Brief Announcement: Capacity of Byzantine Agreement with Finite Link Capacity - Complete Characterization of Four-Node Networks *

Guanfeng Liang
Department of Electrical and Computer Engineering, and Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Illinois, USA
gliang2@illinois.edu

Nitin Vaidya
Department of Electrical and Computer Engineering, and Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Illinois, USA
nhv@illinois.edu

ABSTRACT

In this paper, we consider the problem of maximizing the throughput of Byzantine agreement, when communication links have finite capacity. Byzantine agreement is a classical problem in distributed computing, with initial solutions presented in the seminal work of Pease, Shostak and Lamport. The notion of throughput here is similar to that used in the networking/communications literature on unicast or multicast traffic. We identify necessary conditions for an agreement throughput of R to be achievable. We also provide tight sufficient conditions by construction for agreement throughput R in four-node networks.

Categories and Subject Descriptors: C.2.4 [COMPUTER-COMMUNICATION NETWORKS]: Distributed Systems

General Terms: Algorithms, Reliability, Security, Theory.
Keywords: Byzantine agreement.

1. INTRODUCTION

We consider the problem of characterizing the capacity of Byzantine agreement, given finite-capacity links between nodes in the system. Byzantine agreement is a classical problem in distributed computing, with initial solutions presented in the seminal work of Pease, Shostak and Lamport [1]. Many variations on the Byzantine *agreement* problem have been introduced in the past, with some of the variations also called *consensus*. We will use the following definition of Byzantine agreement: Consider a network with one node designated as the *sender* or *source* (S), and the other nodes designated as the *peers*. The goal of Byzantine agreement is for all the fault-free nodes to "agree on" the value being sent by the sender, despite the possibility that some of the nodes may be faulty. In particular, the following conditions must be satisfied:

- **Agreement:** All fault-free peers must agree on an identical value.
- **Validity:** If the sender is fault-free, then the agreed value must be identical to the sender's value.
- **Termination:** Agreement between fault-free peers is eventually achieved.

Our goal in this work is to design algorithms that can achieve optimal *throughput* of agreement. When defining throughput, the "value" referred in the above definition of agreement is viewed as an infinite sequence of *information* bits. We assume that the information bits have already been compressed, such that for any subsequence of length $n > 0$, the 2^n possible sequences are sent by the sender with equal probability. Thus, no set of information bits sent by the sender contains useful information about other bits. This assumption comes from the observation about "typical sequences" in Shannon's work [3].

We also adopt the notion of *channel capacity* from the information theory literature [3]: tightest upper bound on the amount of *information* that can be reliably transmitted over a communications channel, independent of how the bits are encoded (e.g. the bits could be encoded as a specific waveform, or as silenced interval). In the existing works on Byzantine agreement, the *capacity* of links between the nodes are assumed to be infinite implicitly. To the best of our knowledge, we are the first one to study the problem of Byzantine agreement when the links in the network have finite, and maybe different, capacity.

At each peer, we view the agreed information as being represented in an array of infinite length. Initially, none of the bits in this array at a peer have been agreed upon. As time progresses, the array is filled in with agreed bits. In principle, the array may not necessarily be filled sequentially. For instance, a peer may agree on bit number 3 before it is able to agree on bit number 2. Once a peer agrees on any bit, that agreed bit cannot be changed.

We assume that an agreement algorithm begins execution at time 0. The system is assumed to be synchronous. In a given execution of an agreement algorithm, suppose that by time t all the fault-free peers have agreed upon bits 0 through $b(t)-1$, and at least one fault-free peer has not yet agreed on bit number $b(t)$. Then, the agreement *throughput* is defined as $\lim_{t\to\infty} \frac{b(t)}{t}$.

Capacity of agreement in a given network, for a given sender and a given set of peers, is defined as the supremum of all achievable agreement throughputs.

*This research is supported in part by Army Research Office grant W-911-NF-0710287. Any opinions, findings, and conclusions or recommendations expressed here are those of the authors and do not necessarily reflect the views of the funding agencies or the U.S. government.

PODC'10, July 25–28, 2010, Zurich, Switzerland.
ACM 978-1-60558-888-9/10/07.

2. NECESSARY CONDITIONS FOR AGREEMENT THROUGHPUT R

It is known that a network must contain at least 4 nodes for agreement to be achievable with a single Byzantine failure. In this work, we only consider the case of 4 nodes when at most 1 node may suffer Byzantine failure. The characterization of agreement capacity for the four-node network is non-trivial and cannot be generalized to larger networks directly. The design of capacity achieving algorithms in larger networks (possibly with multiple failures) is substantially more complex than the four-node case.

We consider a synchronous network of 4 nodes named S, A, B and C, with node S acting as the sender, and the others being the peers. At most one of these four nodes may be faulty. The network is viewed as a directed graph, formed by directed links between the nodes in the network, with the capacity of each link being finite. The capacity of some links may be 0, which implies that these links do not exist. Let us call the incoming links at S as the *uplinks* (links AS, BS and CS). We identify the following *necessary* conditions for achieving agreement throughput of R bits/unit time.

- **Necessary condition NC1:** If any one peer is removed from the network, the min-cut from the source S to each remaining peer must be at least R.
- **Necessary condition NC2:** The max-flow to each of the peers from the other peers, with the source removed from the network, must be at least R.
- **Necessary condition NC3:** All incoming links to the peers must exist (capacity> 0).
- **Necessary condition NC4:** The capacity of every out-going link from S must be at least R, when S there is no uplink.

Our main results are the tightness of these conditions:

- **With uplink(s):** Agreement capacity of a four-node network is the supremum over all throughputs R that satisfy necessary conditions NC1, NC2, and NC3.
- **With no uplink:** Agreement capacity of a four-node network is the supremum over all throughputs R that satisfy necessary conditions NC1, NC2, NC3 and NC4.

3. SKETCH OF CAPACITY ACHIEVING ALGORITHMS

We prove our results by providing agreement algorithms that can achieve throughput arbitrarily close to R, given the corresponding conditions are satisfied. The algorithm for the case of complete graph with all 3 uplinks exist is slightly different from the one for the case with fewer uplinks, and is easier to describe. For brevity, we will only sketch the algorithm for the complete graph. Interested readers are referred to our technical report [2] for more details.

The proposed Byzantine agreement algorithm for the complete graph proceeds in rounds. The units for rate R and the various link capacities are assumed to be *bits/time unit*, for a convenient choice of the *time unit*. We assume that by a suitable choice of the *time unit*, the number R and the various link capacities can be turned into integers. The algorithm executes in multiple rounds, with the duration of each round being approximately c time units. Note that in c time units, a link with capacity z bits/time unit can carry z symbols (or packets) from Galois field GF(2^c). Computation is assumed to require 0 time.

In Round 1, the source S transmits as many coded packets as possible to the peers, such that each coded packet is a linear combination of R packets of data, and any subset of R coded packets constitutes **independent** linear combinations of the R data packets. As we know from the design of Reed-Solomon codes, if c is chosen large enough, this linear independence requirement can be satisfied. In round 2, each peer relays as many distinct packets it receives from S in round 1 to each of the other two peers. Then, each fault-free peer check if any node has misbehave by trying to find a unique solution for **each** subset of R packets from among the packets received from the other three nodes in rounds 1 and 2. We can show that if a faulty node misbehaves, it will be detected by at least one fault-free peer. If a failure is detected, a broadcast phase is triggered, and every node including S **broadcasts** all packets it has sent and received during rounds 1 and 2 to the remaining 3 nodes, using the traditional Byzantine agreement algorithm, in particular the algorithm by Pease, Shostak and Lamport [1]. This is possible in the complete graph. For incomplete graph with fewer uplinks, this part is more complicated and is described in our technical report [2]. Based on the broadcast information, the fault-free nodes will be able to identify the faulty node if it misbehaves for more than a finite number of times. Once the faulty node is identified, each fault-free peer can recover the correct data from the packets from the other two fault-free nodes, or terminates the algorithm if S is faulty.

In achieving throughput R, it will be necessary to have multiple "generations" of packets in the network, with the algorithm operating in a pipelined manner (one round per pipeline stage). Agreement algorithm for one new generation of data of size Rc bits (or R symbols from GF(2^c)) starts per round. By using a suitably large c, the overhead for disseminating detection results and a finite number of broadcast phases diminishes to 0 as time goes to infinity. Hence, the throughput can be made arbitrarily close to R.

4. DISCUSSION

While NC1 and NC2 can be easily generalized to larger networks with multiple failures, they are not sufficient for networks with more than 4 nodes. The following condition must also be satisfied for achieving throughput R:

- **Necessary condition NC5:** If any node is removed from the network, the sum capacity of all links in both direction on any cut must be at least R.

In four-node networks, NC5 is implied by NC1 and NC2 in four-node, but in larger networks.

In four-node networks, we used a very simple scheme: Reed-Solomon code at the source, and the peers just forward packets from the source. Unfortunately, this scheme can not be generalized directly to larger networks with multiple failures. Our preliminary results on larger networks show that a capacity achieving algorithm is substantially more complex than the one for four-node networks.

5. REFERENCES

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

[2] G. Liang and N. Vaidya. Capacity of byzantine agreement: Complete characterization of the four node network. *Technical Report, CSL, UIUC*, April 2010.

[3] C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27:379–423, 623–656, 1948.