Byzantine Geoconsensus

Joseph Oglio, Kendric Hood, Gokarna Sharma, and Mikhail Nesterenko(✉)

Department of Computer Science, Kent State University, Kent, OH 44242, USA
{joglio,khood5}@kent.edu, {sharma,mikhail}@cs.kent.edu

Abstract. We define and investigate consensus for a set of N processes embedded in the d-dimensional plane, $d \geq 2$, which we call the *Geoconsensus Problem*. The processes have unique coordinates and can communicate with each other through oral messages. Faulty processes are covered by a finite-size convex fault area F. The correct processes know the fault area size but not its location. We prove that the geoconsensus is impossible if all processes may be covered by at most three areas the size of the fault area.

On the constructive side, for $M \geq 1$ fault areas F of arbitrary shape with diameter D, we present a consensus algorithm *BASIC* that tolerates $f \leq N - (2M + 1)$ Byzantine processes provided that there are $9M + 3$ processes with pairwise distance between them greater than D. We present another consensus algorithm *GENERIC* that lifts this distance requirement. For square F with side ℓ, *GENERIC* tolerates $f \leq N - 15M$ Byzantine processes given that all processes are covered by at least $22M$ axis aligned squares of the same size as F. For a circular F of diameter ℓ, *GENERIC* tolerates $f \leq N - 57M$ Byzantine processes if all processes are covered by at least $85M$ circles. We then estimate the tolerance of *GENERIC* for various size combinations of fault and non-fault areas as well as d-dimensional process embeddings, where $d \geq 3$.

1 Introduction

The problem of *Byzantine consensus* [17,24] has been attracting extensive attention from researchers and engineers in distributed systems since its initial statement. The problem has applications in distributed storage [1,2,5,6,16], secure communication [8], safety-critical systems [26], blockchain [21,27,29], and Internet of Things (IoT) [18].

Pease *et al.* [24] defined the problem as follows. Consider a set of N processes with unique identifiers. The processes communicate in synchronous rounds. Each process can communicate with all other processes. Some number $f < N$ of these processes are faulty. The fault is Byzantine which means that the faulty process may behave arbitrarily. The correct processes know the number of the faults f but not the identifiers of the faulty processes. The Byzantine Consensus Problem requires all $N - f$ correct processes to agree on a single value.

Pease *et al.* proved that the maximum number of faults f that can be tolerated by a deterministic algorithm depends on the communication assumptions. Unauthenticated *oral messages* may be modified upon retransmission. If only oral messages are allowed, Pease *et al.* showed that a consensus algorithm may tolerate up $f < N/3$ faults. In case of unforgeable authenticated *written messages*, the consensus is solvable with an arbitrary number of faults $f \leq N$ [24]. It is shown that Byzantine consensus requires

K. Echihabi and R. Meyer (Eds.): NETYS 2021, LNCS 12754, pp. 19–35, 2021.
https://doi.org/10.1007/978-3-030-91014-3_2

at least f rounds of communication [10] and $O(N^2)$ messages [12]. Faster and more efficient solutions are possible if randomized algorithms are allowed [3,14,20].

The original Byzantine Consensus Problem requires the processes to have unique identifiers but does not restrict their location. One way to add some location information to the system is by limiting process communication. In the original problem statement, any pair of processes may communicate directly. Therefore, the communication topology is a complete graph, i.e. a clique. A number of studies relax this connectivity assumption and investigate the problem in arbitrary graphs [24,28] and wireless networks [22]. Several papers study a related problem of *Byzantine broadcast* in incomplete graphs [15,25].

Recently, Lao *et al.* [18] proposed a Byzantine consensus protocol for IoT and blockchain applications, called Geographic-PBFT or simply G-PBFT, which extends a well-known PBFT algorithm [5] to geographic setting. They considered the case of fixed IoT devices embedded in geographic locations for data collection and processing. The location data for these IoT devices can be recorded at deployment, obtained using low-cost GPS receivers or through location estimation algorithms [4,13]. They argued that the fixed IoT devices have more computational power than other mobile IoT devices (e.g., mobile phones and sensors) and are less likely to exhibit Byzantine behavior. Therefore, they exploited the geographical location information of fixed IoT devices to reach consensus. They argued that G-PBFT avoids Sybil attacks [9], reduces the overhead for validating and recording transactions, achieves consensus with high efficiency and low traffic intensity. However, no formal analysis of G-PBFT is given and it is only experimentally validated. Yet, we believe that these developments warrant a study of Byzantine consensus in devices that are aware of their locations.

Our Contribution. In this paper, we formally define and investigate the problem of reaching consensus among processes in fixed geographical locations. We call this variant the *Byzantine Geoconsensus Problem*. We retain all other parameters of the original problem statement. However, if fault locations are not constrained, Geoconsensus differs little from the classic Byzantine consensus: the geographic location of each process can serve as its identifier. Hence, we consider a variant where the faults are constrained geometrically. Specifically, they are limited to a fixed-size *fault area* F. This limitation allows more effective solutions and makes Geoconsensus an interesting problem to study. We are not aware of prior work in Byzantine consensus where processes are embedded in a geometric plane while faulty processes are located in a fixed area.

Let us enumerate the contributions of this paper. Denote by N the number of processes, M the number of fault areas F, D the diameter of F, and f the number of faulty processes. In other words, f is the number of processes covered by fault areas F. Assume that each process can communicate with all other $N-1$ processes and the communication is through oral messages only. Assume that any process covered by a faulty area F may be Byzantine. The correct processes know the size of each faulty area, such as its diameter, number of edges, etc. but do not know their exact locations. In this paper, we make the following five major contributions:

(i) We prove that Geoconsensus is not solvable deterministically if all N processes may be covered by 3 equal size areas F and one of them may be the fault area. This extends to the case of N processes being covered by $3M$ areas F with M

areas being faulty. This is done by adapting the impossibility proof of Pease *et al.* [24] to Geoconsensus.

(ii) We present algorithm *BASIC* that solves Geoconsensus tolerating $f \leq N-(2M+1)$ Byzantine processes, provided that there are $9M+3$ processes with pairwise distance between them greater than D. The idea is for each process to deterministically select a leader in each independent coverage area. Once the leaders are selected, any generic Byzantine consensus algorithm can be run. We use the classic algorithm by Pease *et al.* [24]. Non-leader processes accept the result chosen by the leaders.

(iii) We present algorithm *GENERIC* that removes the pairwise distance assumption of *BASIC* and solves Geoconsensus tolerating $f \leq N-15M$ Byzantine processes, provided that all N processes are covered by $22M$ axis-aligned squares of the same size as the fault area F. For *GENERIC*, we start with covering processes by axis-aligned squares and studying how these squares may intersect with fault areas of various shapes and sizes. We show that determining optimal axis-aligned square coverage is NP-hard and provide constant-ratio approximation algorithms.

(iv) We extend *GENERIC* to circular F tolerating $f \leq N-57M$ Byzantine processes if all N processes are covered by $85M$ circles of same size as F.

(v) We further extend results of (iii) and (iv) to various shape and alignment combinations of fault and non-fault areas and to d-dimensional process embeddings, $d \geq 3$.

Notice that we considered only square and circular fault areas. However, our results can be immediately extended to more complex shapes as they can be inscribed into simple ones. Providing better bounds on more sophisticated analysis of complex shapes beyond simple inscription is left for future research.

Geoconsensus vs. Generic Byzantine Consensus. Let us contrast the results obtained for Geoconsensus to those of the original Byzantine Consensus Problem. The Geoconsensus provides potentially tighter bounds on the number of faults. The original problem establishes the relationship only between N and f, while Geoconsensus also factors the number of fault areas M. Thus, in the original problem, at most $f < N/3$ faulty processes may be tolerated, whereas our results show that as many as $f \leq N - \alpha M$ faults can be tolerated provided that the processes are placed such that at most βM areas (same size as F) are needed to cover them. Here, α and β are both integers with $\beta \geq c \cdot \alpha$ for some constant c.

Geoconsensus also allows to increase the speed and reduce message complexity of the solution. The original consensus requires at least f consecutive rounds of message exchanges and $O(f \cdot N^2)$ messages. The algorithms presented in this paper rely on the selection of a single leader per coverage area. Since each process knows the location of all other processes, this selection is done without message exchanges. Then, the leaders communicate to achieve consensus. Let N processes be covered by X areas of the same size as fault area F. Then, in one round, at most $O(X^2)$ messages need to be exchanged. To reach consensus the algorithm runs for $O(M)$. Thus, in the worst case, at most $O(M \cdot X^2) < O(f \cdot N^2)$ messages are exchanged.

Pease *et al.* [24] showed that it is impossible to solve consensus through oral messages when $N = 3f$ but provided a solution for $N \geq 3f+1$. That is, their impossibility bound is tight. In this paper, however, we were able to show that it is impossible to solve consensus if all N processes are covered by $3M$ areas that are the same size as

Table 1. Notation used throughout the paper.

Symbol	Description
N; $\mathcal{P}$; (x_i, y_i)	Number of processes; $\{p_1, \ldots, p_N\}$; planar coordinates of process p_i
F; D; $\mathcal{F}$	Fault area; diameter of F; a set of fault areas F with $\|\mathcal{F}\| = M$
f	Number of faulty processes
$\mathcal{P}_D$	Processes in $\mathcal{P}$ such that pairwise distance between them is more than D
A (or $A_j(R_i)$); $\mathcal{A}$	Cover area that is of same shape and size as F; a set of cover areas A
$n(F)$	Number of cover areas $A \in \mathcal{A}$ that a fault area F overlaps

F. Yet, for the axis-aligned squares case, the provide the solution where N processes are covered by at least $22M$ areas. Narrowing the gap between the impossible and the achievable is left for further research.

2 Notation, Problem Definition, and Impossibility

Processes. A computer system consists of a set $\mathcal{P} = \{p_1, \ldots, p_N\}$ of N processes. Every process p_i is embedded in the 2-dimensional plane and has unique planar coordinates (x_i, y_i). Each process is aware of coordinates of all the other processes of $\mathcal{P}$ and is capable of sending a message to any of them. The sender of the message may not be spoofed. The communication between processes is through unauthenticated oral messages. This communication is synchronous.

Byzantine Faults. Every process is either permanently correct or faulty. The fault is Byzantine. A faulty process may behave arbitrarily. To simplify the presentation, we assume that all faulty processes are controlled by a unique adversary trying to prevent the system from achieving its task.

Fault Area. The adversary controls the processes as follows. Let the *fault area* F be a finite-size convex area in the plane. Let D be the diameter of F, i.e. the maximum distance between any two points of F. The adversary may place F in any location on the plane. A process p_i is *covered* by F if the coordinates (x_i, y_i) of p_i is either in the interior or on the boundary of F. Any process covered by F may be faulty.

A *fault area set* or just *fault set* is the set $\mathcal{F}$ of identical fault areas F. The size of this set is M, i.e., $|\mathcal{F}| = M$. The adversary controls the placement of all areas in $\mathcal{F}$. Correct processes know the shape and size of the fault areas F. However, correct processes do not know the precise placement of the fault areas $\mathcal{F}$. For example, if $\mathcal{F}$ contains 4 square fault areas F with the side ℓ, then correct processes know that each fault area is of square with side ℓ but do not know where they are located. Table 1 summarizes the notation used in this paper.

Byzantine Geoconsensus. Consider the binary consensus where every correct process is input a value $v \in \{0, 1\}$ and must output an irrevocable decision with the following three properties.

agreement – no two correct processes decide differently;
validity – every correct process outputs a value input to some correct process;
termination – every correct process eventually decides.

Definition 1. *An algorithm solves* the Byzantine Geoconsensus Problem *(or* Geoconsensus *for short) for fault area set $\mathcal{F}$, if every computation produced by this algorithm satisfies the three consensus properties.*

Impossibility of Geoconsensus. Given a certain set of embedded processes $\mathcal{P}$ and single area F, the *coverage number* k of $\mathcal{P}$ by F is the minimum number of such areas required to cover each process of $\mathcal{P}$. We show that Geoconsensus is not solvable if the coverage number k is less than 4. When the coverage number is 3 or less, the problem is reducible to the classic consensus with 3 sets of peers where one of the sets is faulty. Pease *et al.* [24] proved the solution for the latter problem to be impossible. The intuition is that a group of correct processes may not be able to distinguish which of the other two groups is Byzantine and which one is correct. Hence, the correct groups may not reach consensus.

Theorem 1 (Impossibility of Geoconsensus). *Given a set $\mathcal{P}$ of $N \geq 3$ processes and an area F, there exists no algorithm that solves the Byzantine Geoconsensus Problem if the coverage number k of $\mathcal{P}$ by F is less than* 4.

Proof. Set $N = 3 \cdot \kappa$, for some positive integer $\kappa \geq 1$. Place three areas A on the plane in arbitrary locations. To embed processes in $\mathcal{P}$, consider a bijective placement function $f : \mathcal{P} \to \mathcal{A}$ such that κ processes are covered by each area A. Let v and v' be two distinct input values 0 and 1. Suppose one area A is fault area, meaning that all κ processes in that area are faulty.

This construction reduces the Byzantine Goeconsensus Problem to the impossibility construction for the classic Byzantine consensus problem given in the theorem in Section 4 of Pease *et al.* [24] for the 3κ processes out of which κ are Byzantine. □

3 Geoconsensus Algorithm *BASIC*

In this section, we present the algorithm we call *BASIC* that solves Geoconsensus for up to $f < N - (2M + 1), M \geq 1$ faulty processes located in fault area set $\mathcal{F}$ of size $|\mathcal{F}| = M$ provided that $\mathcal{P}$ contains at least $9M + 3$ processes such that the pairwise distance between them is greater than the diameter D of the fault areas $F \in \mathcal{F}$.

The pseudocode of *BASIC* is shown in Algorithm 1. It contains two parts: the leaders selection and the consensus procedure. Let us discuss the selection of leaders. If the distance between two processes is less than the D, they may be covered by a single fault area F. Therefore, the leaders need to be selected such that, pairwise, they are at least D away from each other. Finding the largest set of such leaders is equivalent to computing the maximum independent set in a unit disk graph. This problem is known to be NP-hard [7]. We, therefore, employ a greedy heuristic.

Denote by $Is(G)$ a distance D maximal independent set of a planar graph G. It is defined as a subset of processes of G such that the distance between any pair of processes of Is is more than D, and every process of G that does not belong to Is is at most D away from a process in Is. That is, $p_i \in Is(G)$ if $\forall p_j \neq p_i \in Is, d(p_i, p_j) > D$ and $\forall p_k \in G \setminus Is$, $\exists p_m \in Is$ such that $d(p_k, p_m) \leq D$. Denote by $Nb(p_i, D)$, the distance D neighborhood of process p_i. That is, $p_j \in Nb(p_i, D)$ if $d(p_i, p_j) \leq D$. It is known [19, Lemma 3.3] that in every distance D planar graph, there exists a neighborhood whose induced subgraph contains an independent set of size at most 3.

Algorithm 1: Geoconsensus algorithm *BASIC*.

```
1  Setting: A set $\mathcal{P}$ of $N$ processes positioned at distinct coordinates. Each process can
   communicate with all other processes and knows their coordinates. There are $M \geq 1$
   identical fault areas $F$. The diameter of a fault area is $D$. The locations of any area $F$ is
   not known to correct processes. Each process covered by any $F$ is Byzantine.
2  Input: Each process has initial value either 0 or 1.
3  Output: Each correct process outputs decision subject to Geoconsensus.
4  Procedure for process $p_k \in \mathcal{P}$
5  // leaders selection
6  Let $P_D \leftarrow \emptyset$, $P_C \leftarrow \mathcal{P}$;
7  while $P_C \neq \emptyset$ do
8      let $P_3 \subset P_D$ be a set of processes such that $\forall p_j \in P_3$, $Nb(p_j, D)$ has distance $D$
       independent set of at most 3;
9      let $p_i \in P_3$, located in $(x_i, y_i)$ be the lexicographically smallest process in $P_3$, i.e.
       $\forall p_j \neq p_i \in P_3$ : located in $(x_j, y_j)$ either $x_i < x_j$ or $x_i = x_j$ and $y_i < y_j$;
10     add $p_i$ to $P_D$;
11     remove $p_i$ from $P_C$;
12     $\forall p_j \in Nb(p_i, D)$ remove $p_j$ from $P_C$;
13 // consensus
14 if $p_k \in P_D$ then
15     run PSL algorithm, achieve decision $v$, broadcast $v$, output $v$;
16 else
17     wait for messages with identical decision $v$ from at least $2M + 1$ processes from $\mathcal{P}_D$,
       output $v$;
```

The set of leaders $\mathcal{P}_D \subset \mathcal{P}$ selection procedure operates as follows. A set P_C of leader candidates is iteratively processed. At first, all processes are candidates. All processes whose distance D neighborhood induces a subgraph with an independent set no more than 3 are found. Among those, the process p_i with lexicographically smallest coordinates, i.e. the process in the bottom left corner, is added to the leader set $\mathcal{P}_D$. Then, all processes in $Nb(p_i, D)$ are removed from the leader candidate set $\mathcal{P}_C$. This procedure repeats until $\mathcal{P}_C$ is exhausted.

The second part of *BASIC* relies on the classic consensus algorithm of Pease *et al.* [24]. We denote this algorithm as *PSL*. The input of *PSL* is the set of $3f + 1$ processes such that at most f of them are faulty as well as the initial value 1 or 0 for each process. As output, the correct processes provide the decision value subject to the three properties of the solution to consensus. *PSL* requires $f + 1$ communication rounds.

The complete *BASIC* operates as follows. All processes select leaders in P_D. Then, the leaders run *PSL* and broadcast their decision. The rest of the correct processes, if any, adopt this decision.

Analysis of *BASIC*. The observation below is immediate since all processes run exactly the same deterministic leaders selection procedure.

Observation 1. *For any two processes $p_i, p_j \in \mathcal{P}$, set P_D computed by p_i is the same as set P_D computed by p_j.*

Lemma 1. *If $\mathcal{P}$ contains at least $3x$ processes such that the distance between any pair of such processes is greater than D, then the size of P_D computed by processes in BASIC is at least x.*

Proof. In [19, Theorem 4.7], it is proven that the heuristic we use for the leaders selection provides a distance D independent set P_D whose size is no less than a third of optimal size. Thus, $x \leq |P_D|$. The lemma follows. □

Lemma 2. *Consider a fault area F with diameter D. No two processes in $\mathcal{P}_D$ are covered by F.*

Proof. For any two processes $p_i, p_j \in \mathcal{P}_D$, $d(p_i, p_j) > D$. Since any area F has diameter D, no two processes $> D$ away can be covered by F simultaneously. □

Theorem 2. *Algorithm BASIC solves the Byzantine Geoconsensus Problem for a fault area set $\mathcal{F}$, the size of $M \geq 1$ with fault areas F with diameter D for N processes in $\mathcal{P}$ tolerating $f \leq N-(2M+1)$ Byzantine faults provided that $\mathcal{P}$ contains at least $9M+3$ processes such that their pairwise distance is more than D. The solution is achieved in $M+2$ communication rounds.*

Proof. If $\mathcal{P}$ contains at least $9M+3$ processes whose pairwise distance is more than D, then, according to Lemma 1, each process in *BASIC* selects P_D such that $|P_D| \geq 3M+1$. We have $M \geq 1$ fault areas, i.e., $|\mathcal{F}| = M$. From Lemma 2, a process $p \in \mathcal{P}_D$ can be covered by at most one fault area F. Therefore, if $|P_D| \geq 3M+1$, then it is guaranteed that even if M processes in $\mathcal{P}_D$ are Byzantine, $2M+1$ correct processes in $\mathcal{P}_D$ can reach consensus using *PSL* algorithm.

In the worst case, the adversary may position fault areas of $\mathcal{F}$ such that all but $2M+1$ processes in $\mathcal{P}$ are covered. Hence, *BASIC* tolerates $N-(2M+1)$ faults.

Let us address the number of rounds that *BASIC* requires to achieve Geoconsensus. It has two components executed sequentially: leaders selection and *PSL*. Leaders selection is done independently by all processes and requires no communication. *PSL* takes $M+1$ rounds for the $2M+1$ leaders to arrive at the decision. It takes another round for the leaders to broadcast their decision. Hence, the total number of rounds is $M+2$. □

4 Covering Processes

In this section, in preparation for describing the *GENERIC* Geoconsensus algorithm, we discuss techniques of covering processes by axis-aligned squares and circles. These techniques vary depending on the shape and alignment of the fault area F.

Covering by Squares. The algorithm we describe below covers the processes by square areas A of size $\ell \times \ell$, assuming that the fault areas F are also squares of the same size. Although F may not be axis-aligned, we use axis-aligned areas A to cover processes. Later, we determine the maximum number of such areas A, that non-axis-aligned F may overlap.

Let A be positioned on the plane such that the coordinate of its bottom left corner is (x_1, y_1). The coordinates of its top left, top right, and bottom right corners are respectively $(x_1, y_1 + \ell), (x_1 + \ell, y_1 + \ell)$, and $(x_1 + \ell, y_1)$. Let process p_i be at coordinate (x_i, y_i). We say that p_i is *covered* by A if and only if $x_1 \leq x_i \leq x_1 + \ell$ and $y_1 \leq y_i \leq y_1 + \ell$. We assume that A is *closed*, i.e., process p_i is assumed to be covered by A even if p_i is positioned on the boundary of A.

Let us formally define the problem of covering processes by square areas, which we denote by SQUARE-COVER. Denote by $\mathcal{A}$ a set of square areas A. We say that $\mathcal{A}$ completely covers all N processes if each $p_i \in \mathcal{P}$ is covered by at least one square $A \in \mathcal{A}$.

Definition 2 (The SQUARE-COVER problem). *Suppose N processes are embedded into a 2d-plane such that the coordinates of each process are unique. Given a number* $k \geq 1$*, is there a set* $\mathcal{A}$ *of cardinality k composed of identical square areas* $A = \ell \times \ell$ *that completely covers these N processes?*

Theorem 3. *SQUARE-COVER is NP-Complete.*

Proof. To prove the theorem, we demonstrate that SQUARE-COVER is reducible to the BOX-COVER problem which was shown to be NP-Complete by Fowler *et al.* [11]. BOX-COVER is defined as follows: There is a set of N points on the plane such that each point has unique integer coordinates. A closed box (rigid but relocatable) is set to be a square with side 2 and is axis-aligned. The problem is to decide whether a set of $k \geq 1$ identical axis-aligned closed boxes are enough to completely cover all N points. Fowler *et al.* provided a polynomial-time reduction of 3-SAT to BOX-COVER such that k boxes will suffice if and only if the 3-SAT formula is satisfiable. In this setting, SQUARE-COVER reduces to BOX-COVER for $\ell = 2$. Therefore, the NP-Completeness of BOX-COVER extends to SQUARE-COVER. □

A Greedy Square Cover Algorithm. Since SQUARE-COVER is NP-Complete, we use an efficient greedy approximation algorithm to find a set $\mathcal{A}$ of k_{greedy} axis-aligned square areas $A = \ell \times \ell$ that completely cover all N processes in $\mathcal{P}$. We prove that $k_{greedy} \leq 2 \cdot k_{opt}$, where k_{opt} is the optimal number of axis-aligned squares in any algorithm to cover those N processes. That is, our heuristic is a 2-approximation of the optimal algorithm. We call this algorithm *GSQUARE*. Each process p_i can run *GSQUARE* independently, because p_i knows all required input parameters for *GSQUARE*.

GSQUARE operates as follows. Suppose the coordinates of process $p_i \in \mathcal{P}$ are (x_i, y_i). Let $x_{min} = \min_{1 \leq i \leq N} x_i, x_{max} = \max_{1 \leq i \leq N} x_i, y_{min} = \min_{1 \leq i \leq N} y_i$, and $y_{max} = \max_{1 \leq i \leq N} y_i$. Let R be an axis-aligned rectangle with the bottom left corner at (x_{min}, y_{min}) and the top right corner at (x_{max}, y_{max}). It is immediate that R is the smallest axis-aligned rectangle that covers all N processes. The width of R is $width(R) = x_{max} - x_{min}$ and the height is $height(R) = y_{max} - y_{min}$. See Fig. 1 for illustration.

Cover rectangle R by a set $\mathcal{R}$ of m *slabs* $\mathcal{R} = \{R_1, R_2, \ldots, R_m\}$. The height of each slab R_i is ℓ, except for possibly the last slab R_m whose height may be less than ℓ. The width of each slab is $width(R)$. That is this width is the same is the width of R.

This slab-covering is done as follows. Place slab R_1 at the bottom of R such that its bottom side aligns with the bottom of R and left and right sides align with the corresponding sides of R. Slide R_1 up so that the bottom-most process $p_{min} = (x_{min}, y_{min}) \in \mathcal{P}$ is on the bottom side of R_1. See Fig. 1 for illustration. Now consider

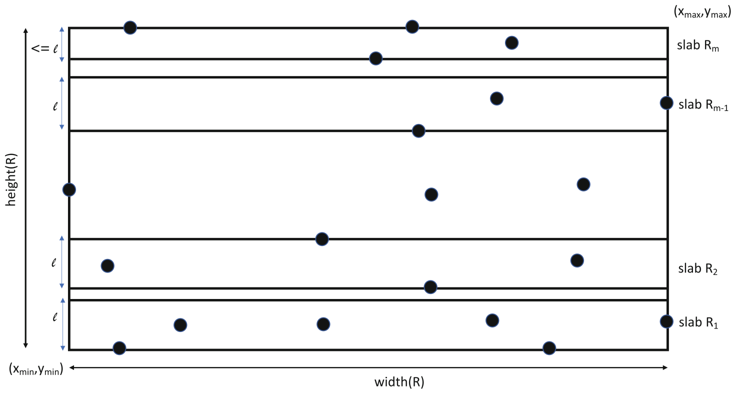

Fig. 1. Selection of axis-aligned smallest enclosing rectangle R covering all N processes in $\mathcal{P}$ and coverage of R by axis-aligned slabs R_i of height ℓ and width $width(R)$. The slabs are selected such that the at least one process is positioned on the bottom side of each slab.

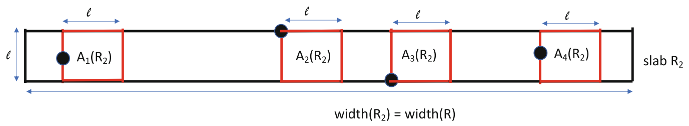

Fig. 2. Selection of axis-aligned areas $A_j(R_2)$ (shown in red) to cover the processes in the slab R_2 of Fig. 1. At least one process is positioned on the left side of each area. (Color figure online)

only the processes in R that are not covered by R_1. Denote this process set by $\mathcal{P}'$. Consider the bottom-most process $y_{min'}$ of $\mathcal{P}'$. Slide the next slab R_2 up so that $p_{min'}$ is on its bottom side. Continue placing slabs over R in this manner until all processes of $\mathcal{P}$ are covered.

We now cover each such slab by axis-aligned square areas $A = \ell \times \ell$. See Fig. 2 for illustration. This square-covering is done similar to slab-covering. Let R_i be a slab to cover. Place the first area A, call it $A_1(R_i)$, on R_i such that the top left corner of A overlaps with the top left corner of slab R_i. Slide $A_1(R_i)$ horizontally to the right until the left-most process in R_i is positioned on the left side of $A_1(R_i)$. Now consider only the processes in R_i not covered by $A_1(R_i)$. Slide the next area A, called $A_2(R_i)$, such that the left-most process in R_i is positioned on the left side of A. Note that there are no uncovered processes between $A_1(R_i)$ and $A_2(R_i)$. Continue to cover all the points in R_i in this manner. The last square may extend past the right side of the slab. Repeat this procedure for every slab of R.

Lemma 3. *Consider any two slabs $R_i, R_j \in \mathcal{R}$ produced by GSQUARE. R_i and R_j do not overlap, i.e., if some process $p \in R_i$, then $p \notin R_j$.*

Proof. It is sufficient to prove this lemma for adjacent slabs. Suppose slabs R_i and R_j are adjacent, i.e., $j = i+1$. According to algorithm *GSQUARE*, after the location of R_i is selected, only processes that are not covered by the slabs so far are considered for the selection of R_j. The first such process lies above the top (horizontal) side of R_i. Hence, there is a non-empty gap between the top side of R_i and the bottom side of R_j. □

Lemma 4. *Consider any two square areas $A_j(R_i)$ and $A_k(R_i)$ selected by GSQUARE in slab $R_i \in \mathcal{R}$. $A_j(R_i)$ and $A_k(R_i)$ do not overlap, i.e., if some process $p \in A_j(R_i)$, then $p \notin A_k(R_i)$.*

See [23] for the proof of the lemma.

Lemma 5. *Consider slab $R_i \in \mathcal{R}$. Let $k(R_i)$ be the number of squares $A_j(R_i)$ to cover all the processes in R_i using GSQUARE. There is no algorithm that can cover the processes in R_i with $k'(R_i)$ number of squares $A_j(R_i)$ such that $k'(R_i) < k(R_i)$.*

Proof. Assume the opposite: there exists an algorithm *X* that can cover processes in R_i with a set of squares whose cardinality k' is less than k used by *GSQUARE*. *GSQUARE* operates such that it places each square A so that some process p lies on the left side of this square. Consider a sequence of such processes: $\sigma \equiv \langle p_1 \cdots p_u, p_{u+1} \cdots p_j \rangle$. Consider any pair of subsequent processes p_u and p_{u+1} in σ with respective coordinates (x_u, y_u) and (x_{u+1}, y_{u+1}). *GSQUARE* covers them with non-overlapping squares with side ℓ. Therefore, $x_u + \ell < x_{u+1}$. That is, the distance between consequent processes in σ is greater than ℓ. Any pair of such processes may not be covered by a single square. Therefore, the number of squares required by the posited algorithm X is at least as large as the number of processes in σ. Since the number of squares placed by *GSQUARE* in slab R_i is k, the number of processes in σ is also k. Therefore, the number of squares required by X is no less than k. This contradicts our initial assumption. □

Let $k_{opt}(\mathcal{R})$ be the number of axis-aligned square areas $A = \ell \times \ell$ to cover all N processes in R in the optimal cover algorithm. We now show that $k_{greedy}(\mathcal{R}) \leq 2 \cdot k_{opt}(\mathcal{R})$, i.e., *GSQUARE* provides 2-approximation. We divide the slabs in the set $\mathcal{R}$ into two sets $\mathcal{R}_{odd}$ and $\mathcal{R}_{even}$. For $1 \leq i \leq m$, let

$$\mathcal{R}_{odd} := \{R_i, i \bmod 2 \neq 0\} \text{ and } \mathcal{R}_{even} := \{R_i, i \bmod 2 = 0\}.$$

Lemma 6. *Let $k(\mathcal{R}_{odd})$ and $k(\mathcal{R}_{even})$ be the total number of (axis-aligned) square areas $A = \ell \times \ell$ to cover the processes in the sets $\mathcal{R}_{odd}$ and $\mathcal{R}_{even}$, respectively. Let $k_{opt}(\mathcal{R})$ be the optimal number of axis-aligned squares $A = \ell \times \ell$ to cover all the processes in $\mathcal{R}$. $k_{opt}(\mathcal{R}) \geq \max\{k(\mathcal{R}_{odd}), k(\mathcal{R}_{even})\}$.*

Proof. Consider two slabs R_i and R_{i+2} for $i \geq 1$. Consider a square $A_j(R_i)$ placed by *GSQUARE*. Consider also two processes $p \in R_i$ and $p' \in R_{i+2}$, respectively. The distance between p and p' is $d(p, p') > \ell$. Therefore, if $A_j(R_i)$ covers p, then it cannot cover $p' \in R_{i+2}$. Hence, no algorithm can produce the number of squares $k_{opt}(\mathcal{R})$ less than the maximum between $k(\mathcal{R}_{odd})$ and $k(\mathcal{R}_{even})$. □

Lemma 7. $k_{greedy}(\mathcal{R}) \leq 2 \cdot k_{opt}(\mathcal{R})$.

Proof. From Lemma 5, we obtain that *GSQUARE* is optimal for each slab R_i. From Lemma 6, we get that for any algorithm $k_{opt}(\mathcal{R}) \geq \max\{k(\mathcal{R}_{odd}), k(\mathcal{R}_{even})\}$. Moreover, the *GSQUARE* produces the total number of squares $k_{greedy}(\mathcal{R}) = k(\mathcal{R}_{odd}) + k(\mathcal{R}_{even})$. Comparing $k_{greedy}(\mathcal{R})$ with $k_{opt}(\mathcal{R})$, we get

$$\frac{k_{greedy}(\mathcal{R})}{k_{opt}(\mathcal{R})} \leq \frac{k(\mathcal{R}_{odd}) + k(\mathcal{R}_{even})}{\max\{k(\mathcal{R}_{odd}), k(\mathcal{R}_{even})\}} \leq \frac{2 \cdot \max\{k(\mathcal{R}_{odd}), k(\mathcal{R}_{even})\}}{\max\{k(\mathcal{R}_{odd}), k(\mathcal{R}_{even})\}} \leq 2.$$

□

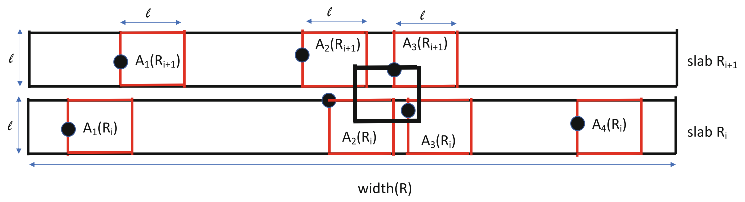

Fig. 3. The maximum overlap of an axis-aligned fault area F with the identical axis-aligned cover squares A of same size.

Covering by Circles. Let $\mathcal{A}$ be the set of identical circles of diameter ℓ. We say that $\mathcal{A}$ completely covers all the processes if every process $p_i \in \mathcal{P}$ is covered by at least one of the circles in $\mathcal{A}$. The problem CIRCLE-COVER of completely covering processes by $\mathcal{A}$ may be formally stated similar to SQUARE-COVER in Definition 2. The following theorem, in turn, can be proven similar to Theorem 3 for SQUARE-COVER.

Theorem 4. *CIRCLE-COVER is NP-Complete.*

A Greedy Circle Cover Algorithm. We call this algorithm *GCIRCLE*. Select the square cover set $\mathcal{A}$ as produced by *GSQUARE*. Consider an individual square $A \in \mathcal{A}$. For each side of A, find a midpoint and place a circle of diameter ℓ there. Observe that thus placed four circles completely cover the area of the square A.

Lemma 8. *Let $k^C_{greedy}(\mathcal{R})$ be the number of circles C of diameter ℓ needed to cover all the processes in $\mathcal{P}$ by algorithm GCIRCLE. Let also $k^C_{opt}(\mathcal{R})$ be the minimum number of such circles used by any algorithm. Then, $k^C_{greedy}(\mathcal{R}) \le 8 \cdot k^C_{opt}(\mathcal{R})$.*

See [23] for the proof of the lemma.

Overlapping Fault Area. The adversary may place the fault area F in any location in the plane. This means that F may not necessarily be axis-aligned. Algorithms *GSQUARE* and *GCIRCLE* produce a cover set $\mathcal{A}$ of axis-aligned squares and circles, respectively. The algorithm we present in the next section needs to know how many areas in $\mathcal{A}$, fault area F overlaps. We now compute the bound for this number. The bound considers both square and circle areas A under various size combinations of fault and non-fault areas. The lemma below is for each $A \in \mathcal{A}$ and F being either squares of side ℓ or circles of diameter ℓ.

Lemma 9. *For the processes of $\mathcal{P}$, consider the cover set $\mathcal{A}$ consisting of the axis-aligned square areas $A = \ell \times \ell$. Place a relocatable square area $F = \ell \times \ell$ in any orientation (not necessarily axis-aligned). F overlaps no more than 7 squares A. If the cover set consists of circles $C \in \mathcal{A}$ of diameter ℓ and F is a circle of diameter ℓ, then F overlaps no more than 28 circles C.*

Proof. Suppose F is axis-aligned. F may overlap at most two squares A horizontally. Indeed, the total width covered by two squares in $\mathcal{A}$ is $> 2\ell$ since the squares do not overlap. Meanwhile, the total width of F is ℓ. Similarly, F may overlap at most two squares vertically. Thus, F may overlap at most 4 distinct axis-aligned areas A. See Fig. 3 for illustration.

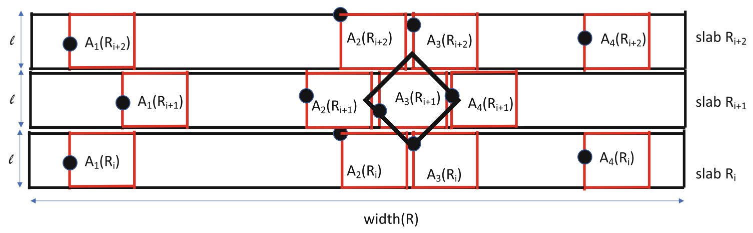

Fig. 4. The maximum overlap of a non-axis-aligned fault area F with the identical axis-aligned cover squares A of the same size.

Consider now that F is not axis-aligned. F can span at most $\sqrt{2}\ell$ horizontally and $\sqrt{2}\ell$ vertically. Therefore, horizontally, F can overlap at most three areas A. Vertically, F can overlap three areas as well. However, not all three areas on the top and bottom rows can be overlapped at once. Specifically, not axis-aligned F can only overlap 2 squares in the top row and 2 in the bottom row. Therefore, in total, F may overlap at most 7 distinct axis-aligned areas. Figure 4 provides an illustration.

Let us consider circular F of size ℓ. It can be inscribed in a square of side ℓ. The square may overlap at most 7 square areas A of side ℓ. Therefore, a circular F can also overlap at most 7 squares. One square area A can be completely covered by 4 circles C. Hence, the circular F may overlap at most $7 \times 4 = 28$ circles C. $\square$

The first lemma below is for each A being an axis-aligned square of side ℓ or a circle of diameter ℓ while F being either a square of side $\ell/\sqrt{2}$ or a circle of diameter $\ell/\sqrt{2}$. The second lemma below considers circular fault area F of diameter $\sqrt{2}\ell$.

Lemma 10. *For the processes in $\mathcal{P}$, consider the cover set $\mathcal{A}$ consisting of the axis-aligned squares $A = \ell \times \ell$. Place a relocatable square area $F = \ell/\sqrt{2} \times \ell/\sqrt{2}$ in any orientation (not necessarily axis-aligned). F overlaps no more than 4 squares A. If the cover set $\mathcal{A}$ consists of circles C of diameter ℓ each, and F is a circle of diameter $\ell/\sqrt{2}$, then F overlaps no more than 16 circles C.*

See [23] for the proof of the lemma.

Lemma 11. *For the processes in $\mathcal{P}$, consider the cover set $\mathcal{A}$ consisting of the axis-aligned square areas $A = \ell \times \ell$. Place a relocatable circular fault area F of diameter $\sqrt{2}\ell$. F overlaps no more than 8 squares A. If $\mathcal{A}$ consists of circles C of diameter ℓ, then circular F of diameter $\sqrt{2}\ell$ overlaps no more than 32 circles C.*

See [23] for the proof of the lemma.

5 Geoconsensus Algorithm *GENERIC*

We are now ready to present an algorithm for solving Geoconsensus that we call *GENERIC*. *GENEREC* follows the same logic as *BASIC* but uses the *GSQUARE* or *GCIRCLE* algorithms, described in the previous section, to obtain the coverage of

Algorithm 2: Geoconsensus algorithm *GENERIC*.

```
1  Setting: A set $\mathcal{P}$ of $N$ processes positioned at distinct planar coordinates. Each process
   can communicate with all other processes and knows the coordinates of all other
   processes. The processes covered by a fault area $F$ at unknown location may be
   Byzantine. There are $M \geq 1$ of identical fault areas $F$ and processes know $M$.
2  Input: Each process has initial value either 0 or 1.
3  Output: Each correct process outputs decision subject to Geoconsensus
4  Procedure for process $p_k$
5  // leaders selection
6  compute the set $\mathcal{A}$ of covers $A_j(R_i)$ using either GSQUARE or GCIRCLE;
7  for every cover $A_j(R_i) \in \mathcal{A}$ do
8      $\mathcal{P}_{min} \leftarrow$ a set of processes with minimum $y$-coordinate among covered by $A_j(R_i)$;
9  if $|\mathcal{P}_{min}| = 1$ then
10     $l_j(A_j(R_i)) \leftarrow$ the only process in $\mathcal{P}_{min}$;
11 else
12     $l_j(A_j(R_i)) \leftarrow$ the process in $\mathcal{P}_{min}$ with minimum $x$-coordinate;
13 // consensus
14 Let $P_L$ be the set of leaders, one for each $A_j(R_i) \in \mathcal{A}$;
15 if $p_k \in \mathcal{P}_L$ then
16     run PSL algorithm, achieve decision $v$, broadcast $v$, output $v$
17 else
18     wait for messages with identical decision $v$ from at least $2M + 1$ processes from $\mathcal{P}_L$,
       output $v$
```

processes in $\mathcal{P}$ by a set $\mathcal{A}$. $\mathcal{A}$ is a set of axis-aligned squares separated such that at most a bounded number of them can be covered by a fault area. A single process per square then participates in the classic consensus.

The pseudocode for *GENERIC* is given in Algorithm 2. The algorithm operates as follows. Each process p_k computes a set $\mathcal{A}$ of covers $A_j(R_i)$ that are of same size as F. Then p_k determines the leader $l_j(A_j(R_i))$ in each cover $A_j(R_i)$. The process in $A_j(R_i)$ with smallest y-coordinate is selected as a leader. If there exist two processes with the same smallest y-coordinate, then the process with the smaller x-coordinate between them is picked. If p_k is selected leader, it participates in running *PSL* [24] (or any other Byzantine consensus algorithm). The leaders run *PSL* then broadcast the achieved decision. The non-leader processes adopt it.

Analysis of *GENERIC*. Let us discuss the correctness and fault-tolerance guarantees of *GENERIC*. In all theorems of this section, *GENERIC* achieves the solution in $M + 2$ communication rounds. The proof for this claim is similar to that for *BASIC* in Theorem 2. Let the fault area $F = \ell \times \ell$ be a, not necessarily axis-aligned, square.

Theorem 5. *Given a set $\mathcal{P}$ of N processes and one square area F positioned at an unknown location such that any process of $\mathcal{P}$ covered by F may be Byzantine. Algorithm GENERIC solves Geoconsensus with the following guarantees:*

- *If $F = \ell \times \ell$ and not axis-aligned and $A = \ell \times \ell$, $f \leq N - 15$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 22$.*
- *If $F = \ell \times \ell$ and axis-aligned and $A = \ell \times \ell$, $f \leq N - 9$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 13$.*
- *If $F = \ell/\sqrt{2} \times \ell/\sqrt{2}$ but $A = \ell \times \ell$, then even if F is not axis aligned, $f \leq N - 9$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 13$.*

Proof. We start by proving the first case. *GSQUARE* produces the cover set $\mathcal{A}$ of at least $|\mathcal{A}| = 22$ areas. From Lemma 9, we obtain that a square fault area $F = \ell \times \ell$, regardless of orientation and location, can overlap at most $n(F) = 7$ axis-aligned squares $A = \ell \times \ell$. *GENERIC* runs *PSL* algorithm using the single leader process in each area A. For its correct operation, *PSL* requires the number of correct processes to be more than twice the number of faulty ones. This is guaranteed since at least $2 \cdot |\mathcal{A}|/3 + 1 = 2 \cdot 22/3 + 1 \geq 2 \cdot n(F) + 1 = 2 \cdot 7 + 1$ leader processes are correct and they can reach consensus using *PSL*.

Let us address the second case. An axis-aligned square F can overlap at most $n(F) = 4$ axis-aligned squares A. Therefore, when $|\mathcal{A}| \geq 13$, we have that $|\mathcal{A}| - 9 \geq 2 \cdot n(F) + 1$ leader processes are correct and they can reach consensus. In this case, $f \leq N - 9$ processes can be covered by F and still they all can be tolerated.

Let us now address the third case, when $F = \ell/\sqrt{2} \times \ell/\sqrt{2}$ but $A = \ell \times \ell$. Regardless of its orientation, F can overlap at most $n(F) = 4$ squares A. Therefore, $|\mathcal{A}| \geq 13$ is sufficient for consensus and total $f \leq N - 9$ processes can be tolerated. □

For the set $\mathcal{F}$ of multiple fault areas F with $|\mathcal{F}| = M$, Theorem 5 extends as follows.

Theorem 6. *Given a set $\mathcal{P}$ of N processes and a set of $M \geq 1$ of square areas F positioned at unknown locations such that any process of $\mathcal{P}$ covered by any F may be Byzantine. Algorithm GENERIC solves Geoconsensus with the following guarantees:*

- *If each $F = \ell \times \ell$ and not axis-aligned and $A = \ell \times \ell$, $f \leq N - 15M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 22M$.*
- *If each $F = \ell \times \ell$ and axis-aligned and $A = \ell \times \ell$, $f \leq N - 9M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 13M$.*
- *If each $F = \ell/\sqrt{2} \times \ell/\sqrt{2}$ but $A = \ell \times \ell$, then even if F is not axis-aligned, $f \leq N - 9M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 13M$.*

Proof. The proof for the case of $M = 1$ extends to the case of $M > 1$ as follows. Theorem 5 gives the bounds $f \leq N - \gamma$ and $|\mathcal{A}| \geq \delta$ for one fault area for some positive integers γ, δ. For M fault areas, M separate $|\mathcal{A}|$ sets are needed, with each set tolerating a single fault area F. Therefore, the bounds of Theorem 5 extend to multiple fault areas with a factor of M, i.e., *GENERIC* needs $M \cdot \delta$ covers and $f \leq N - M \cdot \gamma$ faulty processes can be tolerated. Using the appropriate numbers from Theorem 5 provides the claimed bounds. □

We have the following theorem for the case of circular fault set $\mathcal{F}$, $|\mathcal{F}| = M \geq 1$.

Theorem 7. *Given a set $\mathcal{P}$ of N processes and a set of $M \geq 1$ circles F positioned at unknown locations such that any process of $\mathcal{P}$ covered by F may be Byzantine. Algorithm GENERIC solves Geoconsensus with the following guarantees:*

- *If each F and A are circles of diameter ℓ, $f \leq N - 57M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 85M$.*
- *If each F is a circle of diameter $\sqrt{2}\ell$ and A is a circle of diameter ℓ, $f \leq N - 65M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 97M$.*
- *If each F is a circle of diameter $\ell/\sqrt{2}$ and A is a circle of diameter ℓ, $f \leq N - 33M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 49M$.*

See [23] for the proof of the theorem.

6 Extensions to Higher Dimensions

Our approach can be extended to solve Geoconsensus in d-dimensions, $d \geq 3$. *BASIC* extends as is. *GENERIC* runs correctly so long as we determine (i) the cover set $\mathcal{A}$ of appropriate dimension and (ii) the overlap bound – the maximum number of d-dimensional covers A that the fault area F may overlap. The bound on f then depends on M and the cover set size $|\mathcal{A}|$. In what follows, we discuss 3-dimensional space. The still higher dimensions can be treated similarly.

If $d = 3$, the objective is to cover the embedded processes of $\mathcal{P}$ by cubes of size $\ell \times \ell \times \ell$ or spheres of diameter ℓ. It can be shown that the greedy cube (sphere) cover algorithm, let us call it *GCUBE* (*GSPHERE*), provides $2^{d-1} = 4$ (16) approximation of the optimal cover. The idea is to appropriately extend the 2-dimensional slab-based division and axis-aligned square-based covers discussed in Sect. 4 to 3-dimensions with rectangular cuboids and cube-based covers. See [23] for detailed discussion. We summarize the results for cubic covers and cubic fault areas in Theorem 8.

Theorem 8. *Given a set $\mathcal{P}$ of N processes embedded in 3-d space and a set of $M \geq 1$ of cubic areas F at unknown locations, such that any process of $\mathcal{P}$ covered by F may be Byzantine. Algorithm GENERIC solves Geoconsensus with the following guarantees:*

- *If F is a cube of side ℓ and not axis-aligned and A is also a cube of side ℓ, $f \leq N - 55M$ faulty processes can be tolerated given that the cover set $|\mathcal{A}| \geq 82M$.*
- *If F is a cube of side ℓ and axis-aligned and A is also a cube of side ℓ, $f \leq N - 17M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 25M$.*
- *If F is a sphere of diameter ℓ and A is a sphere of diameter ℓ, $f \leq N - 433M$ faulty processes can be tolerated given that $|\mathcal{A}| \geq 649M$.*

7 Concluding Remarks

In light of the recent development of location-based consensus protocols, such as G-PBFT [18], we have formally defined and studied the consensus problem of processes that are embedded in a d-dimensional plane, $d \geq 2$, on fixed locations known to every other process. We have explored both the possibility as well bounds for a solution to this

Geoconsensus. Our results establish trade-offs between the three parameters N, M, and f, in contrast to the trade-off between only two parameters N and f in the Byzantine consensus literature. Our results also show the dependency of the tolerance guarantees on the shapes and alignment of the fault areas.

Acknowledgement. This research was supported in part by National Science Foundation under Grants No. CCF-1936450 and CAREER CNS-2045597.

References

1. Abd-El-Malek, M., Ganger, G.R., Goodson, G.R., Reiter, M.K., Wylie, J.J.: Fault-scalable Byzantine fault-tolerant services. ACM SIGOPS Oper. Syst. Rev. **39**(5), 59–74 (2005)
2. Adya, A., et al.: Farsite: federated, available, and reliable storage for an incompletely trusted environment. ACM SIGOPS Oper. Syst. Rev. **36**(SI), 1–14 (2002)
3. Ben-Or, M., Kelmer, B., Rabin, T.: Asynchronous secure computations with optimal resilience. In: Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing, pp. 183–192 (1994)
4. Bulusu, N., Heidemann, J., Estrin, D., Tran, T.: Self-configuring localization systems: design and experimental evaluation. ACM Trans. Embed. Comput. Syst. (TECS) **3**(1), 24–60 (2004)
5. Castro, M., Liskov, B.: Practical Byzantine fault tolerance and proactive recovery. ACM Trans. Comput. Syst. (TOCS) **20**(4), 398–461 (2002)
6. Castro, M., Rodrigues, R., Liskov, B.: Base: using abstraction to improve fault tolerance. ACM Trans. Comput. Syst. (TOCS) **21**(3), 236–269 (2003)
7. Clark, B.N., Colbourn, C.J., Johnson, D.S.: Unit disk graphs. Discret. Math. **86**(1–3), 165–177 (1990)
8. Cramer, R., Gennaro, R., Schoenmakers, B.: A secure and optimally efficient multi-authority election scheme. Eur. Trans. Telecommun. **8**(5), 481–490 (1997)
9. Douceur, J.R.: The Sybil attack. In: Druschel, P., Kaashoek, F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45748-8_24
10. Fischer, M.J., Lynch, N.A.: A lower bound for the time to assure interactive consistency. Inf. Process. Lett. **14**(4), 183–186 (1982)
11. Fowler, R.J., Paterson, M., Tanimoto, S.L.: Optimal packing and covering in the plane are NP-complete. Inf. Process. Lett. **12**(3), 133–137 (1981)
12. Hadzilacos, V., Halpern, J.Y.: Message-optimal protocols for Byzantine agreement. Math. Syst. Theory **26**(1), 41–102 (1993)
13. Hightower, J., Borriello, G.: Location systems for ubiquitous computing. Computer **34**(8), 57–66 (2001)
14. King, V., Saia, J.: Breaking the O(n^2) bit barrier: scalable Byzantine agreement with an adaptive adversary. J. ACM (JACM) **58**(4), 1–24 (2011)
15. Koo, C.Y.: Broadcast in radio networks tolerating Byzantine adversarial behavior. In: PODC, pp. 275–282 (2004)
16. Kubiatowicz, J., et al.: OceanStore: an architecture for global-scale persistent storage. ACM SIGOPS Oper. Syst. Rev. **34**(5), 190–201 (2000)
17. Lamport, L., Shostak, R., Pease, M.: The Byzantine generals problem. ACM Trans. Program. Lang. Syst. **4**(3), 382–401 (1982)
18. Lao, L., Dai, X., Xiao, B., Guo, S.: G-PBFT: a location-based and scalable consensus protocol for IoT-blockchain applications. In: IPDPS, pp. 664–673 (2020)

19. Marathe, M.V., Breu, H., Hunt, H.B., III., Ravi, S.S., Rosenkrantz, D.J.: Simple heuristics for unit disk graphs. Networks **25**(2), 59–68 (1995)
20. Martin, J.P., Alvisi, L.: Fast Byzantine consensus. IEEE Trans. Dependable Secure Comput. **3**(3), 202–215 (2006)
21. Miller, A., Xia, Y., Croman, K., Shi, E., Song, D.: The honey badger of BFT protocols. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, pp. 31–42 (2016)
22. Moniz, H., Neves, N.F., Correia, M.: Byzantine fault-tolerant consensus in wireless ad hoc networks. IEEE Trans. Mob. Comput. **12**(12), 2441–2454 (2012)
23. Oglio, J., Hood, K., Sharma, G., Nesterenko, M.: Byzantine geoconsensus. Technical report. 2010.02436 [cs.DC], arXiv, October 2020. http://arxiv.org/abs/2010.02436
24. Pease, M., Shostak, R., Lamport, L.: Reaching agreement in the presence of faults. J. ACM **27**(2), 228–234 (1980). https://doi.org/10.1145/322186.322188
25. Pelc, A., Peleg, D.: Broadcasting with locally bounded Byzantine faults. Inf. Process. Lett. **93**(3), 109–115 (2005). https://doi.org/10.1145/322186.322188
26. Rushby, J.: Bus architectures for safety-critical embedded systems. In: Henzinger, T.A., Kirsch, C.M. (eds.) EMSOFT 2001. LNCS, vol. 2211, pp. 306–323. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45449-7_22
27. Sousa, J., Bessani, A., Vukolic, M.: A Byzantine fault-tolerant ordering service for the hyperledger fabric blockchain platform. In: DSN, pp. 51–58. IEEE (2018)
28. Vaidya, N.H., Tseng, L., Liang, G.: Iterative approximate Byzantine consensus in arbitrary directed graphs. In: PODC, pp. 365–374 (2012)
29. Zamani, M., Movahedi, M., Raykova, M.: Rapidchain: scaling blockchain via full sharding. In: CCS, pp. 931–948 (2018)