protocol for virtual players to be executed by the virtual player committees. We use the following notation. Messages that a player sends in exchange $i, i = 1, 2, 3$ of the j^{th} phase are denoted respectively by m_i^j. The protocol is listed in Figure 4.

During j^{th} phase do

- *Universal Exchange 1 :* PIA(m_1^j), i.e. virtual player committees execute PIA on the m_1^j values of representatives.
- *Universal Exchange 2 :* PIA(m_2^j), i.e. virtual player committees execute PIA on the m_2^j values of representatives.
- *King's Broadcast :* no action.

Figure 4: Modified virtual player protocol

We now show the correctness of the modified protocol.

Proof of correctness(sketch): We note that in protocols where, (1) the messages sent in a round depend solely (and in the same fashion) on the messages received from players in the previous round and, (2) any player sends the same value to all other players, a honest player's message can be replaced with any other honest player's message without affecting the correctness of the protocol. This is because the messages that they receive the previous round only differ in the faulty players' values and a different set of faulty player messages should not affect the correctness of the protocol. The Phase King protocol satisfies the above conditions. A honest virtual player's messages in such a protocol can, therefore, always be replaced by the messages of any of its honest representatives. This can be done by achieving consensus on the representatives' messages. The modification in Figure 4, therefore, does not affect the correctness of the protocol. □

We now analyze the communication complexity of the protocol. As usual, virtual player operations dominate the cost. Each virtual player performs O(1) operations per phase, each operation involving a (global) invocation of PIA, which takes $O(n^3)$ bits. Overall, the protocol therefore involves $O(n^4)$ communication.

6. CASE OF GENERALIZED SECTIONAL ADVERSARY

We now extend the efficient protocol for Sectional adversary, as described in Figure 4, to the case of the Generalized Sectional adversary. We retain the same structure of the final protocol and present an equivalent protocol for the PIA primitive.

The PIA primitive in the protocol for Sectional adversaries (Section 5) can be replaced by the Generalized PIA primitive to get the protocol for Generalized Sectional adversaries. This primitive can be taken to be the protocol for sufficiency given in Theorem 3.3, with players p_1, p_2 and p_3 (as defined in the proof of Theorem 3.3) simultaneously broadcasting their value to all other players.

We now analyse the communication complexity of a global invocation of this primitive. Round 1 can be done using $O(n)$ messages per each possible p_i, which means $O(n^2)$ messages in all. In Round 2, each of the representatives send the values they have received from the players p_1, p_2 and p_3. This takes $O(n^3)$ messages in all. In Round 3, the verifiers broadcast the messages they receive from all representatives. This takes a total of $O(n^4)$ messages. Hence, a global invocation of this primitive requires $O(n^4)$ messages.

THEOREM 6.1. *The final version of the protocol of Section 5 with the PIA replaced by the Generalized Public Internal Agreement achieves agreement tolerating the Generalized Sectional adversary with a communication cost of $O(n^5)$.*

7. CONCLUSION

We have defined a new fault model which incorporates additional knowledge about the fault. We have presented necessary and sufficient conditions under which consensus can be achieved in this model. We have made a beginning in the direction of efficient protocols for consensus in this model with the $O(n^4)$ communication complexity protocol of Section 5. The following theorem expresses the results of this paper in a nutshell.

THEOREM 7.1 (MAIN THEOREM). *Consider a set of n players, any two of which can communicate with each other, under the corrupting influence of a Byzantine Sectional adversary characterized by by $(\mathcal{A}, \bar{\Pi})$.*

1. *Consensus is possible if and only if $\mathcal{A}$ satisfies $\mathcal{Q}^{(2)}$ and either of the following conditions hold for every three adversary sets A_1, A_2, A_3 of $\mathcal{A}$:*
   (a) *$A_1 \cup A_2 \cup A_3 \neq \mathcal{P}$, or*
   (b) *Each of A_1, A_2, A_3 is verifiable w.r.t the adversary structure $\{A_1, A_2, A_3\}$.*
2. *Whenever the conditions in (1) hold, there exists a protocol with the following properties:*
   (a) *The protocol achieves consensus with certainty.*
   (b) *The protocol communicates $O(n^4)$ bits, irrespective of the size of the adversary structure. Its computation complexity is polynomial in the input size.*

The significance of the results can be understood by the following implications.

1. There exist (a lot of) scenarios for which consensus was deemed impossible (in the extant fault models), whereas consensus is achievable in the Sectional adversary model.
2. Usually, the more general the model the less efficient the (worst-case) protocol. Notwithstanding this, the additional power of our model does not hinder the efficiency of the protocol (which is comparable to that of any known protocol for the non-threshold model).
3. Implicit in our characterization is the minimal network primitives required to achieve the desired fault-tolerance.

4. Our characterization strictly generalizes the $\mathcal{Q}^{(3)}$ characterization of [7]. Note that $\mathcal{Q}^{(3)}$ is identical to the $1(a)$ condition, whereas we require either $1(a)$ or $1(b)$ to be satisfied.

One evident future work in this area is to concentrate on developing more efficient protocols. We note that all our protocols are 'built' on known protocols like the Phase King protocol. Hence, any efficiency improvement for consensus tolerating Byzantine faults in the non-threshold adversary model would directly trigger an improvement in the complexity of our algorithms. Not much work has been done in the area of achieving optimal consensus in the non-threshold fault model (unlike the threshold fault model [3]) and optimal non-threshold Byzantine agreement remains a significant open problem.

The success of this effort (and the ones like in [11, 7]) to improve fault tolerance by increasing the description of the fault model should serve as a motivation to further efforts in this direction. It looks worthwhile to study "Fault-knowledge versus Fault-tolerance" in general. Intuitively, there exists a direct correlation between them, namely, as the knowledge about the faults increases, the fault-tolerance increases; however, the exact relationship is to be explored.

8. REFERENCES

[1] B. Altmann, M. Fitzi, and U. Maurer. Byzantine agreement secure against general adversaries in the dual failure model. In *Proceedings of DISC 99*, volume 1693 of *Lecture Notes in Computer Science (LNCS)* pages 123–137, 1999.

[2] O. Babaoglu and R. Drummond. Streets of Byzantium: Network Architectures for Reliable Broadcast. *IEEE Transactions on Software Engineering*, SE-11(6):546–554, June 1985.

[3] P. Berman and J. A. Garay Asymptotically optimal distributed consensus. In *Proceedings of ICALP*, volume 372 of *Lecture Notes in Computer Science (LNCS)* pages 80–94, 1989.

[4] R. Canetti, U. Feige, O. Goldreich, and M. Naor. Adaptively secure multi-party computation. In *Proceedings of 28th ACM Symposium on Theory of Computing (STOC)*, pages 639–648, 1996.

[5] D. Dolev and H. R. Strong. Authenticated algorithms for byzantine agreement. *SIAM Journal of Computing*, 12(4):656–666, November 1983.

[6] M. Fitzi, D. Gottesman, M. Hirt, T. Holenstein, and A. Smith. Detectable Byzantine Agreement secure against faulty majorities. In *Proceedings of 21st ACM Symposium on Principles of Distributed Computing (PODC)*, pages 118–126, 2002.

[7] M. Fitzi and U. Maurer. Efficient byzantine agreement secure against general adversaries. In *Proceedings of Distributed Computing (DISC '98)*, volume 1499 of *Lecture Notes in Computer Science (LNCS)*, pages 134–148. Springer-Verlag, 1998.

[8] M. Fitzi and U. Maurer. From partial consistency to global broadcast. In *Proceedings of the 32th Annual ACM Symposium on Theory of Computing (STOC '00)*, pages 494–503, 2000.

[9] J. A. Garay. Reaching (and maintaining) agreement in the presence of mobile faults. In *Proc. 8th International Workshop on Distributed Algorithms*, volume 857 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pages 253–264, 1994.

[10] J. A. Garay and K. Perry. A continuum of failure models for distributed computing. In *Workshop on Distributed Algorithms*, volume 647 of *Lecture Notes in Computer Science (LNCS)*, Springer-Verlag, pages 153–165, 1992.

[11] M. Hirt and U. Maurer. Complete characterization of adversaries tolerable in secure multiparty computation. In *16th ACM Symposium on Principles of Distributed Computing (PODC)*, pages 25–34, August 1997.

[12] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

[13] N. Lynch. *Distributed Algorithms*, chapter 6. Morgan Kaufmann, San Mateo, CA, USA, 1996.

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

[15] R. Reischuk. A new solution for the Byzantine generals problem. *Information and Control*, 64:23–42, 1985.