Brief Announcement: Byzantine Agreement with Unknown Participants and Failures

Pankaj Khanchandani
ETH Zurich
kpankaj@ethz.ch

Roger Wattenhofer
ETH Zurich
wattenhofer@ethz.ch

ABSTRACT

A set of participants that want to agree on a common opinion despite the presence of malicious or Byzantine participants need to solve an instance of a Byzantine agreement problem. This classic problem has been well studied but most of the existing solutions assume that the participants are aware of n — the total number of participants in the system — and f — the upper bound on the number of Byzantine participants. In this paper, we examine a synchronous system with Byzantine faults where the participants are neither aware of n nor f. The participants have unique identifiers, which are not necessarily consecutive. For such a system, we give algorithms for rotor-coordinator and consensus, both with the resiliency of $n > 3f$, which is also the optimal resiliency for solving consensus when the participants know n and f. Thus, resiliency is unaffected even if the Byzantine participants can lie about n and f.

ACM Reference Format:
Pankaj Khanchandani and Roger Wattenhofer. 2020. Brief Announcement: Byzantine Agreement with Unknown Participants and Failures. In *Symposium on Principles of Distributed Computing (PODC '20), August 3–7, 2020, Virtual Event, Italy*. ACM, New York, NY, USA, 3 pages. https://doi.org/10.1145/3382734.3405740

1 INTRODUCTION

Many modern networks have to function without knowing the size of the system in advance. Consider, for example, a database cluster that requires frequent node scaling because of changing load, or a wireless sensor network that experiences a changing number of faulty or disconnected nodes over time. Nakamoto's blockchain [10] is a prominent example where the network is permissionless, i.e., the network is open to *any* number of nodes. So, the number of participants and consequently, the number of failures also change over time. Agreement is a fundamental distributed computing primitive for fault-tolerant networks, however, much of the existing literature assumes that the size n of the network and/or the upper bound f on the number of failures is known to every node [2, 5, 9, 12].

In this paper, we consider fault-prone systems where the nodes do not know the number of nodes n and the maximum number of Byzantine nodes f and give algorithms for *rotor-coordinator* — selects a unique leader in every round — and *consensus* — every correct node has a binary input and the correct nodes output a common binary value that is an input of some correct node [8]. Since a correct node does not know n and f and a Byzantine node may not announce itself to everyone, there might be more Byzantine nodes in the system than what a correct node thinks, and so a larger proportion of good nodes might be required. We show, however, that these problems can be solved without affecting resiliency. Specifically, we give algorithms for solving the above problems in synchronous systems with the resiliency of $n > 3f$, which is optimal for consensus.

PODC '20, August 3–7, 2020, Virtual Event, Italy

ACM ISBN 978-1-4503-7582-5/20/08…$15.00
https://doi.org/10.1145/3382734.3405740

2 RELATED WORK

If the nodes do not know n and f and the network is asynchronous, i.e., the message delays are unbounded, then agreement with probabilistic termination is impossible since the nodes do not know how many messages to receive before deciding. There is a line of work that deals with this problem using oracles or failure detectors. They either assume f [1, 4, 7] or the failure detector eventually removes the Byzantine nodes [11].

The Byzantine agreement problems have been studied since the work by Lamport et al. [8]. The *king* algorithm [3] solves consensus in $O(f+1)$ rounds with polynomial message complexity and optimal resiliency of $n > 3f$. The round complexity can be improved to $f + 1$ [6]. A *rotor-coordinator*, as also used in [3], rotates through the opinions of at most $f + 1$ coordinator nodes before selecting a correct one. The rotor-coordinator can be easily implemented by rotating through $f + 1$ nodes when f is known and the identifiers are consecutive, unlike in our model, where it is non-trivial.

3 MODEL

The system consists of n nodes, out of which at most f are faulty. The faulty nodes can behave in anyway whatsoever, also known as *Byzantine* behavior. We call the non-Byzantine nodes *correct*. The nodes have unique identifiers, which are not necessarily consecutive. Each node only knows its identifier at initialization apart from a possible input and does not know any global information like n or f. The system is *synchronous* and the computation proceeds in *rounds*. In each round, every node receives the messages that were sent to it in the previous round, does some local computations, and then sends messages to the other nodes to be consumed in the following round. A correct node can broadcast a message to all the nodes or send a message to a specific node that sent a message to the node before. The identifier of a node is included in the message it sends so the receiver of the message can decipher its sender. Thus, a Byzantine node cannot forge its identifier when communicating directly. However, it can help other Byzantine nodes to do so indirectly by claiming to have received messages from other, possibly non-existent nodes. Byzantine nodes can send duplicate

messages across rounds but duplicate messages from the same node in a round are simply discarded.

Note that the only way for a correct node to know about the existence of another node is to receive a message from that node. A Byzantine node may get itself known to only a subset of nodes, however, it can behave as if it already knows all the nodes without having received messages from them. We refer to the above model as the *id-only* model. We give the rotor-coordinator and the consensus algorithms for the *id-only* model in Sections 4 and 5, both with the resiliency of $n > 3f$.

4 ROTOR-COORDINATOR

A *rotor-coordinator* selects the opinion of a unique leader or a coordinator in every round. As there are at most f faulty nodes, a correct node is selected as a leader after at most $f + 1$ rounds. Algorithm 1 gives the pseudocode. The idea is that every correct node broadcasts its willingness to become a coordinator initially, when the faulty nodes may or may not participate (Line 3). Every correct node v keeps a set of candidate coordinators C_v, which it updates in a reliable broadcast fashion (Lines 9 and 12). In each round, a correct node v selects the coordinator with the next larger identifier, say p, from the set C_v and adds it to the set of selected coordinators S_v (Lines 14 and 22). The node v accepts the opinion from p in the next round as the coordinator's opinion (Line 17) and broadcasts its own opinion as the coordinator's opinion in case v was selected as the coordinator from the set C_v (Line 25). The node v terminates when it reselects the same node as the coordinator (Line 20). The hope is that by the time a correct node terminates, it has already witnessed a round in which every correct node accepts the opinion of a common and a correct coordinator. We start by showing that if a correct node adds p to its set of candidate coordinator C_v, then another correct node u adds p to its set C_u as well by the next round.

Algorithm 1 Rotor-coordinator algorithm for a node v. The sets C_v and S_v are used by v to store process identifiers. The set C_v is ordered by the process identifiers in increasing order. We use $|C_v|$ for the size of C_v and $C_v[i]$ for its i^{th} member, where $i \geq 0$. The set B_v holds messages before they are broadcast by v at the end of a round. Note that each iteration of the loop is a single round. Coordinator's opinion is accepted in Line 17.

1: $C_v \leftarrow \phi$ ▷ Set of *candidate* coordinators
2: $S_v \leftarrow \phi$ ▷ Set of *selected* coordinators
3: Broadcast *init* ▷ Round 1
4: Broadcast *echo*(p) if received *init* from p ▷ Round 2
5: **for** $r \leftarrow 0 \rightarrow \infty$ **do** ▷ Rounds 3 up to termination
6: $B_v \leftarrow \phi$
7: Let n_v be the number of nodes that sent at least one message to v until the round r.
8: **if** Received at least $n_v/3$ *echo*(p) and $p \notin C_v$ **then**
9: $B_v \leftarrow B_v \cup \{echo(p)\}$
10: **end if**
11: **if** Received at least $2n_v/3$ *echo*(p) and $p \notin C_v$ **then**
12: $C_v \leftarrow C_v \cup \{p\}$
13: **end if**
14: $p \leftarrow C_v[r \bmod |C_v|]$ ▷ Select the next coordinator as p.
15: Let p' be the coordinator selected in the previous round.
16: **if** Received *opinion*(x) from p' **then**
17: Accept x as the coordinator's opinion
18: **end if**
19: **if** $p \in S_v$ **then**
20: **break**
21: **end if**
22: $S_v \leftarrow S_v \cup \{p\}$
23: **if** $v = p$ **then** ▷ Check if v itself is the coordinator.
24: Let o_v be v's current opinion.
25: $B_v \leftarrow B_v \cup \{opinion(o_v)\}$ ▷ To broadcast v's opinion.
26: **end if**
27: Broadcast B_v if its non-empty
28: **end for**

Lemma 1. *If $n > 3f$ and a correct node v receives at least $n_v/3$ copies of a message m from distinct nodes in a round r, then at least one of those messages was sent by a correct node.*

Proof. Let f_v'' be the number of faulty nodes that sent m to v in the round r. Since every correct node transmits a message in the first round (Line 3), we have $n_v \geq g$, where g is the number of good nodes. So, we can write $n_v = g + f_v'$, where f_v' is the number of faulty nodes that sent at least one message to v until the round r. Using $f_v'' \leq f_v'$ and $n_v = g + f_v'$, the number of correct nodes G that sent a message to v in the round r are at least $n_v/3 - f_v'' \geq (g - 2f_v')/3$. As $g > 2f$, we have $G > 2(f - f_v')/3$ or at least one as $f \geq f_v'$. So, at least one correct node sent the message m to v in the round r. □

Lemma 2. *If $n > 3f$ and a correct node v receives at least $2n_v/3$ copies of a message m in a round r, then every correct node u receives at least $n_u/3$ copies of m in the round r.*

Proof. As v receives at least $2n_v/3$ messages, at least $2n_v/3 - f_v''$ of them were sent by the correct nodes, where f_v'' is the number of messages received by v from the faulty nodes in the round r. Let f_v' be the number of faulty nodes from which v received at least one message until the round r. Then, we have $2n_v/3 - f_v'' = 2(g + f_v')/3 - f_v''$, where g is the number of good nodes. As $f_v'' \leq f_v'$ and $f_v' \leq f$ by definition, we have $2(g + f_v')/3 - f_v'' \geq (2g - f)/3$.

Using $n > 3f$ or $g > 2f$, we have $(2g - f)/3 = (g + (g - f))/3 > (g + f)/3$. Thus, at least $(g + f)/3$ correct nodes broadcast the message m and every correct node receives at least $(g + f)/3$ copies of m in the round r. For a correct node u, we have $(g + f)/3 \geq (g + f_u)/3 = n_u/3$, where f_u is the number of faulty nodes from which u has received at least one message until the round r. □

Lemma 3. *If a correct node v adds p to the set C_v in a round r, then any correct node $u \neq v$ adds p to the set C_u by the round $r + 1$.*

Proof. Let r be the first round in which a correct node v adds p to the set C_v. Thus, the node v received at least $2n_v/3$ *echo*(p) messages. Using Lemma 2, each correct node u receives at least $n_u/3$ *echo*(p) messages in the round r. So, every correct node broadcasts *echo*(p) message at the end of round r (Line 27) and each one of them receives g *echo*(p) messages in the round $r + 1$. As $g > 2f$, we have $3g > 2(f + g) = 2n$. Thus, we have $g > 2n/3 \geq 2n_u/3$ for every correct node u. Therefore, every correct node u adds p to the set C_u in the round $r + 1$. □

We call a round a *good round* if the same node p was selected as a coordinator by every correct node and the node p is correct. In the following, we show that every correct node witnesses a good round before it terminates, if $n > 3f$. We will call a round as a *silent* round if the set C_v remains unchanged for every correct node v, i.e, no correct node executes Line 12 in that round. A *non-silent* round is a round that is not silent. We observe that in a silent round, the value of C_v is identical for every correct node v. If they were not, then there is a silent round between a correct node v adding an identifier p to its set C_v and another correct node $u \neq v$ adding p to its set C_u. This contradicts *Lemma* 3. With $n > 3f$, a good round is ensured as follows.

Lemma 4. *If $n > 3f$, then every correct node witnesses at least one good round until it terminates.*

Proof. Assume for contradiction that a node v terminates in the round with $r = r_t$ without witnessing a good round. Consider a round with $r = r_c \leq r_t$. Let $F_v \subseteq C_v$ and $G_v \subseteq C_v$, respectively, be the set of faulty node identifiers and the set of good or correct node identifiers in C_v when the coordinator node is selected in the round r_c (Line 14). Thus, we have $|C_v| = |F_v| + |G_v|$.

A correct node p sends *init* to all the nodes (Line 3). So, every correct node broadcasts *echo*(p) (Line 4) and consequently, every correct node v receives g *echo*(p) messages. As $n > 3f$, we have $g > 2f$ or $3g > 2(f + g) = 2n$. Thus, we have $g > 2n/3 \geq 2n_v/3$. So, all the correct identifiers are added to C_v before the first coordinator is selected.

So, we have $|G_v| = n - f$ and $|C_v| = |F_v| + n - f$. Using $n > 3f$, we get $|C_v| > |F_v| + 2f$. Say that there is *no* correct node u that added a faulty identifier to its set C_u in the round with $r = 0$. Then, every correct node selects a common coordinator from the set G_v and v witnesses a good round before termination, a contradiction. Thus, there is a correct node u that adds a faulty identifier to its set C_u in the round with $r = 0$. For every non-silent round afterwards, at least one faulty node identifier is added to the set C_u of some correct node u. Using Lemma 3, if a faulty node identifier p is added to C_u, every correct node $w \neq u$ adds p to C_w by the next round. Thus, we have $2f \geq n_{ns}$, where n_{ns} is the number of non-silent rounds prior to the round r_c and starting from the round $r = 0$. Therefore, we have $|C_v| > |F_v| + n_{ns}$.

Moreover, until the round r_c, node v has neither witnessed a good round, nor it has selected the same node again as a coordinator by our assumption. So, in all the silent rounds prior to the round r_c, a unique faulty node was selected as a coordinator by v. Therefore, if n_s is the number of silent rounds prior to the round r_c, then $|F_v| \geq n_s$ since v selects a node as a coordinator only after adding it to the set C_v. So, we have $|C_v| > n_s + n_{ns}$.

Since r starts from 0, we have $n_s + n_{ns} = r_c$. So, we have $|C_v| > r_c$ and $r_c \bmod |C_v| = r_c$. Since the above inequality is true for every round $r_c \leq r_t$, a node that was already selected as a coordinator, is in the set $\{C_v[r \bmod |C_v|] : r < r_c\}$. Therefore, for selecting the same identifier as a coordinator again, it must be that $r > |C_v| > r_c$, a contradiction. □

Theorem 1. *If $n > 3f$, then every correct node terminates in $O(n)$ rounds and there is a round in which every correct node accepts the opinion of a common and a correct coordinator node.*

Proof. As a node terminates as soon as it selects the same node as a coordinator and there are n nodes in total, the node terminates in at most n rounds. Using Lemma 4, the node also witnesses a good round before termination and accepts the corresponding opinion in the next round (Line 17). □

5 CONSENSUS

Formally, each node has a binary input and must output a binary value (termination) such that all the correct nodes output the same value (agreement) and that value is an input of some correct node (validity). It is possible to use the rotor-coordinator to design a consensus algorithm in the *id-only* model based on the king algorithm [3]. We use the rotor-coordinator's opinion as the king's opinion and the thresholds of $n - f$ and $n - 2f$ are replaced respectively by $2n_v/3$ and $n_v/3$. This results in an $O(n)$ round consensus algorithm in the *id-only* model. In the full version, we give the full algorithm and analysis.

6 DISCUSSION

In this paper, we considered distributed systems where the participants are neither aware of the size n nor the safe estimate f of Byzantine failures. We show that selecting a correct leader, and consequently consensus, can be solved in this model with the resiliency of $n > 3f$, which is optimal for consensus. The affect on resiliency of other agreement problems such as early terminating consensus is also explored in the full version. It is unclear if the resiliency of the rotor-coordinator is optimal, a question left for future work.

Acknowledgements: We would like to thank Christoph Lenzen for the discussions and giving feedback on an earlier draft.

REFERENCES

[1] Eduardo A. P. Alchieri, Alysson Neves Bessani, Joni da Silva Fraga, and Fabíola Greve. Byzantine Consensus with Unknown Participants. In *International Conference On Principles Of Distributed Systems (OPODIS), Luxor, Egypt*, December 2008.
[2] Hagit Attiya and Jennifer Welch. *Distributed Computing: Fundamentals, Simulations, and Advanced Topics*, chapter 5. John Wiley & Sons, 2004.
[3] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. Towards Optimal Distributed Consensus. In *30th Annual Symposium on Foundations of Computer Science (FOCS), Research Triangle Park, NC*, October 1989.
[4] David Cavin, Yoav Sasson, and André Schiper. Consensus with Unknown Participants or Fundamental Self-Organization. In *International Conference on Ad-Hoc Networks and Wireless (ADHOC-NOW), Vancouver, BC, Canada*, July 2004.
[5] Bernadette Charron-Bost, Matthias Függer, and Thomas Nowak. Approximate Consensus in Highly Dynamic Networks: The Role of Averaging Algorithms. In *42nd International Colloquium on Automata, Languages, and Programming (ICALP), Kyoto, Japan*, July 2015.
[6] Juan A. Garay and Yoram Moses. Fully Polynomial Byzantine Agreement for $n > 3t$ Processors in $t + 1$ Rounds. *SIAM Journal on Computing (SICOMP)*, 1998.
[7] Fabiola Greve and Sebastien Tixeuil. Knowledge Connectivity vs. Synchrony Requirements for Fault-Tolerant Agreement in Unknown Networks. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Edinburgh, UK*, June 2007.
[8] Leslie Lamport, Robert Shostak, and Marshall Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 1982.
[9] Hammurabi Mendes, Maurice Herlihy, Nitin Vaidya, and Vijay K. Garg. Multidimensional Agreement in Byzantine Systems. *Distributed Computing*, 2015.
[10] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008.
[11] Erfan Taheri and Mohammad Izadi. Byzantine Consensus for Unknown Dynamic Networks. *The Journal of Supercomputing*, 2015.
[12] Lewis Tseng and Nitin H. Vaidya. Fault-Tolerant Consensus in Directed Graphs. In *Symposium on Principles of Distributed Computing (PODC), Donostia-San Sebastián, Spain*, July 2015.