Byzantine Convex Consensus: An Optimal Algorithm*

Lewis Tseng[1,3], and Nitin Vaidya[2,3]

[1] Department of Computer Science,
[2] Department of Electrical and Computer Engineering, and
[3] Coordinated Science Laboratory
University of Illinois at Urbana-Champaign

Email: {ltseng3, nhv}@illinois.edu

Technical Report

July 9, 2013

Abstract

Much of the past work on asynchronous approximate Byzantine consensus has assumed *scalar* inputs at the nodes [4, 8]. Recent work has yielded approximate Byzantine consensus algorithms for the case when the input at each node is a d-dimensional vector, and the nodes must reach consensus on a vector in the convex hull of the input vectors at the fault-free nodes [9, 13]. The d-dimensional vectors can be equivalently viewed as *points* in the d-dimensional Euclidean space. Thus, the algorithms in [9, 13] require the fault-free nodes to decide on a point in the d-dimensional space.

In our recent work [12], we proposed a generalization of the consensus problem, namely *Byzantine convex consensus* (BCC), which allows the decision to be a *convex polytope* in the d-dimensional space, such that the decided polytope is within the convex hull of the input vectors at the fault-free nodes. We also presented an asynchronous approximate BCC algorithm.

In this paper, we propose a new BCC algorithm with optimal fault-tolerance that also agrees on a convex polytope that is as *large* as possible under adversarial conditions. Our prior work [12] does not guarantee the optimality of the output polytope.

*This research is supported in part by National Science Foundation award CNS 1059540. 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.

1 Introduction

Much of the past work on asynchronous approximate Byzantine consensus has assumed *scalar* inputs at the nodes [4, 8]. Recent work has yielded approximate Byzantine consensus algorithms for the case when the input at each node is a d-dimensional vector, and the nodes must reach consensus on a vector in the convex hull of the input vectors at the fault-free nodes [9, 13]. The d-dimensional vectors can be equivalently viewed as *points* in the d-dimensional Euclidean space. Thus, the algorithms in [9, 13] require the fault-free nodes to decide on a point in the d-dimensional space. In our recent work [12], we considered a generalized problem, namely *Byzantine convex consensus* (BCC), which allows the decision to be a *convex polytope* in the d-dimensional space, such that the decided polytope is within the convex hull of the input vectors at the fault-free nodes. In this paper, we propose an asynchronous BCC algorithm with optimal fault-tolerance that reaches consensus on the convex polytope with an *optimal* output polytope (as defined later). This is an improvement over our previous algorithm in [12] that does not guarantee optimality of the output polytope.

The system under consideration is an *asynchronous* system consisting of n nodes, of which at most f may be Byzantine faulty. The Byzantine faulty nodes may behave in an arbitrary fashion, and may collude with each other. Each node i has a d-dimensional vector of reals as its *input* x_i. All nodes can communicate with each other directly on reliable and FIFO (first-in first-out) channels. Thus, the underlying communication graph can be modeled as a *complete graph*, with the set of nodes being $V = \{1, 2, \cdots, n\}$. The impossibility of *exact* consensus in asynchronous systems [5] applies to BCC as well. Therefore, we consider the *Approximate BCC* problem with the following requirements:

- **Validity**: The *output* (or *decision*) at each fault-free node must be a convex polytope in the convex hull of the d-dimensional input vectors at the fault-free nodes. (In a degenerate case, the output polytope may simply be a single *point*.)
- **ϵ-Agreement**: For any $\epsilon > 0$, the *Hausdorff distance* (defined below) between the output polytopes at any two fault-free nodes must be at most ϵ.
- **Termination:** Each fault-free node must terminate within a finite amount of time.

The motivation behind reaching consensus on a convex polytope is that a solution to BCC is expected to also facilitate solutions to a large range of consensus problems (e.g., Byzantine vector consensus [9, 13], or convex function optimization over a convex hull of the inputs at fault-free nodes). Future work will explore these potential applications.

Definition 1 *For two convex polytopes h_1, h_2, the* Hausdorff distance *is defined as [7]*

$$\boldsymbol{d}_H(h_1, h_2) \quad = \quad \max \quad \left\{ \quad \max_{p_1 \in h_1} \ \min_{p_2 \in h_2} d(p_1, p_2), \qquad \max_{p_2 \in h_2} \ \min_{p_1 \in h_1} d(p_1, p_2) \quad \right\}$$

where $d(p, q)$ is the Euclidean distance between points p and q.

Optimality of the Output Polytope: The BCC algorithm proposed in this paper allows the nodes to agree on an output polytope that is "optimal" in the sense defined below.

Definition 2 *Let A be an algorithm that solves Byzantine convex consensus (BCC). Algorithm A is said to reach consensus on an* optimal *convex polytope if for any BCC algorithm B, there* *exist a behavior of the faulty nodes and a message delay pattern such that, at each fault-free node, the output polytope obtained using algorithm B is contained in the output polytope obtained using algorithm A.*

We show that the BCC algorithm proposed here allows the nodes to agree on a polytope that is guaranteed to contain a polytope that is named I_Z in later analysis. I_Z is a function of the inputs at some of the fault-free nodes. We show that, for any correct BCC algorithm, there exists an execution in which the fault-free nodes must agree on a polytope that is equal to or contained in I_Z. Thus, as per Definition 2, the output polytope chosen by our algorithm is optimal.

Lower Bound on n: As noted above, [9, 13] consider the problem of reaching approximate Byzantine consensus on a vector (or a point) in the convex hull of the d-dimensional input vectors at the fault-free nodes, and show that $n \geq (d+2)f+1$ is necessary. [10] generalizes the same lower bound to colorless tasks. The lower bound proof in [9, 13] also implies that $n \geq (d+2)f+1$ is necessary to ensure that BCC is solvable. We do not reproduce the lower bound proof here, but in the rest of the paper, we assume that $n \geq (d+2)f+1$, and also that $n \geq 2$ (because consensus is trivial when $n = 1$).

2 Preliminaries

Some notations introduced throughout the paper are summarized in Appendix A. In this section, we introduce operations $\mathcal{H}$, H_l, H, and two communication primitives, *reliable broadcast* and *stable vector*, used later in the paper.

Definition 3 *Given a set of points X, $\mathcal{H}(X)$ is defined as the convex hull of the points in X.*

Definition 4 *Suppose that ν convex polytopes $h_1, h_2, \cdots, h_\nu$, and ν constants $c_1, c_2, \cdots, c_\nu$ are given such that (i) $0 \leq c_i \leq 1$ and $\sum_{i=1}^{\nu} c_i = 1$, and (ii) for $1 \leq i \leq \nu$, if $c_i \neq 0$, then $h_i \neq \emptyset$. Linear combination of these convex polytopes, $H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$, is defined as follows:*

- *Let $Q := \{i \mid c_i \neq 0,\ 1 \leq i \leq \nu\}$.*
- *$p \in H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$ if and only if*

$$\text{for each } i \in Q, \text{ there exists } p_i \in h_i, \quad \text{such that } \ p = \sum_{i \in Q} c_i p_i \tag{1}$$

Note that a convex polytope may possibly consist of a single point. Because h_i's above are all convex, $H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$ is also a convex polytope (proof included in Appendix B for completeness). The parameters for H_l consist of two lists, a list of polytopes $h_1, \cdots, h_\nu$, and a list of weights $c_1, \ldots, c_\nu$. With a slight abuse of notation, we will specify one or both of these lists as either a *row vector* or a *multiset*, with the understanding that the *row vector* or *multiset* here represent an ordered list of its elements.

Function H below is called in our algorithm with parameters $(\mathcal{V}, t)$ wherein t is a round index ($t \geq 0$) and $\mathcal{V}$ is a set of tuples of the form $(h, j, t-1)$, where j is a node identifier; when $t = 0$, h is a set of received messages in the previous round, and when $t > 0$, h is a convex polytope.

Function $H(\mathcal{V}, t)$, $t \geq 0$:

If $t = 0$**:**

- For each tuple (x,k), where x is a point and k is a node identifier, define $N(x,k) := |\{\, l \mid (\mathcal{I}, l, -1) \in \mathcal{V} \text{ and } (x,k,-1) \in \mathcal{I}\}|$.
- Define set $Y := \{\,(x,k) \mid N(x,k) \geq f+1\}$.
- Define multiset $X := \{\, x \mid (x,k) \in Y\}$. Size of multiset X is identical to the size of set Y. In a multiset, same element may appear multiple times.
- `temp` $:= \cap_{C \subseteq X, |C|=|X|-f} \ \mathcal{H}(C)$.
  The intersection above is over the convex hulls of the subsets of X of size $|X| - f$.
- Return `temp`.

If $t > 0$**:**

- Define multiset $X := \{h \mid (h, j, t-1) \in \mathcal{V}\}$. In our use of function H, each $h \in X$ is always non-empty.
- `temp` $:= H_l(X; \frac{1}{|X|}, \cdots, \frac{1}{|X|})$. Note that all the weights here are equal to $\frac{1}{|X|}$.
- Return `temp`.

Communication Primitives: As seen later, our algorithm proceeds in asynchronous rounds. We label the *preliminary round* as round -1, and the remaining rounds as rounds 0, 1, 2, etc. For communication between the nodes, we use the *reliable broadcast* primitive [1] and *stable vector* primitive [2, 10], which are also used in other related work [9, 13, 10]. Note that we adopt the version of stable vector presented in [10]. In particular, in round -1 (preliminary round), we use *stable vector* and *reliable broadcast* both, as explained below. In rounds 0 and larger, we only use *reliable broadcast.*

Round t**,** $t \geq 0$**:** In round t, $t \geq 0$, each node performs reliable broadcast of one message using `RBSend`. Each message sent using `RBSend` consists of a 3-tuple of the form (v, i, t): here, i denotes the sender node's identifier, t is round index, and v is message value (the value v itself is often a tuple). The operation `RBSend`(v, i, t) is used by node i to perform *reliable broadcast* of (v, i, t) in round t. Each such message may be eventually reliably received by a fault-free node. When message (v, j, t) is *reliably received* by some node i, the event `RBRecv`(v, j, t) is said to have occurred at node i (note that i may possibly be equal to j). The second element in a reliably received 3-tuple message, namely j above, is always identical to the identifier of the node that performed the corresponding reliable broadcast. An appropriate handler is executed on each such `RBRecv` event, as described in the algorithm.

Round -1**:** In round -1, each node i performs reliable broadcast of message $(x_i, i, -1)$ using `RBSend`$(x_i, i, -1)$, where x_i is the input vector at node i. The *stable vector* primitive `SVRecv`(-1) is then invoked via a blocking call. `SVRecv`(-1) eventually returns at each fault-free node with a set containing at least $(n-f)$ messages of the form $(\cdot, \cdot, -1)$. These sets have the desirable property that the sets returned to *all* the fault-free nodes contain at least $(n-f)$ messages in common. Messages sent by some of the nodes using `RBSend` in round -1 may not be included in the set returned to a fault-free node by `SVRecv`(-1). Each such message may be later delivered to the fault-free node via a `RBRecv` event. Thus, for round -1, at fault-free node i, the `RBRecv` events may occur only for messages that are not returned by `SVRecv`. An appropriate handler is executed on each such `RBRecv` event, as described in the algorithm below.

With a slight abuse of terminology, when we say that node j reliably receives $(v, i, -1)$, we mean that either (i) $(v, i, -1)$ is included the set returned by `SVRecv`(-1) to node j, or (ii) event `RBRecv`$(v, i, -1)$ occurs at node j after `SVRecv`(-1) had already returned.

Properties of Communication Primitives: Each fault-free node performs one reliable broadcast (`RBSend`) in each round of our algorithm. *Reliable broadcast* and *stable vector* achieve the properties listed below, as proved previously [1, 10]. In the properties below, round index $r \geq -1$.

- **Fault-Free Integrity:** If a fault-free node i *never* reliably broadcasts (v, i, r), then no fault-free node ever reliably receives (v, i, r).
- **Fault-Free Liveness:** If a fault-free node i performs reliable broadcast of (v, i, r), then each fault-free node eventually reliably receives (v, i, r).
- **Global Uniqueness:** If two fault-free nodes i, j reliably receive (v, k, r) and (w, k, r), respectively, then $v = w$, even if node k is faulty.
- **Global Liveness:** For any two fault-free nodes i, j, if i reliably receives (v, k, r), then j will eventually reliably receive (v, k, r), even if node k is faulty.
- **Fault-free Containment**: For fault-free nodes i, j, let R_i, R_j be the set of messages returned to nodes i, j by stable vector primitive `SVRecv`(-1) in round -1, respectively. Then, $|R_i| \geq n - f$, $|R_j| \geq n - f$, and either $R_i \subseteq R_j$ or $R_j \subseteq R_i$.

The last above property ensures that, in round -1, all the fault-free nodes receive at least $(n - f)$ identical messages. In addition to the above property, the following property is also ensured:

- Any fault-free node i, for any t and j, reliably receives (either via `SVRecv` or `RBRecv`) at most one message of the form $(*, j, t)$.

This property is implemented easily by requiring each node to, after receiving the first message of the form $(*, j, t)$, to simply ignore any further messages of that form. In our algorithm, each fault-free node j reliably broadcasts exactly one message of the form $(*, j, t)$ in any round t. Thus, the above property is useful to avoid responding to multiple messages with the same round index from a faulty node.

3 Proposed Algorithm: Optimal Verified Averaging

The proposed algorithm (named *Optimal Verified Averaging*) proceeds in asynchronous rounds. The input at each node i is a d-dimensional vector of reals, denoted as x_i. The initial round is called a *preliminary round*, and also referred to as round -1. Subsequent rounds are named round 0, 1, 2, etc. In each round $t \geq 0$, each node i computes a state variable h_i, which represents a convex polytope in the d-dimensional Euclidean space. We will refer to the value of h_i at the *end* of the t-th round performed by node i as $h_i[t]$, $t \geq 0$. Thus, for $t \geq 1$, $h_i[t-1]$ is the value of h_i at the *start* of the t-th round at node i.

Similar to the algorithm in our prior work [12], we use a technique named *verification* to ensures that if a faulty node deviates from the algorithm specification (except possibly choosing an invalid input vector), then its incorrect messages will be ignored by the fault-free nodes. The *verification* mechanism is motivated by prior work by other researchers [3]. With *verification*, aside from choosing a bad input, a faulty node cannot cause any other damage to the execution.

Before we present the proposed algorithm, we introduce a convention for the brevity of presentation:

- When we say that $(*, i, t) \in \mathcal{V}$, we mean that *there exists* z *such that* $(z, i, t) \in \mathcal{V}$.
- When we say that $(*, i, t) \notin \mathcal{V}$, we mean that $\forall z,\ (z, i, t) \notin \mathcal{V}$.

The proposed *Optimal Verified Averaging* algorithm for node $i \in V$ is presented below. All references to line numbers in our discussion refer to numbers listed on the right side of the algorithm pseudo-code. Recall that in round $t \geq 0$, whenever a message is reliably received by any node, a handler is called to process that message. In round -1, messages that are not delivered by `SVRecv`(-1) may be later reliably received via a `RBRecv` event, invoking the corresponding handler. Multiple such handlers may execute *concurrently* at a given node. For correct behavior, line 7, and lines 11-16 in the algorithm are atomically executed in a *critical section*. Thus, even though multiple event handlers may execute simultaneously, execution of line 7 in one instance of the handler is not interleaved with execution of any other handler instance; similarly, execution of lines 11-16 in one instance of the handler is not interleaved with execution of any other handler instance.

- **Round** -1: In round -1, each node i uses `RBSend` to reliably broadcast $(x_i, i, -1)$ where x_i is its input (line 1). Each node then calls the primitive `SVRecv`(-1), which eventually returns with a set of messages tagged with index -1. These messages are stored in $\mathit{Verified}_i[-1]$ and $\mathit{Verified}_i^c[-1]$ both (lines 2 and 3). At this point, $\mathit{Verified}_i^c[-1] = \mathit{Verified}_i[-1]$. At line 4, node i also sets $h_i[-1]$ to be equal to a default value $\emptyset$ (because $h_i[-1]$ does not affect future computations). Afterwards, each node can proceed to round 0 (line 5).

  Note that reliable broadcast of a message by some node j may not be received by node i using `SVRecv` at line 2; however, the message may be later reliably received by node i via a `RBRecv` event (line 6). Line 7 specifies the behavior of the event handler for event `RBRecv`$(x, j, -1)$ at node i. Whenever a message of the form $(x, j, -1)$ is reliably received via event `RBRecv`$(x, j, -1)$ (line 6), the set $\mathit{Verified}_i[-1]$ is updated (line 7). Since line 7 is performed atomically, $\mathit{Verified}_i[-1]$ may continue to grow even after node i has proceeded to round 0; however, $\mathit{Verified}_i^c[-1]$ is not modified again. Note that a message received by node i via `SVRecv` (at line 2) or `RBRecv` (at line 6) may possibly have been reliably broadcast by node i itself.

- **Round** $t \geq 0$**:** In round $t \geq 0$, *Optimal Verified Averaging* adopts a similar structure to round -1, with one key difference: stable vector (`SVRecv`) is not used in these rounds, and all messages are received via `RBRecv` events. In round $t \geq 0$, node i first reliably broadcasts message $((h_i[t-1], \mathit{Verified}_i^c[t-1]), i, t)$ (line 8). Lines 9-16 specify the event handler for event `RBRecv`$((h, \mathcal{V}), j, t)$ at node i. Whenever a message of the form $((h, \mathcal{V}), j, t)$ is reliably received from node j (line 9), node i first waits until its own set $\mathit{Verified}_i[t-1]$ becomes large enough to contain $\mathcal{V}$. Note that $\mathit{Verified}_i[t-1]$ is initially computed in the round $t-1$, but it may continue to grow even after node i proceeds to round t. If the condition $\mathcal{V} \subseteq \mathit{Verified}_i[t-1]$ never becomes true, then this message is not processed further.

  Recall that lines 11-16 are performed atomically. The message $((h, \mathcal{V}), j, t)$ is considered to be *verified* if Procedure `Verify`$(h, \mathcal{V}, j, t)$ returns `TRUE` (line 11). As shown in the pseudo-code for `Verify`, the verification checks performed are different for $t = 0$, $t = 1$ and $t \geq 2$. If a message is thus verified, then some elements in the message are added to $\mathit{Verified}_i[t]$ via Procedure `Add`$(\cdot)$ (line 12). As shown in the pseudo-code for `Add`, the elements added are different for $t = 0$, and $t \geq 1$.

Procedure `Proceed`(t) at line 13 determines whether set $\mathit{Verified}_i[t]$ has grown to a point where it is appropriate to compute the new state $h_i[t]$ (line 15) and proceed to round $t+1$ (line 16). The checks performed in `Proceed`(t) are different for $t = 0$ and $t \geq 1$, as shown in the pseudo-code for `Proceed`. The value of $\mathit{Verified}_i[t]$ used to compute $h_i[t]$ is stored in $\mathit{Verified}_i^c[t]$ (line 14).

New messages may still be added to $\mathit{Verified}_i[t]$ if events of the form $\texttt{RBRecv}((h, \mathcal{V}), j, t)$ occur after node i has proceeded to round $t+1$. Thus, $\mathit{Verified}_i[t]$ may continue to grow even after node i has proceeded to round 1; however, $\mathit{Verified}_i^c[t]$ is not modified again, and remains unchanged after it is set at line 14.

Optimal Verified Averaging Algorithm: Steps performed at node i shown below.

The algorithm terminates after t_{end} rounds, where t_{end} is a constant, defined in (19).

Initialization: All sets used below are initialized to $\emptyset$.

Preliminary Round (Round -1) at node i:

1 • $\texttt{RBSend}(x_i, i, -1)$

2 • $\mathit{Verified}_i[-1] := \texttt{SVRecv}(-1)$

3 • $\mathit{Verified}_i^c[-1] := \mathit{Verified}_i[-1]$

4 • $h_i[-1] := \emptyset$

5 • Proceed to Round 0

Comment: Message sent by j using `RBSend` may not be received by i using `SVRecv` at line 2. Due to Fault-free Liveness property of the primitive, this message will later be received by i using `RBRecv` at line 6 below.

6 • **Event handler for event $\texttt{RBRecv}(x, j, -1)$ at node i :**

Line 7 is performed atomically.

7 – $\mathit{Verified}_i[-1] := \mathit{Verified}_i[-1] \cup \{(x, j, -1)\}$

Round $t \geq 0$ at node i:

8 • $\texttt{RBSend}((h_i[t-1], \mathit{Verified}_i^c[t-1]), i, 0)$

9 • **Event handler for event $\texttt{RBRecv}((h, \mathcal{V}), j, t)$ at node i :**

10 – Wait until $\mathcal{V} \subseteq \mathit{Verified}_i[t-1]$

<u>*Lines 11-16 are performed atomically.*</u>

11 – If `Verify` $(h, \mathcal{V}, j, t)$ returns `TRUE` then

12 $\mathit{Verified}_i[t] :=$ `Add` $(\mathit{Verified}_i[t], h, \mathcal{V}, j, t)$

13 – When `Proceed` (t) returns `TRUE` <u>for the first time</u>

14 $\mathit{Verified}_i^c[t] := \mathit{Verified}_i[t]$

15 $h_i[t] := H(\mathit{Verified}_i^c[t], t)$

16 Proceed to Round $t+1$

Procedure `Verify` $(h, \mathcal{V}, j, t)$ at node i:

- Case $t = 0$: If $|\mathcal{V}| \geq n - f$, then return `TRUE`, else return `FALSE`.
- Case $t = 1$: If $|\mathcal{V}| \geq n - f$ and $h = H(\mathcal{V}, 0)$, then return `TRUE`, else return `FALSE`.
- Case $t \geq 2$: If $|\mathcal{V}| \geq n - f$ and $h = H(\mathcal{V}, t-1)$ and $(*, j, t-2) \in \mathcal{V}$,
  then return `TRUE`, else return `FALSE`.

Procedure `Add` $(\mathit{Verified}_i[t], h, \mathcal{V}, j, t)$ at node i:

- Case $t = 0$: return $\mathit{Verified}_i[t] \cup \{(\mathcal{V}, j, -1)\}$.
- Case $t \geq 1$: return $\mathit{Verified}_i[t] \cup \{(h, j, t-1)\}$.

Procedure `Proceed` (t) at node i:

- Case $t = 0$: if $|\mathit{Verified}_i[0]| \geq n - f$,
  then return `TRUE`,
  else return `FALSE`.
- Case $t \geq 1$: if $|\mathit{Verified}_i[t]| \geq n - f$ and $(h_i[t-1], i, t-1) \in \mathit{Verified}_i[t]$,
  then return `TRUE`,
  else return `FALSE`.

The algorithm terminates after t_{end} rounds, where t_{end} is a constant, defined in (19). The state $v_i[t_{end}]$ of each node i is its output when the algorithm terminates after t_{end} iterations.

Definition 5 *A node k's execution of round r, $r \geq 0$, is said to be* <u>**verified by a fault-free node** i</u> *if, eventually node i reliably receives message of the form $((h, \mathcal{V}), k, r+1)$ from node k, and subsequently adds (h, k, r) to $\mathit{Verified}_i[r+1]$ (at line 12). Note that node k may possibly be faulty. Node k's execution of round r is said to be* <u>**verified**</u> *if it is verified by at least one fault-free node.*

We now introduce some more notations (which are also summarized in Appendix A):

- For a *given* execution of the proposed algorithm, let F denote the *actual* set of faulty nodes in the execution. Let $|F| = \phi$. Thus, $0 \leq \phi \leq f$.

- For $r \geq 0$, let $F_v[r]$ denote the set of faulty nodes whose round r execution is verified by at least one fault-free node, as per Definition 5. Note that $F_v[r] \subseteq F$.
- Define $\overline{F_v}[r] = F - F_v[r]$, for $r \geq 0$.

For each faulty node $k \in F_v[r]$, by Definition 5, there must exist a fault-free node i that eventually reliably receives a message of the form $((h, \mathcal{V}), k, r+1)$ from node k, and adds (h, k, r) to $\mathit{Verified}_i[r+1]$. Given these h and $\mathcal{V}$, for future reference, let us define

$$h_k[r] = h \quad (2)$$

$$\mathit{Verified}^c_k[r] = \mathcal{V} \quad (3)$$

Node i verifies node k's round r execution after node i has entered its round $r+1$. Since round r execution of faulty node k above is verified by fault-free node i, due to the checks performed in procedure `Verify`, the equality below holds for $h_k[r]$ and $\mathit{Verified}^c_k[r]$ defined in (2) and (3).

$$h_k[r] = H(\mathit{Verified}^c_k[r], r) \quad \text{for} \quad r \geq 0 \quad (4)$$

(The proof of Claim 5 in Appendix E elaborates on the above equality.) While the algorithm requires each node k to maintain variables $h_k[r]$ and $\mathit{Verified}^c_k[r]$, we cannot assume correct behavior on the part of the faulty nodes. However, from the perspective of each fault-free node that verifies the round r execution of faulty node $k \in F_v[r]$, node k behaves "as if" these local variable take the values specified in (2) and (3) that satisfy (4). Note that if the round r execution (where $r \geq 0$) of a faulty node k is verified by more than one fault-free node, due to the *Global Uniqueness* of reliable broadcast, all these fault-free nodes must have reliably received identical round $r+1$ messages from node k.

Proofs of Lemmas 1, 2 and 3 below are presented in Appendices D, F, and H, respectively. These lemmas are used to prove the correctness of the *Optimal Verified Averaging* algorithm.

Lemma 1 *Optimal Verified Averaging ensures* progress*: (i) all the fault-free nodes will eventually progress to round 0; and, (ii) if all the fault-free nodes progress to the start of round t, $t \geq 0$, then all the fault-free nodes will eventually progress to the start of round $t+1$.*

Lemma 2 *For each node $i \in V - \overline{F_v}[0]$, the polytope $h_i[0]$ is non-empty.*

Lemma 3 *For $r \geq 0$, if $b \in \overline{F_v}[r]$, then for all $\tau \geq r$,*

- *$b \in \overline{F_v}[\tau]$, and*
- *for all $i \in V - \overline{F_v}[\tau+1]$, $(*, b, \tau) \notin \mathit{Verified}^c_i[\tau+1]$.*

4 Correctness

We first introduce some terminology and definitions related to matrices. Then, we develop a *transition matrix* representation of the proposed algorithm, and use that to prove its correctness. Note that the technique is identical to the one present in our prior work [12]. We include the proof here for completeness.

4.1 Matrix Preliminaries

We use boldface upper case letters to denote matrices, rows of matrices, and their elements. For instance, $\mathbf{A}$ denotes a matrix, $\mathbf{A}_i$ denotes the i-th row of matrix $\mathbf{A}$, and $\mathbf{A}_{ij}$ denotes the element at the intersection of the i-th row and the j-th column of matrix $\mathbf{A}$.

Definition 6 *A vector is said to be stochastic if all its elements are non-negative, and the elements add up to 1. A matrix is said to be row stochastic if each row of the matrix is a stochastic vector.*

For matrix products, we adopt the "backward" product convention below, where $a \le b$,

$$\Pi_{\tau=a}^{b}\mathbf{A}[\tau] = \mathbf{A}[b]\mathbf{A}[b-1]\cdots\mathbf{A}[a] \tag{5}$$

For a row stochastic matrix $\mathbf{A}$, coefficients of ergodicity $\delta(\mathbf{A})$ and $\lambda(\mathbf{A})$ are defined as follows [14]:

$$\begin{aligned}\delta(\mathbf{A}) &= \max_j \max_{i_1,i_2} \|\mathbf{A}_{i_1 j} - \mathbf{A}_{i_2 j}\| \\ \lambda(\mathbf{A}) &= 1 - \min_{i_1,i_2}\sum_j \min(\mathbf{A}_{i_1 j}\ , \mathbf{A}_{i_2 j})\end{aligned}$$

Claim 1 *For any p square row stochastic matrices $\mathbf{A}(1), \mathbf{A}(2), \ldots, \mathbf{A}(p)$,*

$$\delta(\Pi_{\tau=1}^{p}\mathbf{A}(\tau)) \ \le\ \Pi_{\tau=1}^{p}\ \lambda(\mathbf{A}(\tau)).$$

Claim 1 is proved in [6]. Claim 2 below follows directly from the definition of $\lambda(\cdot)$.

Claim 2 *If there exists a constant γ, where $0 < \gamma \le 1$, such that, for any pair of rows i, j of matrix $\mathbf{A}$, there exists a column g (that may depend on i, j) such that, $\min(\mathbf{A}_{ig}, \mathbf{A}_{jg}) \ge \gamma$, then $\lambda(\mathbf{A}) \le 1 - \gamma < 1$.*

Let $\mathbf{v}$ be a column vector with n elements, such that the i-th element of vector $\mathbf{v}$, namely $\mathbf{v}_i$, is a convex polytope in the d-dimensional Euclidean space. Let $\mathbf{A}$ be a $n \times n$ row stochastic square matrix. Then multiplication of matrix $\mathbf{A}$ and vector $\mathbf{v}$ is performed by multiplying each row of $\mathbf{A}$ with column vector $\mathbf{v}$ of polytopes. Formally,

$$\mathbf{A}\mathbf{v} = [H_l(\mathbf{v}^T; \mathbf{A}_1) \qquad H_l(\mathbf{v}^T; \mathbf{A}_2) \qquad \ldots \qquad H_l(\mathbf{v}^T; \mathbf{A}_n)]^T \tag{6}$$

where ^{T} denotes the transpose operation (thus, $\mathbf{v}^T$ is the transpose of $\mathbf{v}$). H_l is defined in Definition 4. Thus, the result of the multiplication $\mathbf{A}\mathbf{v}$ is a column vector consisting of n convex polytopes. Similarly, product of row vector $\mathbf{A}_i$ and above vector $\mathbf{v}$ is obtained as follows, and it is a polytope.

$$\mathbf{A}_i\mathbf{v} \ = \ H_l(\mathbf{v}^T\ ;\ \mathbf{A}_i) \tag{7}$$

4.2 Transition Matrix Representation of *Optimal Verified Averaging*

Let $\mathbf{v}[t]$, $t \ge 0$, denote a column vector of length $|V| = n$. In the remaining discussion, we will refer to $\mathbf{v}[t]$ as the state of the system at the end of round t. In particular, $\mathbf{v}_i[t]$ for $i \in V$ is viewed as the state of node i at the end of round t. We define $\mathbf{v}[0]$ as follows:

(I1) For each fault-free node $i \in V - F$, $\mathbf{v}_i[0] := h_i[0]$.

(I2) For each faulty node $k \in F_v[0]$, $\mathbf{v}_k[0] := h_k[0]$, where $h_k[0]$ is defined in (2).

(I3) For each faulty node $k \in \overline{F_v}[0]$, $\mathbf{v}_k[0]$ is *arbitrarily* defined as the origin in the d-dimensional Euclidean space. We will justify this arbitrary choice later.

We will show that the state evolution can be represented in a matrix form as in (8) below, for a suitably chosen $n \times n$ matrix $\mathbf{M}[t]$. $\mathbf{M}[t]$ is said to be the *transition matrix* for round t.

$$\mathbf{v}[t] = \mathbf{M}[t] \ \mathbf{v}[t-1], \qquad t \geq 1 \tag{8}$$

For all $t \geq 0$, Theorem 1 below proves that, for each $i \in V - \overline{F_v}[t]$, $h_i[t] = \mathbf{v}_i[t]$.

Given a particular execution of the algorithm, we construct the transition matrix $\mathbf{M}[t]$ for round $t \geq 1$ using the following procedure.

Construction of the Transition Matrix for Round t $(t \geq 1)$

- For each node $i \in V - \overline{F_v}[t]$, and each $k \in V$:

  If $(*, k, t-1) \in \mathit{Verified}_i^c[t]$, then

$$\mathbf{M}_{ik}[t] := \frac{1}{|\mathit{Verified}_i^c[t]|} \tag{9}$$

  Otherwise,

$$\mathbf{M}_{ik}[t] := 0 \tag{10}$$

  Comment: For a faulty node $i \in F_v[t]$, $h_i[t]$ and $\mathit{Verified}_i^c[t]$ are defined in (2) and (3).

- For each node $j \in \overline{F_v}[t]$, and each $k \in V$,

$$\mathbf{M}_{jk}[t] \quad := \quad \frac{1}{n} \tag{11}$$

Theorem 1 *For $r \geq 0$, with state evolution specified as $\mathbf{v}[r+1] = \mathbf{M}[r+1]\mathbf{v}[r]$ using $\mathbf{M}[r+1]$ constructed above, for all $i \in V - \overline{F_v}[r]$, (i) $h_i[r]$ is non-empty, and (ii) $h_i[r] = \mathbf{v}_i[r]$.*

Proof:

The proof of the theorem is by induction. The theorem holds for $r = 0$ due to Lemma 2, and the choice of the elements of $\mathbf{v}[0]$, as specified in (I1), (I2) and (I3) above.

Now, suppose that the theorem holds for $r = t-1$ where $t-1 \geq 0$, and prove it for $r = t$. Thus, by induction hypothesis, for all $i \in V - \overline{F_v}[t-1]$, $h_i[t-1] = \mathbf{v}_i[t-1] \neq \emptyset$. Now, $\mathbf{v}[t] = \mathbf{M}[t]\mathbf{v}[t-1]$.

- In round $t \geq 1$, each fault-free node $i \in V - F$ computes its new state $h_i[t]$ at line 15 using function $H(\mathit{Verified}_i^c[t], t)$. The function $H(\mathit{Verified}_i^c[t], t)$ for $t \geq 1$ then computes a linear combination of $|\mathit{Verified}_i^c[t]|$ convex hulls, with all the weights being equal to $\frac{1}{|\mathit{Verified}_i^c[t]|}$. Also, by Definition 5 and the definition of $\overline{F_v}[t-1]$, if $(h, j, t-1) \in \mathit{Verified}_i^c[t]$, then $j \notin \overline{F_v}[t-1]$ (i.e., $j \in V - \overline{F_v}[t-1]$). Therefore, if $(h, j, t-1) \in \mathit{Verified}_i^c[t]$, then either j is fault-free, or it is faulty and its round $t-1$ execution is verified: thus, $h = h_j[t-1]$. Also, by induction hypothesis, $h = h_j[t-1] \neq \emptyset$. This implies that $h_i[t] = H(\mathit{Verified}_i^c[t], t)$ is non-empty.

Then observe that, by defining $\mathbf{M}_{ik}[t]$ elements as in (9) and (10), we ensure that $\mathbf{M}_i[t]\mathbf{v}[t-1]$ equals $H(\mathit{Verified}_i^c[t], t)$, and hence equals $h_i[t]$.

- For $i \in F_v[t]$ as well, as shown in (4), $h_i[t] = H(\mathit{Verified}_i^c[t], t)$, where $h_i[t]$ and $\mathit{Verified}_i^c[t]$ are as defined in (2) and (3). The function $H(\mathit{Verified}_i^c[t], t)$ for $t \geq 1$ then computes a linear combination of $|\mathit{Verified}_i^c[t]|$ convex hulls, with all the weights being equal to $\frac{1}{|\mathit{Verified}_i^c[t]|}$. Consider an element $(h, j, t-1)$ in $\mathit{Verified}_i^c[t]$. We argue that $j \in V - \overline{F_v}[t-1]$. Suppose this is not true, i.e., $j \in \overline{F_v}[t-1]$. By Definition 5, node i's round t execution is verified by some fault-free node k, which implies that eventually, $\mathit{Verified}_i^c[t] \subseteq \mathit{Verified}_k[t]$. However, since k is fault-free, and $(h, j, t-1) \notin \mathit{Verified}_k[t]$, a contradiction. Hence, if $(h, j, t-1) \in \mathit{Verified}_i^c[t]$, then $j \in V - \overline{F_v}[t-1]$. That is, if $(h, j, t-1) \in \mathit{Verified}_i^c[t]$, then either j is fault-free, or it is faulty and its round $t-1$ execution is verified: thus, $h = h_j[t-1]$.

  Also, by induction hypothesis, $h = h_j[t-1] \neq \emptyset$. This implies that $h_i[t] = H(\mathit{Verified}_i^c[t], t)$ is non-empty.

  Then observe that, by defining $\mathbf{M}_{ik}[t]$ elements as in (9) and (10), we ensure that $\mathbf{M}_i[t]\mathbf{v}[t-1]$ equals $H(\mathit{Verified}_i^c[t], t)$, and hence equals $h_i[t]$.

□

Now, we argue that for $t \geq 0$, the state $\mathbf{v}_j[t]$ for each node $j \in \overline{F_v}[t]$ does not affect the state of the nodes $V - \overline{F_v}[\tau]$, for $\tau \geq t+1$. From the discussion in the above proof, we see that for $j \in \overline{F_v}[t]$, $(*, j, t) \notin \mathit{Verified}_i^c[t+1]$ for $i \in V - \overline{F_v}[t+1]$. Thus, the sate $\mathbf{v}_j[t]$ does not affect the state $h_i[t+1]$. Then, by Lemma 3, if $j \in \overline{F_v}[t]$, then $j \in \overline{F_v}[\tau]$, for $\tau \geq t+1$. Thus, by the same argument, the sate $\mathbf{v}_j[\tau]$ does not affect the state $h_i[t+1]$. This justifies the somewhat arbitrary choice of $\mathbf{v}_j[0]$ for $j \in \overline{F_v}[0]$, and $\mathbf{M}_{jk}[t]$ in (11) for $j \in \overline{F_v}[t],\ t \geq 1$. This choice does simplify the remaining proof somewhat.

The above discussion shows that, for $t \geq 1$, the evolution of $\mathbf{v}[t]$ can be written as in (8), that is, $\mathbf{v}[t] = \mathbf{M}[t]\mathbf{v}[t-1]$. Given the matrix product definition in (6), it is easy to verify that

$$\mathbf{M}[\tau+1]\ \left(\mathbf{M}[\tau]\mathbf{v}[\tau-1]\right) \ = \ \left(\mathbf{M}[\tau+1]\mathbf{M}[\tau]\right)\ \mathbf{v}[\tau-1] \ \text{ for } \tau \geq 1.$$

Therefore, by repeated application of (8), we obtain:

$$\mathbf{v}[t] \ = \ \left(\Pi_{\tau=1}^{t}\mathbf{M}[\tau]\right)\ \mathbf{v}[0], \qquad t \geq 1 \tag{12}$$

Recall that we adopt the "backward" matrix product convention presented in (5).

Lemma 4 *For $t \geq 1$, transition matrix $\mathbf{M}[t]$ constructed using the above procedure satisfies the following conditions.*

- *For $i, j \in V$, there exists a fault-free node $g(i,j)$ such that $\mathbf{M}_{ig(i,j)}[t] \geq \frac{1}{n}$.*
- *$\mathbf{M}[t]$ is a row stochastic matrix, and $\lambda(\mathbf{M}[t]) \leq 1 - \frac{1}{n}$.*

The proof of Lemma 4 is presented in Appendix J.

4.3 Correctness of ***Optimal Verified Averaging***

Definition 7 *A convex polytope h is said to be* valid *if every point in h is in the convex hull of the inputs at the fault-free nodes.*

Lemmas 5 and 6 below are proved in Appendices K and L, respectively.

Lemma 5 *$h_i[0]$ for each node $i \in V - \overline{F_v}[0]$ is valid.*

Lemma 6 *Suppose non-empty convex polytopes $h_1, h_2, \cdots, h_k$ are all valid. Consider k constants $c_1, c_2, \cdots, c_k$ such that $0 \le c_i \le 1$ and $\sum_{i=1}^{k} c_i = 1$. Then the linear combination of these convex polytopes, $H_l(h_1, h_2, \cdots, h_k; c_1, c_2, \cdots, c_k)$, is valid.*

Theorem 2 *Optimal Verified Averaging satisfies the* validity*, ϵ-*agreement *and* termination *properties after a large enough number of asynchronous rounds.*

Proof: Repeated applications of Lemma 1 ensures that the fault-free nodes will progress from the preliminary round through round r, for any $r \ge 0$, allowing us to use (12). Consider round $t \ge 1$. Let

$$\mathbf{M}^* \;=\; \Pi_{\tau=1}^{t} \mathbf{M}[\tau] \tag{13}$$

(To simplify the presentation, we do not include the round index $[t]$ in the notation $\mathbf{M}^*$ above.) Then $\mathbf{v}[t] = \mathbf{M}^* \mathbf{v}[0]$. By Lemma 4, each $\mathbf{M}[t]$ is a *row stochastic* matrix, therefore, $\mathbf{M}^*$ is also row stochastic. By Lemma 5, $h_i[0] = \mathbf{v}_i[0]$ for each $i \in V - \overline{F_v}[0]$ is valid. Therefore, by Lemma 6, $\mathbf{M}_i^* \mathbf{v}[0]$ for each $i \in V - F$ is valid. Also, by Theorem 1 and (12), $h_i[t] = \mathbf{M}_i^* \mathbf{v}[0]$ for $i \in V - F$. Thus, $h_i[t]$ is valid for $t \ge 1$. This observation together with Lemma 5 implies that *Optimal Verified Averaging* satisfies the validity condition for all round $r \ge 0$.

Let us define $\alpha = 1 - \frac{1}{n}$. By Lemma 4, $\lambda(\mathbf{M}[t]) \le 1 - \frac{1}{n} = \alpha$. Then by Claim 1,

$$\delta(\mathbf{M}^*) = \delta(\Pi_{\tau=1}^{t} \mathbf{M}[\tau]) \;\le\; \lim_{t \to \infty} \Pi_{\tau=1}^{t} \lambda(\mathbf{M}[\tau]) \;\le\; \left(1 - \frac{1}{n}\right)^t \;=\; \alpha^t \tag{14}$$

Consider any two fault-free nodes $i, j \in V - F$. By (14), $\delta(\mathbf{M}^*) \le \alpha^t$. Therefore, by the definition of $\delta(\cdot)$, for $1 \le k \le n$,

$$\|\mathbf{M}_{ik}^* - \mathbf{M}_{jk}^*\| \le \alpha^t \tag{15}$$

By Lemma 3, and construction of the transition matrices, it should be easy to see that $\mathbf{M}_{ib}^* = 0$ for $b \in \overline{F_v}[0]$. Then, for any point p_i^* in $h_i[t] = \mathbf{M}_i^* \mathbf{v}[0]$, there must exist, for all $k \in V - \overline{F_v}[0]$, $p_k \in h_k[0]$, such that

$$p_i^* = \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{ik}^* p_k = \left(\sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{ik}^* p_k(1), \quad \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{ik}^* p_k(2), \cdots, \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{ik}^* p_k(d) \right) \tag{16}$$

where $p_k(l)$ denotes the value of p_k's l-th coordinate. Now choose point p_j^* in $h_j[t]$ defined as follows.

$$p_j^* = \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{jk}^* p_k = \left(\sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{jk}^* p_k(1), \quad \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{jk}^* p_k(2), \cdots, \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{jk}^* p_k(d) \right) \tag{17}$$

Then the Euclidean distance between p_i^* and p_j^* is $d(p_i^*, p_j^*)$. The following derivation is obtained by simple algebraic manipulation, using (15), (16) and (17). The omitted steps in the algebraic manipulation are shown in Appendix M.

$$\begin{aligned} d(p_i^*, p_j^*) &= \sqrt{\sum_{l=1}^d (p_i^*(l) - p_j^*(l))^2} = \sqrt{\sum_{l=1}^d \left(\sum_{k\in V-\overline{F_v}[0]} \mathbf{M}_{ik}^* p_k(l) - \sum_{k\in V-\overline{F_v}[0]} \mathbf{M}_{jk}^* p_k(l)\right)^2} \\ &\leq \alpha^t \sqrt{\sum_{l=1}^d \left(\sum_{k\in V-\overline{F_v}[0]} \|p_k(l)\|\right)^2} \leq \alpha^t \Omega \end{aligned} \tag{18}$$

where $\Omega = \max_{p_k \in h_k[0], k\in V-\overline{F_v}[0]} \sqrt{\sum_{l=1}^d (\sum_{k\in V-\overline{F_v}[0]} \|p_k(l)\|)^2}$. Because the $h_k[0]$'s in the definition of Ω are all valid (by Lemma 5), Ω can itself be upper bounded by a function of the input vectors at the fault-free nodes. In particular, under the assumption that each element of fault-free nodes' input vectors is upper bounded by U and lower bounded by μ, Ω is upper bounded by $\sqrt{dn^2 \max(U^2, \mu^2)}$. Observe that the upper bound on the right side of (18) monotonically decreases with t, because $\alpha < 1$. Define t_{end} as the smallest positive integer t for which

$$\alpha^t \sqrt{dn^2 \max(U^2, \mu^2)} < \epsilon \tag{19}$$

Recall that the algorithm terminates after t_{end} rounds. (18) and (19) together imply that, for fault-free i, j, for each point $p_i^* \in h_i[t_{end}]$ there exists a point $p_j^*[t] \in h_j[t_{end}]$ such that $d(p_i^*, p_j^*) < \epsilon$ (and, similarly, vice-versa). Thus, by Definition 1, Hausdorff distance $\mathbf{d}_H(h_i[t_{end}], h_j[t_{end}]) < \epsilon$. Since this holds true for any pair of fault-free nodes i, j, the ϵ-agreement property is satisfied at termination. □

5 Optimality of *Optimal Verified Averaging*

Due to the *Fault-free Containment* property of *Stable Vector*, all fault-free nodes share at least $(n-f)$ messages in $\mathit{Verified}_i^c[-1]$ (see lines 2-3). Let Z denote the set of these shared messages, that is,

$$Z \quad := \quad \cap_{j\in V-F} \mathit{Verified}_j^c[-1] \tag{20}$$

Define $X_Z =: \{x \mid (x, k, -1) \in Z\}$. Then, define a convex polytope I_Z as follows.

$$I_Z \quad := \quad \cap_{D\subset X_Z, |D|=|X_Z|-f} \mathcal{H}(D) \tag{21}$$

The following lemma establishes a "lower bound" on the convex polytope that the fault-free nodes decide on. Recall that $\overline{F_v}[0]$ is defined as all the faulty nodes that are not verified by any fault-free nodes in Round 1. The proof is presented in Appendix N.

Lemma 7 *For all $i \in V - \overline{F_v}[t]$ and $t \geq 0$, $I_Z \subseteq h_i[t]$.*

Then, the following key theorem shows that the presented algorithm is optimal. The proof is presented in Appendix O. This theorem closes an open question raised in our prior work [12].

Theorem 3 *The output convex polytope at fault-free node i using* Optimal Verified Averaging *is optimal as per Definition 2.*

6 Summary

This paper considers Byzantine Convex Consensus (BCC), wherein each node has a d-dimensional vector as its input, and each fault-free node should agree on an output polytope that is in the convex hull of the input vectors at the fault-free nodes. We present an asynchronous approximate BCC algorithm with optimal fault tolerance that reaches consensus on an *optimal* output polytope.

References

[1] I. Abraham, Y. Amit, and D. Dolev. Optimal resilience asynchronous approximate agreement. In *OPODIS*, pages 229–239, 2004.

[2] H. Attiya, A. Bar-Noy, D. Dolev, D. Peleg, and R. Reischuk. Renaming in an Asynchronous Environment. Journal of the ACM, July 1990.

[3] H. Attiya and J. Welch. *Distributed Computing: Fundamentals, Simulations, and Advanced Topics*. Wiley Series on Parallel and Distributed Computing, 2004.

[4] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl. Reaching approximate agreement in the presence of faults. *J. ACM*, 33:499–516, May 1986.

[5] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32:374–382, April 1985.

[6] J. Hajnal. Weak ergodicity in non-homogeneous markov chains. In *Proceedings of the Cambridge Philosophical Society*, volume 54, pages 233–246, 1958.

[7] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[8] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.

[9] H. Mendes and M. Herlihy. Multidimensional approximate agreement in byzantine asynchronous systems. In *ACM Symposium on Theory of Computing (STOC)*, 2013.

[10] H. Mendes, C. Tasson, and M. Herlihy. The topology of asynchronous byzantine colorless tasks. *CoRR*, abs/1302.6224, 2013.

[11] M. A. Perles and M. Sigorn. A generalization of Tverberg's theorem. *CoRR*, abs/0710.4668, 2007.

[12] L. Tseng and N. H. Vaidya. Byzantine Convex Consensus: Preliminary Version. *CoRR*, abs/1307.1051, 2013.

[13] N. H. Vaidya and V. K. Garg. Byzantine vector consensus in complete graphs. *CoRR*, abs/1302.2543, 2013. To appear at ACM PODC 2013.

[14] J. Wolfowitz. Products of indecomposable, aperiodic, stochastic matrices. In *Proceedings of the American Mathematical Society*, volume 14, pages 733–737, 1963.

A Notations

This appendix summarizes some of the notations and terminology introduced throughout the paper.

- n = number of nodes. We assume that $n \geq 2$.
- f = maximum number of Byzantine nodes.
- $V = \{1, 2, \cdots, n\}$ is the set of all nodes.
- d = dimension of the input vector at each node.
- $d(p, q)$ = the function returns the Euclidean distance between points p and q.
- $d_H(h_1, h_2)$ = the Hausdorff distance between convex polytopes h_1, h_2.
- $\mathcal{H}(C)$ = the convex hull of a multiset C.
- $H_l(h_1, h_2, \cdots, h_k;\ c_1, c_2, \cdots, c_k)$, defined in Section 2, is a linear combination of convex polytopes $h_1, h_2, ..., h_k$ with weights $c_1, c_2, \cdots, c_k$.
- $H(\mathcal{V}, t)$ is a function defined in Section 2.
- $|X|$ = the size of a *multiset* or *set* X.
- $\|a\|$ = the absolute value of a real number a.
- F denotes the *actual* set of faulty nodes in an execution of the algorithm.
- $\phi = |F|$. Thus, $0 \leq \phi \leq f$.
- $F_v[t]$, $t \geq 0$, denotes the set of faulty nodes whose round t execution is verified by at least one fault-free node, as per Definition 5.
- $\overline{F_v}[t] = F - F_v[t]$, $t \geq 0$.
- $\alpha = 1 - \frac{1}{n}$.
- We use boldface upper case letters to denote matrices, rows of matrices, and their elements. For instance, $\mathbf{A}$ denotes a matrix, $\mathbf{A}_i$ denotes the i-th row of matrix $\mathbf{A}$, and $\mathbf{A}_{ij}$ denotes the element at the intersection of the i-th row and the j-th column of matrix $\mathbf{A}$.

B $H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$ is Convex

Claim 3 *$H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$ defined in Definition 4 is convex.*

Proof:

The proof is straightforward.

Let

$$h_L := H_l(h_1, h_2, \cdots, h_\nu;\ c_1, c_2, \cdots, c_\nu)$$

and

$$Q := \{i \mid c_i \neq 0,\ 1 \leq i \leq \nu\}.$$

Given any two points x, y in h_L, by Definition 4, we have

$$x = \sum_{i \in Q} c_i p_{(i,x)} \quad \text{for some } p_{(i,x)} \in h_i, \quad i \in Q \tag{22}$$

and

$$y = \sum_{i \in Q} c_i p_{(i,y)} \quad \text{for some } p_{(i,y)} \in h_i, \quad i \in Q \tag{23}$$

Now, we show that any convex combination of x and y is also in h_L. Consider a point z such that

$$z = \theta x + (1-\theta) y \quad \text{where } 0 \le \theta \le 1 \tag{24}$$

Substituting (22) and (23) into (24), we have

$$\begin{aligned} z &= \theta \sum_{i \in Q} c_i p_{(i,x)} + (1-\theta) \sum_{i \in Q} c_i p_{(i,y)} \\ &= \sum_{i \in Q} c_i \left(\theta p_{(i,x)} + (1-\theta) p_{(i,y)} \right) \end{aligned} \tag{25}$$

Define $p_{(i,z)} = \theta p_{(i,x)} + (1-\theta) p_{(i,y)}$ for all $i \in Q$. Since h_i is convex, and $p_{(i,z)}$ is a convex combination of $p_{(i,x)}$ and $p_{(i,y)}$, $p_{(i,z)}$ is also in h_i. Substituting the definition of $p_{(i,z)}$ in (25), we have

$$z = \sum_{i \in Q} c_i \; p_{(i,z)} \quad \text{where } p_{(i,z)} \in h_i, \quad i \in Q$$

Hence, by Definition 4, z is also in h_L. Therefore, h_L is convex.

□

C Claim 4

Claim 4 *Consider fault-free nodes $i, j \in V - F$. Then*

- *If $(h, k, -1) \in \mathit{Verified}_i[-1]$ at some point of time, then eventually, $(h, k, -1) \in \mathit{Verified}_j[-1]$.*
- *For $t \ge 0$, if $(h, k, t-1) \in \mathit{Verified}_i[t]$ at some point of time, then eventually $(h, k, t-1) \in \mathit{Verified}_j[t]$.*

Proof:

First Part:

In the preliminary round ($t = -1$), node i adds (h, k) to *Verified*$_i[-1]$ whenever it reliably receives message $(h, k, -1)$, i.e., $(h, k, -1)$ is either received by using stable vector or `RBRecv`$(h, k, -1)$

occurred. (For messages in the preliminary round, h is just a single point.) Then by *Global Liveness* property, node j will eventually reliably receive the same message, and add $(h, k, -1)$ to $\mathit{Verified}_j[-1]$.

Second Part:

The proof is by induction.

Induction basis: Suppose that in round $t = 0$, at some real time μ, $(h, k, -1) \in \mathit{Verified}_i[0]$. Thus, node i must have reliably received (at line 9 of round 0) a message of the form $((h, \mathcal{V}), k, -1)$ such that the following conditions are true at time μ:

- Condition 1: $\mathcal{V} \subseteq \mathit{Verified}_i[-1]$ (due to line 10, and the fact that $\mathit{Verified}_i[-1]$ can only grow with time)
- Condition 2: $|\mathcal{V}| \geq n - f$ (due to Case $t = 0$ in Procedure `Verify`)

The *Global Liveness* property implies that eventually node j will also reliably receive the message $((h, \mathcal{V}), k, 0)$ that was reliably received by node i. Also, the correctness of the first part implies that eventually each element of $\mathit{Verified}_i[-1]$ will be included in $\mathit{Verified}_j[-1]$. Thus, because $\mathcal{V} \subseteq \mathit{Verified}_i[-1]$ at time μ, eventually $\mathcal{V} \subseteq \mathit{Verified}_j[-1]$. As in Condition 2 above, node j will also find that $|\mathcal{V}| \geq n - f$. Therefore, by lines 10-12, it follows that eventually $(h, k, -1) \in \mathit{Verified}_j[0]$.

Induction: Consider round $t \geq 1$. Assume that the second part of the lemma holds true through rounds $t - 1$. Therefore, if $(h, k, t-2) \in \mathit{Verified}_i[t-1]$ at some point of time, then eventually $(h, k, t-2) \in \mathit{Verified}_j[t-1]$.

Now we will prove that the second part of the lemma holds for round t. Suppose that at some time μ, $(h, k, t-1) \in \mathit{Verified}_i[t]$. Thus, node i must have reliably received (at line 9 of round t) a message of the form $((h, \mathcal{V}), k, t)$ such that the following conditions are true at time μ:

- Condition 1: $\mathcal{V} \subseteq \mathit{Verified}_i[t-1]$ (due to 10, and the fact that $\mathit{Verified}_i[t-1]$ can only grow with time)
- Condition 2.1: when $t = 1$, $|\mathcal{V}| \geq n - f$ and $h = H(\mathcal{V}, 0)$ (due to Case $t = 1$ in Procedure `Verify`)
- Condition 2.2: when $t \geq 2$, $|\mathcal{V}| \geq n - f$, $h = H(\mathcal{V}, t-1)$, and $(*, k, t-2) \in \mathcal{V}$ (due to Case $t \geq 2$ in Procedure `Verify`)

The correctness of the second part of the lemma through round $t - 1$ implies that eventually each element of $\mathit{Verified}_i[t-1]$ will be included in $\mathit{Verified}_j[t-1]$. Thus, because $\mathcal{V} \subseteq \mathit{Verified}_i[t-1]$ at time μ, eventually $\mathcal{V} \subseteq \mathit{Verified}_j[t-1]$. Also, the *Global Liveness* property implies that eventually node j will reliably receive the message $((h, \mathcal{V}), k, t)$ that was reliably received by node i; then, consider two cases:

- When $t = 1$, as in Condition 2.1 above, node j will also find that $|\mathcal{V}| \geq n - f$, and $h = H(\mathcal{V}, 0)$. Therefore, by lines 10-12, it follows that eventually $(h, k, t-1) \in \mathit{Verified}_j[t]$.
- When $r \geq 2$, as in Condition 2.2 above, node j will also find that $|\mathcal{V}| \geq n - f$, $(*, k, t-2) \in \mathcal{V}$ and $h = H(\mathcal{V}, t-1)$. Therefore, by lines 10-12, it follows that eventually $(h, k, t-1) \in \mathit{Verified}_j[t]$.

Therefore, the proof for the second part is complete.

□

D Proof of Lemma 1

Lemma 1: *Optimal Verified Averaging ensures* progress*: (i) all the fault-free nodes will eventually progress to round 0; and, (ii) if all the fault-free nodes progress to the start of round t, $t \geq 0$, then all the fault-free nodes will eventually progress to the start of round $t+1$.*

Proof: First Part:

By assumption, all fault-free nodes begin the preliminary round eventually, and perform reliable broadcast of their input (line 1). Since the $(n-f)$ fault-free nodes follow the algorithm correctly, `SVRecv`(-1) will eventually return (line 2). Therefore, node i will eventually proceed to round 0 (line 5).

Second Part:

The proof is by induction. By the first part, each fault-free node i begins round 0 eventually, and performs reliable broadcast of $((h_i[-1], \mathit{Verified}_i^c[-1]), i, 0)$ on line 8. Consider fault-free nodes i, j. By *Fault-Free Liveness* property of the primitives, node i will eventually reliable receive message $((h_j[-1], \mathit{Verified}_j^c[-1]), j, 0)$ from fault-free node j. By Claim 4, eventually, $\mathit{Verified}_j^c[-1] \subseteq \mathit{Verified}_i[-1]$; therefore, node i will progress past line 10. Moreover, since node j is fault-free, it follows the algorithm specification correctly. Therefore, `Verify` will return `TRUE`, and node i will eventually include $(\mathit{Verified}_j^c[-1], j, -1)$. Since the above argument holds for all fault-free nodes i, j, it implies that each fault-free node i eventually adds $(\mathit{Verified}_j^c[-1], j, -1)$ to $\mathit{Verified}_i[0]$, for each fault-free node j (including $j = i$). Therefore, at each fault-free node i, eventually, $|\mathit{Verified}_i[0]| \geq n - f$, thus satisfying the condition in Case $t = 0$ of Procedure `Proceed`. Thus, Procedure `Proceed` will return `TRUE`, and each fault-free node i will eventually proceed to round 1 (lines 13-16).

Now we assume that all the fault-free nodes have progressed to the start of round t, where $t \geq 1$, and prove that all the fault-free nodes will eventually progress to the start of round $t+1$.

Consider fault-free nodes $i, j \in V - F$. At line 8 of round t, fault-free node j performs reliable broadcast of $((h_j[t-1], \mathit{Verified}_j^c[t-1]), j, t)$. By *Fault-free Liveness* of reliable broadcast, fault-free node i will eventually reliably receive message $((h_j[t-1], \mathit{Verified}_j^c[t-1]), j, t)$ from fault-free node j. By Claim 4, eventually $\mathit{Verified}_j^c[t-1] \subseteq \mathit{Verified}_i[t-1]$; therefore, node i will progress past line 10 in the handler for message $((h_j[t-1], \mathit{Verified}_j^c[t-1]), j, t)$. Moreover, since node j is fault-free, it follows the algorithm specification correctly. Therefore, Procedure `Verify` will return `TRUE` in the handler at node i for message $((h_j[t-1], \mathit{Verified}_j^c[t-1]), j, t)$ will all be correct. Therefore, by lines 11-12, node i will eventually include $(h_j[t-1], j, t-1)$ in $\mathit{Verified}_i[t]$. Since the above argument holds for all fault-free nodes i, j, it implies that each fault-free node i eventually adds $(h_j[t-1], j, t-1)$ to $\mathit{Verified}_i[t]$, for each fault-free node j (including $j = i$). Therefore, at each fault-free node i, eventually, $|\mathit{Verified}_i[t]| \geq n - f$, and $(h_i[t-1], i, t-1) \in \mathit{Verified}_i[t]$ (because the previous statement holds for $j = i$ too), thus satisfying both the conditions in Case $t \geq 1$ of Procedure `Proceed`. Thus, Procedure `Proceed` will return `TRUE`, and each fault-free node i will eventually proceed to round $t+1$ (lines 13-16).

□

E Claims 5 and 6

Claim 5 *If faulty node i's round t execution is verified by a fault-free node j, then the following statements hold:*

(i) For $t \geq 0$, $\mathit{Verified}_i^c[t] \geq n - f$ and $h_i[t] = H(\mathit{Verified}_i^c[t], t)$,

(ii) For $t \geq 0$, eventually $\mathit{Verified}_i^c[t] \subseteq \mathit{Verified}_j[t]$, and

(iii) For $t \geq 1$, node i's round $t-1$ execution is also verified by node j.

Proof: Let $t \geq 0$. Suppose that node i's round t execution is verified by a fault-free node j. In this case, we can use definitions (2) and (3) of $h_i[t]$ and $\mathit{Verified}_i^c[t]$. Definition 5 implies that node j eventually reliably receives message $((h_i[t], \mathit{Verified}_i^c[t]), i, t+1)$ from node i, and subsequently adds (at line 12 in its round $t+1$) $(h_i[t], i, t)$ to $\mathit{Verified}_j[t+1]$. This implies that this message satisfies the checks done by node j at lines 10 and 11: Specifically, (a) $|\mathit{Verified}_i^c[t]| \geq n - f$, (b) $h_i[t] = H(\mathit{Verified}_i^c[t], t)$, and (c) for $t \geq 1$, $(*, i, t-1) \in \mathit{Verified}_i^c[t]$. Also, by the time node j adds $(h_i[t], i, t)$ to $\mathit{Verified}_j[t+1]$, the condition checked at line 10 also hold: specifically, $\mathit{Verified}_i^c[t] \subseteq \mathit{Verified}_j[t]$, proving claim *(ii)* stated above. Also, (a) and (b) above prove claim *(i)*.

For $t \geq 1$, $(*, i, t-1) \in \mathit{Verified}_i^c[t]$ and eventually $\mathit{Verified}_i^c[t] \subseteq \mathit{Verified}_j[t]$ together imply that eventually $(*, i, t-1) \in \mathit{Verified}_j[t]$. Then this observation together with Definition 5 imply that round $t-1$ execution of node i is verified by node j. This proves claim *(iii)*. □

Claim 6 *If faulty node i's round t execution is verified by a fault-free node j, $t \geq 0$, then for all r such that $0 \leq r \leq t$, node i's round r execution is verified by node j.*

Proof: The claim is trivially true for $t = 0$. The proof of the claim for $t > 0$ follows by repeated application of Claim 5(iii) above.

□

F Proof of Lemma 2

The proof of Lemma 2 uses the following theorem by Tverberg [11]:

Theorem 4 *(Tverberg's Theorem [11]) For any integer $f \geq 0$, for every multiset Y containing at least $(d+1)f+1$ points in a d-dimensional space, there exists a partition $Y_1, .., Y_{f+1}$ of Y into $f+1$ non-empty multisets such that $\cap_{l=1}^{f+1} \mathcal{H}(Y_l) \neq \emptyset$.*

Now we prove Lemma 2.

Lemma 2: *For each node $i \in V - \overline{F_v}[0]$, the polytope $h_i[0]$ is non-empty.*

Proof: Note that $V - \overline{F_v}[0] = (V - F) \cup F_v[0]$.

- For a fault-free node $i \in V - F$, since it behaves correctly, $|\mathit{Verified}_i^c[0]| \geq n - f$ (due to the checks performed in `Verify`), and $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$ (due to line 15).
- For faulty node $i \in F_v[0]$ as well, by Claim 5(i) in Appendix E, $|\mathit{Verified}_i^c[0]| \geq n - f$ and $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$.

Thus, for each $i \in V - \overline{F_v}[0]$, $|\mathit{Verified}_i^c[0]| \geq n - f$ and $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$.

Consider any $i \in V - \overline{F_v}[0]$. Consider the computation of polytope $h_i[0]$ as $H(\mathit{Verified}_i^c[0], 0)$. By step 3 of Case $t = 0$ in function H in Section 2, $|X| = |\mathit{Verified}_i^c[0]| \geq n - f$. Recall that, due to the lower bound on n discussed in Section 1, we assume $n \geq (d+2)f + 1$. Thus, in function H, $|X| \geq n - f \geq (d+1)f + 1$. By Theorem 4 above, there exists a partition $X_1, X_2, \cdots, X_{f+1}$ of X into multisets X_j such that $\cap_{j=1}^{f+1} \mathcal{H}(X_j) \neq \emptyset$. Let us define

$$J = \cap_{i=1}^{f+1} \mathcal{H}(X_j) \tag{26}$$

Thus, J is non-empty. In item (i.e., step) 4 of Case $t = 0$ in function H, because $|X| \geq n - f$, each multiset C used in the computation of function H is of size at least $n - 2f$. Thus, each C excludes only f elements of X, whereas there are $f + 1$ multisets in the above partition of X. Therefore, each set C in step 4 of item 1 of function H will fully contain at least one multiset X_j from the partition. Therefore, $\mathcal{H}(C)$ will contain J. Since this holds true for all C's, J is contained in the convex polytope computed by $H(\mathit{Verified}_i^c[0], 0)$. Since J is non-empty, $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$ is non-empty.

□

G Claim 7

Claim 7 *For $t \geq 0$, if $b \in \overline{F_v}[t]$, then for all $i \in V - \overline{F_v}[t+1]$, $(*, b, t) \notin \mathit{Verified}_i^c[t+1]$.*

Proof: Consider faulty node $b \in \overline{F_v}[t]$. Note that $V - \overline{F_v}[t+1] = (V - F) \cup F_v[t+1]$.

- Consider a fault-free node $i \in V - F$. Since $b \in \overline{F_v}[t]$, node b's round t execution is *not* verified by *any* fault-free node. Therefore, by Definition 5, for fault-free node $i \in V - F$, **at all times**, $(*, b, t) \notin \mathit{Verified}_i[t+1]$. Therefore, by line 14, $(*, b, t) \notin \mathit{Verified}_i^c[t+1]$.
- Consider a faulty node $i \in F_v[t+1]$. In this case, the proof is by contradiction. In particular, for some h, assume that $(h, b, t) \in \mathit{Verified}_i^c[t+1]$. Since $i \in F_v[t+1]$, there exists a fault-free node j that verifies the round $t+1$ execution of node i. Therefore, by Claim 5(ii) in Appendix E, eventually $\mathit{Verified}_i^c[t+1] \subseteq \mathit{Verified}_j[t+1]$. This observation, along with the above assumption that $(h, b, t) \in \mathit{Verified}_i^c[t+1]$, implies that eventually $(h, b, t) \in \mathit{Verified}_j[t+1]$. Since node j is fault-free, Definition 5 implies that execution of node b in round t is verified, and hence $b \in F_v[t]$. This is a contradiction. Therefore, $(*, b, t) \notin \mathit{Verified}_i^c[t+1]$.

□

H Proof of Lemma 3

Lemma 3: *For $r \geq 0$, if $b \in \overline{F_v}[r]$, then for all $\tau \geq r$,*

- *$b \in \overline{F_v}[\tau]$, and*
- *for all $i \in V - \overline{F_v}[\tau+1]$, $(*, b, \tau) \notin \mathit{Verified}_i^c[\tau+1]$.*

Proof: Recall that $F_v[r] \subseteq F$, and $\overline{F_v}[r] = F - F_v[r]$.

For $r \geq 0$, consider a faulty node $b \in \overline{F_v}[r]$. Thus, $b \in F$.

We first prove that $b \in \overline{F_v}[\tau]$, for $\tau \geq r$. This is trivially true for $\tau = r$. So we only need to prove this for $\tau > r$. The proof is by contradiction.

Suppose that there exists $\tau > r$ such that $b \notin \overline{F_v}[\tau]$. Thus, $b \in F_v[\tau]$. The definition of $F_v[\tau]$ implies that node b's round τ execution is verified by some fault-free node j. Then Claim 6 implies that node b's round r execution is verified by node j. Hence by the definition of $F_v[r]$, $b \in F_v[r]$. This is a contradiction. This proves that $b \in \overline{F_v}[\tau]$.

Now, since $b \in \overline{F_v}[\tau]$, by Claim 7, for all $i \in V - \overline{F_v}[\tau+1]$, $(*, b, \tau) \notin \mathit{Verified}_i^c[\tau+1]$.

□

I Claims 8, 9 and 10

Claim 8 *For $t \geq -1$, a fault-free node i adds at most one message from node j to $\mathit{Verified}_i[t]$, even if j is faulty.*

Proof: As stated in the properties of the communication primitives in Section 2, each fault-free node i will reliably receive at most one message of the form $(*, j, t)$ from node j (either via `SVRecv` or via `RBRecv`). Since $\mathit{Verified}_i[t]$ only contains tuples corresponding to reliably received messages, the claim follows. □

Claim 9 *For $t \geq 1$, consider nodes $i, j \in V - \overline{F_v}[t]$. If $(h, k, t) \in \mathit{Verified}_i^c[t]$ and $(h', k, t) \in \mathit{Verified}_j^c[t]$, then $h = h'$.*

Proof: We consider four cases:

- $i, j \in V - F$: In this case, due to *Global Uniqueness* property of the primitive, nodes i and j cannot reliably receive different round t messages from the same node. Hence the claim follows.

- $i \in V - F$ and $j \in F_v[t]$: Suppose that fault-free node p verifies round t execution of node j. Then by Claim 5(ii), eventually $\mathit{Verified}_j^c[t] \subseteq \mathit{Verified}_p[t]$. Since nodes i and p are both fault-free, similar to the previous case, due to the *Global Uniqueness* property, nodes i and p cannot reliably receive distinct round t messages. Thus, if $(h, k, t) \in \mathit{Verified}_i^c[t]$ and $(h', k, t) \in \mathit{Verified}_j^c[t] \subseteq \mathit{Verified}_p[t]$, then $h = h'$.

- $j \in V - F$ and $i \in F_v[t]$: This case is similar to the previous case.

- $i, j \in F_v[t]$: In this case, there exist fault-free nodes k_i and k_j that verify round t execution of nodes i and j, respectively. Thus, by Claim 5(ii), eventually $(h, i, t) \in \mathit{Verified}_i^c[t] \subseteq \mathit{Verified}_{k_i}[t]$ and $(h', i, t) \in \mathit{Verified}_j^c[t] \subseteq \mathit{Verified}_{k_j}[t]$. Since k_i, k_j are fault-free, *Global Uniqueness* implies that $h = h'$.

□

Claim 10 *For $t \geq 1$, consider nodes $i, j \in V - \overline{F_v}[t]$. There exists a fault-free node $g \in V - F$ such that $(h_g[t-1], g, t-1) \in \mathit{Verified}_i^c[t] \cap \mathit{Verified}_j^c[t]$.*

Proof: For any fault-free node, say p, due to the conditions checked in Procedure `Proceed`, $|\mathit{Verified}_p^c[t]| \geq n - f$. For a node $k \in F_v[t]$, recall that $h_k[t]$ and $\mathit{Verified}_k^c[t]$ are defined in (2) and (3). Thus, by Definition 5, there exists some fault-free node, say q, that reliably receives message $((h_k[t], \mathit{Verified}_k^c[t]), k, t+1)$ from node k in round $t+1$, and after performing checks in Procedure `Verify`, adds $(h_k[t], k, t)$ to $\mathit{Verified}_q^c[t+1]$. The checks in Procedure `Verify`, performed by fault-free node q, ensure that $|\mathit{Verified}_k^c[t]| \geq n - f$.

Above argument implies that for the nodes $i, j \in V - \overline{F_v}[t]$, $\mathit{Verified}_i^c[t]$ and $\mathit{Verified}_j^c[t]$ both contain at least $n-f$ messages. Therefore, by Claims 8 and 9, there will be at least $n-2f \geq df+1 \geq f+1$ elements in $\mathit{Verified}_i^c[t] \cap \mathit{Verified}_j^c[t]$. Since f is the upper bound on the number of faulty nodes, at least one element in $\mathit{Verified}_i^c[t] \cap \mathit{Verified}_j^c[t]$ corresponds to a fault-free node, say node $g \in V - F$. That is, there exists $g \in V - F$ such that $(h_g[t-1], g, t-1) \in \mathit{Verified}_i^c[t] \cap \mathit{Verified}_j^c[t]$.

□

J Proof of Lemma 4

Lemma 4: *For $t \geq 1$, transition matrix $\mathbf{M}[t]$ constructed using the above procedure satisfies the following conditions.*

- *For $i, j \in V$, there exists a fault-free node $g(i,j)$ such that $\mathbf{M}_{ig(i,j)}[t] \geq \frac{1}{n}$.*
- *$\mathbf{M}[t]$ is a row stochastic matrix, and $\lambda(\mathbf{M}[t]) \leq 1 - \frac{1}{n}$.*

Proof:

- To prove the first claim in the lemma, we consider four cases for node pairs i, j.
  - (i) $i, j \in V - \overline{F_v}[t]$: By Claim 10, there exists a node $g(i,j)$ such that $(h_{g(i,j)}[t-1], g(i,j), t-1) \in \mathit{Verified}_i^c[t] \cap \mathit{Verified}_j^c[t]$. By (9) in the procedure to construct $\mathbf{M}[t]$, $\mathbf{M}_{ig(i,j)}[t] = \frac{1}{|\mathit{Verified}_i^c[t]|} \geq \frac{1}{n}$ and $\mathbf{M}_{jg(i,j)}[t] = \frac{1}{|\mathit{Verified}_j^c[t]|} \geq \frac{1}{n}$.
  - (ii) $i \in \overline{F_v}[t]$ and $j \in V - \overline{F_v}[t]$: $|\mathit{Verified}_j^c[t]| \geq n - f$ elements of $\mathbf{M}_j[t]$ are equal to $\frac{1}{|\mathit{Verified}_j^c[t]|} \geq \frac{1}{n}$. Since $n - f \geq (d+1)f + 1 \geq 2f + 1$, there exists a fault-free node $g(i,j)$ such that $\mathbf{M}_{jg(i,j)} \geq \frac{1}{n}$. By (11), all elements of $\mathbf{M}_i[t]$, including $\mathbf{M}_{ig(i,j)}[t] = \frac{1}{n}$.
  - (iii) $j \in \overline{F_v}[t]$ and $i \in V - \overline{F_v}[t]$: Similar to case (ii).
  - (iv) $i, j \in \overline{F_v}[t]$: By (11) in the procedure to construct $\mathbf{M}[t]$, all n elements in $\mathbf{M}_i[t]$ and $\mathbf{M}_j[t]$ both equal $\frac{1}{n}$. Choose a fault-free node as node $g(i,j)$. Then $\mathbf{M}_{ig(i,j)}[t] = \mathbf{M}_{ig(i,j)}[t] = \frac{1}{n}$.
- Observe that, by construction, for each $i \in V$, the row vector $\mathbf{M}_i[t]$ is stochastic. Thus, $\mathbf{M}[t]$ is row stochastic. Also, due to the claim proved in the previous item, and Claim 2, $\lambda(\mathbf{M}[t]) \leq 1 - \frac{1}{n} < 1$.

□

K Proof of Lemma 5

Lemma 5: *$h_i[0]$ for each node $i \in V - \overline{F_v}[0]$ is valid.*

Proof: Recall that $V - \overline{F_v}[0] = (V - F) \cup F_v[0]$. Now, consider two cases:

- $i \in V - F$: Recall that $h_i[0]$ is obtained using function $H(\mathit{Verified}_i^c[0], 0)$. Note that the function $H(\mathit{Verified}_i^c[0], 0)$ first computes frequency counts $N(x, k)$ for each (x, k), and then computes sets Y and X using $N(x, k)$ values.

  For X and Y computed in $H(\mathit{Verified}_i^c[0], 0)$, consider a value $x^* \in X$. Since $x^* \in X$, there must exist k^* such that $(x^*, k^*) \in Y$. This, in turn, implies that there must exist at least $f + 1$ tuples of the form $(\mathcal{I}, j, -1) \in \mathit{Verified}_i^c[0]$ such that $(x^*, k^*, -1) \in \mathcal{I}$. When j is a fault-free node, $\mathcal{I}$ above must be equal to $\mathit{Verified}_j^c[-1]$ due to the algorithm specification. Since $(x^*, k^*, -1)$ appears in at least $f + 1$ tuples as observed above, there exists at least one fault-free node j such that $(x^*, k^*, -1) \in \mathit{Verified}_j^c[-1]$. Therefore, if k^* is fault-free, then x^* must be the input vector at node k^*.

  Also, by Claim 8, for any faulty node b, at most one tuple of the form (b, v) may appear in set Y above. Therefore, except for at most f values in X (which may correspond to faulty nodes), all the other values in X must be equal to inputs at fault-free nodes. Therefore, at least one set C used to compute `temp` in step 4 in Case $t = 0$ of function $H(\mathit{Verified}_i^c[0], 0)$ must contain only the inputs at fault-free nodes. Therefore, $h_i[0] = \texttt{temp}$ is in the convex hull of the inputs at fault-free nodes. That is, $h_i[0]$ is valid.

- $i \in F_v[0]$: Suppose that round 0 execution of node i is verified by a fault-free node j. By Claim 5 in Appendix E, $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$, $|\mathit{Verified}_i^c[0]| \geq n - f$, and eventually $\mathit{Verified}_i^c[0] \subseteq \mathit{Verified}_j[0]$. Suppose that at some time τ, $\mathit{Verified}_i^c[0] \subseteq \mathit{Verified}_j[0]$. Let $\mathit{Verified}_j[0]$ at real time τ be denoted as $\mathit{Verified}_j^\tau[0]$. Then, $\mathit{Verified}_i^c[0] \subseteq \mathit{Verified}_j^\tau[0]$. By an argument similar to the previous item, it should be easy to see that $H(\mathit{Verified}_j^\tau[0], 0)$ is *valid*. Also, observe that if $\mathcal{V}_1 \subseteq \mathcal{V}_2$, then $H(\mathcal{V}_1, 0) \subseteq H(\mathcal{V}_2, 0)$. Thus, $H(\mathit{Verified}_i^c[0], 0) \subseteq H(\mathit{Verified}_j^\tau[0], 0)$, and since $H(\mathit{Verified}_j^\tau[0], 0)$ is valid, $H(\mathit{Verified}_i^c[0], 0)$ is also valid. Thus, $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$ is valid.

□

L Proof of Lemma 6

The proof is straightforward, but included here for completeness.

Lemma 6: *Suppose non-empty convex polytopes $h_1, h_2, \cdots, h_k$ are all valid. Consider k constants $c_1, c_2, \cdots, c_k$ such that $0 \leq c_i \leq 1$ and $\sum_{i=1}^k c_i = 1$. Then the linear combination of these convex polytopes, $H_l(h_1, h_2, \cdots, h_k; c_1, c_2, \cdots, c_k)$, is valid.*

Proof:

Observe that the points in $H_l(h_1, \cdots, h_k; c_1, \cdots, c_k)$ are convex combinations of the points in $h_1, \cdots, h_k$, because $\sum_{i=1}^k c_i = 1$ and $0 \leq c_i \leq 1$, for $1 \leq i \leq k$. Let G be the set of input vectors at the fault-free nodes in $V - F$. Then, $\mathcal{H}(G)$ is the convex hull of the inputs at the fault-free nodes. Since h_i, $1 \leq i \leq k$, is valid, each point $p \in h_i$ is in $\mathcal{H}(G)$. Since $\mathcal{H}(G)$ is a convex polytope, it follows that any convex combination of the points in $h_1, \cdots, h_k$ is also in $\mathcal{H}(G)$.

□

M Algebraic Manipulation in the Proof of Theorem 2

$$
\begin{aligned}
d(p_i^*, p_j^*) &= \sqrt{\sum_{l=1}^{d} (p_i^*(l) - p_j^*(l))^2} \\
&= \sqrt{\sum_{l=1}^{d} \left(\sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{ik}^* p_k(l) - \sum_{k \in V - \overline{F_v}[0]} \mathbf{M}_{jk}^* p_k(l) \right)^2} \qquad \text{by (16) and (17)} \\
&= \sqrt{\sum_{l=1}^{d} \left(\sum_{k \in V - \overline{F_v}[0]} (\mathbf{M}_{ik}^* - \mathbf{M}_{jk}^*) p_k(l) \right)^2} \\
&\leq \sqrt{\sum_{l=1}^{d} \left[\alpha^{2t} \left(\sum_{k \in V - \overline{F_v}[0]} \| p_k(l) \| \right)^2 \right]} \qquad \text{by (15)} \\
&= \alpha^t \sqrt{\sum_{l=1}^{d} \left(\sum_{k \in V - \overline{F_v}[0]} \| p_k(l) \| \right)^2} \qquad (27)
\end{aligned}
$$

N Proof of Lemma 7

We first prove a claim that will be used in the proof of Lemma 7.

Claim 11 *For $t \geq 1$, define $\mathbf{M}'[t] = \Pi_{\tau=1}^{t} \mathbf{M}[\tau]$. Then, for all nodes $j \in V - \overline{F_v}[t]$, and $k \in \overline{F_v}[0]$, $\mathbf{M}'_{jk}[t] = 0$.*

Proof: The proof is by induction on t.

Induction Basis: Consider the case when $t = 1$. Recall that $V - \overline{F_v}[1] = (V - F) \cup F_v[1]$. Consider any $j \in V - \overline{F_v}[1]$, and $k \in \overline{F_v}[0]$. Then by Lemma 3, $(*, k, 0) \notin \mathit{Verified}_j^c[1]$. Then, due to (10), $\mathbf{M}_{jk}[1] = 0$, and hence $\mathbf{M}'_{jk}[1] = \mathbf{M}_{jk}[1] = 0$.

Induction: Consider $t \geq 2$. Assume that the claim holds true through $t-1$. Then, $\mathbf{M}'_{jk}[t-1] = 0$ for all $j \in V - \overline{F_v}[t-1]$ and $k \in \overline{F_v}[0]$. Recall that $\mathbf{M}'[t-1] = \Pi_{\tau=1}^{t-1} \mathbf{M}[\tau]$.

Now, we will prove that the claim holds true for t. Consider $j \in V - \overline{F_v}[t]$ and $k \in \overline{F_v}[0]$. Note that $\mathbf{M}'[t] = \Pi_{\tau=1}^{t} \mathbf{M}[\tau] = \mathbf{M}[t] \Pi_{\tau=1}^{t-1} \mathbf{M}[\tau] = \mathbf{M}[t] \mathbf{M}'[t-1]$. Thus, $\mathbf{M}'_{jk}[t]$ can be non-zero only if there exists a $q \in V$ such that $\mathbf{M}_{jq}[t]$ and $\mathbf{M}'_{qk}[t-1]$ are both non-zero.

For any $q \in \overline{F_v}[t-1]$, by Lemma (3), $(*, q, t-1) \notin \mathit{Verified}_j^c[t]$. Then, due to (10), $\mathbf{M}_{jq}[t] = 0$ for all $q \in \overline{F_v}[t-1]$. Additionally, by the induction hypothesis, for all $q \in V - \overline{F_v}[t-1]$ and $k \in \overline{F_v}[0]$, $\mathbf{M}'_{qk}[t-1] = 0$. Thus, these two observations together imply that there does not exist any $q \in V$ such that $\mathbf{M}_{jq}[t]$ and $\mathbf{M}'_{qk}[t-1]$ are both non-zero. Hence, $\mathbf{M}'_{jk}[t] = 0$. □

Lemma 7: *For all $i \in V - \overline{F_v}[t]$ and $t \geq 0$, $I_Z \subseteq h_i[t]$.*

Proof: Recall that Z and I_Z are defined in (20) and (21), respectively. We first prove that for all $i \in V - \overline{F_v}[0]$, $I_Z \subseteq h_i[0]$.

Recall that $V - \overline{F_v}[0] = (V - F) \cup F_v[0]$. Now, consider two cases:

- $i \in V - F$:

  We first make the following observations for each fault-free node i:

  - *Observation 1*: $\mathit{Verified}_i^c[0]$ contains at least $f+1$ messages from fault-free nodes (at line 14). This is due to the *Fault-free Integrity* property of the primitive, and the fact that $|\mathit{Verified}_i^c[0]| \geq n - f$ (due to the condition checked in procedure `Proceed` before $\mathit{Verified}_i^c[0]$ is set equal to $\mathit{Verified}_i[0]$).
  - *Observation 2*: $\mathit{Verified}_i^c[0]$ contains tuples of the form $(\mathcal{V}, *, -1)$. We will say that a tuple $(\mathcal{V}, *, -1) \in \mathit{Verified}_i^c[0]$ contains Z if $Z \subseteq \mathcal{V}$. Due to Observation 1, at least $f+1$ tuples in $\mathit{Verified}_i^c[0]$ contain Z, because tuples corresponding to all the fault-free nodes contain Z.
  - *Observation 3*: Observation 2 and the definition of H imply that multiset X defined in in step 3 of Case $t = 0$ of function H contains X_Z defined in Section 5.
  - *Observation 4*: Let A and B be sets of points in the d-dimensional space, where $|A| \geq n - f$, $|B| \geq n - f$ and $A \subseteq B$. Define $h_A := \cap_{C_A \subseteq A, |C_A| = |A| - f} \ \mathcal{H}(C_A)$ and $h_B := \cap_{C_B \subseteq B, |C_B| = |B| - f} \ \mathcal{H}(C_B)$. Then $h_A \subseteq h_B$. This observation follows directly from the fact that every multiset C_A in the computation of h_A is contained in some multiset C_B used in the computation of h_B, and the property of $\mathcal{H}$.

  Now, consider the computation of $h_i[0]$ at line 13. By Observation 3 and Observation 4, $I_Z \subseteq H(\mathit{Verified}_i^c[0], 0) = h_i[0]$.

- $i \in F_v[0]$:

  Suppose that round 0 execution of node i is verified by a fault-free node j. By Claim 5, eventually $\mathit{Verified}_i^c[0] \subseteq \mathit{Verified}_j[0]$. Since node j is fault-free, $\mathit{Verified}_j[0]$, and therefore, $\mathit{Verified}_i^c[0]$, contains messages from at most f faulty nodes. This together with the fact that $|\mathit{Verified}_i^c[0]| \geq n - f$ (by Claim 5), implies that $\mathit{Verified}_i^c[0]$ contains messages from at least $f+1$ fault-free nodes. Then by this observation and the fact that $h_i[0] = H(\mathit{Verified}_i^c[0], 0)$ (by Claim 5), we can show that $I_Z \subseteq h_i[0]$ using the same argument as in the previous case.

Thus, $I_Z \in h_i[0]$ for all $i \in V - \overline{F_v}[0]$.

Now we make several observations for each fault-free node $i \in V - F$:

- As shown above, $I_Z \in h_j[0]$ for all $j \in V - \overline{F_v}[0]$.
- From (13), for $t \geq 1$,
  $$\mathbf{v}[t] = \mathbf{M}^* \mathbf{v}[0]$$
  where $\mathbf{v}_j[0] = h_j[0]$ for $j \in V - \overline{F_v}[0]$.
- By Theorem 1, $\mathbf{v}_i[t] = h_i[t]$.
- Observe that $\mathbf{M}^*$ equals $\mathbf{M}'[t]$ defined in Claim 11. Thus, due to Claim 11, $\mathbf{M}^*_{ik} = 0$ for $k \in \overline{F_v}[0]$ (i.e,. $k \notin V - \overline{F_v}[0]$).

- $\mathbf{M}^*$ is the product of row stochastic matrices; therefore, $\mathbf{M}^*$ itself is also row stochastic. Thus, for fault-free node i, $\mathbf{v}_i[t] = h_i[t]$ is obtained as the product of the i-th row of $\mathbf{M}^*$, namely $\mathbf{M}^*_i$, and $\mathbf{v}[0]$: this product yields a linear combination of the elements of $\mathbf{v}[0]$, where the weights are non-negative and add to 1 (because $\mathbf{M}^*_i$ is a stochastic row vector).

- From (7), recall that $\mathbf{M}^*_i\mathbf{v}[0] = H_l(\mathbf{v}[0]^T\ ;\ \mathbf{M}^*_i)$. Function H_l ignores the input polytopes for which the corresponding weight is 0. Finally, from the previous observations, we have that when the weight in $\mathbf{M}^*[i]$ is non-zero, the corresponding polytope in $\mathbf{v}[0]^T$ contains I_Z. Therefore, the linear combination also contains I_Z.

Thus, I_Z is contained in $h_i[t] = \mathbf{v}_i[t] = \mathbf{M}^*_i\mathbf{v}[0]$. □

O Proof of Theorem 3

Theorem 3: *The output convex polytope at fault-free node i using* Optimal Verified Averaging *is optimal as per Definition 2.*

Proof: Consider set X_Z defined in Section 5. Due to Claim 8 in Appendix E and the fact that set X_Z contains at least $(n-f)$ tuples, at least $(n-2f)$ tuples in X_Z correspond to inputs at fault-free nodes. Let V_Z denote the set of fault-free nodes whose tuples appears in X_Z. Let $S = V - F - V_Z$. Since $|X_Z| \geq n - f$, $|S| \leq f$.

Now consider the following execution of any algorithm ALGO that correctly solves Byzantine convex consensus. Suppose that the faulty nodes in F follow the algorithm correctly except choosing an incorrect input (in acceptable range for inputs). Consider the case when nodes in $V - V_Z$, including fault-free nodes in S, are so slow that the other fault-free nodes must terminate before receiving any messages from the nodes in $V - V_Z$. The fault-free nodes in V_Z cannot determine whether the nodes in $V - V_Z$ are just slow, or faulty (crashed).

Nodes in V_Z must be able to terminate without receiving any messages from the nodes in $V - V_Z$, including fault-free nodes in S. Thus, the output must be in the convex hull of inputs at the fault-free nodes whose tuples are included in X_Z. However, any f of the nodes whose values are in X_Z may be faulty. Therefore, the output obtained by ALGO must be contained in I_Z as defined in Section 5. On the other hand, by Lemma 7, the output obtained using *Optimal Verified Averaging* contains I_Z. This proves the theorem. □